

**NICOLAU LEAL WERNECK**

**ESTIMAÇÃO DE ORIENTAÇÃO DE CÂMERA EM  
AMBIENTES ANTRÓPICOS A PARTIR DE EDGELS**

**São Paulo  
2012**



**NICOLAU LEAL WERNECK**

**ESTIMAÇÃO DE ORIENTAÇÃO DE CÂMERA EM  
AMBIENTES ANTRÓPICOS A PARTIR DE EDGELS**

Tese apresentada à Escola Politécnica da  
Universidade de São Paulo para obtenção do  
Título de Doutor em Ciências.

**São Paulo  
2012**



**NICOLAU LEAL WERNECK**

**ESTIMAÇÃO DE ORIENTAÇÃO DE CÂMERA EM  
AMBIENTES ANTRÓPICOS A PARTIR DE EDGELS**

Tese apresentada à Escola Politécnica da  
Universidade de São Paulo para obtenção do  
Título de Doutor em Ciências.

Área de Concentração:  
Sistemas Digitais

Orientadora:  
Profa. Dra. Anna Helena Reali Costa

**São Paulo  
2012**

## FICHA CATALOGRÁFICA

Werneck, Nicolau Leal

Estimação de orientação de câmera em ambientes antrópicos a partir de edgels / N.L. Werneck. -- São Paulo, 2012.

244 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Inteligência artificial 2.Visão computacional 3. Reconhecimento de padrões 4. Processamento de imagens I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II. t.

# AGRADECIMENTOS

Para a Professora Dra. Anna Helena Reali Costa, pela melhor orientação que eu poderia desejar. A Professora Anna me ensinou muito, o que não é pouco.

À CAPES pela bolsa de doutorado, e à USP e ao CNPq por outras formas de financiamento e disponibilização de recursos utilizados nesta pesquisa.

Para Carina, quasares do meu céu profundo, que me acompanhou como Virgílio por todos os círculos desta formidável jornada.

Para a minha família e amigos, pelos indispensáveis apoio e incentivo, especialmente meus pais e meu irmão pela influência na escolha do tema desta tese, e ainda Sylvio e Giocondo *in memoriam*.

Aos professores Jaime Sichman e Anarosa Brandão, e demais companheiros do LTI: Antônio, Valdinei, Guillaume, Sara, Alan, Allan Diego, Diego, Tiago, Gustavo, Rodrigo, Fábio, Annibal, Rafael, Kadu, Marcelo, e principalmente Flávio e Camila pelas colaborações. E aos colegas dos corredores da USP pelas muitas conversas iluminadoras, principalmente Yannick e os professores João José Neto, João Kögler e Humberto Sandmann.

A todos professores e funcionários da USP pelo estimado trabalho, e alunos que pude acompanhar durante o curso.

To all the great people I met during my internship at Google, specially Dan Bloomberg, Jeff Breidenbach, David Gluss, Paul, Dany, Nemanja, David E., Rika, Oded, Kristy.

To all researchers and software developers from all around the world that I was able to interact with during this research, or who provided me data or software that I used. And a special thanks to all free software developers.



*In this phantasy of prismatic distortion it moved anomalously  
in a diagonal way, so that all the rules of matter and  
perspective seemed upset.* —**The Madness from the Sea,**  
H. P. Lovecraft

*A estranha ordem geométrica de tudo.*  
—**A Máquina do Mundo**, Carlos Drummond de Andrade



# RESUMO

Esta tese apresenta o *Corisco*, um método para estimar a orientação de uma câmera a partir de uma única imagem capturada de um ambiente antrópico. O *Corisco* foi desenvolvido com o objetivo de atender às necessidades de aplicações de Robótica Móvel e da análise de grandes conjuntos de imagens, o que significa que o método deve não só apresentar um bom desempenho computacional, mas também deve poder utilizar diferentes modelos de câmera, permitir realizar um comprometimento entre a velocidade de cálculo e precisão dos resultados, e ainda deve poder tanto aproveitar estimativas iniciais da solução, quanto dispensá-las. O *Corisco* apresenta todas estas características. Os ambientes considerados possuem um sistema referencial natural com três eixos ortogonais, e contêm conjuntos de retas paralelas a estes eixos. A orientação estimada é uma rotação tridimensional entre o referencial natural e o sistema referencial da câmera. O *Corisco* requer o conhecimento do modelo de câmera para funcionar, mas qualquer modelo de câmera pode ser utilizado, e não apenas a projeção perspectiva. As imagens são analisadas no *Corisco* por um processo de extração de edgels, que são pontos localizados sobre as projeções das retas do ambiente, associados à direção tangencial da projeção da reta naquele ponto. Esta extração de edgels utiliza uma máscara em forma de grade que permite sub-amostrar os dados, criando um comprometimento entre velocidade e precisão. A orientação é estimada através de um processo de otimização em dois passos que minimiza uma função objetivo definida com a técnica de M-estimação, com uma função de erro redescendente. Esta técnica substitui a aplicação de estimação MAP ou ME nos métodos similares existentes. O primeiro passo da otimização utiliza o algoritmo RANSAC, permitindo ao *Corisco* funcionar sem estimativas iniciais, e o segundo passo é uma otimização contínua com restrições que permite ao *Corisco* parametrizar a orientação utilizando um quaternion. O *Corisco* foi testado com diferentes modelos de câmera, incluindo a projeção perspectiva, um modelo com distorção radial, e duas projeções onidirecionais, a polar equidistante e a equiretangular. A comparação dos resultados obtidos com o *Corisco* e com métodos alternativos atingiram níveis de precisão próximos a 2°. O tempo médio de cálculo pode ser controlado através de um par de parâmetros, o que pode requerer sacrificar a precisão. As durações observadas nos experimentos foram de 0.3 até 60 segundos. As características apresentadas pelo *Corisco* atendem os objetivos estabelecidos, e os resultados dos experimentos foram satisfatórios e motivadores.

**Palavras-chave:** Inteligência artificial. Visão computacional. Reconhecimento de padrões. Processamento de imagens.



# ABSTRACT

This thesis presents *Corisco*, a method to estimate the orientation of a camera from a single image captured from an anthropic environment. *Corisco* was developed with the objective of answering the needs of Mobile Robotics applications, and of the analysis of large set of images, what means the method should not only present a good computational performance, but must also be able to use different camera models, allow to control the compromise between calculation speed and result precision, and must also be capable of both exploiting initial estimated of the result, and of operating without any initial estimates. *Corisco* presents all of these characteristics. The considered environments have a natural reference system with three orthogonal axes, and contain sets of lines parallel to these axes. The estimated orientation is a three-dimensional rotation between the natural frame and the camera reference frame. *Corisco* requires the knowledge of the camera model to work, but any camera model can be used, and not just the perspective projection. The images are analyzed in *Corisco* by a process that extracts edgels, which are points located over the projections of the environment lines, associated with the tangential direction of the line projection at that point. This edgel extraction uses a grid mask that can sub-sample the data, creating a compromise between speed and precision. The orientation is estimated through a two-step optimization process that minimizes an objective function defined with the technique of M-estimation, using a redescending error function. This technique replaces the application of the MAP or EM estimation in the similar existing methods. The first optimization step uses the RANSAC algorithm, allowing *Corisco* to work without initial estimates, and the second step is a continuous and constrained optimization that allows *Corisco* to parameterize the orientation using a quaternion. *Corisco* was tested with different camera models, including the perspective projection, a model with radial distortion, and two omnidirectional projections, the polar equidistant and the equirectangular. The comparison between the results obtained with *Corisco* and with alternative methods reached precision levels near  $2^\circ$ . The mean calculation time can be controlled through a couple of parameters, what may require sacrificing precision. The durations observed in the experiments ranged from 0.3 to 60 seconds. The characteristics presented by *Corisco* meet the established objectives, and the experiment results were satisfactory and motivating.

**Keywords:** Artificial intelligence. Computer vision. Pattern recognition. Image processing.



# LISTA DE FIGURAS

|     |  |    |
|-----|--|----|
| 1.1 | Uma imagem capturada por uma câmera da sonda Spirit (esq.), e o modelo de elevação obtido nesta ocasião (dir.). A pedra próxima ao centro da foto corresponde à mancha distinta próxima ao centro do modelo. Imagem retirada de (MAIMONE; LEGER, 2007). . . . .  | 32 |
| 1.2 | Exemplos de segmentos de reta, edgels e linhas retas que compartilham alguns parâmetros. . . . .   | 41 |
| 1.3 | À esquerda, um exemplo de rastreamento de contorno, reproduzido do livro de Blake e Isard (1998). O contorno é rastreado a partir de pontos individuais contidos dentro dele, que são edgels. Os pontos sobre as linhas escuras ortogonais ao contorno são edgel extraídos da imagem, porém suas direções não estão representadas no gráfico. À direita, uma imagem reproduzida do artigo de Eade e Drummond (2006) onde edgels são rastreados sobre as imagens para realizar a localização de câmera e modelagem do ambiente. Neste gráfico cada edgel é representado através de um pequeno segmento de reta. . . . . | 43 |
| 1.4 | Exemplo de edgels sendo utilizados no reconhecimento de um protótipo do caractere 'h', reproduzido de Smith (2007). . . . .  | 44 |
| 1.5 | Uma imagem criada com a projeção estereográfica, à esquerda, e à direta a imagem retificada, ou seja, com projeção perspectiva. Apesar das linhas retas na imagem da direita, existem algumas características inconvenientes. Imagem reproduzida de (FLECK, 1995). . . . .   | 46 |
| 2.1 | Objeto em um ambiente refletindo raios de luz capturados por uma câmera, e a imagem resultante. . . . .  | 57 |
| 2.2 | Modelo de câmera <i>pinhole</i> (esq.) e de uma lente olho-de-peixe (dir.). . .  | 59 |
| 2.3 | Homem desenhando um alaúde, Albert Dürer (1525). . . . .   | 60 |
| 2.4 | Imagens de câmeras comerciais, que exemplificam o modelo de câmera <i>pinhole</i> . Estas imagens possuem na realidade distorções muito sutis, difíceis de observar. . . . .   | 61 |
| 2.5 | Uma imagem obtida com uma lente grande-angular, com distorção radial aparente (esq.) (CERIANI et al., 2009), e uma imagem com distorções intensas obtida através de uma lente olho-de-peixe com mais de 180 graus de campo de visão (dir.). . . . .  | 61 |
| 2.6 | Uma imagem com projeção equiretangular, criada a partir de um arranjo de câmeras montado sobre um carro (fonte: Google Inc., projeto StreetView). .  | 62 |
| 2.7 | Demonstração de edgels retirados de uma grade, distorcidos pelo modelo de Harris em ambos sentidos. . . . .  | 66 |

---

|      |   |     |
|------|---|-----|
| 2.8  | Comparação da projeção de Harris com diferentes valores de $\kappa$ , e a projeção polar equidistante. . . . .  | 68  |
| 2.9  | Projeção de um ponto contido em uma reta do ambiente produzindo um edgel na imagem. . . . .   | 70  |
| 2.10 | Imagen de um poliedro, com regiões de cor uniforme limitadas por bordas. . . . .  | 74  |
| 2.11 | Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente. . . . .   | 78  |
| 2.12 | Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente. . . . .   | 78  |
| 2.13 | Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente. . . . .   | 79  |
| 2.14 | Valores de cor de uma imagem amostrados sobre uma linha que corta uma borda. As curvas em cada canal exibem a forma de um degrau cuja descontinuidade se encontra no local onde a borda corta a linha percorrida. O sentido de variação destes degraus varia entre diferentes canais neste exemplo. . . . .   | 80  |
| 2.15 | Retas paralelas produzindo um ponto de fuga através da projeção pontual. . . . .  | 83  |
| 2.16 | Evolução da estimação dos parâmetros de uma mistura de duas gaussianas em duas dimensões (WIKIPEDIA, 2012). . . . .   | 96  |
| 2.17 | Diferentes funções $\rho$ que podem ser utilizadas para realizar M-estimação. . . . .   | 99  |
| 2.18 | À esquerda, uma função Gaussiana não-normalizada comparada com a função de pesos correspondente à função biquadrada de Tukey com $s = 1$ . E à direita, as funções biquadrada de Tukey com $s = 1$ e a função de erro que resulta do uso da distribuição Gaussiana como uma função de pesos no algoritmo IRLS. . . . .  | 101 |
| 3.1  | Diferentes abordagens para a medição de erros. No primeiro caso os erros são distâncias entre pontos sobre a imagem. No segundo caso os erros são angulares, relacionados a diferenças entre direções medidas sobre diferentes pontos da imagem. No terceiro caso os erros são distâncias entre pontos e retas, incluindo o caso em que há segmentos de retas parametrizados apenas por um par de pontos. O último caso é o alinhamento espacial, que leva em consideração o afastamento dos vetores normais de cada reta do plano tridimensional ao qual deveriam se restringir. . . . . | 136 |

---

|      |  |     |
|------|--|-----|
| 4.1  | Diagrama de blocos do processo desenvolvido que constitui o <i>Corisco</i> . Os parâmetros de orientação $\Psi$ ótimos são encontrados por um processo de otimização iterativo e de dois estágios. A função de erro que é otimizada utiliza parâmetros obtidos a partir do modelo de câmera e dos edgels extraídos da imagem de entrada. A classificação dos edgels também pode ser obtida como resultado do processo. . . . .   | 144 |
| 4.2  | Exemplo da análise da imagem e do resultado do processo de estimação de orientação. No topo e à esquerda encontra-se a imagem de entrada, e à direita a intensidade do gradiente. Em baixo e à esquerda encontra-se a saída do extrator de edgels, e à direita algumas direções preditas a partir da orientação estimada. . . . .  | 146 |
| 4.3  | Detalhe dos edgels extraídos da Figura 4.2. . . . .  | 147 |
| 4.4  | Exemplos do resultado da estimação de orientação pelo <i>Corisco</i> . . . . .   | 148 |
| 4.5  | Exemplos do resultado da estimação de orientação pelo <i>Corisco</i> para mais uma imagem da base de dados YorkUrbanDB, e um exemplo com uma imagem de projeção polar equidistante, obtida com uma lente olho-de-peixe.  | 149 |
| 4.6  | Classificação de edgels e direções preditas, calculadas pelo <i>Corisco</i> a partir da solução encontrada para as imagens da Figura 4.5. Cada gráfico contém dados relativos a apenas uma das direções possíveis do ambiente, permitindo uma visualização melhor. A localização aproximada dos pontos de fuga foi assinalada nos gráficos das direções preditas. No caso da imagem de projeção aproximadamente há duas direções aproximadamente paralelas ao plano da imagem, e apenas um ponto de fuga pode ser visto. No caso da imagem da lente olho-de-peixe é possível observar ambos pontos de fuga para as duas primeiras direções nos dois gráficos mais abaixo e à esquerda, enquanto para a terceira direção o ponto de fuga se localiza aproximadamente no centro da imagem. . . . . | 150 |
| 4.7  | Máscara em forma de grade e edgels extraídos da imagem da Figura 2.11. . . . .   | 152 |
| 4.8  | Máscara em forma de grade e edgels extraídos da imagem da Figura 2.12. . . . .   | 152 |
| 4.9  | Valores das componentes do gradiente e posições onde edgels foram detectados na segunda linha varrida na imagem da Figura 4.7. . . . .   | 154 |
| 4.10 | Valores das componentes do gradiente e posições onde edgels foram detectados na quarta linha varrida na imagem da Figura 4.8. . . . .  | 155 |
| 4.11 | Criação de uma função de erro robusta em um modelo de mistura com Gaussiana e uma distribuição uniforme. A curva contínua é o erro quadrático convencional obtido de uma distribuição Gaussiana pura. A curva tracejada é obtida de um modelo de mistura de uma Gaussiana e uma distribuição uniforme. . . . .   | 166 |
| 4.12 | Exemplos de uma trajetória seguida em uma otimização por FilterSQP. . .  | 180 |

---

|      |   |     |
|------|---|-----|
| 4.13 | Trajetórias de passos possíveis na minimização de um modelo quadrático para dois casos diferentes de matriz Hessiana. O círculo representa a estimativa de solução atual, e a próxima estimativa se localizará em algum ponto da trajetória.  | 184 |
| 4.14 | Passos de uma iteração do método SQP.   | 188 |
| 4.15 | Linhas extraídas de uma imagem de projeção equiretangular utilizando o método de extração proposto, baseado no <i>Corisco</i> . As cores indicam a classe de cada linha de acordo com sua direção no ambiente.  | 197 |
| 5.1  | Distribuições de erro e tempo de cálculo exibidos pelo <i>Corisco</i> na análise do YorkUrbanDB com 10.000 (gráfico de cima) e 1.000 (gráfico de baixo) iterações na etapa RANSAC. As distribuições estão representadas através de diagramas de caixa, que indicam a mediana, os pontos de primeiro e terceiro quartil, e limites. Os pontos isolados em cada linha são amostras individuais que apresentaram valores extremos de erro, e foram plotadas como se fossem <i>outliers</i> . No gráfico de tempo as barras de erro mostram o desvio padrão das medições, e os pontos os valores médios. A curva tracejada indica o tempo gasto apenas pelo passo RANSAC. | 204 |
| 5.2  | Distribuições de erro e tempo de cálculo exibidos pelo <i>Corisco</i> na análise do YorkUrbanDB com 200 iterações na etapa RANSAC (gráfico de cima), e utilizando um erro quadrático com 1000 iterações do RANSAC (gráfico de baixo).   | 205 |
| 5.3  | Duas imagens do conjunto ApaSt. A da esquerda, em orientação paisagem, foi obtida com a câmera Sony α230, e a da direita, em orientação retrato, com o Nokia N8.  | 208 |
| 5.4  | Modelo tridimensional do ambiente reconstruído pelo <i>Bundler</i> . Trata-se de uma nuvem de pontos que delineiam os contornos dos edifícios observados na cena.   | 208 |
| 5.5  | Valores das componentes dos quaternions de referência transformados, e dos quaternions estimados pelo <i>Corisco</i> para as imagens da série ApaSt.  | 210 |
| 5.6  | Distribuição de erros de estimação do <i>Corisco</i> para as imagens do conjunto ApaSt analisadas com $i = 10.000$ e $g = 1$ .  | 211 |
| 5.7  | Distribuições de erro e tempo de cálculo exibidos pelo <i>Corisco</i> na análise da base ApaSt com 10.000 (gráfico de cima) e 1.000 (gráfico de baixo) iterações na etapa RANSAC. Nestes diagramas de caixotes os limites inferiores mostram a posição da amostra de menor erro, e os limites superiores a posição da quarta pior amostra.  | 212 |
| 5.8  | Distribuições de erro e tempo de cálculo exibidos pelo <i>Corisco</i> na análise da base ApaSt com 200 iterações na etapa RANSAC.   | 213 |

---

|   |     |
|---|-----|
| 5.9 Valor da função objetivo do <i>Corisco</i> na solução encontrada para diferentes valores de distância focal. Os gráficos à esquerda se referem à câmera Nokia N8, e os da direita à câmera <i>Sony α230</i> . Os gráficos do topo mostram a soma dos valores para todas as imagens de cada câmera, e os gráficos abaixo mostram exemplos de curvas obtidas para uma única imagem de cada subconjunto. . . . . | 215 |
| 5.10 Valores das componentes dos quaternions de referência transformados, e dos quaternions estimados para as imagens da série estudada. . . . .  | 218 |
| 5.11 Curva de distribuição de erro para as orientações estimadas com o <i>Corisco</i> e com um modelo de orientação constante da sequência de imagens equirretangulares do Google StreetView. . . . .   | 219 |
| 5.12 Imagem obtida com lente olho-de-peixe. Esta imagem é a fonte utilizada para a sintetização das imagens retificadas sem e com estimativa da orientação das Figuras 5.13 e 5.14. . . . .   | 221 |
| 5.13 Imagens retificadas sem estimativa de orientação. Estas imagens simplesmente utilizam o modelo de câmera para produzir imagens com projeção perspectiva a partir da imagem da Figura 5.12. Os eixos a que estão alinhados cada plano destes exemplos são os próprios eixos do referencial de câmera da imagem original. . . . .  | 223 |
| 5.14 Imagens retificadas com estimativa de orientação. Assim como na Figura 5.13 a imagem da Figura 5.12 foi utilizada como fonte para produzir imagens retificadas, com projeção perspectiva, porém desta vez os planos de cada imagem de saída estão alinhados às direções do referencial natural do ambiente, estimadas pelo <i>Corisco</i> . . . . .  | 224 |



# LISTA DE TABELAS

|     |   |     |
|-----|---|-----|
| 2.1 | Funções de erro utilizadas na M-estimação. . . . .  | 98  |
| 3.1 | Tabela comparativa do método proposto, o <i>Corisco</i> , e alternativas existentes. O símbolo $\circ$ indica que o método possui a propriedade da coluna correspondente, e o símbolo $\times$ indica a ausência da propriedade. . . . .  | 133 |
| 5.1 | Desempenho apresentado pelo <i>Corisco</i> e outros métodos na análise do YorkUrbanDB. Diferentes combinações de tamanho de grade $g$ e número de iterações do RANSAC $i$ foram testadas no caso do <i>Corisco</i> . Para cada método foi medido o tempo médio de cálculo da estimativa de orientação e levantada uma distribuição do erro de predição, determinado a partir de orientações de referência contidas no banco de dados. . . . . | 201 |



# LISTA DE SIGLAS

**MAP** — Probabilidade máxima *a posteriori*

**EM** — Algoritmo Esperança-Maximização (*expectation maximization algorithm*)

**RANSAC** — *Random sample consensus*

**SFM** — Estrutura a partir do movimento (*structure from motion*)

**SLAM** — Mapeamento e localização simultâneos (*simultaneous localization and mapping*)

**PDF** — Função de densidade de probabilidade (*probability density function*)

**CDF** — Função de densidade de probabilidade acumulada



# LISTA DE SÍMBOLOS

- $y^x$  — A componente  $x$  do vetor  $y$ .
- $y_x$  — O  $x$ -ésimo vetor  $y$  em uma lista de vetores.
- $q$  — Um vetor no espaço tridimensional, no referencial da câmera.
- $p$  — Uma posição sobre a imagem. Um mapeamento entre pontos  $q$  e suas projeções  $p$  na imagem constitui um modelo de câmera.
- $J$  — Matriz Jacobiana de um modelo de câmera.
- $f$  — Distância focal de um modelo de câmera.
- $\kappa$  — Parâmetro de distorção radial de um modelo de câmera.
- $\Psi$  — Um quaternion que modela a rotação entre o sistema referencial da câmera e o referencial natural do ambiente, ao qual se alinham as retas presentes no ambiente.
- $r_k$  — Um vetor tridimensional na direção  $k$ . Calculado a partir dos componentes de  $\Psi$ .
- $p_n$  — As coordenadas de imagem do edgel extraído com índice  $n$ .
- $v_n$  — Direção do edgel extraído com índice  $n$ .
- $v_{nk}$  — Direção calculada da projeção em  $p_n$  de uma reta na direção  $r_k$  no espaço. Calculada a partir da matriz Jacobiana da projeção.
- $u_n$  — Direção ortogonal a  $v_n$ , paralela ao gradiente da imagem na posição  $p_n$ .
- $n_n$  — Vetor normal do plano de interpretação de um edgel ou reta definidos sobre a imagem. Calculado a partir de  $p_n$ ,  $v_n$  e do modelo de câmera.
- $k$  — Classes das retas do ambiente e de seus edgels correspondentes. Em um ambiente antrópico há apenas três classes e os vetores  $r_k$  são mutuamente ortogonais.
- $c_n$  — Classe do edgel de índice  $n$ .
- $\rho(x)$  — Função de erro. Necessariamente uma função com simetria par e monotonicamente crescente para valores de entrada positivos. Por exemplo, a função biquadrada de Tukey.
- $s$  — Parâmetro de escala de uma função  $\rho$ .
- $p(a|b)$  — Probabilidade condicional de  $a$  dado  $b$ . Por exemplo, probabilidade de observar a evidência  $a$  dado o parâmetro de modelo  $b$ .
- $L(b; a)$  — Verossimilhança dos parâmetros  $b$  dada a evidência  $a$ .
- $F(\Psi)$  — Função de erro que depende dos parâmetros a serem estimados, além dos dados extraídos, do modelo de camera e outros fatores. O  $\Psi$  que minimiza esta função é a estimativa ótima da orientação.



# SUMÁRIO

|              |  |     |
|--------------|--|-----|
| <b>1</b>     | <b>Introdução</b>  | 31  |
| <b>1.1</b>   | <b>Panorama de Visão Computacional</b>                                 | 31  |
| <b>1.1.1</b> | <i>Modelos e restrições</i>  | 34  |
| <b>1.1.2</b> | <i>Extração de características</i>                                     | 36  |
| <b>1.1.3</b> | <i>Edgels</i>  | 40  |
| <b>1.2</b>   | <b>Proposta da tese</b>  | 44  |
| <b>1.2.1</b> | <i>Motivações</i>  | 45  |
| <b>1.2.2</b> | <i>Objetivos</i>   | 46  |
| <b>1.2.3</b> | <i>Justificativas</i>  | 48  |
| <b>1.2.4</b> | <i>Principais contribuições</i>  | 50  |
| <b>1.3</b>   | <b>Estrutura do documento</b>  | 52  |
| <b>2</b>     | <b>Fundamentos técnicos</b>  | 55  |
| <b>2.1</b>   | <b>Formação de imagens</b>   | 56  |
| <b>2.1.1</b> | <i>Exemplos de modelo de câmera</i>                                    | 58  |
| <b>2.1.2</b> | <i>Equações de modelos de câmera</i>                                   | 62  |
| <b>2.1.3</b> | <i>Cálculo da projeção de um edgel</i>                                 | 69  |
| <b>2.2</b>   | <b>Segmentação e extração de bordas</b>                                | 72  |
| <b>2.2.1</b> | <i>Detecção de borda</i>   | 75  |
| <b>2.2.2</b> | <i>Extração de retas</i>   | 81  |
| <b>2.2.3</b> | <i>Extração de pontos de fuga</i>                                      | 82  |
| <b>2.3</b>   | <b>Estimação simultânea de parâmetros e associações de observações</b> | 84  |
| <b>2.3.1</b> | <i>RANSAC</i>  | 84  |
| <b>2.3.2</b> | <i>Transformada de Hough</i>   | 88  |
| <b>2.3.3</b> | <i>Outras técnicas</i>   | 89  |
| <b>2.4</b>   | <b>Estimação de parâmetros por otimização contínua</b>                 | 91  |
| <b>2.4.1</b> | <i>Esperança da verossimilhança</i>                                    | 94  |
| <b>2.4.2</b> | <i>M-estimação</i>   | 97  |
| <b>3</b>     | <b>Trabalhos correlatos</b>  | 103 |
| <b>3.1</b>   | <b>Técnicas de visão baseadas em pontos</b>                            | 104 |
| <b>3.1.1</b> | <i>Extração de pontos de interesse</i>                                 | 106 |
| <b>3.1.2</b> | <i>Aplicações baseadas em pontos</i>                                   | 108 |
| <b>3.2</b>   | <b>Técnicas de visão baseadas em bordas</b>                            | 112 |
| <b>3.2.1</b> | <i>Extração de ponto de fuga</i>                                       | 114 |
| <b>3.2.2</b> | <i>Aplicações baseadas em linhas</i>                                   | 117 |

---

|            |   |     |
|------------|---|-----|
| 3.2.3      | <i>A busca por estrutura</i>  | 122 |
| <b>3.3</b> | <b>Estimação de orientação a partir de edgels e comparações com o Corisco</b> | 123 |
| 3.3.1      | <i>O Corisco comparado a técnicas diretamente relacionadas</i>                | 127 |
| 3.3.2      | <i>O Corisco considerado em um contexto mais amplo</i>                        | 131 |
| 3.3.3      | <i>Retrospectiva</i>  | 138 |
| <b>4</b>   | <b>Detalhamento do Corisco</b>  | 143 |
| <b>4.1</b> | <b>Extração de edgels</b>   | 151 |
| 4.1.1      | <i>Procedimento para a extração de edgels</i>                                 | 153 |
| 4.1.2      | <i>Consideração sobre o espaçamento da grade</i>                              | 156 |
| <b>4.2</b> | <b>Função da direção de um edgel</b>  | 157 |
| 4.2.1      | <i>Parametrização da orientação da câmera</i>                                 | 158 |
| 4.2.2      | <i>Cálculo das direções preditas dos edgels</i>                               | 160 |
| <b>4.3</b> | <b>Cálculo da função objetivo</b>   | 161 |
| 4.3.1      | <i>Aplicações de MAP e EM à estimação de orientação a partir de edgels</i>    | 162 |
| 4.3.2      | <i>Função objetivo utilizada no Corisco</i>                                   | 169 |
| <b>4.4</b> | <b>Estimação de parâmetros por busca aleatória</b>                            | 174 |
| <b>4.5</b> | <b>Estimação de parâmetros por otimização não-linear</b>                      | 178 |
| 4.5.1      | <i>Otimização pelo método de Newton com região de confiança</i>               | 181 |
| 4.5.2      | <i>Otimização com restrições e o algoritmo FilterSQP</i>                      | 184 |
| 4.5.3      | <i>Derivadas da função objetivo</i>   | 188 |
| <b>4.6</b> | <b>Revisão e algumas aplicações imediatas</b>                                 | 191 |
| 4.6.1      | <i>Rascunho do algoritmo</i>  | 193 |
| 4.6.2      | <i>Estimação de parâmetros intrínsecos</i>                                    | 196 |
| 4.6.3      | <i>Extração de linhas por agrupamento de edgels</i>                           | 196 |
| <b>5</b>   | <b>Experimentos</b>   | 199 |
| <b>5.1</b> | <b>Análise do banco de dados YorkUrbanDB</b>                                  | 200 |
| <b>5.2</b> | <b>Comparação com um método baseado em pontos</b>                             | 207 |
| <b>5.3</b> | <b>Estimação de distância focal baseada no Corisco</b>                        | 213 |
| <b>5.4</b> | <b>Análise de imagens de projeção equiretangular</b>                          | 217 |
| <b>5.5</b> | <b>Análise de imagens de projeção polar equidistante</b>                      | 219 |
| <b>5.6</b> | <b>Discussões finais</b>  | 222 |
| <b>6</b>   | <b>Conclusão</b>  | 227 |
| <b>6.1</b> | <b>Principais contribuições alcançadas com o Corisco</b>                      | 228 |
| <b>6.2</b> | <b>Trabalhos futuros</b>  | 230 |
| <b>6.3</b> | <b>Considerações finais</b>   | 230 |





# 1 Introdução

*Now here we go dropping science dropping it all over. (...)*

*Postulating theorems formulating equations. (...)*

*Dropping science like when Galileo dropped the orange.*

—**The Sounds of Science**, Adam Yauch et alii

A pesquisa relatada nesta tese se insere na área de Visão Computacional, mais especificamente na área em que se aplicam técnicas de Processamento de Sinais e Reconhecimento de Padrões para realizar uma extração de características geométricas das imagens sendo analisadas e então estimar parâmetros relacionados à estrutura do ambiente, ao posicionamento das câmeras no espaço e ao modelo da câmera. Esta área da Visão Computacional lida com questões puramente espaciais e geométricas, o que se contrapõe a problemas de mais alto nível tal como o reconhecimento de objetos ou classificação de imagens.

A Seção 1.1 a seguir traça um breve panorama desta área da Visão Computacional. A Seção 1.2 descreve então dentro deste contexto, e de forma introdutória, o método que foi desenvolvido durante esta pesquisa, ressaltando as contribuições alcançadas. A Seção 1.3 descreve por fim a estrutura do restante deste documento.

## 1.1 Panorama de Visão Computacional

As entidades envolvidas nos problemas puramente geométricos de Visão Computacional são objetos situados no espaço tridimensional, compondo o *ambiente*, e câmeras que capturam imagens deste ambiente. O que se deseja é determinar os parâmetros dos modelos destas entidades utilizando como dados de entrada principais as imagens obtidas pelas câmeras. Para isto são exploradas as relações funcionais existentes entre as imagens formadas e a localização e orientação dos objetos e da câmera no espaço, além do chamado *modelo de câmera* que rege a forma como pontos do espaço são projetados sobre o plano da imagem. O processo de formação de imagens será descrito com mais detalhe no Capítulo 2. Os parâmetros que modelam a posição e orientação da câmera são também chamados os seus *parâmetros extrínsecos*, ou *externos*, enquanto os parâmetros relativos ao modelo de câmera e outros fatores restantes que podem influenciar na formação da imagem constituem os *parâmetros intrínsecos*, ou *internos* de uma câmera.

Estes problemas de natureza espacial são resolvidos através de algum tipo de processo de estimação de parâmetros, que pode utilizar fórmulas fechadas ou algoritmos iterativos de otimização. Diferentes situações surgem de acordo com o quanto se conhece a princípio acerca das diferentes entidades envolvidas no problema, e quais são os parâmetros desconhecidos que precisam ser estimados. Alguns exemplos de situações comuns são:

- *Reconstrução* — Quando todos os parâmetros de um arranjo de câmeras são conhe-

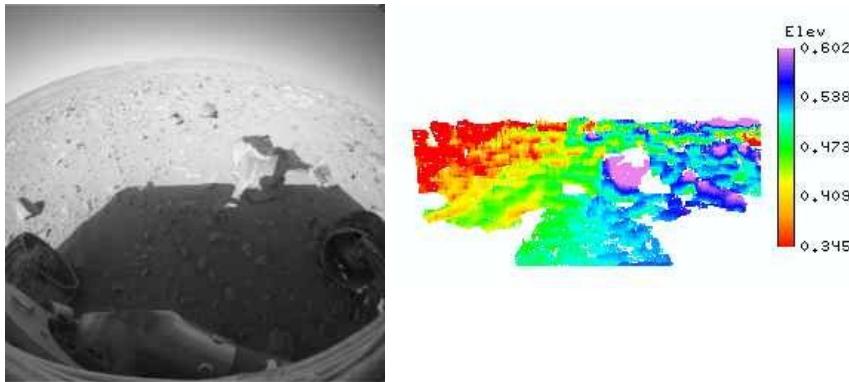


Figura 1.1: Uma imagem capturada por uma câmera da sonda Spirit (esq.), e o modelo de elevação obtido nesta ocasião (dir.). A pedra próxima ao centro da foto corresponde à mancha distinta próxima ao centro do modelo. Imagem retirada de (MAIMONE; LEGER, 2007).

cidos temos o problema básico de reconstrução do ambiente, onde um conjunto de imagens é analisada para se encontrar os parâmetros de um modelo do ambiente. Este é o caso para muitas aplicações bem-sucedidas de Visão Computacional, tal como a criação de sensores visuais para o controle de máquinas industriais, sistemas de captura de movimentos para uso em Computação Gráfica, ou ainda arranjos estereoscópicos para a modelagem de ambientes. A Figura 1.1 mostra um exemplo deste último caso. Trata-se de um par de câmeras de parâmetros conhecidos utilizadas para obter modelos de elevação de solo, instaladas nas sondas robóticas enviadas pela NASA para explorar Marte.

- *Localização* — No problema da localização da câmera são conhecidos tanto os parâmetros internos da câmera quanto de um modelo do ambiente. O modelo do ambiente por ser, por exemplo, uma lista de marcos pontuais, um conjunto de poliedros ou uma planta baixa do interior de uma casa. Dadas estas informações resta apenas estimar a posição da câmera. O processo de estimativa geralmente começa com uma análise das imagens que identifica nelas as projeções de elementos que fazem parte do modelo do ambiente. Este é o problema usualmente encontrado no funcionamento de um robô móvel, por exemplo, para determinar a sua localização conforme ele se desloca por um ambiente na realização de alguma tarefa qualquer. A localização da câmera também é necessária em aplicações de Realidade Aumentada, onde figuras artificiais são superpostas às imagens de maneira a simular a existência de um objeto na cena. O problema de localizar um objeto conhecido que se movimenta no espaço em relação a uma câmera parada constitui um problema equivalente a este.
- *Calibração* — Certas técnicas requerem o conhecimento do modelo de câmera para serem utilizadas, como costuma ser o caso nos dois cenários de aplicação citados anteriormente. Isto torna necessário um procedimento de calibração que permita

analisar um conjunto de imagens e, a partir de modelos de ambiente e de localização de câmera conhecidos ou estimados, calcular os parâmetros do modelo de câmera que podem então ser utilizados por outros métodos. Uma forma de fazer isto é construir um alvo de calibração que produza imagens fáceis de analisar, permitindo uma fácil localização de marcos visuais, por exemplo. Em alguns casos é possível apenas dispor o alvo em uma posição pré-determinada, ou obter de alguma outra forma a informação da sua localização em relação à câmera, e então determinar apenas os parâmetros do modelo de câmera após a análise da imagem. Mas geralmente é melhor estimar simultaneamente os parâmetros intrínsecos e a localização do alvo em relação à câmera, a partir dos mesmos dados retirados das imagens, porque é raro poder confiar na qualidade da construção de um alvo de calibração ou na precisão de estimativas externas de localização.

Mesmo quando a localização também é calculada dentro do processo, além dos parâmetros intrínsecos, o conhecimento do formato do alvo de calibração ainda torna este problema bem mais simples do que o caso em que não se conhece nada a respeito da cena. Por exemplo, a análise das imagens pode ser bastante facilitada quando se utiliza um padrão xadrez como alvo e o número de quadrados e as suas posições relativas são conhecidas. Por mais que um processo de calibração possa envolver a estimação de parâmetros extrínsecos, é a estimação dos parâmetros intrínsecos de uma câmera que constitui o seu principal objetivo.

- *Rastreamento* — Várias aplicações de Visão Computacional se caracterizam por ter parâmetros que variam de forma contínua ao longo do tempo. É o que ocorre no casos do rastreamento de pessoas em sistemas de vigilância, por exemplo, ou no rastreamento de câmera, onde são conhecidos o modelo do ambiente, o modelo de câmera e ainda se possui uma estimativa inicial da posição da câmera. O que se deseja é determinar a movimentação da câmera ao longo do tempo a partir disto. Se os parâmetros de localização de um objeto ou da câmera são conhecidos em um determinado instante de tempo, esta informação pode ser utilizada para facilitar a estimação em instantes futuros, permitindo a aplicação de algoritmos específicos para realizar uma atualização dos dados. Isto contrasta com casos onde não há qualquer forma de estimativa inicial, como a reconstrução e a localização puras, que requerem algoritmos mais poderosos.
- *Rastreamento de câmera em ambientes desconhecidos* — Neste caso não se conhece a princípio nada ou quase nada a respeito do ambiente ou da movimentação da câmera, e conforme esta se desloca pelo ambiente um modelo é construído, e este mesmo modelo do ambiente em atualização constante é utilizado para estimar as posições da câmera ao longo do tempo. Como a câmera se desloca constantemente, é possível utilizar técnicas de rastreamento, e são geralmente empregadas

câmeras calibradas. Na área de robótica é possível encarar isto como uma instância, utilizando câmeras, do chamado problema de *localização e mapeamento simultâneos*, também conhecido pela sigla SLAM do inglês *simultaneous localization and mapping*.

- *Problema geral* — O cenário mais desafiador possível é aquele no qual não se conhece nada a respeito do ambiente ou das câmeras que produziram cada imagem de entrada. Existem até casos em que não se sabe nem mesmo se as imagens sendo analisadas são fotografias tiradas de um mesmo ambiente, mas isto já envolveria um problema superior de classificação de imagens. Apesar da maior complexidade, este é um problema que se pode resolver com técnicas similares a todas as mencionadas anteriormente. As técnicas utilizadas nos outros problemas podem inclusive ser consideradas especializações das técnicas que são empregadas neste problema mais geral possível.

Cada problema de Visão Computacional é caracterizado portanto por quais informações estão disponíveis ou precisam ser determinadas, o que também restringe quais técnicas podem ser utilizadas na solução do problema. Outras condições de uma aplicação também podem influenciar na escolha das técnicas. Uma aplicação pode requerer funcionamento em tempo real, por exemplo, e podem haver tolerâncias maiores ou menores quanto à precisão dos valores que são estimados. Os problemas também podem diferir muito em relação à quantidade de câmeras e de objetos envolvidos, o que pode viabilizar técnicas mais simples, ou pode requerer técnicas mais sofisticadas capazes de lidar com grandes quantidades de dados.

### 1.1.1 Modelos e restrições

As informações conhecidas a princípio em um determinado problema de Visão Computacional não possuem apenas a forma de valores numéricos que são parâmetros de modelos matemáticos. A própria escolha dos modelos a serem utilizados em um problema também faz parte destas informações. Existem diversos modelos de câmera, por exemplo, e na maior parte das aplicações é necessário escolher apenas um para ser utilizado. Os modelos utilizados para os objetos do ambiente também devem ser escolhidos, e assim como no modelo de câmera é usual selecionar apenas um determinado modelo durante o desenvolvimento de uma aplicação. Esta escolha é bastante importante, porque o modelo deve representar adequadamente características relevantes do ambiente, e além disso há técnicas que só podem ser utilizadas com alguns modelos específicos, por mais que haja formas alternativas de se resolver um mesmo problema com diferentes modelos de objetos e técnicas compatíveis correspondentes.

Um modelo de objeto bastante utilizado é o poliedro, também frequentemente utilizado em Computação Gráfica, mas existem também modelos mais complexos e poderoso-

sos, tais como superfícies contínuas de forma livre, ou sólidos primitivos como esferas. Por outro lado, existem também muitas aplicações de Visão Computacional para as quais bastam alguns modelos ainda mais simples do que poliedros. Este é o caso dos retalhos de planos, curvas, segmentos de reta, retas, e até mesmo simples pontos.

A escolha de um modelo geométrico a ser utilizado para representar um ambiente pode envolver também a definição de *restrições*. Um modelo que possui uma determinada restrição pode ser visto como apenas um modelo um pouco mais específico do que o modelo geral utilizado para defini-lo. Uma restrição é, portanto, outra forma de informação preexistente que define o problema sendo estudado, e que também pode ser uma condição necessária para o uso de técnicas específicas.

Uma restrição que é utilizada em algumas aplicações é a de que os objetos do ambiente se localizam apenas sobre um único plano. Este pode ser o caso do rastreamento de pessoas ou de veículos em sistemas de vigilância, por exemplo, em ambientes onde há um solo sobre o qual se assume que os objetos se localizam. Esta restrição também é utilizada em alguns robôs móveis para localizar obstáculos visualmente. Os modelos dos alvos de calibração de câmeras também são muitas vezes pontos localizados sobre uma superfície plana. Esta restrição pode beneficiar a precisão das estimativas obtidas, ou também o desempenho dos algoritmos, mas mais do que isso, ela é uma condição necessária para algumas destas aplicações. A localização de um objeto no espaço com uma única câmera só é possível com o uso de alguma restrição deste tipo. Não fosse esta restrição, seria necessária uma segunda câmera, por exemplo, para tornar possível uma localização de objetos.

Um outro exemplo de restrição bastante utilizada é limitar a orientação da câmera de forma que o seu eixo vertical esteja sempre ortogonal a um determinado plano, reduzindo os graus de liberdade da orientação da câmera para apenas um ao invés dos três de uma orientação genérica de um objeto no espaço tridimensional. Movimentos de câmera também podem ser limitados a uma reta, por exemplo, ou outro tipo de curva.

A pesquisa relatada neste documento envolve o uso de uma outra forma bastante comum de restrição sobre o modelo do ambiente. Trata-se da hipótese de que o ambiente contém um grande número de linhas retas que são paralelas ou ortogonais entre si. Isto implica que estas retas estão orientadas de acordo com apenas três direções mutuamente ortogonais possíveis. Estas direções constituem um *sistema referencial natural*, ou *canônico*, cujos eixos são paralelos a cada um destes conjuntos de retas encontradas no ambiente. Muitos ambientes construídos por humanos possuem esta propriedade, tal como o interior de casas e escritórios onde paredes e móveis contêm tais linhas. Também é possível constatar esta propriedade em ambientes urbanos, onde edifícios e ruas estão frequentemente orientados de forma consistente. Estes ambientes que possuem esta propriedade são denominados *ambientes antrópicos* nesta tese. Outros nomes utilizados na literatura para esta hipótese são *Manhattan world*, do inglês para “mundo Manhattan”

em referência ao formato dos quarteirões do famoso distrito novaiorquino, ou ainda *Lego land*, ou “Terra de Lego”, em referência ao popular brinquedo de origem dinamarquesa.

O uso desta restrição de ambiente antrópico implica que o modelo do ambiente deve envolver a existência de linhas retas, ou talvez planos. Por exemplo, algumas aplicações baseadas em um modelo de ambiente constituído por pontos utilizam a restrição ao determinar que cada ponto deve fazer parte de um grupo de pontos que são coplanares, e a orientação destes planos deve obedecer à restrição.

Assim como a restrição da posição dos objetos em um plano pode permitir utilizar uma única câmera para localizar os objetos, a restrição de ambiente antrópico é uma condição necessária para o tipo de técnica desenvolvida nesta pesquisa, onde determina-se a orientação da câmera em relação ao ambiente a partir de uma única imagem. Esta estimativa é possível porque existe uma relação funcional entre a direção de uma reta no ambiente e a direção de uma possível projeção desta reta em cada ponto da imagem. Esta relação depende apenas da direção da reta em relação à câmera, e não depende de sua posição. Esta independência torna estas técnicas atraentes para várias aplicações, porque permite descobrir a orientação da câmera sem que o ambiente precise ser completamente modelado. Ou seja, a restrição pode ser utilizada para desacoplar os problemas de reconstrução do ambiente e de determinação da orientação da câmera. A orientação estimada nestas técnicas é mais especificamente a rotação tridimensional que constitui a transformação de base entre o sistema referencial da câmera e o referencial natural do ambiente.

### 1.1.2 Extração de características

Como já foi dito, as restrições que podem ser exploradas em um problema determinam quais técnicas podem ser aplicadas, e o mesmo ocorre com a escolha do modelo para os objetos. Além de influenciar o processo de estimativa de parâmetros, a escolha do modelo de objetos também é um fator determinante da forma como as imagens da entrada podem ser analisadas para produzir os primeiros dados utilizados no restante do processo.

Aplicações que visam em primeiro lugar produzir modelos detalhados de objetos, utilizando algum modelo específico selecionado *a priori*, não podem escapar de fazer uso de tal modelo em algum momento. Mas nos casos em que se deseja apenas realizar tarefas tal como a calibração ou a localização de câmera, o modelo utilizado para o ambiente não é relevante, desde que permita a estimativa destes parâmetros desejados. Portanto, pode ser possível nestes casos fazer uso de modelos bastante simples do ambiente, desde que haja no ambiente as condições que tornem viável o uso do modelo. Isto dá origem a diversas técnicas de Visão Computacional onde estes modelos simples são utilizados para o ambiente mesmo que se saiba que modelos bem mais completos poderiam ser empregados, porque o modelo mais simples pode bastar para atender as demandas de várias aplicações, e isto pode beneficiar o desempenho dos algoritmos.

A técnica estudada nesta pesquisa faz parte de uma grande família de técnicas que se caracterizam por possuir dois estágios separados de processamento. Modelos de ambiente relativamente simples são utilizados para simplificar o processo de análise das imagens, e esta análise se limita ao primeiro estágio de processamento. Neste primeiro estágio é realizada uma extração das chamadas *características geométricas* da imagem. Este processo instancia um número de objetos localizados sobre a imagem, tal como áreas, curvas, retas ou pontos, que buscam representar características encontradas na imagem.

No funcionamento destas técnicas baseadas em características assume-se que estas características geométricas possuem relações diretas com objetos da cena. Por exemplo, pontos encontrados sobre a imagem podem ser a projeção de pontos localizados no ambiente, e uma curva sobre a imagem pode ser a projeção de uma curva do ambiente, ou uma área da imagem pode representar a projeção de um determinado volume do ambiente. Uma vez que estas características geométricas tenham sido obtidas, esta assunção de relação é utilizada no estágio seguinte onde ocorre um processo de estimação de parâmetros que produz as estimativas finais desejadas.

No caso da localização de objetos restritos a uma superfície no espaço, um ponto encontrado na imagem definirá uma reta no espaço, cujos parâmetros são dados pelo modelo da câmera, que seria conhecido. A posição original daquele ponto no ambiente será dada pelo ponto onde esta reta corta a superfície do ambiente. No caso de um arranjo de câmeras calibrado, um par de pontos, um em cada imagem, que se sabe corresponderem a um mesmo ponto do ambiente, produzirão duas retas. A posição do ponto no ambiente será dada pela intersecção destas retas, que é o princípio utilizado na técnica de triangulação.

Exemplos similares a estes podem ser facilmente definidos utilizando como características retas ao invés de pontos. No primeiro caso, um segmento de reta seria mapeado em um segmento de reta sobre o plano do ambiente. No segundo caso dois segmentos de reta correspondentes sobre as imagens definiriam um segmento no espaço. É fácil imaginar estas variações ao observar que um segmento de reta pode ser facilmente definido através dos pontos em suas extremidades. Mas também seria possível fazer o mesmo com retas ilimitadas, um par de retas correspondentes nas duas imagens pode ser utilizado para encontrar os parâmetros que definem uma reta ilimitada no espaço tridimensional do problema.

O processo que produz estas características geométricas para uma dada imagem de entrada é chamado extração, como em extração de pontos ou extração de linhas. Existem diversos algoritmos para extrair pontos de imagens, algumas vezes denominados *pontos de interesse*, além de curvas, polígonos, retas e segmentos de reta. A escolha do algoritmo de extração específico a ser utilizado em uma aplicação depende das condições do problema sendo estudado, assim como da própria escolha dos modelos de objeto no ambiente e de características geométricas extraídas.

Extratores de pontos geralmente se baseiam em processos que encontram máximos locais de funções definidas sobre o plano da imagem. A imagem em si é modelada através de uma função, mas os extremos locais desta função não costumam ser uma boa alternativa na prática, exceto em aplicações como na captura de movimentos onde existem marcadores luminosos no ambiente que produzem picos bastante distintos na imagem, por exemplo. As funções utilizadas nos extratores geralmente calculam valores a partir de pequenos trechos da imagem ao redor de uma dada posição, e levantam alguma forma de estatística ou momento da função definida sobre esta pequena região.

O problema da extração de curvas e linhas é um pouco mais complexo. A maioria dos métodos inicia com o uso de um *detector de borda* que produz pontos que se acredita fazerem parte de alguma curva sobre a imagem. A seguir há um procedimento que associa estes pontos em grupos de pontos que fazem parte de uma mesma curva, e os parâmetros de cada curva são então encontrados com técnicas de regressão. A extração de linhas retas, especificamente, é utilizada em muitas aplicações, e para isto podem ser utilizadas técnicas clássicas de regressão para ajuste de linhas a conjuntos de pontos. A estimação dos parâmetros de uma linha pode ser feita simultaneamente a esta determinação de quais pontos detectados fazem parte da linha, o que constitui um processo mais complexo e robusto, ou seja, resistente a falhas tal como a introdução de um ponto incorreto no conjunto analisado. Há casos em que se sabe quantas linhas devem haver no ambiente, e até quais são aproximadamente os seus parâmetros. Em outros casos não se conhece nem mesmo o número de linhas, e isto constitui um problema bem mais desafiador.

O uso de pontos em uma técnica qualquer pode ser vantajoso por alguns motivos. Em primeiro lugar, estas entidades são muito simples de representar, necessitando apenas um vetor com os valores de suas coordenadas. O cálculo da projeção sobre a imagem de um ponto do espaço também é relativamente simples, bastando apenas substituir as coordenadas nas expressões que constituem o modelo de câmera. Retas podem ser mais desafiadoras. Há várias maneiras de se representar uma mesma reta, utilizando parâmetros com significados diferentes, e cada possibilidade pode ter suas vantagens e desvantagens. Uma reta também pode se tornar uma curva não-reta em alguns tipos de mapeamento, enquanto pontos sempre são mapeados em pontos. Além disso, há o chamado *problema da abertura*, que se refere à possibilidade de não se poder diferenciar dois pontos sobre uma reta apenas através do aspecto da imagem ao redor destes pontos.

As dificuldades no uso de linhas ao invés de pontos devem portanto ser reconhecidas. Porém as vantagens do seu uso não deixam de existir. Algumas vantagens são:

- Linhas permitem modelar o ambiente de uma forma mais rica em significados, o que pode ser necessário para várias aplicações. Enquanto modelos baseados em pontos são esparsos e podem deixar vazias regiões do espaço que na realidade contêm algum objeto, linhas possuem o potencial para modelar os contornos de objetos com perfeição, e representar de forma completa regiões que podem ter significado

específico dentro de uma aplicação. Por exemplo, um conjunto de linhas pode delimitar com exatidão uma região do espaço dentro de onde um robô móvel deve planejar suas trajetórias, enquanto um conjunto de pontos pode apenas servir como base para definir tal região.

- Linhas permitem explorar restrições estruturais que frequentemente existem nos ambientes. A mais importante restrição possível deste tipo é justamente a condição de ambiente antrópico discutida anteriormente. Muitas aplicações que utilizam uma única câmera, chamadas *monoculares*, dependem de algum tipo de restrição relacionada ao uso de linhas. E além de permitir que estas técnicas existam, estas restrições também podem trazer benefícios computacionais, como também foi dito anteriormente. Estas restrições estruturais do ambiente também podem possuir um valor semântico de interesse. Por exemplo, pode ser útil para uma aplicação que um mapa seja construído com seus eixos já alinhados às direções das retas existentes no ambiente. Isto pode facilitar a interação entre sistemas, bem como com os usuários.
- Do ponto de vista de extração de características, as linhas possuem um grande potencial para aumentar a robustez das detecções e qualidade das estimativas de parâmetros. O reconhecimento de um determinado ponto em uma imagem conforme o aspecto de sua projeção varia entre diferentes pontos de vista é um problema que pode se mostrar muito desafiador em condições mais gerais. Linhas constituem um modelo mais flexível, capaz de produzir técnicas mais robustas. Isto é especialmente verdadeiro em aplicações baseadas em imagens fortemente distorcidas, o que inclui desde imagens produzidas por algumas câmeras convencionais e a maioria das câmeras de segurança, até lentes com distorções intensas tal como as olho-de-peixe, ou ainda arranjos catadióptricos, que utilizam espelhos curvos. Qualquer forma de produzir imagens que capturam grandes regiões do espaço, como imagens onidirecionais, fatalmente produz fortes distorções em alguma região da imagem. Imagens com a projeção chamada *equiretangular* estão se tornando uma forma popular de armazenar imagens em aplicações de visão onidirecional, e são utilizadas em projetos como o *Google StreetView* ou o *360cities*, e esta projeção ainda é suportada pela maior parte dos programas que lidam com imagens panorâmicas. Outras projeções cilíndricas também são utilizadas em algumas aplicações.

Tanto no caso da projeção equiretangular quanto nas distorções radiais a aplicação de técnicas baseadas em pontos geralmente conta apenas com a robustez dos métodos para funcionar apesar das distorções, que são raramente levadas em consideração durante a extração e associação dos pontos. Linhas representam uma forma de característica visual que facilita a realização de análises que levam em consideração as distorções das imagens, como foi realizado no método proposto nesta tese.

Com relação ao último ponto, é importante observar que, enquanto o modelo do ambiente pode conter apenas linhas retas, as projeções destas retas na imagem podem ser curvas dependendo do modelo de câmera. Isto é um fator que dificulta o trabalho com imagens deste tipo, e como será mostrado no Capítulo 3, técnicas de extração de retas têm sido utilizadas analisar estas curvas, mesmo que o modelo geométrico adequado não seja empregado e o desempenho destes algoritmos de extração seja consequentemente afetado de forma negativa pelas distorções. As dificuldades surgidas nestas situações são eventualmente tratadas por processos secundários que analisam os segmentos de reta extraídos e finalmente levam em consideração as curvaturas existentes. Um dos objetivos do método proposto aqui é encontrar uma forma mais natural de trabalhar com imagens distorcidas que contêm projeções de linhas retas.

### 1.1.3 *Edgels*

O método desenvolvido nesta pesquisa é baseado nos chamados *edgels*, um neologismo da língua inglesa criado a partir do termo “elemento de borda”. Edgels constituem uma forma relativamente pouco utilizada de característica geométrica. Um motivo para isto é que em um contexto qualquer onde existem edgels, quase sempre existirão curvas ou linhas retas, e a extração destas outras características costuma receber mais atenção. Entretanto, edgels surgem naturalmente em diversos processos. Eles são facilmente produzidos a partir de pontos detectados para uma posterior extração de curvas, por exemplo. Logo edgels não são uma entidade incomum, eles apenas são raramente encarados como características geométricas a serem utilizadas da mesma forma que ocorre com segmentos de retas ou pontos extraídos das imagens.

Um edgel é um ponto sobre a imagem associado a uma informação de direção bidimensional. Ele pode ser visto portanto como um vetor acoplado a um ponto, como existe em um campo vetorial. Para que este vetor possa representar uma direção ele deve possuir norma unitária, e o seu sentido deve ser irrelevante. Um vetor com a mesma direção e sentido oposto constituiria um edgel equivalente.

Edgels possuem portanto três graus de liberdade, sendo um deles uma direção sobre o plano, e os restantes uma posição em duas dimensões. Ele é fundamentalmente diferente de uma reta, que possui apenas dois graus de liberdade, um deles direcional. Edgels também se diferenciam de segmentos de reta, que possuem quatro graus de liberdade. Um segmento pode ser representado através de dois pontos, e portanto quatro graus de liberdade relacionados a posicionamento, mas também pode ser representado, por exemplo, através de uma posição bidimensional, uma direção e um comprimento.

É possível interpretar um edgel como sendo um segmento de reta que possui comprimento infinitesimal. Uma reta também pode ser produzida a partir de um segmento de reta estendendo-se o seu comprimento indefinidamente a partir de ambas extremidades. De forma semelhante, uma reta também pode ser definida a partir de um edgel estendendo-se

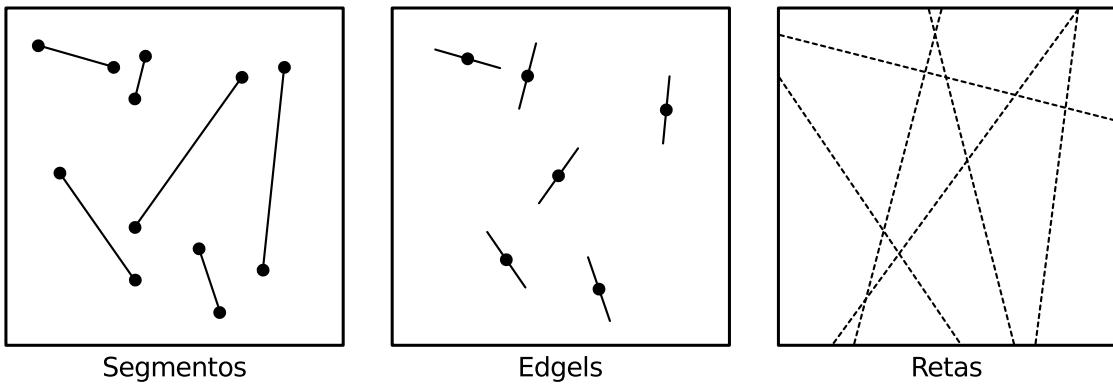


Figura 1.2: Exemplos de segmentos de reta, edgels e linhas retas que compartilham alguns parâmetros.

o seu vetor associado indefinidamente em ambos sentidos. Por outro lado, um edgel pode ser definido a partir de uma reta selecionando-se um ponto contido nela, e utilizando a mesma direção. A Figura 1.2 apresenta três imagens, uma contendo segmentos de reta, outra edgels e outra linhas retas. Estas entidades de cada uma destas imagens podem ser criadas a partir das outras imagens, por exemplo, os edgels neste exemplo poderiam ter sido criados pela seleção de um ponto sobre cada uma das linhas retas na outra imagem. Os segmentos são representados por um par de pontos ligados por uma reta. Os edgels são representados por pontos cortados por um pequeno segmento de tamanho constante orientado na direção do edgel. As retas são simplesmente retas pontilhadas, que se estendem por toda a área da imagem e portanto tocam necessariamente duas de suas margens.

A forma como edgels são utilizados em uma análise é assumindo que eles são a projeção na imagem de pontos que pertencem a uma reta do ambiente. Como foi dito anteriormente, a projeção de um segmento de reta do ambiente em uma imagem pode ser uma curva não-reta. Mas cada ponto da reta no espaço ainda possui um único ponto correspondente nesta curva projetada sobre a imagem. E em cada ponto desta curva é possível calcular uma direção tangencial a ela. Os edgels extraídos de uma imagem são pontos que pertencem a estas curvas, e suas direções são estas direções das retas que tangenciam a curva em cada ponto.

No caso em que o modelo de câmera não cria distorções, todos os edgels situados sobre uma mesma reta possuirão a mesma direção, que é a direção da reta projetada. Esta direção na imagem possui uma relação funcional com a direção da reta original no sistema referencial da câmera, e consequentemente com a orientação da câmera através da restrição de ambiente antrópico. No caso em que há distorções na imagem as direções dos edgels continuam se relacionando com a direção da reta no espaço, mas estas direções dos edgels passam a sofrer também influência do modelo de câmera.

Como foi mencionado, edgels surgem naturalmente durante um processo de extração de curvas, podendo ser definidos a partir dos pontos criados por um detector de borda, e

que seriam normalmente associados em seguida para formar as curvas ou retas. Mas existem técnicas de visão que utilizam edgels (BLAKE; ISARD, 1998; EADE; DRUMMOND, 2006; SMITH, 2007; COUGHLAN; YUILLE, 1999), ou entidades que podem ser interpretadas como sendo edgels, que não realizam esta associação destas entidades para a extração de entidades de maior nível de abstração antes de passar para um passo seguinte de análise.

Um exemplo é o rastreamento de contornos. Se em uma aplicação de visão é conhecida a localização aproximada de um contorno de um objeto, e então o objeto muda gradualmente de posição, a nova posição do contorno pode ser encontrada a partir da anterior através de uma atualização dos parâmetros deste contorno. A forma como esta atualização é geralmente feita começa com a realização de uma busca por pontos situados sobre o novo contorno. Esta busca é feita sobre linhas definidas a partir de pontos situados sobre o contorno antigo. Estes pontos são amostrados ao longo da curva de forma regular e esparsa, ou seja, não se utilizam absolutamente todos os pontos contidos no contorno conhecido, mas apenas um conjunto pequeno e suficiente de pontos (BLAKE; ISARD, 1998). Estas linhas sobre as quais se realizam as buscas são ortogonais ao contorno no ponto em que se cruzam. No caso de uma reta sobre a imagem, por exemplo, seriam calculados pontos sobre a reta antiga, e as buscas por pontos sobre a nova reta seriam feitas a partir de cada um destes pontos amostrados, na direção ortogonal à reta antiga. A Figura 1.3 mostra um exemplo de rastreamento de contorno, onde é possível ver as linhas ortogonais à curva conhecida sobre as quais foram realizadas as buscas. Pontos sobre estas linhas mostram onde ocorre uma detecção de borda. Estes pontos encontrados podem ser interpretados como edgels que foram extraídos da imagem.

Um exemplo de uma técnica onde edgels são utilizados de uma forma mais evidente como características geométricas extraídas é o sistema criado por Eade e Drummond (2006) para o rastreamento de câmera e modelagem de ambiente simultâneos. No método proposto por estes pesquisadores o próprio ambiente é modelado como sendo um conjunto de edgels situados no espaço, porém com posições e direções tridimensionais, e as projeções destes edgels espaciais produzem edgels bidimensionais sobre a imagem. Edgels que já fazem parte do modelo do ambiente são rastreados de forma semelhante ao que é feito no rastreamento de contorno, porém apenas uma busca é feita para cada edgel. Além deste rastreamento existe um processo que busca descobrir novos edgels sobre a imagem, e o passo inicial para isto é uma análise de imagem que realiza explicitamente uma extração de edgels. A extração utilizada naquele sistema começa por uma detecção de bordas, seguida por um ladrilhamento da imagem. As bordas detectadas são analisadas dentro de cada um destes ladrilhos, podendo resultar na instanciação de um edgel ou nenhum.

Um outro exemplo interessante de uso de edgels vem de uma outra área de pesquisa, o reconhecimento de caracteres. Trata-se da técnica desenvolvida por Smith (2007). Na téc-

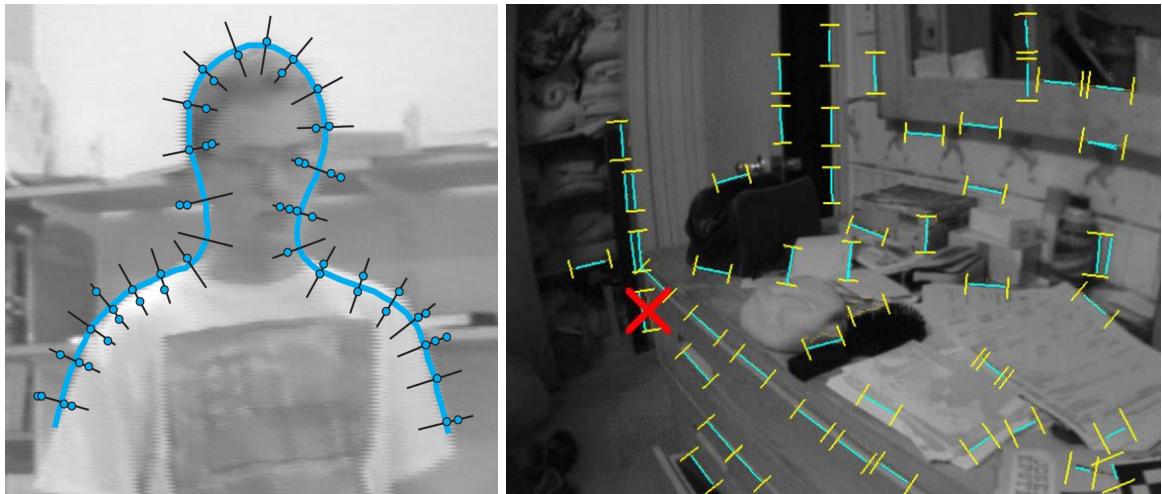


Figura 1.3: À esquerda, um exemplo de rastreamento de contorno, reproduzido do livro de Blake e Isard (1998). O contorno é rastreado a partir de pontos individuais contidos dentro dele, que são edgels. Os pontos sobre as linhas escuras ortogonais ao contorno são edgel extraídos da imagem, porém suas direções não estão representadas no gráfico. À direita, uma imagem reproduzida do artigo de Eade e Drummond (2006) onde edgels são rastreados sobre as imagens para realizar a localização de câmera e modelagem do ambiente. Neste gráfico cada edgel é representado através de um pequeno segmento de reta.

nica original de reconhecimento em que este pesquisador baseou seu trabalho realizava-se uma extração de contornos mais complexos, que eram então comparados com protótipos existentes em uma base de dados. A experiência com esta técnica original conduziu a um processo em que pequenos mas mais numerosos trechos de reta, que podem ser interpretados como edgels, são comparados aos protótipos dos caracteres a serem reconhecidos. O processo desenvolvido é ilustrado pela Figura 1.4, que mostra um conjunto de edgels extraídos de uma imagem sendo comparados a um protótipo do caractere 'h'. O conjunto de edgels provê um conjunto de observações que são utilizadas diretamente como evidências para realizar a classificação. Estas observações se mostraram suficientes para este problema, dispensando a realização de análises mais complexas da entrada. A entidade utilizada na análise e durante o reconhecimento não é portanto a mesma utilizada para definir os protótipos sendo reconhecidos, uma diferença que pode ser considerada incomum em algumas áreas de estudo.

O agrupamento das observações em estruturas como linhas ou curvas acaba criando dificuldades na hora de realizar as comparações com os protótipos, porque estes agrupamentos podem ser equivocados, criando incoerências de alto nível durante a interpretação dos protótipos. Por exemplo, contornos que deveriam ser separados podem se associar em um contorno só, ou um determinado contorno dos protótipos pode estar representado como mais de um contorno na análise. Dada a necessidade de lidar com estas imperfeições quando contornos são extraídos, o trabalho com edgels totalmente dissociados acaba sendo mais simples, eles constituem uma única base de evidências primordiais a que se

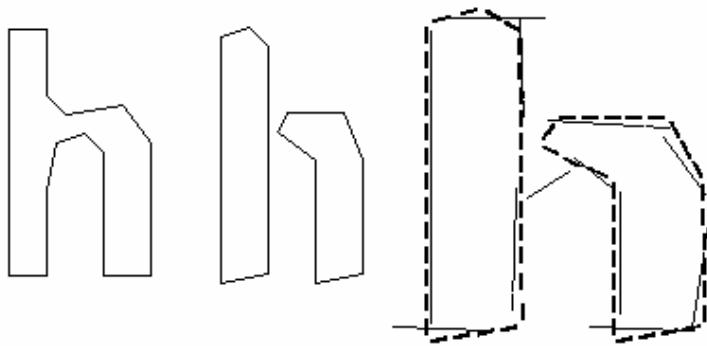


Figura 1.4: Exemplo de edgels sendo utilizados no reconhecimento de um protótipo do caractere 'h', reproduzido de Smith (2007).

reduzem os dados a serem analisados, e a classificação é feita diretamente a partir deles. Esta remoção do passo intermediário de associação das observações que realizaria uma extração de curvas não só simplifica o processo de análise, como acaba resultando em um desempenho superior na classificação.

## 1.2 Proposta da tese

Esta tese consiste na proposta e desenvolvimento de um método para a estimativa de orientação de câmera baseado na restrição de ambiente antrópico. Apesar do ambiente ser modelado a partir de segmentos de linha reta que atendem a restrição, o método aqui proposto tem por base uma técnica já investigada no passado que é baseada em edgels, e que dispensa a extração de retas das imagens. O desenvolvimento desta técnica de base se resume basicamente às pesquisas conduzidas por Coughlan e Yuille (1999, 2003), Deutscher, Isard e MacCormick (2002), Schindler e Dellaert (2004) e Denis, Elder e Estrada (2008). O método proposto foi denominado *Corisco*, e foi implementado na prática em um sistema que foi utilizado para a realização de experimentos e validação deste método. O nome *Corisco* é uma sigla para a expressão inglesa *Camera orientation from infinitesimal segments by constrained optimization*, ou *Orientação de câmera a partir de segmentos infinitesimais através de otimização com restrições* em português.

Os edgels que são extraídos das imagens durante a sua análise inicial representam pontos que se assume serem projeções de pontos situados sobre as retas do ambiente. A direção destes edgels possui uma relação com a direção da reta original no ambiente, o que permite realizar a estimativa de orientação através ainda da restrição de ambiente antrópico. Existem técnicas alternativas baseadas na extração de linhas que utilizam as mesmas relações e restrições para realizar esta estimativa de orientação (CAPRILE; TORRE, 1990; CIPOLLA; DRUMMOND; ROBERTSON, 1999; ROTHER, 2002). Apesar das semelhanças, a mudança do modelo utilizado para a análise implica em grandes diferenças na forma como a estimativa deve ser feita, o que será detalhado nos capítulos a seguir.

Além da possibilidade de realizar a estimativa de orientação de câmera, o interesse em

estudar técnicas que utilizam retas para modelar o ambiente também decorreu da necessidade em robótica móvel de realizar a detecção de obstáculos para a condução segura de robôs por ambientes antrópicos, além do reconhecimento de objetos para a realização de tarefas. O uso deste método baseado em edgels pode ser uma forma de facilitar o eventual desenvolvimento de outros métodos baseados em linhas para uso nestas aplicações.

### 1.2.1 Motivações

Esta pesquisa iniciou-se com o intuito de desenvolver melhores formas de se utilizar imagens obtidas através de equipamentos como lentes olho-de-peixe e arranjos catadióptricos para aplicações de robótica móvel. As dificuldades no uso de extractores de retas são logo percebidas neste cenário, causadas pelas fortes distorções existentes nestas imagens. Uma forma de lidar com isto é realizar uma retificação das imagens (AMARAL; COSTA, 2010; SACHT et al., 2010), ou seja, produzir novas imagens que não possuam as distorções, utilizando a chamada *projeção perspectiva*, ou *projeção pontual*. Mas a retificação pode ser inconveniente por alguns motivos:

- Pode ser custoso sintetizar estas imagens retificadas, tornando a técnica inadequada para cenários que requerem alto desempenho.
- É preciso conhecer o modelo de câmera para realizar a retificação.
- A transformação das imagens cria um passo intermediário entre as análises e os dados originais, e esta prática costuma ter um impacto negativo nos resultados das estimativas.
- Para câmeras com campos de visão muito grandes o resultado de uma retificação pode ser uma imagem que, apesar de possuir linhas retas, é inadequada para análise porque os objetos nas extremidades da imagem adquirem grandes dimensões.

O aspecto distorcido dos objetos nas extremidades de uma imagem retificada torna esta representação inadequada para câmeras com grandes campos de visão, mas pode ser ainda pior do que isto. No caso de imagens onidirecionais, como a projeção equiretangular, a retificação iria produzir imagens de dimensões infinitas, e que ainda assim não conteriam toda a área da imagem original. A Figura 1.5 mostra um exemplo de retificação de uma imagem com campo visual grande. Apesar da projeção perspectiva manter o formato das linhas retas, os formatos dos objetos ainda sofrem distorções.

A projeção estereográfica mostra um exemplo de modelo de câmera onde as linhas retas do ambiente são mapeadas em curvas, mas em compensação é possível ter um campo visual maior sem que o formato dos objetos seja tão distorcido. Métodos baseados em extração de linhas retas seriam a princípio aplicáveis apenas para o caso da projeção perspectiva, como na imagem à direita na Figura 1.5. Métodos como o *Corisco* buscam



Figura 1.5: Uma imagem criada com a projeção estereográfica, à esquerda, e à direta a imagem rectificada, ou seja, com projeção perspectiva. Apesar das linhas retas na imagem da direita, existem algumas características inconvenientes. Imagem reproduzida de (FLECK, 1995).

permitir trabalhar com outros tipos de modelos de câmera, tal como a imagem à esquerda da Figura 1.5.

A técnica introduzida por Coughlan e Yuille (1999) e desenvolvida posteriormente (DEUTSCHER; ISARD; MACCORMICK, 2002; COUGHLAN; YUILLE, 2003; SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008) para a estimativa de orientação de câmera em ambientes antrópicos sem extração de retas pareceu ser uma boa alternativa para uma futura aplicação neste cenário de robótica móvel com visão onidirecional, e esta pesquisa buscou concretizar este potencial com o desenvolvimento do Corisco. Este método é portanto mais atraente para a aplicação em um robô móvel, por evitar as dificuldades associadas à extração de retas ou curvas em imagens distorcidas.

### 1.2.2 Objetivos

A pesquisa realizada pode ser vista como uma tentativa de responder à pergunta: *qual é a melhor forma de se estimar a orientação de câmera em tempo real em um ambiente antrópico a partir de uma única imagem distorcida?* Este problema surge em diferentes aplicações de Visão Computacional. Uma aplicação de destaque para este método seria na análise de grandes conjuntos de imagens, onde a estimativa de orientação das imagens individuais pode funcionar como um estágio inicial que simplifica o processo completo onde as imagens são posteriormente analisadas em conjunto. Neste cenário a possibilidade de analisar as imagens individualmente permite paralelizar este estágio inicial do processo facilmente, e um bom desempenho é desejável quando os conjuntos atingem tamanhos consideráveis, da ordem de milhares ou dezenas de milhares de imagens.

Outra aplicação interessante seria em robôs móveis. Neste caso um método de estimativa de orientação pode auxiliar em primeiro lugar no guiamento do robô, mas a orientação estimada também pode ser útil para a estimativa da localização completa do robô,

além de outras tarefas como o mapeamento do ambiente e reconhecimento de objetos. O alto desempenho é interessante neste cenário por tratar-se de uma aplicação interativa, o que faz com que a velocidade do processo de estimativa limite a velocidade com que o sistema pode operar. Neste cenário de aplicação também é interessante que o método seja capaz de funcionar tanto sem nenhuma estimativa da solução, permitindo ao robô funcionar sem nenhum conhecimento do ambiente, quanto realizando um rastreamento, onde estimativas são atualizadas recursivamente ao longo do tempo. Também é comum em robôs móveis a utilização de sistemas de captura de imagens onidirecionais, baseados em espelhos, conjuntos de câmeras ou em lentes especiais que resultam em imagens com projeções que contém fortes distorções. Isto torna desejável que qualquer método de Visão Computacional empregado seja capaz de lidar com isto.

O problema da estimativa de orientação em ambientes antrópicos impõe naturalmente a necessidade de considerar a existência de retas no ambiente, e impede análises baseadas apenas em características pontuais. A restrição de tempo real implica que o método utilizado deve buscar realizar cálculos com baixo custo computacional, e deve permitir realizar um comprometimento entre precisão e velocidade. Estas duas restrições podem ser atendidas através do uso de edgels, o que ainda permite analisar imagens distorcidas com mais naturalidade do que, por exemplo, realizando extração de retas ou curvas. O uso de edgels se encontra no cerne do *Corisco*, determinando a forma como as imagens são analisadas e permitindo alcançar todos estes objetivos. O comprometimento entre precisão e velocidade é obtido em primeiro lugar através da sub-amostragem dos dados, realizada no *Corisco* por uma máscara em forma de grade.

A estimativa da orientação requer um método de otimização global. O processo utilizado no *Corisco* para esta otimização possui dois passos, e contém mais características benéficas para aplicações interativas. O primeiro passo oferece outro mecanismo de comprometimento entre precisão e qualidade, e também permite iniciar a otimização sem qualquer estimativa inicial da solução. Já o segundo passo do processo é baseado em uma técnica de otimização contínua, mas ao contrário de métodos semelhantes existentes, o *Corisco* utiliza apenas operações matemáticas de custo computacional relativamente baixo, e também realiza o cálculo de derivadas da função objetivo através de fórmulas fechadas. No caso em que se deseja realizar rastreamento também é possível dispensar o primeiro passo da otimização, empregando apenas o método do segundo passo para realizar atualizações de estimativas existentes da orientação ao longo do tempo.

O *Corisco* visa portanto atender os requisitos de um método para estimativa de orientação em ambientes antrópicos que possua características atraentes a aplicações interativas tal como robôs móveis, com um custo computacional relativamente baixo e a possibilidade de realizar um comprometimento entre precisão e velocidade. Além do bom desempenho, o *Corisco* é capaz de funcionar sem nenhuma estimativa inicial da solução, mas pode ser adaptado facilmente para a realização de rastreamento. O *Corisco* tam-

bém é capaz de lidar com imagens que utilizam qualquer tipo de projeção através de um processo unificado, requerendo apenas a modificação de um módulo separado para implementar cada projeção específica. Estes objetivos são alcançados pelo *Corisco* através do uso de edgels como entidade fundamental da análise das imagens, e através da escolha dos algoritmos de otimização específicos selecionados durante o desenvolvimento.

### 1.2.3 Justificativas

Apesar de se basear apenas em edgels, o *Corisco* pode ser aplicado até mesmo em situações em que se deseja extraer curvas ou retas das imagens, já que este processo pode ser bastante beneficiado por uma estimativa prévia da orientação (FLINT et al., 2010; LEBEGUE; AGGARWAL, 1993). A estimação separada da orientação também pode ser empregada para simplificar processos maiores de localização e mapeamento (SINHA; STEEDLY; SZELISKI, 2010; BOSSE et al., 2002), além de permitir realizar tarefas mais simples tal como guiar um robô por um corredor mantendo uma direção constante, sem realizar estimações da localização da câmera ou da estrutura do ambiente. O *Corisco* é portanto um método de baixo nível que pode ser aproveitado em diferentes tarefas de visão de nível mais alto.

Além da maior naturalidade com que esta técnica lida com distorções comparada à extração de retas ou de curvas, há também o atrativo da adequação para funcionamento em tempo real, principalmente devido à possibilidade de controlar com facilidade o comprometimento entre a velocidade e precisão da estimativa através do controle do número de edgels que são analisados. No método proposto os edgels são utilizados diretamente para estimar a orientação, e se pode esperar um comprometimento mais bem-comportado. Já no caso da extração não-informada de linhas uma sub-amostragem afetará primeiro a qualidade das linhas obtidas, e então o resultado final. Se neste caso o modelo de câmera e a restrição de ambiente antrópico não forem exploradas para a extração de linhas, é de se esperar que a qualidade dos resultados piore bem mais rapidamente conforme menos dados de entrada sejam utilizados.

O próprio uso de extração de retas ou curvas para aplicações em tempo real já seria desafiador a princípio pela complexidade destes algoritmos, exceto em um cenário onde ocorre um rastreamento de contornos e estas extrações não seriam frequentes. Mesmo neste cenário, a análise através de edgels ainda pode oferecer um melhor desempenho nos momentos iniciais do processamento, onde não há linhas conhecidas no ambiente a serem rastreadas.

A possibilidade de utilizar o *Corisco* para calibração de câmera, com poucas modificações, também se tornou um atrativo, o que faz com que qualquer aplicação desenvolvida com o método possua a seu dispor um procedimento para resolver este problema. Isto beneficia principalmente possíveis aplicações baseadas em *smartphones* ou outras plataformas portáteis disponíveis para consumo, já que seus usuários não dispõem das

mesmas facilidades que um laboratório de robótica móvel para a realização de calibração de câmera. A calibração fornecida por esta técnica não oferece necessariamente a melhor qualidade possível, mas pode auxiliar na inicialização de métodos baseados em extração de linhas. A calibração por edgels pode ser também interessante em algum cenário em que as características do modelo de câmera se modifiquem muito ao longo do tempo, mas isto é algo bastante incomum.

Podemos resumir as desvantagens do uso de extração de linhas para resolver o problema de estimar a orientação de câmera em um ambiente antrópico através dos seguintes motivos:

- A extração de linhas irá inherentemente produzir informações extras, desnecessárias, que são os parâmetros das linhas que serão extraídas, e mais do que isto, as associações das observações que formam cada linha, além da classificação das linhas de acordo com a sua direção no ambiente.
- O comprometimento entre a qualidade e velocidade da estimação será necessariamente pior com extração de linhas, por causa da forma indireta como a subamostragem irá afetar a estimação de orientação. A condição de ambiente antrópico também não é necessariamente explorada em processos de extração de linhas, o que resulta em uma menor robustez.
- A extração de linhas é diretamente afetada de forma negativa quando as imagens se tornam muito distorcidas, pelo simples fato de que as linhas da imagem se tornam curvas. Os métodos de extração de linhas precisam lidar com isto empregando algoritmos ainda mais complexos na análise dos dados. Métodos de extração de curvas também podem ser utilizados, e estes são inherentemente mais complexos.

O uso de edgels se contrapõe a esta alternativa da seguinte forma:

- Os edgels não são associados em retas de mais alto nível, seus parâmetros são apenas utilizados de forma direta para a estimação da orientação. A classificação dos edgels de acordo com sua direção no ambiente é algo que não se pode evitar determinar, mas isto é realizado de uma forma implícita no algoritmo proposto.
- O comprometimento ocorre de forma natural, já que a lista com todos edgels é diretamente utilizada no procedimento de estimação de parâmetros. A restrição de ambiente antrópico é utilizada em todo o momento durante a estimação de parâmetros.
- O método empregado é o mesmo para imagens distorcidas ou não, apenas as expressões do modelo de câmera são alteradas.

A estimação de orientação através de edgels constitui portanto uma alternativa superior à estimativa através de extração de linhas em muitos cenários. Mas é importante considerar que não se tratam simplesmente de abordagens em competição direta entre si. A estimativa por extração de linhas não é um processo radicalmente diferente do proposto, mas sim um processo mais complexo e de dificuldade por vezes subestimada, e que pode ser auxiliado por um método tal como o *Corisco*, quando não substituído.

A extração de linhas envolve uma classificação das bordas detectadas em um número indeterminado de classes independentes, apenas mutuamente excludentes, enquanto que a classificação de acordo com as três direções possíveis no ambiente, sejam de edgels ou das curvas extraídas, é um problema de classificação bem mais simples pelo fato de que as classes são conhecidas e de que existem restrições geométricas a serem exploradas. Assim, faz sentido tentar resolver o problema mais simples de forma direta — estimar apenas a orientação e classificar os edgels — sem passar pela extração de linhas. A extração de linhas seria justificável apenas por imposição da aplicação, talvez pelo desejo de realizar estimativas ainda mais precisas utilizando-se um cálculo de erro de uma grandeza diferente da utilizada na estimativa por edgels. Mas ainda assim este processo de extração de linhas poderia ser bastante beneficiado pelo aproveitamento das informações obtidas por uma estimativa inicial realizada apenas com os edgels.

#### 1.2.4 Principais contribuições

O desenvolvimento do *Corisco* tomou como ponto de partida a técnica estimativa de orientação baseada em edgels proposta inicialmente por Coughlan e Yuille (1999) e investigada posteriormente por outros pesquisadores (COUGHLAN; YUILLE, 2003; DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008). A principal modificação introduzida com o *Corisco* foi a efetiva adaptação da técnica, pela primeira vez, para a análise de imagens distorcidas. Esta é uma possibilidade que foi apenas reconhecida no passado. Foram estudadas especificamente distorções radiais de câmeras comuns, utilizando o modelo de Harris (TORDOFF; MURRAY, 2004), e de lentes olho-de-peixe com projeção polar equidistante, além de imagens com projeção equiretangular.

Comparado a estes métodos existentes, o passo inicial de análise de imagem do *Corisco* foi modificado para criar um processo de extração de edgels similar ao criado por Eade e Drummond (2006) em que a subamostragem pode ser controlada de uma forma bastante conveniente. Os métodos anteriores utilizavam ou a imagem inteira (COUGHLAN; YUILLE, 2003), ou pontos sorteados aleatoriamente (DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004), ou todos os pontos encontrados por um detector de bordas (DENIS; ELDER; ESTRADA, 2008). Nesta pesquisa foi investigada uma forma de subamostragem baseada em uma máscara em forma de grade que seleciona um conjunto restrito das bordas detectadas na imagem para extrair edgels. Este

procedimento permite sub-amostrar os dados desde o princípio do processamento das imagens, e garante uma amostragem regular de curvas longas na imagem.

A estimação de parâmetros é feita a partir destes edgels extraídos através de um processo de otimização, realizado em dois passos. Uma mesma função objetivo é minimizada nestes dois passos, havendo apenas uma troca do algoritmo de otimização. A orientação da câmera é parametrizada através de quaternions, que são vetores de quatro dimensões que devem possuir norma unitária para representar uma rotação tridimensional. Ou seja, o espaço das soluções é a 3-esfera, o correlato em quatro dimensões de uma esfera. Esta função objetivo é definida a partir da técnica de M-estimação, desenvolvida dentro da área de estudos da Estatística Robusta, e substitui outras técnicas de estimação empregadas nos outros métodos.

Um detalhe importante sobre a função objetivo utilizada no *Corisco* é que a forma como ela é calculada é a mesma para qualquer modelo de câmera utilizado. Os diferentes modelos de câmera são implementados através de uma rotina separada que apenas calcula um conjunto de coeficientes que são utilizados para o cálculo da função objetivo, e que permanecem constantes durante a otimização. Estes coeficientes fazem parte das matrizes Jacobianas do mapeamento de projeção do espaço para as coordenadas de imagem, calculadas para cada edgel. Qualquer projeção pode ser utilizada com o *Corisco*, até mesmo projeções com fortes distorções como a equiretangular foram analisadas com sucesso durante os experimentos.

Outro detalhe importante sobre a função objetivo é que não são realizadas chamadas a procedimentos para o cálculo de funções muito custosas como a exponencial, logaritmo e funções trigonométricas. As funções mais complexas utilizadas são a raiz quadrada e a divisão, e ainda foram empregadas com sucesso no sistema desenvolvido rotinas de alto desempenho que calculam aproximações destas funções.

O primeiro passo do processo de otimização visa melhorar a convergência global do processo completo, e faz com que não seja necessário possuir alguma estimativa inicial da solução para iniciar o processo. Ele é baseado em uma busca aleatória guiada pelos próprios dados de entrada, e produz uma estimativa da solução que pode então ser utilizada para iniciar o segundo passo. O segundo passo é baseado em um algoritmo de otimização contínua com restrições. A função objetivo é definida sobre todo o espaço quadridimensional dos quaternions, e utiliza-se um algoritmo de otimização com restrições para impor a condição de norma unitária para a solução. Esta possibilidade de utilizar quaternions não-normalizados na otimização simplifica as expressões envolvidas, e permite utilizar fórmulas fechadas para as derivadas da função objetivo em relação aos parâmetros do quaternion.

Algumas das principais contribuições que foram alcançadas com o desenvolvimento do *Corisco* que podem ser destacadas são:

- Uso da extração de edgels através de uma máscara em forma de grade.

- Uso de modelos de câmera com distorções, e não apenas projeção perspectiva, com um procedimento único empregado para qualquer modelo de câmera.
- Cálculo de resíduos em cada edgel utilizando uma multiplicação de vetores normalizados ao invés de operações trigonométricas.
- Extração de bordas realizada em imagens coloridas, aproveitando a informação de todos os canais, e não simplesmente convertendo tudo para um único canal em um primeiro passo.
- O uso de M-estimação para a definição da função que é minimizada para a estimação da orientação, substituindo o uso da estimação pelas técnicas MAP e EM.
- A otimização contínua para a estimação de parâmetros foi realizada através de otimização com restrições, o que permite simplificar a expressão da função objetivo. O algoritmo FilterSQP (FLETCHER; LEYFFER, 2002) foi empregado para a otimização com restrições, utilizando fórmulas fechadas para o cálculo de todas derivadas, ou seja, dispensando o uso de técnicas de diferenciação numérica ou outras alternativas.
- Uso de uma busca estocástica, no estilo do algoritmo RANSAC (CHOI; KIM; YU, 2009), para encontrar uma estimativa inicial da solução. Isto melhora a convergência global da otimização, e permite analisar qualquer imagem sem depender de restrições adicionais ou de estimativas iniciais da solução fornecidas pelo usuário ou por algum método anterior.

Os testes realizados com o método desenvolvido apresentaram resultados bastante satisfatórios e motivadores. O tempo de execução e a precisão alcançada foram melhores do que os apresentados por um método similar em um teste com uma base de dados padrão. O método também foi demonstrado com sucesso na análise de imagens distorcidas, inclusive com a projeção equiretangular.

### 1.3 Estrutura do documento

A estrutura do restante deste documento é a seguinte: O Capítulo 2 possui uma revisão de conceitos básicos utilizados em Visão Computacional, incluindo a teoria de formação de imagens. O Capítulo 3 traz uma revisão de diversas técnicas baseadas em características geométricas, e uma análise comparativa de métodos similares ao proposto para justificar de forma mais sólida a pesquisa que foi conduzida. O Capítulo 4 contém enfim a descrição do método desenvolvido, o *Corisco*, incluindo detalhes a respeito de técnicas que foram empregadas, no contexto desta aplicação. O Capítulo 5 relata experimentos realizados para validar o método desenvolvido, e o Capítulo 6 conclui a tese com uma crítica

do trabalho desenvolvido e algumas sugestões de formas como o método proposto ainda poderia ser modificado ou empregado.



## 2 Fundamentos técnicos

*Man's desires are limited by his perceptions,  
none can desire what he has not perceived.*

—**There is No Natural Religion**, William Blake

Este capítulo contém uma introdução técnica à Visão Computacional, restrita aos tópicos mais relevantes a esta tese. São discutidos o processo de formação de imagens em câmeras fotográficas e as relações geométricas existentes entre os parâmetros dos modelos de cena e de câmera, além de como estas relações podem ser utilizadas em processos de estimativa de parâmetros. Esta base de conhecimentos é compartilhada por todos os métodos de Visão Computacional que serão discutidos adiante nesta tese, incluindo o *Corisco* aqui proposto, e o conteúdo deste capítulo é portanto indispensável para compreender o funcionamento de todos eles. Estes métodos compartilham diversas técnicas de análise das imagens e de estimativa de parâmetros, mas se diferem nos algoritmos e modelos específicos que são utilizados, além de quais entidades são analisadas, e quais parâmetros são conhecidos e quais são encontrados ao final de cada processo. Um aspecto bastante peculiar do *Corisco* é o uso da técnica de M-estimação, discutida na conclusão deste capítulo, enquanto métodos similares existentes utilizam outras técnicas de estimativa que também serão apresentadas.

A Visão Computacional, doravante denominada apenas *visão*, pode ser considerada uma área de estudos proveniente da Inteligência Artificial (IA) que busca atender ao desejo de “fazer um computador ver” (HARTLEY; ZISSELMAN, 2003, pg. xi). A visão compartilha com outras áreas da IA dificuldades relacionadas à representação, inferência e aprendizagem, porém envolve simultaneamente muitas questões relativamente complexas de natureza estritamente física ou geométrica, situadas principalmente nos estágios mais fundamentais de qualquer aplicação. A importância da compreensão destas questões para o estudo da área como um todo é ressaltada por Forsyth e Ponce (2003, pg. xvii). Estes autores ainda sintetizam a atividade principal da visão como sendo “extrair descrições do mundo a partir de imagens ou sequências de imagens”, enquanto Trucco e Verri (1998, pg. 1) definem os problemas estudados pela visão como “a computação de propriedades do mundo 3-D a partir de uma ou mais imagens”.

Aplicações completas de visão envolvem necessariamente alguma forma de imagem como entrada de dados. Em uma aplicação contemporânea, na maior parte das vezes esta entrada será composta por uma ou mais imagens capturadas através de uma câmera composta por um arranjo de lentes e um sensor de imagens baseado em tecnologias da eletrônica digital como CCD ou CMOS. Uma vez que as imagens estejam disponíveis, técnicas de processamento de imagens podem ser utilizadas para se obter a partir delas dados que

servem de entrada para estágios superiores de análise. Estes dados são geralmente valores numéricos calculados a partir de filtros lineares e outras formas relativamente simples de processamento de sinais.

Sistemas de visão podem ser divididos de acordo com a natureza dos dados que são extraídos das imagens. Algumas aplicações realizam análises mais sofisticadas tratando as imagens como superfícies contínuas. Alguns exemplos são reconhecimento de faces (FORSYTH; PONCE, 2003, cap. 22), estimativa de forma a partir de sombreado (*shape from shading*) (TRUCCO; VERRI, 1998, cap. 9), a estimativa diferencial de fluxo ótico (TRUCCO; VERRI, 1998, cap. 8) e ainda a reconstrução densa. A outra família é a dos casos como o da presente pesquisa, onde ocorre uma extração de características geométricas, as quais são modelos geométricos primitivos como pontos e linhas que são ajustados às imagens segundo algum critério de minimização de erro ou de maximização de verossimilhança. Nesta classe se encontram aplicações bastante fundamentais de visão, como algoritmos clássicos para calibração de câmeras (TRUCCO; VERRI, 1998, cap. 6), reconhecimento de objetos, localização de objetos e de câmera por estereopsis (HARTLEY; ZISSEMAN, 2003) e a maior parte dos sistemas de rastreamento de câmera em tempo real.

A Seção 2.1 a seguir descreve o processo de formação de imagens e como este fenômeno é modelado matematicamente. Os diferentes modelos de câmera e diferentes formas de distorção possíveis são estudadas, bem como o fenômeno da formação de um edgel pela projeção na imagem de um ponto localizado sobre uma reta do ambiente. A Seção 2.2 aborda a forma como se pode analisar imagens para obter as chamadas *características geométricas* das imagens que podem ser utilizadas para diferentes fins. As Seções 2.3 e 2.4 discutem técnicas que são usualmente empregadas para resolver problemas de estimativa de parâmetros baseados nestas características extraídas e nos modelos de câmera, focando respectivamente em processos de buscas aleatórias baseadas em combinações dos dados, e em processos de otimização contínua baseados em Teoria das Probabilidades.

## 2.1 Formação de imagens

Uma câmera fotográfica é um dispositivo constituído por um arranjo de lentes e um sensor capaz de capturar imagens de um *ambiente*, ou *cena*, existente à frente da câmera. Um ambiente é constituído por um conjunto de objetos sólidos distribuídos pelo espaço. Assumimos que existe uma fonte de luz iluminando este ambiente. A luz é uma forma de energia que se propaga pelo espaço e pode ser absorvida, refletida ou refratada pelos objetos. A intensidade de luz incidindo ou sendo refletida em um determinado ponto na superfície de um objeto é representada por um valor numérico, um número real positivo. A luz pode ser decomposta em diferentes comprimentos de onda, cores primárias, ou *canais*, e valores diferentes de intensidade podem ser atribuídos a cada um destes ca-

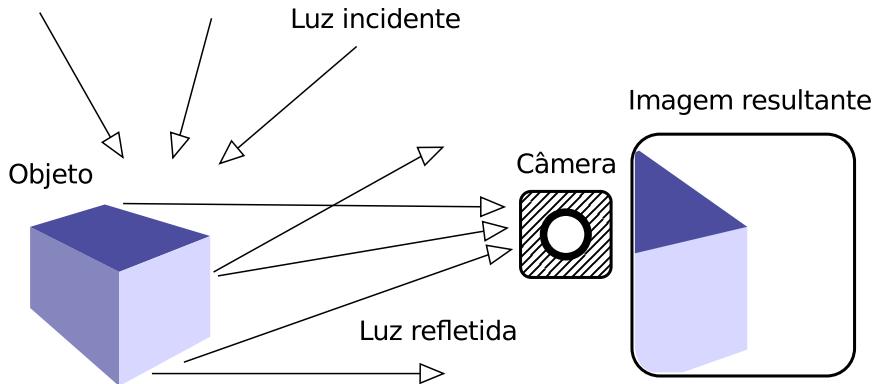


Figura 2.1: Objeto em um ambiente refletindo raios de luz capturados por uma câmera, e a imagem resultante.

nais. Esta combinação das intensidades nos diferentes canais da luz que é emitida por um ponto de uma superfície de um objeto, ou que incide sobre um ponto de uma imagem é denominada simplesmente a *cor* daquele ponto da imagem ou objeto.

Cada ponto das superfícies dos objetos de uma cena pode refletir a luz incidente com diferentes intensidades em cada canal. A intensidade da luz refletida depende de vários fatores, como características da fonte de luz e do material que constitui o objeto, além da geometria da cena. Estas variações na intensidade da luz refletida dão origem às variações nas cores e ao sombreamento que caracterizam o aspecto dos objetos ao serem observados. A Figura 2.1 ilustra todo este processo. Fontes de luz iluminam um objeto, que reflete esta luz para várias direções. Uma cena iluminada produz um chamado *campo de luz*, que é a especificação da quantidade de luz sendo transmitida através de cada ponto do espaço para cada direção possível, ou seja, o conjunto de todos os raios de luz formados pela cena em uma condição específica de iluminação.

Ao capturar uma imagem do ambiente uma câmera se situa em uma posição específica, e capture apenas os raios que atingem um determinado ponto do espaço, denominado *ponto focal* da câmera. O ponto focal da câmera é quase sempre utilizado para modelar a própria posição da câmera em qualquer aplicação onde esta grandeza é relevante. Cada ponto da imagem obtida corresponde a uma das direções dos raios incidentes neste ponto focal, de forma biunívoca. Este mapeamento bijetivo entre as direções dos raios que incidem sobre o ponto focal e as coordenadas do ponto correspondente a este raio sobre a imagem produzida pela câmera constitui o chamado *modelo de câmera*.

As cores de cada ponto de uma imagem capturada por uma câmera são portanto as cores dos raios de luz emitidos por pontos sobre as superfícies dos objetos da cena. Cada um destes pontos sobre os objetos localiza-se a uma determinada direção com relação à câmera, e esta direção possui uma dependência funcional com a posição do ponto na imagem em que será projetado o ponto do ambiente. A distância entre o ponto e a câmera pode ter uma influência na intensidade de luz registrada, mas este e outros

fenômenos mais complexos, tal como a profundidade de campo ou o efeito de vinheta, serão ignorados neste trabalho.

Apesar dos raios de luz propagarem-se da cena para a câmera, é útil enxergar o processo como se ocorresse no sentido oposto. A cor da imagem em um determinado ponto selecionado pode ser descoberta a partir de um raio que parte da câmera apontando na direção dada pelo modelo de câmera. Estendendo-se este raio em direção ao ambiente, ele eventualmente tocará a superfície de algum objeto. A intensidade da luz sendo emitida na direção da câmera por este ponto é a cor atribuída ao ponto da imagem selecionado.

A simulação deste processo de formação de imagens com o intuito de criar imagens artificiais é parte do que constitui a área de estudos da Computação Gráfica. Uma aplicação típica de Computação Gráfica é criar uma imagem a partir de um modelo geométrico de um conjunto de volumes sólidos que representam um ambiente. Além dos modelos das superfícies dos objetos, é preciso conhecer também todos os parâmetros que modelam as fontes de luz e os materiais dos objetos, além dos parâmetros que modelam a câmera. Uma vez que se conhecem todos estes parâmetros, é possível então calcular os valores de cor de qualquer ponto sobre a imagem realizando um processo de busca pelas superfícies que interceptam o seu raio correspondente, como descrito anteriormente. Supondo que todos objetos são opacos, apenas a superfície mais próxima da câmera que é cortada por um raio é considerada para a determinação da cor do ponto da imagem associado a este raio. Isto é necessário para simular o fenômeno da *occlusão* dos objetos mais afastados da câmera pelos mais próximos.

A Visão Computacional é, de certa forma, o oposto da Computação Gráfica. Na Computação Gráfica os parâmetros que modelam o ambiente e a câmera são todos conhecidos, e é preciso produzir uma imagem de saída. Enquanto isso, na visão as imagens são os dados de entrada dos processos e é preciso produzir como saída os parâmetros que modelam a cena e a câmera. A maioria dos problemas de Computação Gráfica envolve cálculos com equações fechadas, e o que se busca são maneiras de acelerar os cálculos e aperfeiçoar os modelos para representar novos materiais e tipos de objetos para alcançar resultados mais realistas e com maiores detalhes, mas com a maior eficiência computacional possível. Já na visão a maioria dos problemas é alguma forma de reconhecimento de padrões, classificação ou estimativa de parâmetros, e suas soluções exigem quase sempre alguma forma de otimização, não só otimização contínua para buscar parâmetros de modelos geométricos, mas também otimização inteira e combinatória para a associação de dados.

### 2.1.1 Exemplos de modelo de câmera

A Figura 2.2 demonstra dois exemplos de modelo de câmera. À esquerda se encontra o chamado modelo de *buraco-de-agulha*, ou *pinhole* em inglês, em referência à técnica de fotografia em que se usam câmeras com um pequeno furo ao invés de lentes. À direita

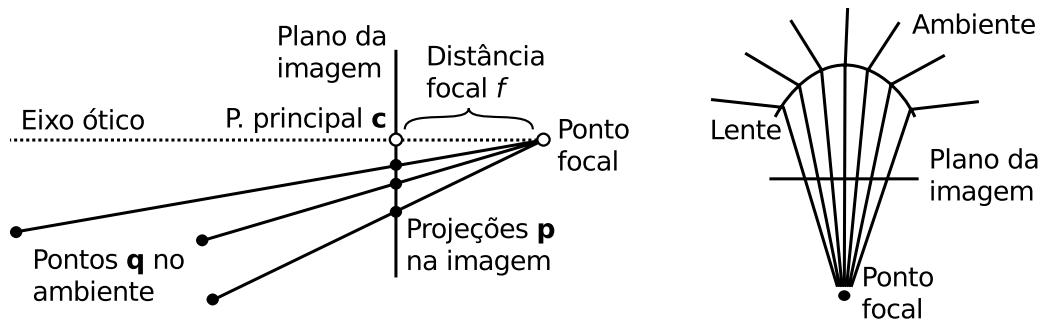


Figura 2.2: Modelo de câmera *pinhole* (esq.) e de uma lente olho-de-peixe (dir.).

da Figura 2.2 encontra-se a ilustração de modelo de câmera de uma lente olho-de-peixe, onde a projeção não segue o modelo de *pinhole*.

O modelo *pinhole* é bastante importante, primeiro em teoria por ser um modelo constituído de forma completa e exclusiva pelo o processo de *projecção pontual*, também chamada *projecção perspectiva*, *projecção retilínea* ou ainda *projecção gnômica*. Esta projeção, no contexto da análise de imagens, dá origem à chamada *distorção de perspectiva*. O modelo de câmera *pinhole* também é bastante útil na prática, e é utilizado para modelar a maior parte das câmeras disponíveis no mercado de consumo. A imagem no modelo *pinhole* é formada em um plano localizado a uma certa distância do ponto focal. Esta distância é denominada *distância focal*. A linha que une o ponto focal a cada ponto da imagem dá exatamente a direção do raio de luz incidente associado àquele ponto da imagem. A projeção pontual faz parte dos ensinamentos fundamentais das artes gráficas, e é conhecida há muito tempo por desenhistas e pintores (KLINE, 1985, Cap. 10). A Figura 2.3 mostra uma gravura de um manual de desenho histórico que ilustra a projeção pontual.

No caso do modelo representado à direita da Figura 2.2 a relação entre as direções dos raios e as coordenadas da imagem não é tão simples quanto no modelo *pinhole*. Este exemplo demonstra algo observado em lentes do tipo olho-de-peixe, por exemplo, onde pontos na extremidade da imagem podem receber a projeção de raios provenientes de direções a até mais do que 90 graus em relação ao centro da imagem. Isto é impossível de conseguir com a projeção pontual. Outros exemplos de imagens que possuem relações mais complexas entre as coordenadas de imagem e as direções dos raios provenientes são os sistemas catadióptricos construídos com espelhos curvos, e imagens construídas a partir de várias câmeras utilizando projeções tal como a equirectangular. Mas até mesmo as lentes grande-angulares convencionais como as utilizadas para vigilância podem apresentar distorções apreciáveis.

A projeção equirectangular, também denominada *projecção cilíndrica equidistante*, *projecção geográfica* ou *plate carrée*, é uma das projeções cartográficas que podem ser utilizadas para construir mapas do globo terrestre com dimensões limitadas (SWART; TOR-

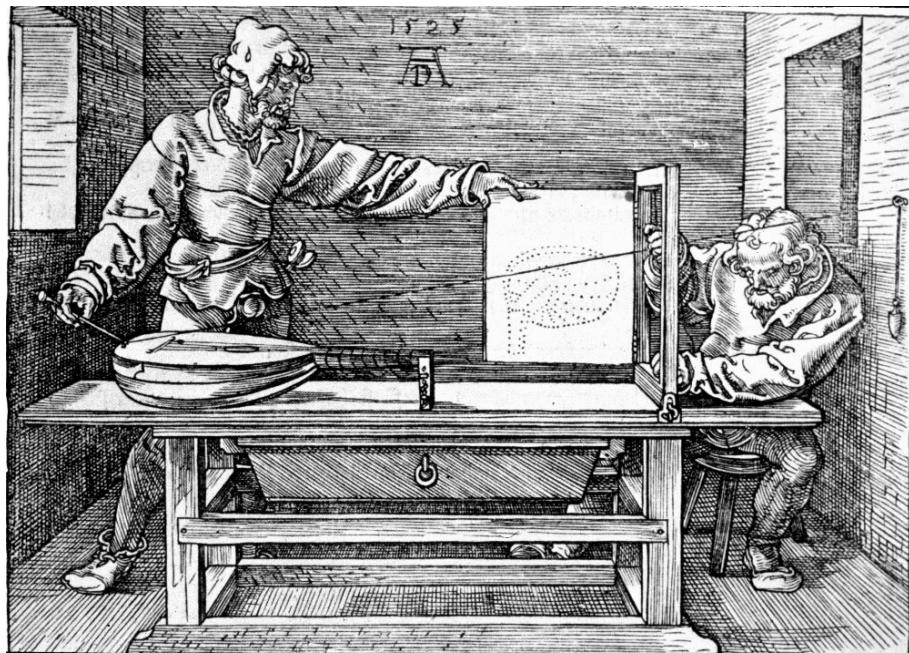


Figura 2.3: Homem desenhando um alaúde, Albert Dürer (1525).

RENCE, 2011; TORII; HAVLENA; PAJDLA, 2009). Nesta projeção as coordenadas cartesianas da imagem estão diretamente ligadas aos parâmetros da direção do raio de luz associado dados em coordenadas esféricas, que seriam a latitude e longitude de um ponto do globo em um mapa. Imagens construídas com esta projeção, assim como as imagens de alguns sistemas catadióptricos e algumas lentes olho-de-peixe podem ser capazes de varrer grandes extensões do espaço, e geralmente buscam varrer 360 graus ao redor de algum eixo. Uma imagem que possui um campo de visão tão grande é geralmente denominada *onidirecional*.

O uso da projeção equiretangular tem se tornado bastante comum em aplicações com imagens onidirecionais, por ser uma forma prática de se armazenar um panorama completo, contendo todas as direções, visível a partir de algum ponto do espaço. Estas imagens são geralmente obtidas através da fusão de imagens de várias câmeras convencionais separadas. A criação de grandes bancos de dados de imagens deste tipo por algumas empresas, além da própria disponibilidade de equipamentos que produzem este tipo de imagem, está fazendo crescer o interesse em técnicas de visão que possam ser empregadas nelas sem problemas. Já os sistemas catadióptricos e lentes olho-de-peixe estão se tornando comuns em robôs móveis e em vigilância, e assim cresce também o interesse em técnicas de visão que possam ser utilizadas em imagens com estes outros tipos de distorção.

Imagens com projeções como a equiretangular são dificilmente produzidas por algum sistema físico e, como foi dito, são geralmente o resultado de alguma forma de processamento de imagens. Sistemas físicos de produção de imagens são quase sempre modelados



Figura 2.4: Imagens de câmeras comerciais, que exemplificam o modelo de câmera *pinhole*. Estas imagens possuem na realidade distorções muito sutis, difíceis de observar.

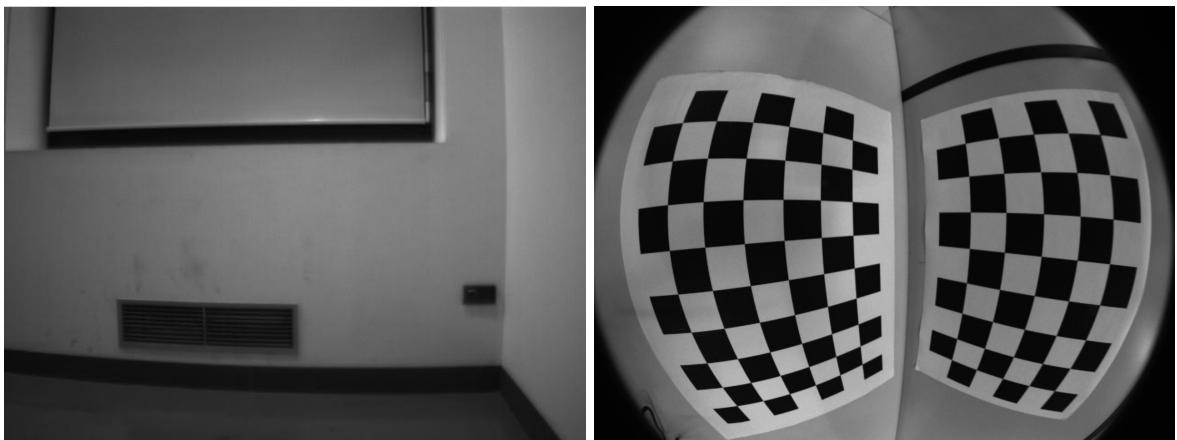


Figura 2.5: Uma imagem obtida com uma lente grande-angular, com distorção radial aparente (esq.) (CERIANI et al., 2009), e uma imagem com distorções intensas obtida através de uma lente olho-de-peixe com mais de 180 graus de campo de visão (dir.).

com base no modelo *pinhole*, porém considerando-se que a imagem dita “ideal” formada no plano daquele modelo sofre ainda uma distorção antes de ser disponibilizada. E esta distorção é quase sempre radial, o que significa que cada ponto da imagem é mapeado sobre uma linha que o liga até um ponto central da distorção, e o deslocamento de cada ponto também é uma função da sua distância até este centro. Um efeito importante deste tipo de distorção é que linhas que são retas na imagem original, de projeção pontual ideal, tornam-se curvas, a não ser que elas passem pelo centro da distorção.

A Figura 2.4 traz dois exemplos de imagens obtidas com câmeras fotográficas digitais comuns, disponíveis para consumidores não-profissionais. A Figura 2.5 demonstra uma câmera com lente grande-angular, com distorções aparentes, e também uma imagem obtida com uma lente olho-de-peixe, com distorções bastante intensas. A Figura 2.6 traz por fim um exemplo de imagem com projeção equiretangular.

A Seção 2.1.2 a seguir descreve de forma mais detalhada como são os modelos de



Figura 2.6: Uma imagem com projeção equiretangular, criada a partir de um arranjo de câmeras montado sobre um carro (fonte: Google Inc., projeto StreetView).

câmera, com as equações necessárias. Vários modelos diferentes são discutidos, mas é importante notar que o *Corisco* não depende do uso de nenhum modelo específico. Estes diferentes modelos são apenas utilizados para produzir coeficientes que são utilizados de uma mesma forma durante a estimativa de parâmetros.

### 2.1.2 Equações de modelos de câmera

O modelo de câmera é constituído por um mapeamento de pontos  $q$  do ambiente ao redor da câmera para pontos  $p$  localizados sobre o plano da imagem criada. O vetor  $q$  se baseia no referencial da câmera, cuja origem é o ponto focal. O mapeamento de  $q$  para  $p$  é uma projeção, logo todos os múltiplos escalares de um dado vetor  $q$  são projetados no mesmo ponto. Por este motivo a norma de  $q$  é irrelevante nestas equações, e este vetor é utilizado menos como uma representação de uma posição em três dimensões, e mais como uma representação redundante de direção, com um grau de liberdade a mais do que o necessário.

Para definir um mapeamento inverso de  $p$  para  $q$  é preciso escolher um conjunto restrito de vetores  $q$  no espaço tridimensional que formam a imagem do mapeamento — “imagem” no sentido matemático, de resultado de uma função. Por exemplo, é possível considerar que o mapeamento ocorre entre o plano da imagem, e a superfície de uma esfera cujo centro é o ponto focal. Este mapeamento inverso calcula uma direção  $q$  no espaço de onde origina o raio projetado sobre um ponto  $p$  da imagem. Esta esfera de raio unitário centrada no ponto focal é um lugar geométrico utilizado em muitas técnicas, e é denominada *esfera Gaussiana*.

No caso da projeção pontual, do modelo de câmera *pinhole*, o mapeamento é calcu-

lado pelas equações

$$\begin{aligned} p^x &= (q^x/q^z)f + c^x \\ p^y &= (q^y/q^z)f + c^y \end{aligned} \quad (2.1)$$

onde o vetor  $\mathbf{c} = (c^x, c^y)$  é o chamado *ponto principal* ou *centro de projeção*. A origem do sistema referencial da imagem é a extremidade superior esquerda da imagem, e o plano da imagem é colocado paralelo aos eixos  $x$  e  $y$ , e ortogonal portanto ao eixo  $z$  do referencial de câmera. O eixo  $z$  neste caso é denominado o *eixo principal* ou *eixo ótico* do arranjo de lentes, e o ponto principal é onde este eixo corta o plano da imagem. Como dito anteriormente, a distância do plano até o ponto focal é a distância focal  $f$ . Esta relação entre  $\mathbf{p}$  e  $\mathbf{q}$  pode ser verificada na Figura 2.2. Os valores de  $\mathbf{c}$  e  $f$  constituem os parâmetros intrínsecos no caso deste modelo de câmera.

O centro de projeção costuma se localizar próximo ao centro das imagens analisadas. A relação entre a distância focal e o tamanho da imagem determina o campo visual, e a distância focal também funciona como um fator de escala. Câmeras com distância focal muito grande produzem imagens com um campo bastante restrito, mas são capazes de “enxergar mais longe”, ampliando a imagem a partir do centro de projeção. Se a distância focal é reduzida a escala da imagem diminui, e o campo visual aumenta. No limite em que a distância focal vai para o infinito obtemos a chamada *projeção ortográfica*, ou *paralela*, que se caracteriza pela independência de  $\mathbf{p}$  em relação à variável  $q^z$ .

Câmeras disponíveis comercialmente que podem ser aproximadas com o modelo *pinhole* possuem tipicamente uma distância focal próxima das dimensões da própria imagem. Lentes que produzem distâncias focais abaixo de três quartos da largura da imagem costumam ser consideradas *grande-angulares*. Uma distância focal de metade da largura da imagem produz um campo visual de noventa graus na direção horizontal. Uma distância focal 4 vezes maior do que a largura da imagem produz um campo visual com menos de 30 graus.

Lentes com grandes campos visuais, e portanto distâncias focais curtas, costumam introduzir alguma forma de distorção nas imagens. Lentes com distâncias focais moderadas também podem causar distorções, mas elas serão tipicamente mais modestas ou até desprezíveis, enquanto no caso de lentes grande-angulares estas distorções costumam ser bastante intensas. A distorção introduzida por uma lente é geralmente modelada através de uma função de simetria radial. O centro de projeção pode ser tomado como sendo o centro da distorção. Existem casos em que o centro de distorção pode não corresponder ao centro de projeção, mas esta é uma simplificação que pode ser feita sem causar muitos problemas na maior parte dos casos, e que não trouxe desvantagens aparentes durante a pesquisa aqui apresentada.

Em um modelo de câmera com distorção radial as coordenadas finais da projeção são calculadas primeiro através de uma projeção ideal  $\mathbf{p}'$ , dada simplesmente pela Equação 2.1 considerando o ponto principal localizado sobre a origem do referencial de ima-

gem, ou seja

$$\begin{aligned} p'^x &= (q^x/q^z)f \\ p'^y &= (q^y/q^z)f. \end{aligned} \quad (2.2)$$

A relação entre esta projeção pontual ideal e o ponto projetado final, distorcido, é então obtida por um fator de escala que depende da distância do ponto projetado ideal ao ponto principal. As coordenadas do ponto principal  $c$  no referencial da imagem final são então incrementadas a esta posição distorcida. Ou seja,

$$\begin{aligned} |\mathbf{p}'| &= \sqrt{p'^x{}^2 + p'^y{}^2} \\ p^x &= p'^x g(|\mathbf{p}'|) + c^x \\ p^y &= p'^y g(|\mathbf{p}'|) + c^y. \end{aligned} \quad (2.3)$$

Quando a função de distorção  $g(x) = 1$ , temos o caso da projeção pontual ideal. Um modelo básico e bastante utilizado em aplicações com pouca distorção é dado por

$$g(x) = 1 + \kappa x^2, \quad (2.4)$$

onde  $\kappa$  representa o coeficiente de distorção. Polinômios de maior ordem também podem ser utilizados (TORDOFF; MURRAY, 2004). Um detalhe importante sobre este modelo é o fato de que o polinômio par produz uma função final ímpar relacionando  $\mathbf{p}'$  a  $\mathbf{p}$ . Isto é uma condição necessária devido à simetria radial da distorção. Este modelo também utiliza uma expressão bastante simples, que não depende do cálculo de funções complexas tal como a raiz quadrada, e isto o torna interessante em cenários de aplicação de alto desempenho que não possam funcionar de outra forma.

Uma desvantagem deste simples modelo polinomial é em primeiro lugar a dificuldade em se calcular a sua função inversa. O cálculo do mapeamento inverso é algo necessário em algumas aplicações, incluindo o método desenvolvido. Outra desvantagem é a má aproximação das funções de distorção mais intensas encontradas em lentes do tipo grande-angular ou olho-de-peixe.

Existem modelos alternativos que atendem melhor a estes requisitos. Um deles é o modelo de divisão de Fitzgibbon (STRAND; HAYMAN, 2005), que utiliza a equação

$$g(x) = \frac{1}{1 + \kappa x^2}. \quad (2.5)$$

Um outro modelo, atraente pela facilidade com que se pode calcular sua função inversa, é o de Harris (TORDOFF; MURRAY, 2004). Neste caso temos a fórmula

$$g(x) = \frac{1}{\sqrt{1 - 2\kappa x^2}}. \quad (2.6)$$

A função inversa do modelo de Harris é dada simplesmente pela troca do sinal do

coeficiente  $\kappa$ . Em ambos casos das Equações 2.4 e 2.6 o coeficiente  $\kappa$  está relacionado tanto à intensidade quanto o tipo de distorção. Se  $\kappa = 0$  não há distorção, e temos simplesmente a projeção pontual ideal. O sinal de  $\kappa$  determina se a distorção será do tipo *barril* ou *almofada*. Se  $\kappa > 0$  os pontos mais distantes da origem se distanciam mais ainda, causando a distorção do tipo almofada. Se  $\kappa < 0$  os pontos são retraídos na proporção da distância ao ponto, o que causa a distorção do tipo barril.

A Figura 2.7 ilustra os dois tipos de distorção. Em cada gráfico há um conjunto de edgels originais, e distorcidos. Os edgels originais foram amostrados de uma grade, em pares nas direções vertical e horizontal sobre as mesmas posições. O ponto onde cada par de edgels se encontram é a posição de ambos edgels, e suas direções são dadas pelos segmentos de reta desenhados. O mapeamento que constitui a distorção desloca os pontos, mas também faz com que toda a imagem ao redor de cada ponto seja deformada, e assim as direções destes edgels são alteradas além de suas posições. O cálculo da nova direção é realizado através de uma aproximação de primeira ordem do mapeamento, baseada no cálculo da matriz Jacobiana da transformação definida como

$$\mathbf{J} = \begin{bmatrix} \frac{\partial p^x}{\partial p'^x} & \frac{\partial p^x}{\partial p'^y} \\ \frac{\partial p^y}{\partial p'^x} & \frac{\partial p^y}{\partial p'^y} \end{bmatrix}. \quad (2.7)$$

No caso do modelo de Harris a matriz Jacobiana se torna

$$\mathbf{J} = \begin{bmatrix} 1 - 2\kappa p'^y^2 & 2\kappa p'^x p'^y \\ 2\kappa p'^x p'^y & 1 - 2\kappa p'^x^2 \end{bmatrix} \frac{1}{(1 - 2\kappa(p'^x^2 + p'^y^2))^{3/2}}. \quad (2.8)$$

Se um edgel se encontra na posição  $\mathbf{p}'$  antes da distorção, e sua direção é dada pelo vetor normalizado  $\mathbf{v}'$ , a direção do edgel distorcido será a direção do vetor  $\mathbf{J} \mathbf{v}'$ , onde  $\mathbf{J}$  deve ser calculado para esta posição  $\mathbf{p}'$  específica. O vetor resultante se encontra na direção que se deseja obter, ou seja

$$\mathbf{v} \propto \mathbf{J} \mathbf{v}', \quad (2.9)$$

onde o símbolo  $\propto$  indica uma relação de proporcionalidade, ou dependência linear. Para encontrar a nova direção  $\mathbf{v}$ , modelada por um vetor normalizado, basta portanto aplicar a matriz  $\mathbf{J}$  a  $\mathbf{v}'$  e então normalizar o vetor resultante. Devido a esta normalização, é possível desprezar o fator constante na Equação 2.8 para este cálculo de  $\mathbf{v}$  a partir de  $\mathbf{v}'$ .

É possível observar na Figura 2.7 os efeitos das distorções nas direções dos edgels. Em primeiro lugar, não há modificação tanto da posição quanto das direções dos edgels localizados sobre o ponto principal. No caso da distorção tipo almofada, a distorção faz com que as direções apontem mais na direção dada pela própria posição de cada edgel. No caso da distorção tipo barril ocorre o oposto, os edgels tendem a apontar para a direção

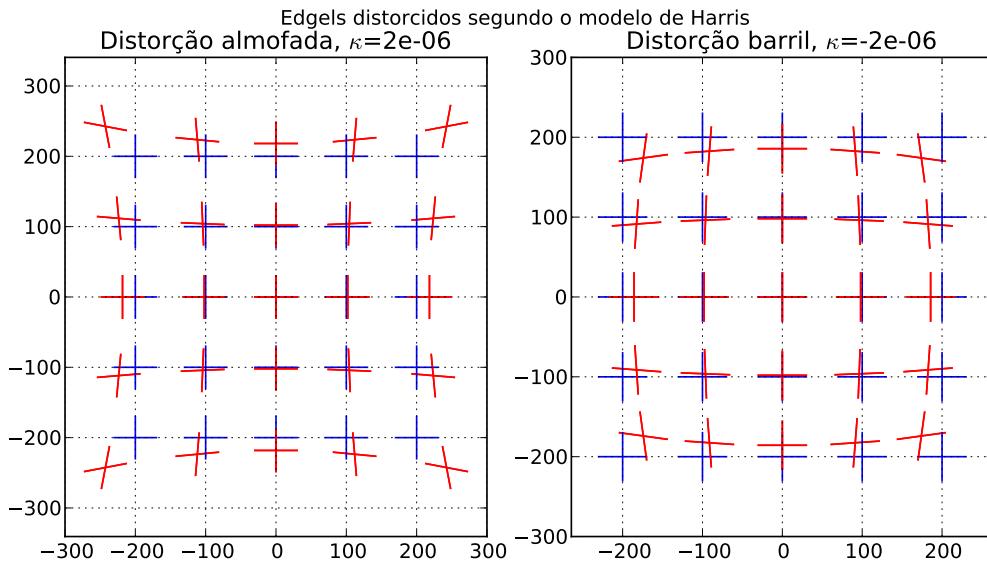


Figura 2.7: Demonstração de edgels retirados de uma grade, distorcidos pelo modelo de Harris em ambos sentidos.

ortogonal a esta, ou seja, a direção pela qual um círculo centrado na origem e com raio  $|p|$  cortaria este edgel. Um detalhe importante a respeito desta modificação das direções dos edgels é que um par de edgels de direções ortogonais podem não manter esta propriedade após o mapeamento, o que é demonstrado de forma clara nestes gráficos. O aspecto de uma grade após sofrer as distorções, demonstrada nestes gráficos, é o que dá origem aos nomes destes dois tipos de distorção.

Enquanto modelos de distorção radial como o de Harris costumam ser adequados para muitas aplicações, há casos em que esta dependência em um modelo ideal subjacente de projeção pontual limita a aplicabilidade deste modelo. Existem modelos mais adequados que devem ser empregados nestes casos, que dispensam esta ligação com a projeção pontual. Este é o caso de algumas lentes do tipo olho-de-peixe, capazes de capturar campos visuais de até mais do que  $180^\circ$ . Muitas lentes deste tipo são desenvolvidas para produzir imagens utilizando a projeção *polar azimuthal equidistante*. Nesta projeção o azimute da direção  $q$  de um determinado raio é definido pelo seu ângulo ao redor do eixo óptico do referencial de câmera. Este azimute é o mesmo ângulo em relação ao centro da projeção em que se encontra o ponto  $p$  na imagem correspondente a  $q$ . Todas as distorções radiais compartilham esta mesma propriedade. Já o ângulo  $\varphi$  entre  $q$  e o eixo óptico, que é o zênite desta projeção, está diretamente ligado à distância do ponto  $p$  até o centro de projeção, por uma relação linear. As fórmulas resultantes para calcular a

projeção de  $\mathbf{q}$  para  $\mathbf{p}$  são

$$\begin{aligned}\varphi &= \cos^{-1} \left( \frac{q^z}{\sqrt{q^{x^2}+q^{y^2}+q^{z^2}}} \right) = |\mathbf{p} - \mathbf{c}|/f \\ p^x &= \varphi \frac{q^x}{\sqrt{q^{x^2}+q^{y^2}}} f + c^x \\ p^y &= \varphi \frac{q^y}{\sqrt{q^{x^2}+q^{y^2}}} f + c^y.\end{aligned}\quad (2.10)$$

É interessante comparar a Equação 2.10 com o caso da projeção pontual, onde a fórmula da projeção em função de  $\varphi$  seria

$$\begin{aligned}p^x &= \tan(\varphi) \frac{q^x}{\sqrt{q^{x^2}+q^{y^2}}} f + c^x \\ p^y &= \tan(\varphi) \frac{q^y}{\sqrt{q^{x^2}+q^{y^2}}} f + c^y.\end{aligned}\quad (2.11)$$

É possível ainda encontrar uma formulação equivalente para o modelo de Harris, onde  $\varphi$  serviria de argumento para uma função mais complexa do que esta. É interessante notar ainda que no caso da projeção pontual temos

$$\varphi = \tan^{-1}(|\mathbf{p} - \mathbf{c}|/f). \quad (2.12)$$

Todos estes modelos de câmera com distorção radial possuem em comum o fato de que próximo ao ponto principal eles se aproximam de uma projeção pontual com distância focal  $f$ . Para ver isto basta analisar as aproximações de primeira ordem das equações ao redor do caso  $\varphi \approx 0$ . É importante lembrar ainda que o valor de  $\varphi$  na Equação 2.10 é dado em radianos. Quando a projeção polar azimutal equidistante é utilizada para fazer mapas, isto significa que uma linha reta traçada sobre o mapa a partir da origem até um ponto qualquer terá um comprimento proporcional à distância que seria medida sobre a superfície da Terra entre estes dois pontos, seguindo o círculo máximo geodésico que os liga sobre o globo. É a esta característica da projeção que se refere o termo *equidistante* neste caso.

Uma característica importante do modelo de Harris é que se  $\kappa < 0$ , existe uma distância limitada até o ponto principal que definirá a projeção de raios incidindo a  $90^\circ$  do eixo ótico. Esta distância será dada pelo ponto em que a distorção inversa vai para o infinito. Esta distância pode ser obtida através da Equação 2.6, e é dada por

$$x = \sqrt{\frac{1}{|2\kappa|}}. \quad (2.13)$$

Enquanto isto, na projeção equidistante a posição onde são projetados os raios vindos a  $90^\circ$  é

$$x = \frac{\pi}{2} f. \quad (2.14)$$

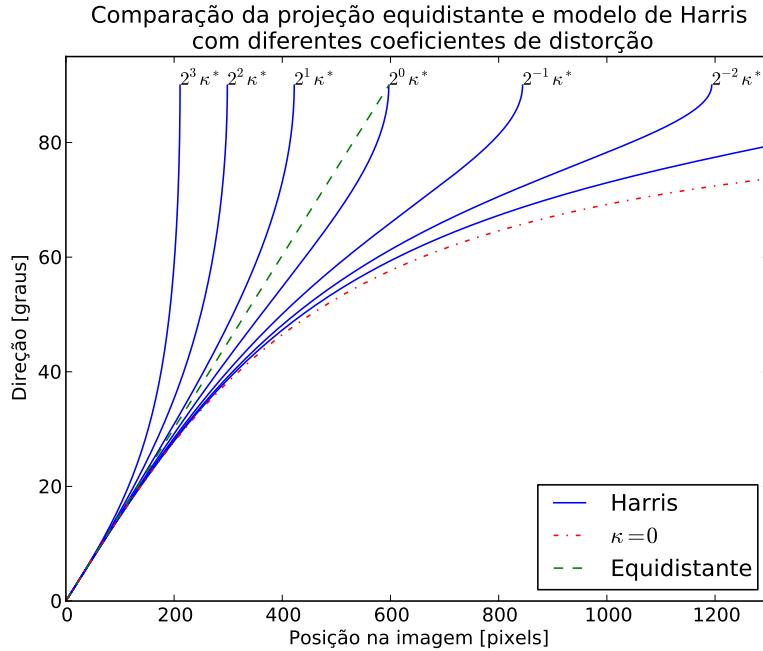


Figura 2.8: Comparação da projeção de Harris com diferentes valores de  $\kappa$ , e a projeção polar equidistante.

Utilizando as Equações 2.13 e 2.14 podemos encontrar o valor de  $\kappa$  que projeta os raios de  $90^\circ$  sobre o mesmo círculo que a projeção equidistante, utilizando o mesmo parâmetro  $f$ . Com este coeficiente de distorção o modelo de Harris se tornaria uma aproximação da projeção equidistante. Este valor do coeficiente é dado por

$$\kappa^* = -\frac{2}{(\pi f)^2}. \quad (2.15)$$

A Figura 2.8 mostra a relação entre o ângulo dos raios incidentes e a distância da projeção ao ponto principal para vários modelos de câmera diferentes. O eixo vertical do gráfico é o ângulo  $\varphi$ , e o horizontal é valor de  $|\mathbf{p} - \mathbf{c}|$ . No caso da projeção equidistante a relação é linear, como mostra a linha tracejada. A curva da extrema direita mostra o caso de uma projeção pontual, sem distorção radial de qualquer tipo. Esta curva é dada pela função arco-tangente, e a projeção vai para o infinito para raios a  $90^\circ$ . As curvas contínuas são distorções através do modelo de Harris. Cada curva foi criada com um valor diferente de  $\kappa$ . Os valores selecionados são o valor de  $\kappa^*$  multiplicado por potências inteiras de 2 ao redor de  $2^0 = 1$ .

É possível observar na Figura 2.8 que quando  $\kappa = \kappa^*$  as curvas se encontram em  $\varphi = 90^\circ$  e  $|\mathbf{p} - \mathbf{c}| = f\frac{\pi}{2}$ , sendo que  $f \simeq 380.0$  neste exemplo. A curva do modelo de Harris se localiza estritamente abaixo da linha da projeção equidistante. Conforme  $\kappa$  aumenta a curva gradualmente transita para a esquerda desta linha. Isto significa que o valor de  $\kappa = \kappa^*$  serve como uma espécie de estimativa superior do valor de  $\kappa$  que

produz a curva que melhor aproximaria a distorção equidistante com o modelo de Harris. Esta relação entre estes dois modelos foi comprovada experimentalmente durante esta pesquisa. Um outro fato relevante mostrado na Figura 2.8 é a similaridade de todos estes modelos na região próxima à origem, tornando-os equivalentes à projeção pontual com distância focal  $f$  como discutido anteriormente.

O modelo de Harris e a projeção polar equidistante constituem dois tipos de distorção radial. Um último modelo de câmera que também foi estudado nesta pesquisa, e que pertence a uma classe diferente, é o da projeção equiretangular. Imagens de projeção equiretangular não são produzidas por lentes, mas sim criadas a partir de outras imagens. Como já foi mencionado na Seção 2.1.1, esta projeção é bastante conhecida na cartografia, e é obtida através do uso dos ângulos de latitude e longitude de um ponto  $q$  para definir as coordenadas Cartesianas de sua projeção  $p$ , ou seja

$$\begin{aligned} p^x &= f \tan^{-1}(q^z, q^x) \\ p^y &= f \sin^{-1} \left( \frac{q^y}{\sqrt{q^{x^2} + q^{y^2} + q^{z^2}}} \right), \end{aligned} \quad (2.16)$$

onde a função  $\tan^{-1}(a,b)$  é a função arco-tangente de  $a/b$  levando em conta os sinais dos argumentos para produzir valores entre  $-\pi$  e  $\pi$ . Esta projeção também é equidistante, sendo que neste caso este termo se refere às distâncias medidas nas direções verticais sobre a imagem, ou ainda sobre a linha horizontal de latitude 0.

### 2.1.3 Cálculo da projeção de um edgel

A Figura 2.9 ilustra a formação de um edgel em uma imagem a partir de uma reta do ambiente. O ambiente possui uma reta vertical, alinhada ao eixo vertical do referencial natural do ambiente. Um ponto  $q$  desta reta foi selecionado no local onde ela cruza com uma reta hipotética, tracejada, alinhada a outra direção válida do ambiente. Este ponto selecionado da reta foi projetado sobre a imagem, sobre o ponto  $p$ . Um pequeno segmento de reta sobre a imagem corta este ponto, orientado na direção  $v$ . Este segmento é um pequeno trecho ao redor de  $p$  da curva sobre a imagem que constitui a projeção da reta completa. O vetor  $u$ , ortogonal a  $v$ , aponta na direção do gradiente da imagem calculado sobre o ponto  $p$ , conforme será discutido na Seção 2.2.

As setas sólidas na Figura 2.9 mostram as direções do referencial da câmera, partindo do ponto focal. As setas pontilhadas partindo deste mesmo ponto mostram as direções  $r$  das retas no ambiente, que são direções dos eixos do referencial do ambiente. A reta hipotética no ambiente produziria um edgel na imagem com a mesma posição do primeiro, mas com outra direção  $v$ . É importante perceber que nesta orientação de câmera estas duas direções  $v$  possíveis para este edgel não se mostraram ortogonais, apesar das direções  $r$  no ambiente serem ortogonais. As diferentes direções ortogonais no ambiente só produzem edgels ortogonais sobre um mesmo ponto em condições bastante específicas.

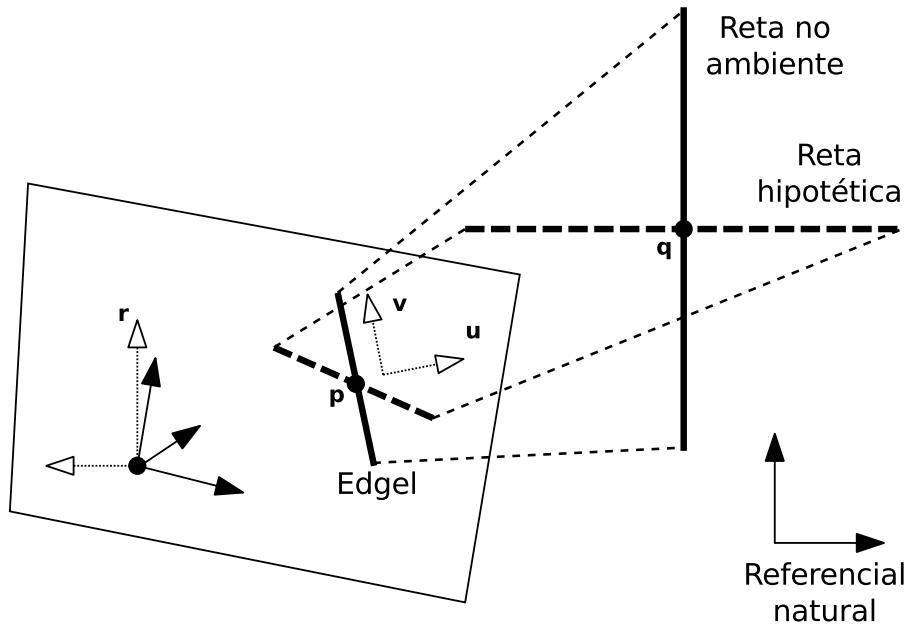


Figura 2.9: Projeção de um ponto contido em uma reta do ambiente produzindo um edgel na imagem.

Assim como a modificação da direção de um edgel pode ser encontrada através da aplicação da matriz Jacobiana da distorção sobre o plano da imagem, a matriz Jacobiana do mapeamento de projeção que constitui o modelo de câmera também pode ser utilizada para calcular as direções de um edgel. O que se deseja é encontrar a direção  $v$  da projeção sobre o ponto  $p$  da imagem de uma reta qualquer orientada na direção  $r$  no espaço. Este ponto  $p$  é a projeção da direção  $q$ , que pode ser encontrada através do mapeamento inverso da projeção. A Jacobiana da projeção é dada por

$$\mathbf{J} = \begin{bmatrix} \frac{\partial p^x}{\partial q^x} & \frac{\partial p^x}{\partial q^y} & \frac{\partial p^x}{\partial q^z} \\ \frac{\partial p^y}{\partial q^x} & \frac{\partial p^y}{\partial q^y} & \frac{\partial p^y}{\partial q^z} \end{bmatrix}. \quad (2.17)$$

A direção  $v$  é proporcional à simples aplicação da matriz  $\mathbf{J}$  a  $r$ , ou seja

$$v \propto \mathbf{J} r. \quad (2.18)$$

Assim para calcular  $v$  basta realizar a multiplicação do vetor pela matriz e normalizar o vetor resultante, exatamente como no caso da Equação 2.9.

Para o modelo *pinhole* dado pela Equação 2.1 a Jacobiana necessária para o cálculo das direções dos edgels seria

$$\mathbf{J} = \frac{f}{q^z} \begin{bmatrix} 1 & 0 & -q^x/q^z \\ 0 & 1 & -q^y/q^z \end{bmatrix}. \quad (2.19)$$

Como é possível aplicar qualquer fator multiplicativo não-nulo a esta matriz sem afetar

o resultado após a normalização, podemos simplificar a matriz desta fórmula tornando-a explicitamente dependente apenas dos valores conhecidos durante a análise de uma imagem

$$\mathbf{v} \propto \begin{bmatrix} f & 0 & -p^x \\ 0 & f & -p^y \end{bmatrix} \mathbf{r}. \quad (2.20)$$

Ou seja, dado um edgel encontrado na posição  $\mathbf{p}$  sobre a imagem, a Equação 2.20 permite dizer qual é a Jacobiana da projeção ao redor da direção  $\mathbf{q}$  associada a  $\mathbf{p}$ . Neste caso não é necessário calcular os valores de  $\mathbf{q}$ , que são apenas valores intermediários para calcular o que é realmente desejado, que são os coeficientes da matriz.

No caso do modelo de Harris a direção de uma projeção pode ser encontrada a partir das Equações 2.20, 2.8 e 2.3. Primeiro as coordenadas de uma projeção ideal, sem a distorção radial, são encontradas pela transformada inversa da Equação 2.3, que no caso do modelo de Harris significa apenas aplicar a mesma fórmula da Equação 2.6 porém com o sinal  $\kappa$  invertido. Conhecendo a posição não-distorcida, a direção do edgel é obtida através da Equação 2.20. A seguir a direção deste edgel após a distorção de Harris é obtida utilizando a Equação 2.8. Assim, dados os valores calculados de  $\mathbf{p}'$  a partir de  $\mathbf{p}$  temos

$$\mathbf{v} \propto \begin{bmatrix} 1 - 2\kappa p'^{y^2} & 2\kappa p'^x p'^y \\ 2\kappa p'^x p'^y & 1 - 2\kappa p'^{x^2} \end{bmatrix} \begin{bmatrix} f & 0 & -p'^x \\ 0 & f & -p'^y \end{bmatrix} \mathbf{r}. \quad (2.21)$$

No caso das projeções polar equidistante e equiretangular o cálculo do mapeamento inverso não envolve a projeção pontual ideal. Para a polar equidistante a direção  $\mathbf{q}$  relativa a um determinado ponto  $\mathbf{p}$  da imagem é dada por

$$\begin{aligned} \varphi &= |\mathbf{p}|/f \\ q^x &= \sin(\varphi)p^x/|\mathbf{p}| \\ q^y &= \sin(\varphi)p^y/|\mathbf{p}| \\ q^z &= \cos(\varphi). \end{aligned} \quad (2.22)$$

Como resultado,  $|\mathbf{q}| = 1$  e  $\sqrt{q^{x^2} + q^{y^2}} = \sin(\varphi)$ . Encontrando as derivadas da Equação 2.10 e substituindo os coeficientes de  $\mathbf{q}$  pelas expressões da Equação 2.22, e ignorando ainda um fator multiplicativo para simplificar os cálculos, a Jacobiana da projeção equidistante em um ponto  $\mathbf{p}$  é dada por

$$\mathbf{J} \propto \begin{bmatrix} (\sin(\varphi) \cos(\varphi) - \varphi)x^2 + \varphi & (\sin(\varphi) \cos(\varphi) - \varphi)xy & -\sin(\varphi)x \\ (\sin(\varphi) \cos(\varphi) - \varphi)xy & (\sin(\varphi) \cos(\varphi) - \varphi)y^2 + \varphi & -\sin(\varphi)y \end{bmatrix}, \quad (2.23)$$

onde  $x = p^x/|\mathbf{p}|$  e  $y = p^y/|\mathbf{p}|$ .

Na projeção equiretangular, as coordenadas  $\mathbf{p}$  da imagem são os ângulos que parame-

trizam a direção  $\mathbf{q}$  em um sistema de coordenadas esféricas. Assim podemos escrever

$$\begin{aligned}\theta &= p^x/f \\ \varphi &= p^y/f \\ q^x &= \cos(\varphi) \sin(\theta) \\ q^y &= \sin(\varphi) \\ q^z &= \cos(\varphi) \cos(\theta).\end{aligned}\tag{2.24}$$

O que resulta em  $|\mathbf{q}| = 1$ . Assim como no caso da projeção polar equidistante, os termos da Jacobiana são dados aqui aquém de um fator multiplicativo. A fórmula da Jacobiana é obtida derivando-se a Equação 2.16, e utilizando também a Equação 2.24. Nas expressões resultantes os coeficientes da Jacobiana podem se obtidos a partir de  $\mathbf{p}$ , calculando-se os ângulos  $\varphi$  e  $\theta$ , e então utilizando a fórmula

$$\mathbf{J} \propto \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ -\sin(\varphi) \cos(\varphi) \sin(\theta) & \cos^2(\varphi) & -\sin(\varphi) \cos(\varphi) \cos(\theta) \end{bmatrix}.\tag{2.25}$$

É interessante notar que, por se tratar de uma projeção cilíndrica,  $\partial p^x / \partial q^y = 0$  no caso da equiretangular.

Esta seção discutiu em detalhes as propriedades matemáticas de um modelo de câmera, e mais especificamente demonstrou como calcular a direção predita para um edgel sobre a imagem que se assume ser a projeção de um ponto de uma reta no ambiente. Este edgel é um ponto que pertence a uma curva sobre a imagem que seria formada pela projeção completa de uma reta do ambiente, e a direção do edgel é tangente a esta curva. Dadas as coordenadas da posição  $\mathbf{p}$  do edgel, a direção  $\mathbf{r}$  da reta no ambiente e o modelo de câmera, o cálculo do vetor  $\mathbf{v}$  na direção do edgel é obtido através da Equação 2.18. Este cálculo depende dos coeficientes da matriz  $\mathbf{J}$ , que é a Jacobiana do mapeamento de projeção que constitui o modelo de câmera. Como apenas a direção de  $\mathbf{v}$  é importante, é possível realizar estes cálculos sem se preocupar com a norma de  $\mathbf{v}$ , o que permite simplificar as expressões. As fórmulas para os coeficientes da matriz relativa aos modelos de câmera de projeção pontual ideal, projeção pontual com distorção radial, projeção polar equidistante e projeção equiretangular foram fornecidas nas Equações 2.20, 2.21, 2.23 e 2.25, respectivamente.

## 2.2 Segmentação e extração de bordas

Na Computação Gráfica é comum modelar os objetos do ambiente através de modelos geométricos controlados por parâmetros numéricos. Algumas formas geométricas usuais são esferas, superfícies bi-cúbicas e poliedros, especialmente poliedros constituídos por faces triangulares. Pontos, curvas e segmentos de linha reta são entidades geométricas ainda mais primitivas que também podem fazer parte do ambiente, e são naturalmente

encontradas dentro destes modelos citados. Os vértices e arestas de um poliedro, por exemplo, são constituídos por pontos e por segmentos de reta.

A projeção de um ponto do ambiente é um ponto em uma imagem. Se um ambiente é constituído por pequenos pontos luminosos, eles podem ser localizados em uma imagem através de uma busca por picos na intensidade luminosa da imagem. Esta é a forma como alguns sistemas de captura de movimentos funcionam: marcadores luminosos são colocados sobre o ator ou objeto sendo modelado e a posição de cada marcador pode ser facilmente reconhecida se as imagens forem capturadas em condições favoráveis (JOSEFSSON; NORDH; ERIKSSON, 1996).

Estruturas mais complexas no ambiente também produzem estruturas distintas na imagem, que podem ser identificadas. Uma superfície lisa e fosca de uma única cor produz na imagem uma região da mesma cor. Um objeto de uma certa cor contra um fundo de cor diferente produz na imagem uma região limitada da cor do objeto, circundada por uma região com a cor do fundo. Uma curva no espaço produz uma curva na imagem. Um segmento de linha reta no espaço produz um segmento de linha reta na imagem no caso do modelo de câmera *pinhole*, mas a reta poderá se tornar uma curva se uma distorção radial for aplicada, por exemplo.

Uma reta em um ambiente pode ser criada por um fio, por exemplo, mas a forma mais comum de se encontrar uma reta em cenas urbanas ou ambientes internos a edifícios é como uma aresta de um poliedro. Suponha que um ambiente possui um hexaedro formado por retângulos, como na Figura 2.1. Mesmo que o objeto seja constituído de forma homogênea por um único material, as condições de iluminação podem causar variações nas cores percebidas em cada face. Isto se deve à relação funcional entre a quantidade de luz que é refletida e o ângulo formado pela luz incidente com a direção normal de cada face. O resultado é que cada face do hexaedro adquire uma cor diferente, e produz em consequência uma região de cor diferente na imagem.

A ilustração no lado esquerdo da Figura 2.10 é uma reprodução da ilustração contida na Figura 2.1, que demonstra a formação de uma imagem de um ambiente com poliedros. Nesta ilustração da esquerda vemos a imagem capturada, e o que se percebe são regiões justapostas com cores uniformes. No caso de um modelo de câmera com projeção pontual perfeita, sem distorção, e objetos com arestas retas, as fronteiras entre estas regiões serão segmentos de reta. A segunda imagem, à direita da Figura 2.10, é uma réplica da imagem à sua esquerda, destacando a localização destes segmentos de reta que são as projeções na imagem das arestas do objeto. A cor da imagem em um dos lados de cada um destes segmentos de reta será a cor de uma das faces do objeto, e do outro lado do segmento encontraremos a cor de outra face, ou do fundo da cena. As setas na Figura 2.10 indicam três destas fronteiras, uma delas entre duas faces do objeto, e outras duas entre o objeto e o fundo branco.

Podemos concluir portanto que objetos de uma cena dão origem a regiões justapostas

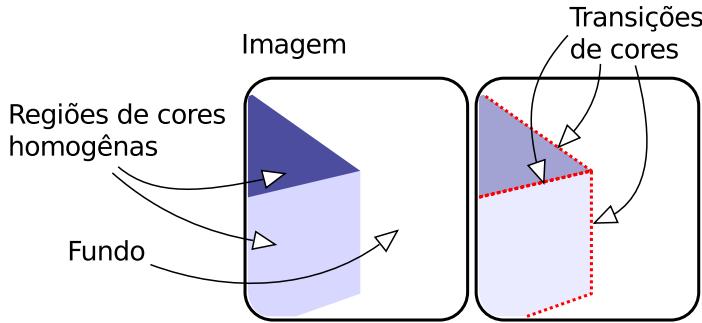


Figura 2.10: Imagem de um poliedro, com regiões de cor uniforme limitadas por bordas.

em imagens que possuem cores particulares. As diferenças de cor podem ser utilizadas para *segmentar* a imagem, o que significa determinar exatamente qual região da imagem corresponde um determinado objeto, ou a uma determinada face ou parte de um objeto. A fronteira entre dois objetos em uma imagem, ou a curva que delimita a região de um objeto ou de uma face, é denominada *uma borda*.

Uma borda pode ser uma curva de formato qualquer, incluindo segmentos de reta ou concatenações de segmentos de reta, ou outros modelos mais genéricos. Se todos objetos de uma cena forem poliedros, e se uma câmera com o modelo *pinhole* for utilizada, todas as bordas de objetos e de faces serão polígonos. No caso geral, o contorno da projeção de um objeto ou face na imagem será uma área contornada por uma borda com um formato qualquer. Um objeto que não possui arestas, constituído por superfícies curvas como no caso de uma esfera, também pode criar bordas em uma imagem. Se uma esfera de uma cor é observada contra um fundo de outra cor, a sua projeção dará origem a uma área curva, usualmente circular ou elíptica.

Bordas também podem ser definidas como sendo as curvas que separam áreas que se distinguem a partir de características mais complexas do que simplesmente o valor de cor. Certos materiais podem criar regiões com diferentes *texturas*, e também é comum que a cor de um objeto se modifique de forma suave ao longo de suas superfícies, mesmo que ainda de forma mais abrupta através de uma aresta. Existem também condições em que um objeto pode formar bordas na imagem sem que estas constituam um contorno fechado. Estes detalhes não serão discutidos a fundo no restante deste trabalho.

É preciso ressaltar que o significado exato do termo *borda* pode depender do contexto. No sentido mais estrito, a borda é todo o contorno fechado de uma área com características que diferem das regiões vizinhas. Mas o termo também pode se referir a cada um dos segmentos de reta que contornam uma face de um poliedro, por exemplo. Neste caso diríamos que a projeção de cada aresta do objeto constitui uma de suas bordas na imagem. No sentido mais amplo, uma borda é qualquer pequena região do espaço em que se verifica uma transição local abrupta em uma característica da imagem em uma determinada direção, mas pouca ou nenhuma variação na direção ortogonal. É neste sentido que se

fala em *detecção de bordas* em algum ponto da imagem, o que será discutido em breve neste texto.

A foto à direita na Figura 2.4 demonstra como é possível observar na prática o fenômeno descrito anteriormente de formação de bordas por objetos poliédricos. Os edifícios observados na foto possuem faces construídas com o mesmo material, pintadas com a mesma tinta. Apesar disto, as condições de iluminação fazem com que faces orientadas em diferentes direções adquiram cores diferentes na imagem, e assim as arestas dos edifícios formam bordas na imagem. É possível também observar bordas ocorrendo no encontro dos edifícios com o céu.

Esta relação íntima entre os modelos geométricos dos objetos e as bordas formadas na imagem dá origem à ideia de que poderíamos tentar localizar as bordas da imagem para então determinar, por exemplo, a localização no espaço dos objetos através do modelo de câmera. Este princípio serve como base para diversas aplicações de Visão Computacional. Esta determinação da localização de um modelo geométrico de uma região da imagem a partir das variações de cor é denominada extração de uma característica geométrica. Um extrator de segmentos de reta, por exemplo, é capaz de determinar as coordenadas das extremidades dos segmentos de reta na imagem que seriam as projeções das arestas de objetos como os edifícios e paredes internas na Figura 2.4. Um extrator de curvas mais genérico seria necessário para se localizar as projeções das arestas dos objetos nas Figuras 2.5 e 2.6, já que as bordas encontradas nestas imagens não são necessariamente compostas por segmentos de reta.

### 2.2.1 Detecção de borda

Há diversas técnicas bem-estabelecidas que podem ser utilizadas para a extração de retas ou outras características geométricas. Todas estas técnicas se baseiam em uma mesma maneira de representar imagens em um computador. Uma imagem digital é fundamentalmente um modelo de uma função definida sobre uma região retangular no plano cartesiano ( $\mathbb{R}^2$ ). Os valores determinados por esta função são a cor da imagem em cada ponto. Na prática, é comum modelar esta função através de uma amostragem regular de seus valores, utilizando para isto uma grade quadrada. Cada um destes pontos em que a função é amostrada é denominado um *pixel* — neologismo criado na língua inglesa a partir de “picture element”, ou elemento de imagem. Cada pixel possui um par de coordenadas que determina sua posição exclusiva na imagem, que são a linha e a coluna daquele pixel na grade utilizada para a amostragem, e portanto são números inteiros variando de zero até os limites vertical e horizontal da imagem. Neste trabalho adotamos a convenção de atribuir ao pixel superior esquerdo a posição (0,0), com os valores das coordenadas crescendo para baixo e para a direita na imagem. As coordenadas destes pixels são dadas no referencial de imagem, onde se localizam os vetores  $p$  discutidos na Seção 2.1.2.

Como imagens são quase sempre disponibilizadas desta forma, como uma tabela de

pixels com valores amostrados de cor, muitas das técnicas de processamento existentes foram criadas a partir disto. Por exemplo, muitas técnicas de segmentação de imagens determinam quais são os pixels que fazem parte de uma mesma região, ao invés de tentar determinar com maior precisão a posição por onde as bordas das regiões passam entre os pixels.

Uma técnica de análise muito comum é a chamada *detecção de borda*. Um detector é um classificador binário, e um detector de borda diz se um determinado pixel é ou não parte de uma borda da imagem. Mais especificamente, o detector diz se uma borda da imagem cruza uma certa região finita ao redor das coordenadas exatas do pixel, pois afinal de contas é bastante improvável que um pixel qualquer possa cair exatamente sobre as mesmas coordenadas por onde passa a curva que define uma borda. Um detector de bordas pode ser utilizado para construir um extrator de bordas. Para isto basta passar todos pixels da imagem pelo detector, e utilizar as coordenadas de cada pixel aceito pelo detector para dizer a posição aproximada de uma borda da imagem. Um extrator mais sofisticado vai analisar a saída do detector de pixels, e levar em consideração a vizinhança de cada pixel para possivelmente uni-los e construir curvas com a concatenação de pixels vizinhos que passaram pelo detector de bordas. Este tipo de técnica costuma receber o nome de *seguimento de contorno*, e é possível impor condições nas características desta borda extraída. Por exemplo, é possível criar seguidores de contorno especificamente para extrair segmentos de reta de uma imagem.

O primeiro passo para realizar qualquer processo que envolve detecção de borda é o cálculo do gradiente da imagem. O gradiente em um determinado ponto é dado por um vetor cuja componente horizontal é a derivada parcial na direção horizontal da função que modela a imagem, e o componente vertical é a derivada parcial na direção vertical. Estas derivadas são calculadas a partir de filtros lineares, aplicados à imagem através de convolução (TRUCCO; VERRI, 1998; ANDO, 2000). Cada canal de uma imagem tem o seu gradiente calculado separadamente. As Equações 2.26, 2.27 e 2.28 trazem as matrizes com os coeficientes de alguns filtros usuais, que foram testados nesta pesquisa. A matriz  $\mathbf{F}_{\text{Sobel}}$  da Equação 2.26 é utilizada com bastante frequência, mas é pouco acurada em comparação com muitas alternativas, e só se justifica em cenários em que não se requer muita precisão nos cálculos. Já a matriz  $\mathbf{F}_{\text{Ando}}$  utiliza valores mais precisos, adequados para aplicações que utilizam ponto-flutuante, e é um filtro que busca minimizar um determinado critério de consistência definido por Ando (2000). A matriz  $\mathbf{F}_{\text{Zernike}}$  não é utilizada exatamente para calcular o gradiente da imagem, mas pode ser interpretada desta forma. Ela é utilizada no método de detecção de bordas desenvolvido por Ghosal e Mehrotra (1993).

$$\mathbf{F}_{\text{Sobel}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} / 8 \quad (2.26)$$

$$\mathbf{F}_{\text{Ando}} = \begin{bmatrix} -0.003776 & -0.010199 & 0. & 0.010199 & 0.003776 \\ -0.026786 & -0.070844 & 0. & 0.070844 & 0.026786 \\ -0.046548 & -0.122572 & 0. & 0.122572 & 0.046548 \\ -0.026786 & -0.070844 & 0. & 0.070844 & 0.026786 \\ -0.003776 & -0.010199 & 0. & 0.010199 & 0.003776 \end{bmatrix} \quad (2.27)$$

$$\mathbf{F}_{\text{Zernike}} = \begin{bmatrix} -146.67 & -468.68 & 0. & 468.68 & 146.67 \\ -933.33 & -640. & 0. & 640. & 933.33 \\ -1253.33 & -640. & 0. & 640. & 1253.33 \\ -933.33 & -640. & 0. & 640. & 933.33 \\ -146.67 & -468.68 & 0. & 468.68 & 146.67 \end{bmatrix} / 19368.05 \quad (2.28)$$

As Figuras 2.11, 2.12 e 2.13 trazem exemplos de imagens e seus gradientes. Todos estes exemplos possuem um único canal. A matriz  $\mathbf{F}_{\text{Ando}}$  foi utilizada em todos os cálculos de gradiente apresentados nesta tese, exceto onde for especificado. A imagem na Figura 2.11 é artificial, e possui apenas um conjunto de círculos concêntricos. As cores dos gráficos que mostram o valor de cada componente do gradiente indicam o sinal do valor, e é possível ver como o gradiente fica positivo ou negativo conforme a imagem transita para uma área clara ou escura. A derivada na direção horizontal é nula nos pontos em que a borda dos círculos é horizontal, já que nestes pontos a derivada da imagem é estritamente vertical. A imagem da intensidade do gradiente, isto é, a norma euclidiana do vetor gradiente em cada pixel, mostra de forma clara que este valor é máximo onde se localizam as bordas dos objetos desta imagem.

As imagens das Figuras 2.12 e 2.13 não são artificiais, porém é possível notar como a intensidade do gradiente ainda indica a localização das bordas de objetos, como nas arestas superiores do edifício e nas prateleiras da estante. E a direção do gradiente, que pode ser obtida a partir de seus componentes, é ortogonal à direção destas bordas. Não é possível apreciar a precisão desta medição da direção nestas imagens, mas ao menos é possível notar que bordas mais aproximadamente verticais ou horizontais possuem valores nulos nos componentes destas mesmas direções, ou seja, bordas em uma destas direções não aparecem nos gráficos das derivadas parciais na mesma direção. Por exemplo, as divisórias verticais da estante não podem ser percebidas na derivada na direção  $y$  na Figura 2.13.

Porque uma borda é uma entidade unidimensional, uma curva definida sobre um plano, o problema da extração de bordas em imagens está intimamente ligado à versão unidimensional do problema, que é a determinação da localização de um “degrau” em

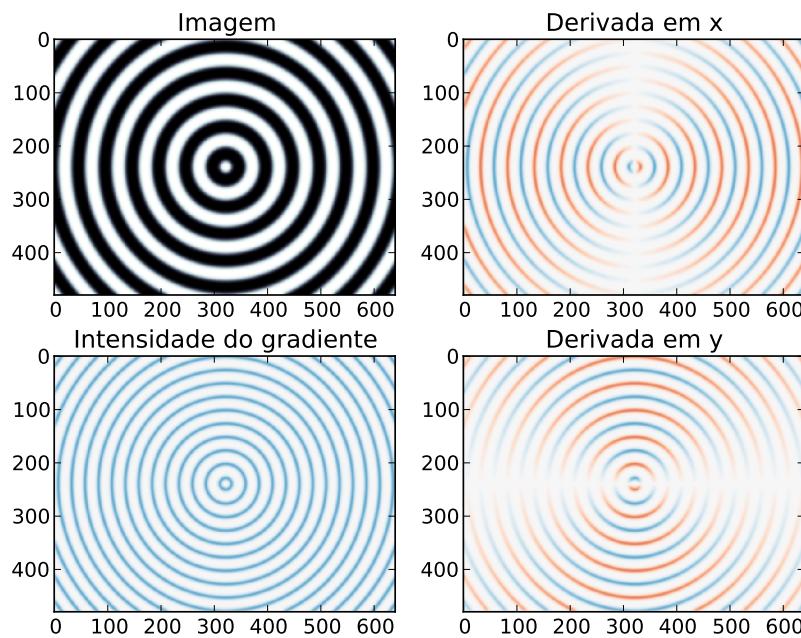


Figura 2.11: Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente.

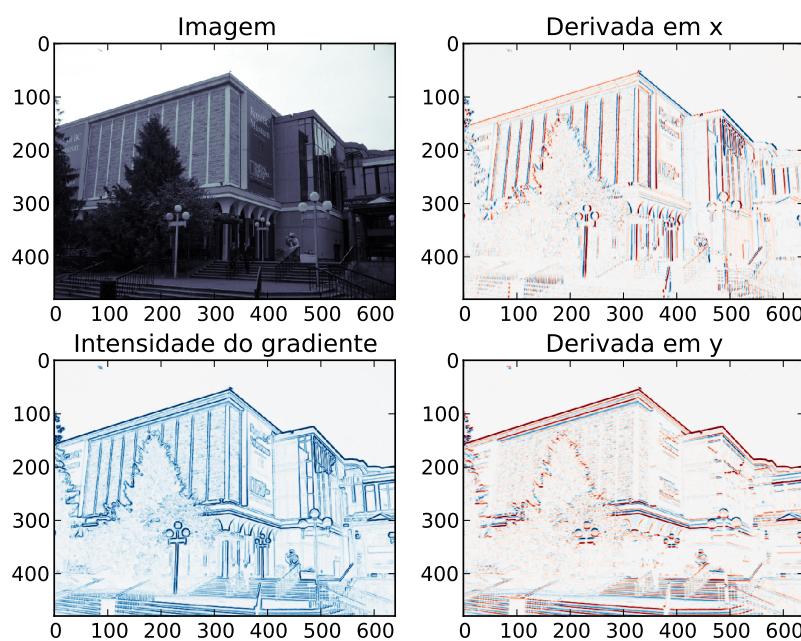


Figura 2.12: Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente.

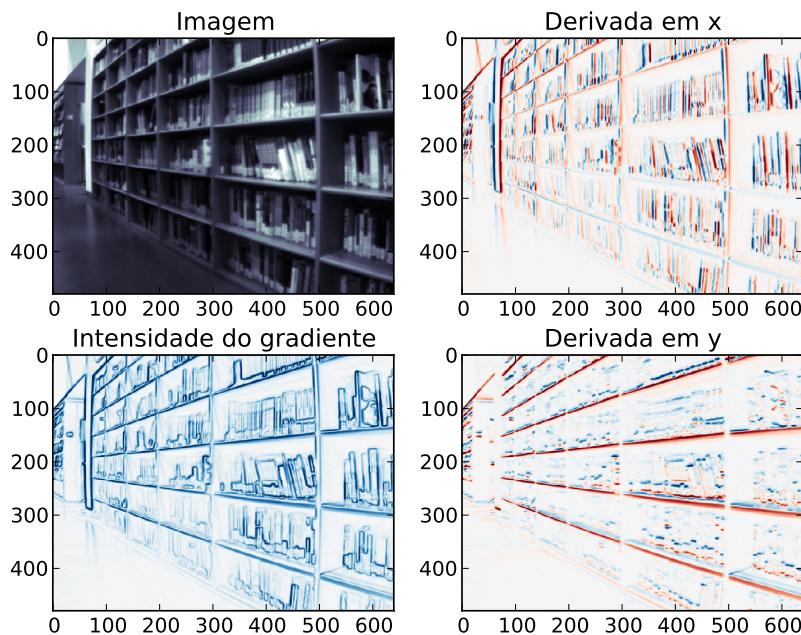


Figura 2.13: Exemplo de imagem, componentes de seu gradiente e a intensidade de gradiente.

uma função, ou o ponto de descontinuidade da função, onde há uma variação brusca entre duas regiões onde o valor é praticamente constante. É isto que se verifica se uma imagem for percorrida em uma trajetória que cruza uma borda da imagem, especialmente em uma direção ortogonal à borda. Percorrer uma imagem em uma trajetória que tangencia a borda não pode revelar nada, já que os valores de cor amostrados da imagem serão aproximadamente constantes.

A Figura 2.14 apresenta um detalhe de uma foto da mesma cena da Figura 2.4. A imagem foi varrida em uma linha vertical, mostrada superposta à imagem no gráfico à esquerda da figura. Esta linha corta uma das bordas da imagem, que é o encontro de uma das faces de um edifício com o céu ao fundo. O gráfico à direita da figura traz os valores da cor da imagem em cada pixel sobre esta trajetória vertical percorrida. As coordenadas verticais dos pixels visitados são os valores no eixo vertical deste gráfico, e o eixo horizontal indica o valor de intensidade luminosa de cada pixel para cada um dos três canais de cor desta imagem.

É possível observar o formato de degrau nos valores do gráfico da Figura 2.14. O caso desta borda representada é interessante porque o degrau é crescente no canal vermelho, crescente também no canal verde, apesar de ter uma variação de valor bem menor, mas decrescente no canal azul. A intensidade da cor azul predomina na extremidade superior do trecho amostrado, o que é coerente com o fato de que o céu é azul. Já o prédio apresenta uma cor levemente amarelada, tendo justamente o canal azul como menos intenso. O resultado desta combinação particular de cores é que se este três canais forem

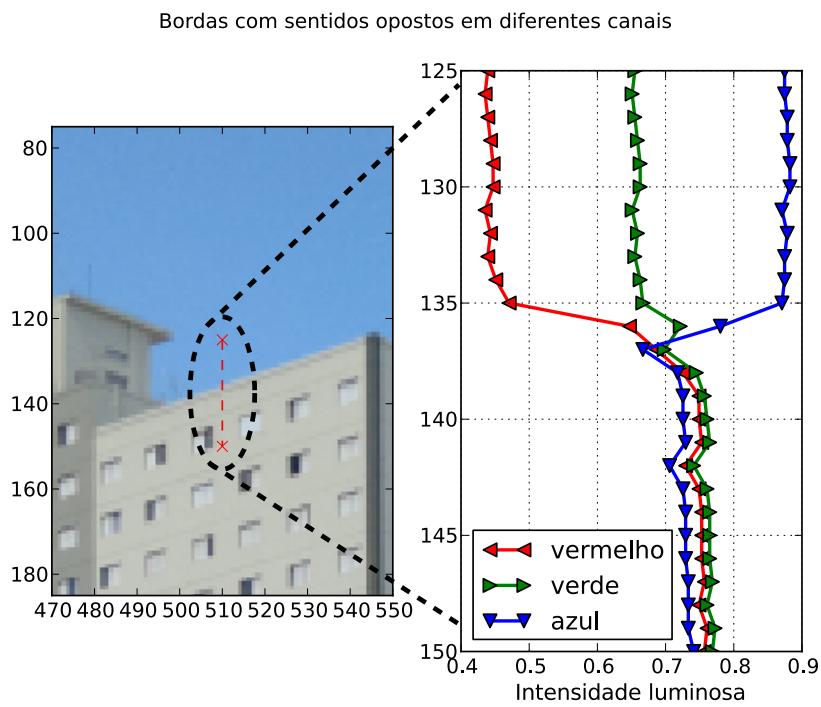


Figura 2.14: Valores de cor de uma imagem amostrados sobre uma linha que corta uma borda. As curvas em cada canal exibem a forma de um degrau cuja descontinuidade se encontra no local onde a borda corta a linha percorrida. O sentido de variação destes degraus varia entre diferentes canais neste exemplo.

simplesmente somados antes da análise para gerar uma imagem com um único canal, o que não é uma prática incomum, o resultado é um degrau com uma baixa variação de nível. Os canais vermelho ou azul sozinhos, ou a subtração deles, permitiriam uma melhor detecção desta borda. Evitar esta possível perda de informação é um desafio peculiar na análise de imagens coloridas.

Técnicas para a detecção de bordas em uma dimensão são geralmente baseadas de alguma forma no cálculo da derivada da função, e por isso a aplicação destas técnicas em imagens envolvem o cálculo do gradiente. Uma forma simples de detecção de bordas é a limiarização do valor do módulo do gradiente, mas pesquisa na área de detecção de bordas produziu diversas técnicas mais sofisticadas do que esta, dentre as quais podemos destacar as desenvolvidas por Canny (1983, 1986), Deriche (1987), Perona (1995) e Elder e Zucker (1998). Duas revisões da pesquisa na área da detecção de bordas em imagens foram compiladas por Papari e Petkov (2011) e Ziou e Tabbone (1998). Esta é uma área de pesquisa relativamente ampla, ainda mais se considerado em conjunto o problema de extrair curvas e contornos de objetos a partir das bordas detectadas. Alguns dos problemas mais complexos estudados são determinar as melhores formas de filtragem para suavização das imagens, detectar bordas difusas e na transição de regiões com texturas diferentes, ou como melhor detectar junções entre bordas, que são questões que atendem a necessidades particulares de diferentes aplicações.

A detecção de uma borda em um ponto de uma imagem depende primeiro de analisar o gradiente da imagem sobre o ponto para determinar a direção de maior variação na intensidade. Note que isto assume uma imagem com um único canal de cor. A direção do gradiente no ponto é utilizada então para determinar a direção em que os valores da imagem devem ser analisados para realizar então um processo detecção de borda unidimensional. Ou seja, uma borda é um ponto que é um máximo local no valor da intensidade do gradiente na direção em que o gradiente aponta. Este processo de detecção é atribuído principalmente a Canny (1983), e é a forma mais comum na atualidade para realizar detecção de borda.

Uma vez que se tenha extraído da imagem um conjunto de pixels rotulados como bordas, é possível então aplicar técnicas como as que serão discutidas a seguir. A técnica de extração de edgels desenvolvida nesta pesquisa também utiliza a detecção de bordas, e será discutida na Seção 4.1.

### 2.2.2 Extração de retas

Já foi dito anteriormente como um detector de bordas pode ser utilizado para realizar uma extração de curva através de seguimento de contorno. Um pixel inicial aceito pelo detector de bordas é tomado para iniciar um processo de busca, que procura por pixels vizinhos que também sejam aceitos pelo detector. Estes pixels são concatenados progressivamente até que se encontre um pixel que obedeça algum critério de terminação, como um pixel que não possua um novo vizinho a ser seguido, ou um pixel com mais de dois vizinhos, que indicaria uma junção de curvas.

O seguimento de contorno é uma técnica pouco robusta, porque depende do bom funcionamento do detector para todos os pixels que sejam tocados por cada curva que se deseja extrair. No caso em que se desejam extraír curvas de formato específico, como linhas retas, também é necessário realizar testes repetidamente a cada novo pixel concatenado para determinar quando parar a busca e instanciar o modelo geométrico.

A determinação dos parâmetros de uma reta para ajustá-la a um conjunto de pontos, ou coordenadas de pixels, é um problema muito bem conhecido em estatística. Uma vez que se saiba as coordenadas de todos os pontos envolvidos, basta executar um procedimento tal como a regressão linear. O desafio na extração de retas sobre as bordas de uma imagem é determinar quais são os pontos obtidos pelo detector de bordas que devem ser levados em conta durante a execução do procedimento que calcula os parâmetros de uma reta. Ou seja, é preciso primeiro agrupar os pontos de uma imagem de acordo com a reta da qual eles fazem parte, para então realizar a regressão.

O seguimento de contorno realiza implicitamente um agrupamento de pixels, utilizando para isto a informação de vizinhança entre os pixels. Pixels são de uma mesma curva se forem vizinhos. Esta técnica não pode ser utilizada portanto em situações em que não há essa informação de vizinhança entre os pontos. É o que ocorre quando há

uma falha na detecção de borda, e um pixel da cadeia não é detectado. Isto impede que o seguimento encontre todos os pontos que deveriam ser parte de um mesmo modelo. O mesmo problema pode ser causado por um objeto na imagem obstruindo uma parte intermediária de uma linha. Também pode ser interessante para uma aplicação considerar como parte de uma mesma linha estruturas que são verdadeiramente desconexas, porém ainda assim colineares, como no caso da análise de múltiplas janelas alinhadas em um edifício, por exemplo. Além destas situações, pode ser desejável evitar varrer todos os pixels de uma imagem. Uma técnica que possa extrair retas apenas a partir de um sub-conjunto de observações pode oferecer um menor tempo de processamento.

Há várias situações, portanto, em que pode não ser possível utilizar uma informação de vizinhança e realizar uma forma de seguimento. Nestas situações é preciso buscar uma outra forma de resolver o problema de agrupamento dos pontos simultâneo ao cálculo dos parâmetros dos modelos. As duas famílias de técnicas mais populares para resolver este problema são a chamada *transformada de Hough*, e o algoritmo RANSAC, sigla de *random sample consensus* ou *consenso de amostra aleatória*. Estes algoritmos serão descritos na Seção 2.3.

### 2.2.3 Extração de pontos de fuga

Um efeito bem-conhecido que pode ser observado na projeção pontual é a formação de um *ponto de fuga* na imagem quando existem várias linhas paralelas no ambiente. Todas as projeções de linhas que se encontram em uma mesma direção no ambiente devem cruzar um determinado ponto de fuga correspondente na imagem. A Figura 2.15 ilustra a formação de um ponto de fuga em uma imagem produzida através da projeção perspectiva, ou seja, em um câmera com o modelo *pinhole*. Um conjunto de três segmentos de reta paralelos no ambiente produz três segmentos na imagem, e a extensão destes segmentos na imagem se encontram em um único ponto, que é o ponto de fuga. A seta pontilhada na figura mostra a direção que o ponto de fuga sobre o plano da imagem se encontra em relação ao referencial da câmera.

Esta direção em que se encontra o ponto de fuga é exatamente a direção das retas no referencial da câmera. Se o plano da imagem se situar de forma paralela à direção de um destes conjuntos de retas, o ponto de fuga associado se localizará “no infinito”, e as projeções sobre a imagem serão paralelas. Se o plano for exatamente perpendicular a um grupo de retas paralelas, o ponto de fuga se localizará no ponto principal da imagem. Em projeções com distorções também se podem observar pontos de fuga, e suas localizações na imagem também correspondem às direções daquelas retas no referencial da câmera, mas como as curvas nestas projeções não serão retas a restrição criada por este ponto nos modelos extraídos é bem menos simples de compreender e explorar do que no caso da projeção pontual.

A extração de pontos de fuga permite realizar diversas tarefas importantes. A principal

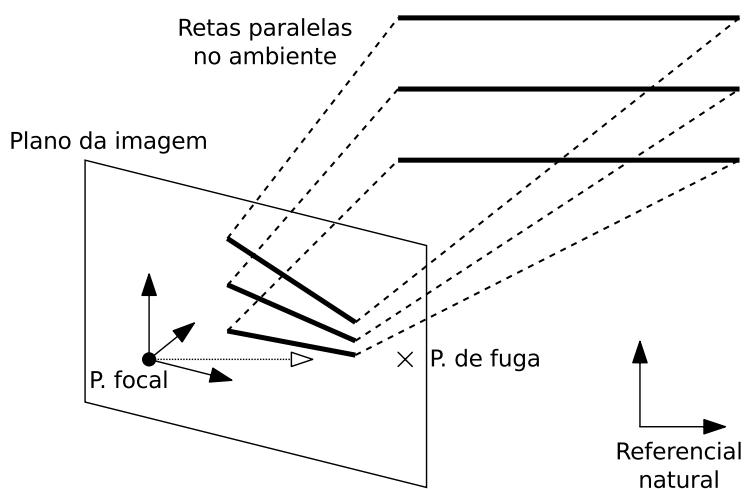


Figura 2.15: Retas paralelas produzindo um ponto de fuga através da projeção pontual.

seria justamente resolver o problema de maior interesse desta pesquisa, que é estimar a orientação da câmera com relação às direções das retas do ambiente. Em um ambiente atrópico assume-se que sempre existirão três pontos de fuga, e as direções destes pontos de fuga no referencial da câmera podem ser utilizadas para determinar a orientação da câmera. É possível, por exemplo, compor uma matriz de rotação a partir dos vetores destas três direções que faz a transformação entre o referencial de câmera e o referencial natural do ambiente.

A orientação da câmera constitui parte dos seus parâmetros extrínsecos. O restante seriam parâmetros relacionados à posição da câmera, que não podem ser obtidos apenas a partir da análise de direções de retas no ambiente sem que mais restrições sejam impostas. Os parâmetros intrínsecos que podem ser obtidos são, por exemplo, a distância focal e coordenadas do ponto principal em um modelo de *pinhole* (CAPRILE; TORRE, 1990; CIPOLLA; DRUMMOND; ROBERTSON, 1999). É importante notar que é possível extrair pontos de fuga de uma imagem sem o conhecimento de parâmetros intrínsecos ou extrínsecos, assim como é possível realizar este processo levando em conta estes parâmetros, inclusive impondo-se ou não a restrição trazida pela hipótese de ambiente antrópico.

O *Corisco* encontra as direções mutuamente ortogonais dos pontos de fuga de uma imagem de um ambiente antrópico explorando as relações geométricas entre as retas do ambiente e as suas projeções na imagem. Porém este método não requer a extração de linhas retas seguida por uma extração desinformada de pontos de fuga. No *Corisco* a hipótese de ambiente antrópico é explorada a todo o momento, e os parâmetros extrínsecos são encontrados sem que seja necessário associar as observações a linhas retas, mas as observações são atribuídas a cada ponto de fuga de uma forma similar a como funcionam outros métodos de extração de pontos de fuga. O *Corisco* também permite realizar uma inferência dos parâmetros intrínsecos, assim como é feito por outros métodos que são

baseados na extração de pontos de fuga.

Encontrar a posição do ponto de fuga de um conjunto de retas constitui um problema muito similar ao de ajustar uma reta a um conjunto de pontos. Em primeiro lugar é necessário determinar quais são as retas que efetivamente fazem parte do grupo. Uma vez que estas associações tenham sido determinadas, os parâmetros do ponto de fuga podem ser encontrados por algum método de regressão. O conhecimento aproximado dos parâmetros do ponto pode ser utilizado para auxiliar na determinação de quais retas devem ser consideradas. A melhor maneira para se resolver um problema de determinar a posição de um ponto de fuga, ou ainda de quais pontos de fuga existem em um grande conjunto de retas de diferentes direções no espaço, é utilizar algum algoritmo como os explicados na Seção 2.3 a seguir.

### 2.3 Estimação simultânea de parâmetros e associações de observações

Esta seção discute algumas técnicas para resolver problemas em que se precisa estimar parâmetros de modelos ao mesmo tempo em que se determina quais são exatamente os pontos dos dados de entrada que devem ser levados em consideração. A Seção 2.3.1 discute o algoritmo RANSAC. A Seção 2.3.2 discute a transformada de Hough, e a Seção 2.3.3 discute algumas outras alternativas.

#### 2.3.1 RANSAC

O algoritmo RANSAC foi desenvolvido nos anos 1980, com o principal objetivo de fornecer uma maneira para identificar e remover *outliers* de conjuntos de dados durante estimações de parâmetros. *Outliers* são pontos que contaminam a amostra, produzidos por fenômenos diferentes do que se pretende estudar, e que devem ser ignorados para permitir uma análise que considera apenas os pontos relevantes. Na língua portuguesa estes pontos são conhecidos pela expressão “ponto fora da curva”. Diversas variações do RANSAC já foram propostas desde a sua introdução, buscando melhorar a sua eficiência, precisão ou robustez (CHOI; KIM; YU, 2009).

O RANSAC funciona produzindo hipóteses de modelos a partir de sub-conjuntos de amostras. Todo modelo geométrico possui uma quantidade mínima de pontos para os quais é possível ajustar o modelo de forma ideal, fazendo o ponto localizar-se perfeitamente sobre o modelo. Por exemplo, dois pontos definem exatamente uma reta, desde que não sejam coincidentes. Três pontos não colineares definem um círculo. No caso de um conjunto de pontos que são medições de um mesmo vetor, em que se deseja descobrir a média, uma única amostra constitui este conjunto mínimo de observações para definir um modelo.

Se um conjunto de observações de um mesmo modelo possuísse valores ideais, sem ruídos, todos estes modelos produzidos a partir de sub-conjuntos mínimos teriam os mesmos parâmetros, e bastaria produzir um destes modelos para encontrar a solução de um

problema. Mas os ruídos adicionados às variáveis, bem como a contaminação do conjunto de dados por *outliers*, faz com que cada sub-conjunto produza uma hipótese diferente para os parâmetros. A variação nestes valores entre várias hipóteses produzidas depende da intensidade do ruído presente nos valores, e da geometria do problema. Para níveis modestos de ruído, haverá pouca variação. E se um dos pontos utilizados na definição do modelo for um *outlier*, é de se esperar que este modelo seja muito pouco coerente com o restante dos dados.

O RANSAC funciona portanto produzindo estas hipóteses de modelo a partir de sub-conjuntos de amostras selecionadas aleatoriamente. Cada hipótese é testada com um procedimento específico, e então a melhor hipótese de um conjunto de tamanho pré-determinado é selecionada como a melhor estimativa dos parâmetros do modelo. O procedimento utilizado tradicionalmente com o RANSAC para avaliar a qualidade de um modelo é apenas contar o número de pontos cuja distância até o modelo é inferior do que um certo limiar. Os pontos além deste limiar são então considerados *outliers*, e os mais próximos são *inliers*. O objetivo do processo é encontrar uma hipótese de modelo que maximiza o número de *inliers*. O RANSAC geralmente é implementado por um algoritmo da seguinte forma:

- Coletam-se as observações, e então o número de iterações a serem realizadas é determinado baseado em parâmetros de entrada.
- Para cada iteração é sorteado um conjunto mínimo de observações, e uma hipótese dos parâmetros do modelo é calculada a partir delas.
- A hipótese é testada de acordo com o procedimento selecionado, por exemplo, contam-se o número de observações a uma distância mínima deste modelo hipotético.
- Se o valor calculado pelo procedimento de teste for o melhor observado até o momento, armazenam-se o valor e os parâmetros atuais.
- Ao final das iterações, a melhor hipótese observada é retornada como a estimativa final da solução.

O desempenho deste algoritmo depende necessariamente da quantidade de *outliers* presentes nos dados. Se não houver nenhum outlier, todas as hipóteses produzidas deverão ter uma qualidade razoável, e seriam necessárias poucas hipóteses, se é que mais do que uma, para encontrar uma hipótese satisfatória. Quando *outliers* começam a ser introduzidos eles interferem no processo causando a criação de hipóteses ruins. Todo o modelo produzido contendo um outlier produz a princípio uma hipótese que será rejeitada.

Esta interferência causada pelos outliers piora de acordo com o número de pontos contidos em um conjunto mínimo para a produção de uma hipótese. No caso de um modelo que depende de um único ponto, se a probabilidade de sortear um ponto *inlier* dentro do conjunto de dados for  $p$ , a probabilidade de sortear uma hipótese satisfatória também será  $p$ . Já no caso de uma linha reta, onde é necessário sortear dois pontos, a probabilidade de sortear um par satisfatório de pontos será reduzida para  $p^2$ . Note que por ser um valor de probabilidade,  $0 \leq p \leq 1$ .

Para garantir uma alta probabilidade de sortear ao menos uma hipótese satisfatória ao longo do processo, é possível utilizar a distribuição de Bernoulli para escolher um número de iterações  $N$  que resulta em uma probabilidade final  $k$  de não se obter sucesso. Não obter sucesso significa produzir apenas hipóteses contaminadas por *outliers*. A probabilidade de se obter sucesso em ao menos uma das  $N$  iterações é  $1 - k$ . Este número de iterações necessárias é portanto dado por

$$k = (1 - p^d)^N \quad (2.29)$$

$$N = \frac{\log(k)}{\log(1 - p^d)}, \quad (2.30)$$

onde  $d$  é o número de pontos utilizado para produzir uma hipótese. O número  $N$  cresce de acordo com  $d$ , e decresce com  $k$  e  $p$ . É preciso notar que os argumentos dos dois logaritmos são menores do que 1, e portanto resultam em valores negativos. O valor  $1 - p^d$  é a probabilidade de sortear um conjunto de  $d$  pontos que contém um ou mais *outliers*.

A Equação 2.30 na prática subestima a quantidade de iterações necessárias, porque não é verdade que todas as hipóteses produzidas a partir apenas de *inliers* são aceitáveis. No caso de um ajuste de retas a qualidade da hipótese amostrada depende da posição relativa dos pontos sobre a reta. Se os dois *inliers* estiverem afastados um do outro, próximos às duas extremidades do conjunto de dados, o modelo produzido será bom porque o ruído somado às coordenadas dos pontos influencia menos na inclinação da reta. Mas se os dois pontos forem próximos um do outro, este ruído pode causar a criação de retas hipotéticas com inclinações muito distantes, que podem acabar se afastando demais dos outros pontos.

Uma das problemas com o RANSAC é portanto a grande quantidade de iterações que podem ser necessárias para se obter sucesso em alguns casos. Modelos com muitas dimensões, com  $d \geq 3$ , são bem mais desafiadores do que nos casos  $d = 1$  e  $d = 2$ . Problemas onde existem muitos outliers, com  $p \simeq 0.5$  por exemplo, também podem requerer valores bastante altos para  $N$ . Um problema relativamente simples pode resultar em um bom desempenho com  $N \simeq 10$ . Já problemas mais complexos tal como encontrar um pequeno segmento de reta dentro de uma grande quantidade de pontos pode resultar em um  $N > 10^4$  para que se garanta uma boa probabilidade de sucesso. Estes são casos de

$p$  muito baixo, em que se poderia dizer que são os pontos procurados que “contaminam” uma grande quantidade de pontos não-estruturados.

Além desta questão do número de iterações necessárias, o RANSAC tradicional também possui uma limitação importante que é o fato de que as soluções que podem ser entregues são estritamente as hipóteses produzidas pelos sub-conjuntos de amostras sorteados. O espaço das soluções possíveis é portanto discreto, e não contínuo, o que significa que a qualidade das soluções será inherentemente limitada.

Umas das primeiras modificações importantes introduzidas ao RANSAC original foi a substituição do procedimento de teste por uma forma de medição de erro baseada em técnicas tradicionais desenvolvidas a partir da teoria de probabilidades e já bastante utilizadas para estimação de parâmetros. Estas técnicas serão discutidas a seguir na Seção 2.4. Ao invés de realizar uma simples contagem dos pontos *inliers*, mede-se o valor de uma função de erro. Esta função deve possuir a propriedade de ser constante para valores de resíduo acima de um limiar para reproduzir as características da função tradicional do RANSAC. Funções de erro com esta característica são denominadas *redescendentes*, e são estudadas dentro da teoria de M-estimação que é discutida na Seção 2.4.2. Esta variação do RANSAC é denominada MSAC.

A teoria de probabilidades pode ser aplicada para determinar uma função de erro ainda melhor, que é o que é feito no método MLESAC de Torr (2000). Além desta melhoria no procedimento de teste das hipóteses, este método demonstra uma outra modificação que pode ser feita. Após a determinação de uma hipótese inicial de boa qualidade, é possível utilizar a mesma função de erro com algum método de otimização contínua que encontra parâmetros ótimos, superando assim a limitação de que as soluções poderiam ser apenas modelos produzidos a partir de conjuntos de observações. Utilizando métodos de otimização, a busca inicial precisa apenas produzir um hipótese boa o suficiente que permita a aplicação da otimização a seguir. Esta mesma estratégia foi empregada no Corisco. Outro exemplo de modificação do RANSAC que utiliza otimização contínua após a busca aleatória é o método apresentado por Chum, Matas e Obdrzalek (2004).

O uso de funções de erro baseadas em teoria de probabilidades e a aplicação de técnicas de otimização contínua beneficiam a precisão e velocidade do processo de estimação. Outra modificação que pode ser realizada para acelerar a estimação é utilizar as melhores estimativas para guiar o sorteio das amostras, dando uma maior probabilidade de ser selecionada para observações que mais se aproximam da estimativa atual da solução (TORDOFF; MURRAY, 2005). Outras formas de modificar o sorteio das observações para tentar aumentar a probabilidade de produzir boas hipóteses são aplicar heurísticas determinadas *a priori* tal como limitar a distância entre observações sorteadas, ou ainda utilizar a técnica de algoritmos genéticos (CHOI; KIM; YU, 2009). O número de iterações realizadas também pode ser calculado de forma interativa, o que irá resultar em terminações mais rápidas já que limites selecionados previamente deverão ser necessariamente

mais pessimistas do que qualquer cálculo realizado ao longo do processo utilizando informações coletadas em cada nova iteração.

### 2.3.2 *Transformada de Hough*

Uma outra técnica para atacar o problema da estimativa de parâmetros com seleção de dados é a transformada de Hough, desenvolvida a partir dos anos 1960 com foco inicial justamente no problema de extrair retas de imagens (HART, 2009). Com o passar do tempo diversos pesquisadores buscaram modificar a técnica tentando torná-la mais precisa, robusta, e principalmente tentando tornar o método mais eficiente, já que em sua versão original o método pode ser bastante custoso tanto em complexidade quanto no uso de memória. Alguns exemplos de aplicações e modificações sugeridas podem ser encontrados no artigo de revisão de Illingworth e Kittler (1988).

A extração de retas de uma imagem pela forma mais tradicional da transformada de Hough ocorre da seguinte forma:

- Primeiro realiza-se uma detecção de bordas para produzir um número de pontos sobre as retas da imagem.
- Define-se então um espaço de parâmetros sobre o plano Cartesiano. Cada ponto neste espaço corresponde a uma linha reta na imagem. Sobre este espaço de parâmetros são definidos pequenas células acumuladoras, cada uma ligada a uma região deste espaço.
- Para cada ponto detectado na imagem é possível definir uma curva sobre este espaço de parâmetros. Esta curva passa por todas as combinações de parâmetros possíveis das retas que cruzam o ponto, ou seja, todas as retas que poderiam existir na imagem que contém aquele ponto. Estas retas podem ser facilmente obtidas através de uma reta que cruze o ponto, rotacionando-a ao redor dele. Realizar esta rotação significa percorrer a curva definida no espaço de parâmetros.
- O algoritmo varre toda a lista de pontos de entrada, e para cada um deles calcula-se a curva correspondente no espaço de parâmetros, e então as células cortadas por esta curva são incrementadas.
- Ao final do processo procura-se quais são as células com valores minimamente grandes, e que sejam máximos locais. Os parâmetros destas células representam os modelos extraídos.

A principal diferença que deve ser ressaltada entre a transformada de Hough e o RANSAC está no conjunto de soluções possíveis. No caso do RANSAC as hipóteses são obtidas dos conjuntos de observações. Na transformada de Hough as soluções possíveis são os

parâmetros de cada célula. A resolução com que as células são criadas cria um comprometimento entre o desempenho do algoritmo e a precisão da solução. A transformada de Hough também permite facilmente extrair um conjunto de retas ao mesmo tempo. A transformada de Hough tende a apresentar um desempenho muito ruim quando o número de parâmetros do modelo cresce para 3 ou mais, devido ao crescimento exponencial do número de células.

Assim como ocorreu com o RANSAC, métodos de estimação de parâmetros baseados na transformada de Hough podem se beneficiar da aplicação da teoria de probabilidades (STEPHENS, 1991). O valor calculado em cada célula pode ser interpretado como a saída de uma função de erro definida a partir da técnica de M-estimação, e o uso de funções de erro criadas a partir desta técnica beneficia a precisão da estimação. Uma forma de tornar o processo mais rápido é realizando uma sub-amostragem aleatória dos dados, além de determinar de forma adaptativa o critério de parada (MATAS; GALAMBOS; KITTNER, 2000). Também é possível utilizar a informação do gradiente da imagem sobre cada ponto para restringir as buscas, o que não é algo tradicionalmente realizado no RANSAC.

Na versão tradicional da transformada de Hough, apesar de haver acumuladores definidos sobre todo o espaço de parâmetros, não há uma varredura de todo este espaço, o que constituiria uma forma bastante ineficiente de extração de linhas. Ao iterar sobre a lista dos pontos de entrada, esta técnica explora a análise dos próprios dados para restringir o espaço de busca, assim como também é feito com o RANSAC. A diferença é que no RANSAC utiliza-se um conjunto mínimo de observações para gerar uma única hipótese que é testada. Na transformada de Hough seleciona-se uma observação que dá origem a um conjunto de hipóteses a serem consideradas. Assim hipóteses que não se aproximam de ao menos um único ponto jamais são consideradas.

Tanto o RANSAC quanto a transformada de Hough utilizam de alguma forma uma função de erro redescendente, que pode ser interpretada como uma aplicação de M-estimação. A principal diferença entre as duas técnicas é a forma como soluções hipotéticas a serem testadas são produzidas. No RANSAC ocorre uma busca aleatória guiada pelos próprios dados. Na transformada de Hough a busca é restrita por amostras individuais, e o espaço de parâmetros é amostrado permitindo alcançar qualquer nível de precisão desejado. Isto evita uma potencial ineficiência do RANSAC devido à amostragem irregular do espaço de parâmetros.

### 2.3.3 Outras técnicas

O algoritmo RANSAC e a transformada de Hough são as duas técnicas mais tradicionalmente utilizadas para a extração de retas. A transformada de Hough também é usualmente aplicada para a extração de outras formas geométricas como círculos. O RANSAC também é empregado com frequência em problemas de multi-visão, por exemplo, em que pontos correspondentes devem ser encontrados entre múltiplas imagens.

Dois bons exemplos de técnicas mais recentes para este tipo de problema são os algoritmos RUDR (OLSON, 2001) e J-linkage (TOLDO; FUSIELLO, 2008). O RUDR funciona a princípio com uma busca aleatória similar à do RANSAC, porém são sorteados conjuntos de pontos com um ponto a menos do que o necessário para especificar um modelo completo. Este conjunto de pontos selecionados define um conjunto no espaço de parâmetros onde um modelo ótimo pode ser procurado. Ou seja, define-se um modelo com um parâmetros livre a ser ajustado. Para determinar este último parâmetro, utiliza-se uma transformada de Hough em uma dimensão. Desta forma o RUDR consegue unir a maior economia de memória e eficiência do RANSAC comparado à transformada de Hough tradicional, mas alcançando uma melhor precisão e determinismo pelo uso da transformada de Hough no final.

O J-linkage funciona a partir do princípio de que pontos que fazem parte de um mesmo modelo tendem a apresentar resíduos similares quando testados por um modelo qualquer. O algoritmo inicia com uma produção de vários modelos hipotéticos, como ocorre no RANSAC. Os pontos são então agrupados de acordo com a sua semelhança. Os vetores de características utilizados para comparar os pontos são criados testando se cada ponto é um *inlier* ou não de cada modelo. Pontos que são aproximadamente *inliers* e *outliers* dos mesmos modelos hipotéticos são portanto associados como sendo pontos de um mesmo modelo, e os parâmetros deste modelo são determinados posteriormente, após o agrupamento das observações.

Outra abordagem que pode ser utilizada para estes problemas de estimação de parâmetros robusta é o algoritmo EM, que é discutido na Seção 2.4.1. O EM é uma técnica de otimização iterativa, fortemente baseado na teoria de probabilidades, que permite lidar com a questão da ausência de informações para a estimação de parâmetros em vários tipos de problemas diferentes. No caso da extração de retas esta ausência de informação se refere ao desconhecimento de quais pontos pertencem a um mesmo modelo. Ao longo das iterações de uma aplicação de EM são determinados tanto os parâmetros dos modelos quanto as classificações de cada ponto de acordo com o modelo de que faz parte.

O EM necessita de uma estimativa inicial dos parâmetros de um conjunto de modelos para funcionar, porém sempre considera todo o espaço contínuo de parâmetros. Estas duas características tornam o EM fundamentalmente diferente de técnicas como o RANSAC ou a Transformada de Hough. Assim a técnica EM é aplicada em alguns métodos de forma complementar a algoritmos como o RANSAC, realizando uma estimação precisa em um estágio final dos algoritmos, porém ainda levando em consideração a possibilidade de alterar as classes dos pontos. Uma alternativa menos atraente do ponto de vista da precisão ou robustez seria simplesmente assumir que uma classificação baseada apenas na saída de um algoritmo como RANSAC seria perfeita, e realizar apenas a estimação de parâmetros dos modelos sem se preocupar mais com as classificações.

## 2.4 Estimação de parâmetros por otimização contínua

Esta seção traz uma breve introdução a alguns conceitos básicos de estimação de parâmetros baseada na Teoria de Probabilidades. Será discutido inicialmente o princípio da maximização de verossimilhança, que é utilizado para criar funções de erro que podem ser utilizadas para a estimação de parâmetros a partir de um conjunto de dados. Uma função deste tipo pode ser tanto utilizada em processos como os apresentados anteriormente, quanto em otimizações contínuas. A Seção 2.4.1 aborda a técnica de estimação EM, que possui algumas vantagens sobre formas mais simples de maximização de verossimilhança. O EM permite realizar uma classificação de dados, o que implica na possibilidade de se eliminar *outliers*. A Seção 2.4.2 discute a M-estimação, que é uma técnica desenvolvida dentro da teoria de Estatística Robusta que generaliza a técnica de maximização de verossimilhança, porém com atenção ao fato de que os modelos das funções de densidade de probabilidade dos problemas podem diferir do esperado, ou podem simplesmente não ser conhecidos. A M-estimação permite em muitos casos criar processos de estimação com propriedades muito semelhantes às oferecidas pelo técnica EM.

A aplicação da técnica de maximização de verossimilhança a algum problema inicia com a definição de um modelo probabilístico para a produção dos dados que são observados em função dos parâmetros que se quer determinar. Este é o chamado *modelo gerativo*. Por exemplo, suponha um problema onde um conjunto de valores reais são amostrados de uma distribuição Gaussiana. O modelo é dado pela função de densidade de probabilidade (PDF) condicional

$$p(x|\mu) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}. \quad (2.31)$$

A variável aleatória da distribuição é  $x$ . O desvio padrão desta PDF, dado por  $\sigma$ , é conhecido, e atuaria apenas como uma constante neste caso. O valor  $\mu$  é o centro da distribuição. Este valor é a variável à qual está condicionada a distribuição, e é também o valor que desejamos estimar.

Uma função de densidade de probabilidade condicional é geralmente entendida da seguinte forma: se forem atribuídos valores específicos para as variáveis condicionais, o resultado será uma PDF comum, não-condicionada. Mas é possível também atribuir valores para as variáveis aleatórias, e a expressão resultante é uma função das varáveis condicionais. Neste caso a função é conhecida como *função de verossimilhança*. Portanto a Equação 2.31 representa tanto a função de densidade de probabilidade condicional de  $x$  condicionada a  $\mu$  quanto a função de verossimilhança de  $\mu$  dado  $x$ . A notação  $L(\mu; x)$  é utilizada algumas vezes para enfatizar que a função estaria sendo utilizada desta forma.

Se forem amostrados  $N$  valores da distribuição acima, teremos um vetor  $\mathbf{x}$  composto pelos valores  $x_n$  sorteados. Este vetor constitui uma amostra de uma Gaussiana multi-variada, cuja PDF será dada por um produto de cada PDF individual, assumindo que os

processos que produzem cada variável são independentes, resultando em

$$p(\mathbf{x}|\mu) = \left( \frac{1}{\sigma\sqrt{2\pi}} \right)^N \prod_n e^{-(x_n - \mu)^2/2\sigma^2}. \quad (2.32)$$

O método da estimativa por maximização da verossimilhança parte do princípio de que uma boa estimativa para o valor de  $\mu$  utilizado para produzir os  $x_n$  de uma amostragem pode ser obtida pelo valor que maximiza a sua verossimilhança. Neste exemplo a expressão da verossimilhança é dada pela própria Equação 2.32, onde cada variável  $x_n$  deve ser substituída pelo valor observado. Para facilitar o processo de encontrar este valor de  $\mu$  que maximiza a expressão, é comum trabalhar na prática com o valor inverso aditivo do logaritmo desta verossimilhança, que é então minimizado. No exemplo da distribuição acima teríamos a expressão

$$-\log(L(\mu; \mathbf{x})) = N \log(\sigma\sqrt{2\pi}) + \sum_n (x_n - \mu)^2/2\sigma^2. \quad (2.33)$$

O valor de  $\mu$  seria então encontrado pela minimização da uma certa função  $F(\mu)$ , produzindo a estimativa

$$\tilde{\mu} = \operatorname{argmin}_{\mu} F(\mu). \quad (2.34)$$

Esta função deve ser utilizada então como função objetivo para um processo de otimização, e pode ser qualquer função cujos mínimos se encontrem nos mesmos pontos de máxima da verossimilhança original. O uso do logaritmo é uma modificação que respeita esta restrição, e outras modificações que podem ser feitas na expressão da Equação 2.33 envolvem a remoção da parcela e dos coeficientes constantes. O resultado é a simples expressão

$$F(\mu) = \sum_n (x_n - \mu)^2. \quad (2.35)$$

A Equação 2.35 demonstra que o resultado da aplicação do método de maximização de verossimilhança ao problema da estimativa de  $\mu$  a partir de um conjunto de pontos com distribuição Gaussiana é uma minimização de uma soma dos quadrados das diferenças entre cada ponto e o parâmetro sendo estimado. A expressão dentro do somatório é geralmente interpretada como sendo um *erro quadrático*, e este processo de estimativa é uma minimização de uma soma de erros quadráticos, ou um problema de *quadrados mínimos*.

Este processo pode ser generalizado a partir desta expressão, dando origem a outros métodos de estimativa baseados em minimizações de erros. A Equação 2.35 deve primeiro ser reescrita como

$$F(\mu) = \sum_n \rho(x_n - \mu). \quad (2.36)$$

A diferença  $x_n - \mu$  que serve como argumento para a função  $\rho$  pode ser chamada de *erro*, mas os termos *resíduo* ou *desvio* podem ser empregados para diferenciar estes valores dos valores calculados pela função  $\rho$ , que são também usualmente denominados *erros*. A função  $\rho$  é denominada *função de erro*, ou também *função de perda*.

Assim como a distribuição Gaussiana leva à minimização de erros quadráticos, com  $\rho(x) = x^2$ , outras distribuições produzem outras funções de erro. A distribuição Laplaciana e a sua função de erro correspondente são dadas por

$$p(x|\mu) = \frac{1}{2b} e^{-|x-\mu|/b} \quad (2.37)$$

$$\rho(x) = |x - \mu|. \quad (2.38)$$

Ou seja, esta distribuição está ligada à minimização de erros absolutos através do método da maximização de verossimilhança, assim como a distribuição Gaussiana está ligada à minimização de erros quadráticos.

É interessante observar que no caso dos erros quadráticos é fácil encontrar a solução do problema. O mínimo daquela função é dado pelo ponto onde a derivada em relação a  $\mu$  é nula, ou seja

$$\frac{\partial F}{\partial \mu} = \sum_n -2(x_n - \mu) = 0 \quad (2.39)$$

$$\mu = \frac{1}{N} \sum_n x_n. \quad (2.40)$$

Assim demonstra-se a relação bem-conhecida entre  $\mu$  e a média de um conjunto de amostras. O caso do erro absoluto linear não é tão simples, a derivação resulta em

$$\frac{\partial F}{\partial \mu} = \sum_n -\frac{x_n - \mu}{|x_n - \mu|}. \quad (2.41)$$

Esta função não é inversível, e portanto não é possível encontrar uma fórmula fechada para o valor de  $\mu$  neste caso. Mas ainda é possível, no entanto, aplicar métodos de otimização não-linear iterativa para encontrá-lo.

O princípio de maximização de verossimilhança é a base de muitos métodos de estimação mais sofisticados. Aplicação desta técnica significa assumir uma certa distribuição de probabilidades para as observações para então encontrar uma expressão para a estimação de parâmetros. As duas principais limitações desta técnica são o fato de que em algumas situações existem variáveis condicionais desconhecidas que também precisam ser levadas em consideração, e também existem situações em que não é possível saber com exatidão qual é a PDF que deve ser utilizada, além de que como demonstrado anteriormente, há diferentes graus de complexidade associados às diferentes expressões

resultantes a serem minimizadas. As seções a seguir discutem técnicas mais sofisticadas de estimação para lidar com estes problemas.

#### 2.4.1 Esperança da verossimilhança

Na Equação 2.31 o único parâmetro condicional é  $\mu$ . Em casos de problemas como o da estimativa de orientação, estudado nesta pesquisa, o problema pode ser visto como um caso de uma função de densidade de probabilidades que é uma mistura de funções, baseada em várias distribuições daquele tipo. Cada função da mistura modela uma possível classe a que pode pertencer cada observação, e a classe se torna um novo parâmetro condicional do modelo. No caso de uma variável aleatória unidimensional e  $K$  classes, com parâmetros correspondentes  $\mu_k$  formando um vetor  $\mu_{1\dots K}$ , a PDF de cada classe e a PDF total são dadas pelas expressões

$$p(x|\mu_{1\dots K}, c) = p(x|\mu_k) \quad \text{se} \quad c = k \quad (2.42)$$

$$p(x|\mu_{1\dots K}) = \sum_k p(x|\mu_k, c = k)p(c = k) \quad (2.43)$$

onde  $\mu_k$  é o parâmetro que modela cada PDF, e a variável  $c$  é a classe a que pertence a amostra  $x$ . No processo de sorteio de uma amostra escolhe-se primeiro uma classe utilizando a distribuição  $p(c)$ . Em certos problemas as probabilidades de cada classe podem ser iguais, o que implicaria em uma PDF uniforme, mas este nem sempre é o caso. Uma vez que se definiu a classe, a PDF  $p(x|\mu_k, c = k)$  é utilizada para sortear  $x$ .

A Equação 2.43 pode ser utilizada para a estimativa de parâmetros de forma similar às Equações 2.31 e 2.32, produzindo a fórmula

$$p(\mathbf{x}|\mu_{1\dots K}) = \prod_n \sum_k p(x_n|\mu_k, c_n = k)p(c_n = k). \quad (2.44)$$

A diferença desta fórmula para o caso da Equação 2.32 é que cada soma ponderada das funções de verossimilhança constitui agora um *valor esperado* da verossimilhança sobre a variável  $c_n$ , que especifica a classe de cada amostra  $x_n$ . Este método de estimativa é chamado *estimação de probabilidade máxima a posteriori*, ou MAP, e pode ser interpretada como uma forma de maximização de verossimilhança regularizada, onde os valores da probabilidades *a priori*, dadas por  $p(c)$  neste exemplo, são utilizados para ponderar a função de verossimilhança. Desta forma a função conjunta é marginalizada sobre as classes, produzindo apenas a distribuição sobre os valores observados.

A necessidade de utilizar um estimador MAP surge pelo desconhecimento das classes das amostras  $x_n$ , ou seja, dos valores das variáveis  $c_n$ . Uma mesma PDF sobre cada  $c_n$  é então utilizada para substituir esta informação ausente. Uma outra alternativa mais interessante poderia ser efetivamente estimar os valores de  $p(c)$ , ou até mesmo o valor de cada  $c_n$ . Isto se torna bastante complicado, no entanto, pelo fato de que estas variáveis

são discretas, o que dificulta o processo de otimização. Seria desejável portanto alguma maneira de transformar este problema em um problema de otimização contínua, para que se possam aplicar técnicas eficientes de otimização.

Uma forma de lidar com esta estimação de  $c_n$  é utilizar a técnica chamada *Expectation-Maximization*, ou *Esperança-Maximização* em português, ou apenas EM (MCLACHLAN; KRISHNAN, 2007). O resultado neste caso é um processo de otimização em que cada iteração é constituída por dois passos. No chamado *passo E* realiza-se uma estimação dos valores de  $p(c_n)$  para cada amostra. Ou seja, existirão funções de distribuição de probabilidade de classe estimadas para cada amostra. No *passo M* a seguir é realizada uma maximização da esperança da verossimilhança, exatamente como no estimador MAP, porém agora com valores de  $p(c_n)$  estimados. Durante este passo é possível tanto utilizar uma distribuição de  $c$  que varia para cada amostra, quanto uma única distribuição média calculada a partir de todos os dados. Os valores de  $p(c_n)$  são então atualizados na iteração seguinte, até que haja uma convergência.

A forma como se calculam os valores de  $p(c_n)$  no passo *E* é através da função de verossimilhança  $p(x_n|\mu_{1\dots K}, c_n)$ , aplicando-se a regra de Bayes, que é enunciada através da equação

$$p(b|a)p(a) = p(a|b)p(b). \quad (2.45)$$

A fórmula resultante permite atualizar os valores  $c_n^t$  na iteração  $t$  da seguinte forma

$$p(c_n^{t+1}|x_n, \mu_{1\dots K}) = \frac{p(x_n|\mu_{1\dots K}, c_n^t)p(c_n^t)}{p(x_n)}. \quad (2.46)$$

O cálculo costuma ser conduzido explorando o fato de que a soma das probabilidades deve ser unitária

$$\sum_k p(c_n^{t+1} = k) = 1, \quad (2.47)$$

e assim o cálculo é facilmente realizado através da ponderação dos valores atuais pelos valores da verossimilhança de  $c_n^t$ , pois como as probabilidades são diretamente proporcionais a estes produtos, ou

$$p(c_n^{t+1}|x_n, \mu_{1\dots K}) \propto p(x_n|\mu_{1\dots K}, c_n^t)p(c_n^t), \quad (2.48)$$

basta uma normalização destes valores para obter a nova estimativa. É preciso ressaltar que os valores de  $\mu_{1\dots K}$  são considerados constantes e conhecidos neste passo.

Um fato importante que pode ser deduzido da Equação 2.46 é que se os parâmetros não se alterarem mais, os valores de  $c_n$  vão eventualmente convergir de forma que uma única classe adquira probabilidade igual a 1, enquanto as outras vão para 0 — um grupo de classes também pode adquirir uma distribuição uniforme entre elas em situações es-

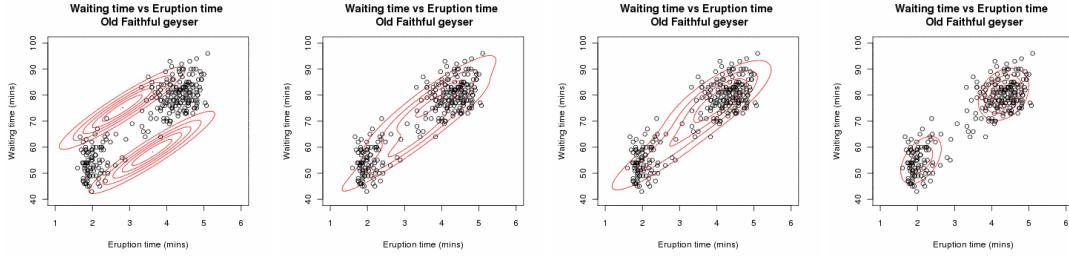


Figura 2.16: Evolução da estimativa dos parâmetros de uma mistura de duas gaussianas em duas dimensões (WIKIPEDIA, 2012).

peciais. Desta forma o algoritmo acaba atribuindo um rótulo de classe para cada dado de entrada. Nestas condições o cálculo dos parâmetros se reduz a uma simples análise separada dos parâmetros de cada modelo, considerando apenas os dados que foram atribuídos a cada classe individualmente. No caso em que se estima uma função  $p(c)$  geral esta convergência para 0 ou 1 não ocorre necessariamente.

Outra consequência do processo de normalização no passo  $E$  é que esta classificação dos pontos leva em consideração todas as classes simultaneamente. Conforme a probabilidade de uma determinada amostra pertencer a uma determinada classe cresce, a probabilidade dela pertencer às outras classes vai diminuindo. Assim não só a amostra começará a afetar mais a estimativa dos parâmetros desta nova classe de probabilidade crescente, como afetará menos também na estimativa das demais. Assim quando uma classificação equivocada é eventualmente corrigida com a atribuição de outra classe para a observação em questão, o impacto negativo desta observação na estimativa dos parâmetros da classe antiga será eliminado.

Nas equações anteriores os parâmetros  $\mu_{1\dots K}$  pretendiam significar a princípio apenas o centro de PDFs individuais em uma mistura de funções, mas é possível realizar o mesmo processo controlando-se qualquer outro tipo de parâmetro. No caso de uma mistura de Gaussianas multivariadas é possível estimar não apenas todas as médias, mas também as matrizes de covariâncias de cada Gaussiana individual. A Figura 2.16 ilustra a estimativa de uma mistura de Gaussianas utilizando um algoritmo como o descrito anteriormente.

Uma variação importante da técnica EM é o chamado *EM generalizado*, ou GEM. Neste caso o passo  $M$  não é necessariamente uma otimização completa, que efetivamente encontra um extremo, mas apenas uma atualização que eleva, ou melhor, não reduz o valor da esperança da verossimilhança naquela iteração. Desta forma é possível criar processos onde um passo  $E$  é seguido de um passo  $M$  constituído por um único passo de algum algoritmo de otimização não-linear iterativo, e os dois juntos constituem uma espécie de passo de otimização modificado.

As técnicas de estimativa MAP e EM já foram aplicadas anteriormente no problema da estimativa de orientação de câmera a partir de edgels abordado nesta pesquisa, permitindo lidar com sucesso com a questão da classificação das observações entre as diferentes

direções possíveis das retas no ambiente que produziram cara edgel. No método desenvolvido aqui foi empregada a técnica de M-estimação, que será descrita a seguir, que possui muitas semelhanças com estas técnicas já empregadas. Esta modificação permite fazer alterações no processo de otimização visando principalmente torná-lo mais rápido, mas também facilitar a proposição da técnica em si, além de permitir a exploração de abordagens diferentes para o problema.

#### 2.4.2 M-estimação

A técnica EM engloba um grande número de métodos de estimação de parâmetros, e vários destes foram desenvolvidos em outras áreas sem uma conexão inicial conhecida com esta técnica. Um caso importante é o algoritmo de quadrados mínimos reponderado iterativamente (IRLS, do inglês *iteratively reweighted least squares*). Este algoritmo permite estimar parâmetros realizando uma minimização de diferentes funções de erro  $\rho$ , e não apenas quadrática, mas é baseado em sucessivas minimizações de erros quadráticos ponderados. Em cada iteração superior do algoritmo os pesos para esta ponderação são recalculados a partir dos parâmetros atuais, e novos parâmetros são encontrados pela solução de um novo problema de quadrados mínimos ponderados. Os novos parâmetros produzem então novos pesos, e assim continua o processo até a convergência (MARONNA; MARTIN; YOHAI, 2006, sec. 4.5.2).

Esta sucessão de estimações intercalada por atualizações no valor dos pesos do IRLS pode ser facilmente interpretada como uma aplicação da técnica EM apresentada anteriormente (MCLACHLAN; KRISHNAN, 2007, 1.8.3). Tendo em vista estas semelhanças, e considerando ainda que tanto o método de estimação MAP quanto o EM já foram aplicados com sucesso no problema da estimação de orientação de câmera baseada em edgels, como será mostrado nos capítulos seguintes, há que se perguntar se não seria possível abordar este problema do ponto de vista da Estatística Robusta, que é a área de estudos onde mais se foca na questão da escolha de diferentes normas de erro, cujo uso levaria por exemplo à aplicação do algoritmo IRLS.

Uma das vantagens de se utilizar as teorias da Estatística Robusta é evitar esta necessidade de se especificar com exatidão as funções de densidade de probabilidade envolvidas em um problema, como é necessário com outras técnicas. Nesta área estuda-se de que forma imprecisões no conhecimento destas funções afetam os processos de estimação. Um problema bastante comum que ilustra este tipo de situação é quando um conjunto de dados é contaminado por *outliers*. Um problema estudado em Estatística Robusta é justamente como lidar com esse tipo de contaminação, identificando estes pontos e impedindo que eles afetem a estimação de parâmetros.

Uma técnica bastante utilizada dentro da Estatística Robusta para este propósito é a chamada M-estimação (MARONNA; MARTIN; YOHAI, 2006) (ou *M-estimation* em inglês), um nome que alude à similaridade da técnica com a maximização de verossimi-

Tabela 2.1: Funções de erro utilizadas na M-estimação.

| Função de erro             | Expressão   | PDF associada |
|----------------------------|---|---------------|
| Erro absoluto              | $\rho(x) =  x $   | Laplaceana    |
| Erro quadrático            | $\rho(x) = x^2$   | Gaussiana     |
| Função de Huber            | $\rho(x) = \begin{cases} x^2 & \text{se }  x  \leq s \\ 2s x  - s^2 & \text{se }  x  > s \end{cases}$       | —             |
| Erro absoluto limitado     | $\rho(x) = \begin{cases}  x  & \text{se }  x  \leq s \\ s & \text{se }  x  > s \end{cases}$                 | —             |
| Erro quadrático limitado   | $\rho(x) = \begin{cases} x^2 & \text{se }  x  \leq s \\ s^2 & \text{se }  x  > s \end{cases}$               | —             |
| Função biquadrada de Tukey | $\rho(x) = \begin{cases} 1 - [1 - (x/s)^2]^3 & \text{se }  x  \leq s \\ 1 & \text{se }  x  > s \end{cases}$ | —             |

lhança clássica. A M-estimação é constituída justamente por uma minimização de erros de acordo com a Equação 2.36, e o que se estuda são as implicações de se selecionar diferentes funções  $\rho$ , e como selecionar os seus parâmetros para cada problema específico de estimação.

Do ponto de vista de maximização de verossimilhança, uma determinada função  $\rho$  define indiretamente uma PDF, e o processo conduzido é uma estimação de parâmetros com este modelo implícito. Do ponto de vista da Estatística Robusta a verdadeira PDF que modela os dados originais pode não ser exatamente a que produz a função  $\rho$ , mas o método estará levando em consideração fenômenos como a contaminação de dados, e outros que podem ser difíceis de modelar. Além disso pode ser possível selecionar funções que possuem propriedades vantajosas do ponto de vista computacional, de forma similar a como a distribuição Gaussiana pode ser escolhida apenas devido ao fato de que ela implica em um problema de mínimos quadrados, para o qual existem algoritmos de otimização mais convenientes que podem ser aplicados.

A Tabela 2.1 lista algumas destas funções  $\rho$  que se utilizam em processos de M-estimação, que substituem o logaritmo da verossimilhança que surge nos processos baseados em maximização de verossimilhança. A Figura 2.17 ilustra algumas destas funções. É possível observar neste gráfico algumas das características compartilhadas por todas estas funções. Em primeiro lugar, tratam-se de funções pares, com simetria espelhada, e por isso é possível mostrar apenas o lado direito de todas elas. O valor inicial de todas as funções também é 0, e elas também são todas monotonicamente crescentes para valores positivos.

É possível observar nestas funções que a quadrática é a que apresenta o crescimento mais rápido. Isto é considerado uma característica negativa, porque assim os *outliers* podem ter uma influência forte no resultado da estimação. Quase todas as funções de erro empregadas na M-estimação costumam apresentar um crescimento mais lento do que quadrático. A função de erro absoluto, por exemplo, cresce de forma linear, mas possui uma descontinuidade na sua derivada no ponto zero, o que é considerado indesejável em

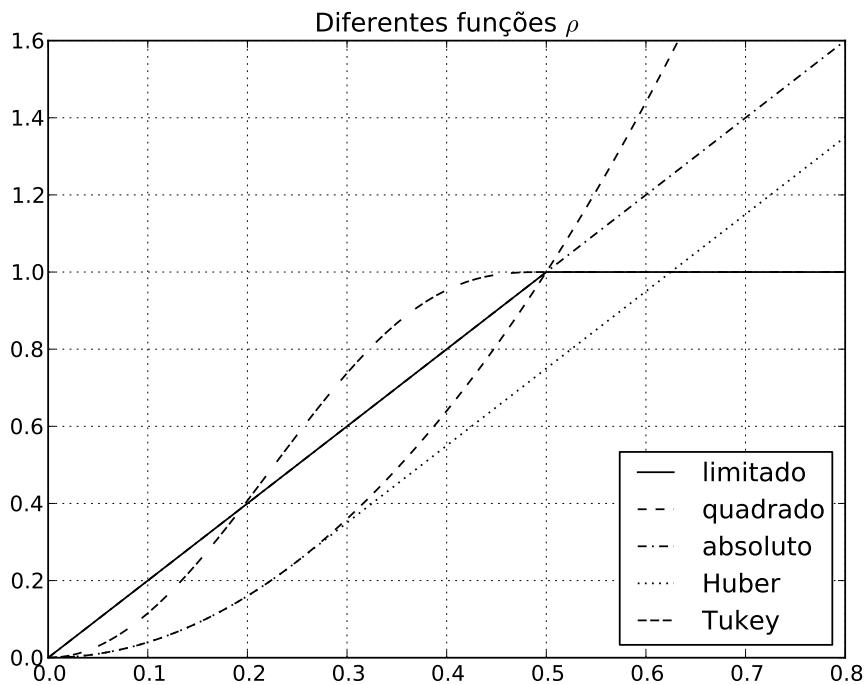


Figura 2.17: Diferentes funções  $\rho$  que podem ser utilizadas para realizar M-estimação.

algumas situações. A função de Huber busca realizar um comprometimento entre estas características. Ela é mais suave próxima da origem, onde a expressão é quadrática, mas para valores mais altos passa a crescer de forma linear.

Do ponto de vista da maximização de verossimilhança, o erro quadrático deveria ser utilizado para modelos com PDF Gaussiana, e o erro absoluto apenas para a PDF Laplaciana. Na prática, porém, a existência de *outliers* torna indesejável o uso da função quadrática no caso da Gaussiana, e a existência destes *outliers* pode ser até mais comum do que se pode imaginar a princípio. Quando se amostra um conjunto pouco numeroso de pontos de uma Gaussiana podem surgir naturalmente pontos bastante distantes do centro da distribuição, e eles podem atuar como *outliers* mesmo que sejam amostras legítimas. Assim, na prática, o uso do erro absoluto pode ser melhor do que o quadrático na estimação do centro de uma PDF Gaussiana.

A função de Huber busca oferecer as virtudes do erro absoluto sem se afastar muito do modelo original de Gaussiana. A PDF resultante da função de Huber não é nenhuma distribuição tradicional, por isso não está listada na Tabela 2.1. Mas esta PDF correspondente é uma em que o centro é dado pela função da Gaussiana, mas partir de um ponto limiar começa a decair de forma exponencial. Ou seja, é uma PDF que admite mais livremente a possibilidade de pontos mais distantes da média, já que na Gaussiana o decaimento é com o quadrado da exponencial. É uma distribuição similar à Gaussiana para a maior parte dos seus valores, próximos ao centro, mas que possui uma cauda mais espessa.

As funções quadrática limitada, absoluta limitada e a função biquadrada de Tukey compartilham uma propriedade bastante significativa. Todas estas funções se tornam constantes a partir de um determinado ponto. Estas funções são chamadas *redescendentes*, devido ao fato de suas derivadas começarem nulas para valores negativos extremos, eventualmente decrescerem, depois subirem até um valor máximo de onde depois decrescem para zero novamente, criando uma trajetória sinusoidal. Isto significa que se um determinado dado produzir um erro de intensidade que ultrapassa este limiar a partir do qual a derivada de  $\rho$  é nula, sua influência no processo de estimativa passará igualmente a ser nula. O valor do erro calculado continua dependendo desta amostra, mas variações pequenas de seus valores não afetarão o valor do erro calculado, assim como a contribuição deste erro no erro total não será mais modificada por variações pequenas dos parâmetros do modelo.

A função de Tukey é a única das apresentadas que é redescendente e possui derivada contínua, tornando-a mais suave. Esta suavidade não é um fator absolutamente necessário para utilizar alguma função, mas tende a ser benéfica para os processos de otimização de variáveis contínuas que são empregados na minimização do erro total.

É interessante observar o que ocorre se o algoritmo IRLS for empregado para minimizar um erro definido com a função biquadrada de Tukey. Como dito anteriormente, em uma estimativa simples do parâmetro de translação de uma PDF unidimensional, a solução em cada passo do IRLS é encontrada pela equação

$$\sum_n W(x_n - \mu) (x_n - \mu) = 0, \quad (2.49)$$

onde a função  $W(x)$  é a função para o cálculo dos pesos que deve ser selecionada de acordo com a função  $\rho$  que se deseja utilizar, através da fórmula

$$W(x) = \begin{cases} \rho'(x)/x & \text{se } |x| > 0 \\ \rho''(0) & \text{se } x = 0 \end{cases}. \quad (2.50)$$

Se  $\rho(x) = x^2$ ,  $W$  é simplesmente igual a 1, e a expressão se torna a solução para uma soma de quadrados dos resíduos, dada na Equação 2.40. No caso da função biquadrada de Tukey, o fato dela ser redescendente faz com que a sua função  $W$  se torne 0 para valores elevados, já que ela é proporcional à derivada de  $\rho$ . O gráfico à esquerda na Figura 2.18 mostra a função  $W$  obtida a partir da função de Tukey para  $s = 1$ .

No mesmo gráfico esquerdo da Figura 2.18 foi traçada também a curva relativa a uma função de distribuição Gaussiana, porém escalada de forma a se ajustar à função  $W$  ali presente. É possível observar que ambas funções compartilham certas características. Ambas são relativamente suaves, decaem rapidamente para zero a partir de certo ponto, e possuem uma elevação arredondada no centro.

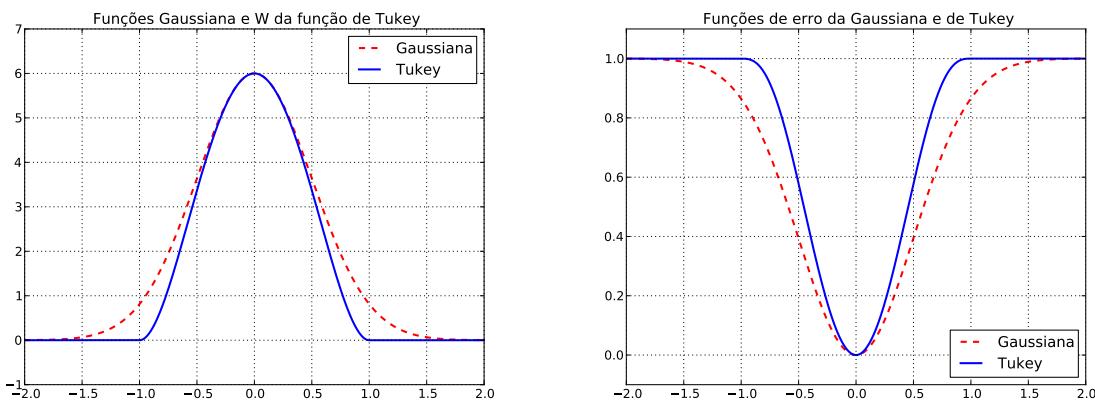


Figura 2.18: À esquerda, uma função Gaussiana não-normalizada comparada com a função de pesos correspondente à função biquadrada de Tukey com  $s = 1$ . E à direita, as funções biquadrada de Tukey com  $s = 1$  e a função de erro que resulta do uso da distribuição Gaussiana como uma função de pesos no algoritmo IRLS.

A semelhança destas curvas sugere que a aplicação do algoritmo EM com uma PDF Gaussiana em um caso como este implica em uma minimização de erros baseada em uma de função de erro mais vantajosa do que a quadrática. O gráfico à direita na Figura 2.18 mostra qual seria esta função  $\rho$  de erro cuja função  $W$  seria uma Gaussiana. A função biquadrada de Tukey também está desenhada para comparação. A diferença entre estas curvas e as da esquerda da Figura 2.18, além do fato de que as concavidades apontam em direções opostas, é bastante sutil. É preciso reforçar que a curva rotulada “Gaussiana” na figura se refere à função  $\rho$  que produz uma função  $W(x)$  Gaussiana. A curva do gráfico foi calculada a partir da integração de uma Gaussiana multiplicada por  $x$ .

O que o gráfico direito da Figura 2.18 procura mostrar é que as vantagens do uso do algoritmo EM com uma PDF Gaussiana na estimação de alguma grandeza podem não ser fruto da escolha desta função em específico, mas apenas do fato que esta função oferece as mesmas propriedades de outras funções de erro redescendentes. Seria possível portanto selecionar outra função para o trabalho, em especial funções mais usualmente empregadas na M-estimação tal como a própria função biquadrada de Tukey. O mesmo argumento vale para o uso de outras funções junto da técnica EM ou MAP, como já foi realizado no estudo da estimação de orientação de câmera. A teoria básica de maximização de verossimilhança pede o uso da fórmula exata da PDF das grandezas no cálculo das expressões. Mas se abordarmos o problema do ponto de vista da Estatística Robusta, esta preocupação é substituída pela simples exigência que se use alguma função de erro minimamente aceitável.

Apesar da M-estimação abandonar o requerimento de que a PDF exata deve ser conhecida, a escolha da função de erro a ser utilizada não pode ser feita de maneira totalmente arbitrária. Características da PDF que gerou os dados ainda devem influenciar a escolha.

No caso da aplicação de uma função tal como a de Tukey ou Huber, o parâmetro  $s$  deve ser selecionado de forma a aproximar a escala da distribuição dos erros ao redor da média. Existem técnicas em Estatística Robusta que permitem estimar este valor se ele não for conhecido. Por exemplo, há casos em que se pode realizar uma estimativa inicial com a norma absoluta, que não depende de escala, e a partir daí levanta-se uma medição da variância para que se possa então utilizar por exemplo a função de Tukey (MARONNA; MARTIN; YOHAI, 2006). A existência deste tipo de técnica torna esta abordagem bastante interessante para aplicações práticas de estimativa de parâmetros, onde pode ser desejável evitar assumir que todos os parâmetros das funções de probabilidade são conhecidos.

Também é possível estimar parâmetros tal como a variância das distribuições em um modelo de mistura utilizando a técnica EM, o que é similar a uma estimativa do fator de escala de uma função de erro na M-estimação. Realizar este tipo de estimativa poderia ser interessante no problema da estimativa de orientação de câmera, mas isto ainda não foi demonstrado por ninguém. Todos os métodos similares ao que será apresentado aqui sempre assumem o conhecimento das variâncias, e realizam uma inferência apenas de parâmetros de natureza translacional, e não de escala ou forma.

No Capítulo 3 a seguir serão descritos vários métodos de estimativa de parâmetros para problemas de Visão Computacional, incluindo problemas de estimativa de orientação de câmera como o estudado nesta pesquisa. Estes métodos aplicam técnicas como as discutidas neste capítulo. Entre estes estão incluídos os métodos já existentes propostos para o problema da estimativa de orientação baseada em edgels, baseados em MAP e EM. O Capítulo 4 discute enfim com mais detalhes as técnicas utilizadas no método proposto, o que inclui uma busca aleatória no estilo do RANSAC, e uma minimização de erros com uma função redescendente baseada na teoria de M-estimação.

### 3 Trabalhos correlatos

*We can think of the past without reference to the future.  
But we can't think of the future without reference to the past.*  
—What Becomes an End, Kylesa

Este capítulo contém uma revisão do desenvolvimento de diversas pesquisas em Visão Computacional que abordaram problemas similares ao estudado nesta tese, envolvendo a estimativa de parâmetros extrínsecos a partir de características geométricas extraídas de imagens. A estimativa de parâmetros intrínsecos e de modelos do ambiente também são abordados por alguns dos métodos que serão discutidos, apesar deste não ser o foco principal do método proposto.

Todos estes problemas de visão abordados aqui compartilham uma mesma natureza, e todos os métodos que serão apresentados a seguir compartilham também um mesmo princípio de funcionamento. Estes problemas podem ser todos enxergados como sendo diferentes versões do problema mais geral e abstrato chamado *bundle adjustment*, que será discutido a seguir. Nestes problemas sempre existe alguma função de erro a ser minimizada que depende de características geométricas que são extraídas das imagens. Estas observações também geralmente precisam ser agrupadas ou classificadas de alguma forma para o cálculo da função.

Em alguns problemas podem haver assunções que podem ser feitas acerca dos modelos de câmera ou do ambiente, que pode torná-los mais simples. Cada problema também possui um conjunto específico de parâmetros sendo buscados, e existem diferentes algoritmos que podem ser empregados para realizar a otimização em cada caso, diferindo principalmente em fatores como haver um funcionamento interativo com atualizações incrementais das estimativas, ou não. Apesar das diferenças entre os vários métodos, a estratégia de mais alto nível é sempre a mesma: alguma forma de otimização de um erro relacionado a características geométricas extraídas das imagens.

Inicialmente serão discutidos na Seção 3.1 métodos baseados em pontos. Estes são fundamentalmente diferentes do método proposto pela forma de característica geométrica utilizada e pelas restrições existentes, porém é útil conhecer esta abordagem alternativa devido aos conceitos em comum que existem entre todas as técnicas, além de ferramentas que também são empregadas em todos estes cenários tal como o algoritmo RANSAC discutido no Capítulo 2. Problemas envolvendo pontos são a epítome do *bundle adjustment*, e os outros casos devem ser vistos como extensões ou modificações deste. A Seção 3.2 apresenta diversas técnicas de visão baseadas em linhas, e portanto mais próximos ao tema desta tese. A Seção 3.3 discute enfim o problema específico da estimativa de orientação baseada em edgels, que é o problema para o qual o Corisco foi desenvolvido para

resolver.

### 3.1 Técnicas de visão baseadas em pontos

Como foi dito no Capítulo 1, a forma mais simples e básica de característica geométrica é o ponto, e pode ser útil do ponto de vista didático apresentar qualquer técnica de visão computacional considerando primeiro como proceder ao utilizar pontos em todas as análises. O desenvolvimento de técnicas baseadas em pontos é interessante tanto do ponto de vista teórico, pela natureza fundamental do ponto na geometria, quanto prático, devido à facilidade com que se pode manipular e interpretar um ponto dentro de um problema. Muitas novas técnicas de visão de alto nível são inicialmente desenvolvidas utilizando pontos devido a isto.

Um bom exemplo de uma pesquisa pioneira que utilizou pontos como entidade fundamental é o trabalho realizado por Moravec (1980), onde pela primeira vez demonstrou-se um veículo capaz de se locomover evitando obstáculos utilizando para isto apenas uma câmera, sem sensores laser ou de qualquer outro tipo. No sistema proposto naquela pesquisa é empregado um extrator de pontos de interesse para analisar imagens de um arranjo de câmeras, e então criar um modelo do ambiente. Este modelo contém as localizações no espaço destes pontos extraídos, e eles são então considerados como sendo obstáculos a serem evitados pelo robô móvel sobre o qual está montado o arranjo de câmeras.

Um exemplo mais recente de técnica baseada em pontos é o trabalho de Davison et al. (2007), que demonstraram um sistema capaz de localizar uma câmera em tempo real em um ambiente previamente desconhecido. Este sistema realiza uma extração de pontos similar ao anterior, porém o deslocamento relativo da câmera entre as imagens obtidas é desconhecido, e os marcos também precisam ser rastreados ao longo do tempo para permitir a estimativa de todos os parâmetros. Os mais populares algoritmos de calibração também são todos basados em pontos (ZHANG, 2000; HORN, 2000). Estes métodos de calibração podem ser utilizados com diferentes formas de extração de pontos das imagens de entrada, além de diferentes técnicas para associar estes pontos. Deve ser observado que estes métodos de calibração citados diferem apenas na forma como é realizada uma aproximação inicial do problema, porém na prática, quando se deseja alcançar a melhor precisão possível e não há restrições fortes no custo computacional, é preciso encarar o problema como uma instância de *bundle adjustment* onde os parâmetros internos também devem ser estimados enquanto se minimiza o erro de reprojeção dos pontos modelados.

Estes exemplos ilustram a forma geral de um sistema de visão baseado em extração de pontos. Nestes sistemas geralmente considera-se que o ambiente possui um conjunto de pontos tridimensionais, também denominados *marcos*, cujas projeções nas imagens podem ser calculadas através do modelo de câmera, ou medidas através de algum processo de extração ou registro de pontos — o *registro* se refere a um processo de busca na imagem por uma observação com características específicas conhecidas. O uso de

pontos correspondentes em múltiplas imagens para resolver problemas de rastreamento de câmera, mapeamento ou calibração é anterior ao surgimento da Visão Computacional. Este problema foi estudado primeiro dentro da Geodésia e Fotogrametria (TRIGGS et al., 2000), e tornou-se alvo de estudos da visão tão cedo quanto esta área surgiu. Pontos podem ser utilizados tanto para modelagem do ambiente, através de um conjunto de marcos pontuais, quanto para a determinação dos parâmetros extrínsecos e intrínsecos da câmera, ou seja, localização e calibração. Ou seja, há uma grande variedade de problemas que podem ser abordados através destas técnicas, e é até mesmo possível estudar problemas tal como o reconhecimento de objetos utilizando apenas marcos pontuais.

Técnicas de visão por pontos para a reconstrução de ambientes são geralmente baseadas em pares ou conjuntos maiores de observações correspondentes que são projeções de um mesmo ponto do espaço, e que são extraídas das diferentes imagens disponíveis. Em um problema em que há apenas duas câmeras com posição relativa conhecida, parâmetros intrínsecos conhecidos, e se conhecem ainda as posições das projeções de um marco em ambas imagens, a localização deste ponto no ambiente pode ser determinada através de triangulação. Esta é uma técnica de simples compreensão, ao alcance de qualquer pessoa com conhecimentos básicos de geometria.

Este problema simples de localização de um único ponto pode se tornar difícil rapidamente quando se começam a considerar imperfeições e generalizações. Quando múltiplas câmeras e pontos precisam ser considerados, e mais parâmetros precisam ser estimados, temos então um exemplo mais completo do problema de *bundle adjustment* (ou BA da sigla em inglês, ou *ajuste de feixes* em português). Este nome se refere ao fato de que cada marco e cada câmera constitui um ponto por onde passam diversos raios de luz, formando feixes que interligam estas entidades formando um grafo. Os raios correspondentes a um único marco projetados a partir das câmeras devem se encontrar sobre a posição do marco, e os raios projetados de uma mesma câmera devem cruzar todos os marcos correspondentes. O problema é encontrar as posições dos marcos e das câmeras, além das orientações das câmeras e parâmetros intrínsecos, de forma a atender estas restrições geométricas.

Como explicado por Triggs et al. (2000), e como já dito anteriormente, a solução para um problema de *bundle adjustment* qualquer segue sempre um certo espírito geral que deve ser adaptado para diferentes aplicações. A forma da solução é sempre um processo de otimização que minimiza uma determinada função de erro. Diferentes aplicações podem empregar diferentes processos de otimização na busca por maior eficiência ou precisão, explorando peculiaridades presentes. As grandezas analisadas nas imagens para definir esta função de erro são geralmente baseadas na análise de características geométricas extraídas, pontuais ou mais complexas, mas é possível utilizar também funções baseadas mais diretamente nos próprios valores de intensidade luminosa da imagem. O uso de características geométricas é empregado quando possível porque permite dividir a

análise em estágios distintos, simplificando o funcionamento dos sistemas e melhorando o desempenho através da redução de dados.

Nos casos em que se decide utilizar características pontuais para realizar o BA, a função de erro mais simples e usual de se empregar é a soma dos erros quadráticos de reprojeção. Este erro é a soma dos quadrados das distâncias entre os pontos extraídos das imagens e as projeções calculadas a partir de um dado conjunto de parâmetros. Estes parâmetros incluem as coordenadas do ponto no espaço e todos os parâmetros extrínsecos e intrínsecos da câmera. Uma importante modificação que pode ser realizada nesta função de erro é o uso dos M-estimadores discutidos na Seção 2.4.2. Uma função de erro robusta definida por esta técnica pode ajudar a enfrentar problemas ligados a vários fenômenos difíceis de modelar, mas são particularmente importantes no caso do BA para lidar com observações associadas incorretamente.

Quando se fala em *bundle adjustment* geralmente assume-se que a questão da associação de observações entre diferentes imagens já foi resolvida. O problema específico do BA seria apenas como calcular os parâmetros considerando-se as características geométricas do problema. Em muitos cenários de aplicação, porém, a determinação destas correspondências pode ter que ocorrer de forma simultânea ao próprio cálculo dos parâmetros, necessitando da aplicação de alguma técnica como as discutidas nas seções 2.3 e 2.4.

Observações associadas incorretamente criam *outliers* nos dados sendo analisados, que podem acarretar no surgimento de erros muito grandes no cálculo da função de erro, mesmo sobre a solução. A existência deste tipo de *outlier* é inerente a processos de estimação em que não se conhecem exatamente as associações, e um estimador robusto tem a capacidade de tornar a estimação menos sensível a este tipo de erro. Um revés desta técnica é a maior dificuldade em se calcular as expressões necessárias para a solução do problema, além de possíveis dificuldades na convergência do processo de otimização. Porém seu uso pode ser indispensável em algumas aplicações.

As seções a seguir discutem técnicas para a extração pontos de interesse, e algumas aplicações de visão baseada em pontos.

### 3.1.1 Extração de pontos de interesse

Como discutido na Seção 2.2, uma imagem digital é uma função de duas dimensões, modelada por um conjunto de pixels que são amostras de seu valor retiradas de pontos de uma grade. Processos de extração de características geométricas varrem os pixels de uma imagem em busca de pontos que obedeçam algum critério de detecção, e modelos geométricos podem então ser ajustados de forma mais precisa sobre estas regiões da imagem onde ocorreram detecções positivas de características geométricas de interesse.

No caso da extração de pontos esta detecção é geralmente realizada com uma busca por extremos locais de uma função definida sobre a área da imagem, que é calculada a

partir dos valores da intensidade da imagem. Mais especificamente, existe uma função auxiliar que deve ser calculada sobre a imagem, e o valor desta função em cada ponto depende dos valores de intensidade na região circundante deste ponto. Os extremos desta função auxiliar são os pontos de interesse a serem extraídos. Técnicas mais simples apenas calculam esta função auxiliar sobre cada pixel da imagem, e seleciona pixels com valores maiores do que seus vizinhos. Mas pode ser possível realizar uma busca mais refinada, com precisão de sub-pixel, simplesmente realizando uma otimização contínua desta função de análise.

Existem diversas abordagens para se criar um extrator de pontos de interesse, variando principalmente na forma como se calcula a função auxiliar. A utilidade de um extrator para problemas de visão computacional vai depender da sua capacidade de produzir valores consistentes para as coordenadas do marco sendo rastreado. Isto é, as coordenadas encontradas nas imagens precisam ser a projeção do mesmo ponto do ambiente conforme a câmera se desloca. Bons detectores costumam se basear em algum tipo de invariância a transformações da imagem. A invariância a translação é uma característica indispensável. A invariância a rotação é bastante desejável, assim como a mais geral invariância a transformações afim. Invariância a alterações de brilho e contraste também são bastante úteis. Estas operações não são capazes contudo de abranger todas as modificações possíveis no aspecto de um objeto conforme ele é observado de diferentes pontos de vista, e qualquer detector de pontos de interesse irá fatalmente apresentar algum tipo de falha em alguma aplicação minimamente genérica. Isto significa que as coordenadas encontradas podem sofrer flutuações, e pontos que são detectados com sucesso sob certas condições podem deixar de ser produzidos em outras, constituindo um falso negativo de detecção.

Uma comparação do desempenho de diferentes extratores de pontos foi realizada por Tuytelaars e Mikolajczyk (2007). Além da extração dos pontos, muitas aplicações fazem uso também da obtenção de um descritor para cada ponto, que pode ser utilizado para auxiliar na identificação da observação entre diferentes imagens. Assim como no caso dos detectores, é desejável que os descritores sejam tão consistentes quanto possível. Uma comparação entre diferentes tipos de descritores foi realizada por Mikolajczyk e Schmid (2005).

É importante destacar que um detector de pontos de interesse envolve por sua natureza uma busca por toda a área da imagem. Uma sub-amostragem da imagem, por exemplo selecionando-se um conjunto reduzido de pontos aleatoriamente antes de realizar a detecção, dificilmente produz um resultado útil, exceto talvez se acoplado a alguma forma de busca por picos local que fatalmente negaria a intenção inicial de realizar uma redução de dados. Como veremos no Capítulo 4 bordas são uma forma de característica geométrica mais adequada para este tipo de redução de dados, que não exigem que toda a área da imagem seja levada em consideração para que se extraia uma quantidade suficiente de dados para serem analisados.

### 3.1.2 Aplicações baseadas em pontos

Aplicações de BA basadas em pontos ou não podem ter diferentes objetivos. No caso mais geral não se conhece nada a respeito do ambiente ou das câmeras, mas apenas os parâmetros dos grupos de características geométricas extraídas das imagens. Se as associações não forem conhecidas algum algoritmo que permite inferir estas informações deve ser utilizado em um nível superior para permitir resolver o problema. Diferentes problemas de BA se caracterizam principalmente por qual conjunto de parâmetros dos modelos deve ser estimado. Há casos onde apenas parâmetros do ambiente precisam ser encontrados, e há casos onde apenas parâmetros da câmera são desconhecidos. Estes diferentes casos de aplicação já foram brevemente discutidos no Capítulo 1.

No caso de armações estereoscópicas a localização relativa das câmeras é conhecida, bem como seus parâmetros intrínsecos, e deseja-se localizar os objetos no espaço. Nestas condições a própria detecção de pontos pode ser realizada de maneira que observações correspondentes nas diferentes imagens sejam determinadas com facilidade, explorando a chamada restrição de *geometria epipolar* (HARTLEY; ZISSEMAN, 2003; MORAVEC, 1980; SE; LOWE, 2001). A possibilidade deste tipo de análise das imagens resolve o problema da determinação de associação de dados, e faz com que uma armação estereoscópica constitua um sistema que funciona como um confiável sensor de distâncias. Uma armação com parâmetros internos conhecidos, o que inclui a determinação das posições relativas das câmeras, exige um pouco de trabalho para ser obtida, mas feito isto o problema de localização que precisa ser resolvido para produzir as observações é bem mais simples do que um BA completo.

A estimativa dos parâmetros intrínsecos de uma câmera, ou de um conjunto de câmeras como em uma armação estereoscópica, constitui um outro tipo de problema que se pode encaixar no BA, que é o processo conhecido por *calibração de câmera*. Aqui a prioridade é obter os parâmetros intrínsecos da câmera para o uso em outras aplicações, e assim um sistema de calibração pode contar por exemplo com alvos de calibração conhecidos e confiáveis para facilitar o processo. A calibração realizada sem auxílio de alvos artificiais é conhecida por *auto-calibração*, e é mais desafiadora pela necessidade de modelar todo o ambiente. A calibração é fundamental para o funcionamento de diversas técnicas de visão computacional que assumem que o modelo da câmera é conhecido com perfeição. Apesar da importância, processos de calibração podem ainda hoje se mostrar bastante laboriosos, e são frequentemente obstáculos para a realização de pesquisas e criação de aplicações. Algumas técnicas bem-conhecidas de calibração baseada em pontos são o algoritmo de Tsai (LENZ; TSAI, 1988; HORN, 2000), e o algoritmo de Zhang (2000). Um exemplo mais recente é a técnica criada por Furukawa e Ponce (2008), em que a calibração é realizada através um modelo denso de ambiente. Todos estes algoritmos se baseiam na minimização do erro médio quadrático das reprojeções.

Outra possibilidade relevante de especialização do BA é determinar a localização das

câmeras para algum fim. Como exemplo de um trabalho recente interessante em que a localização das câmeras é a maior prioridade, apesar de todos os parâmetros serem inicialmente desconhecidos, podemos citar o trabalho desenvolvido por Snavely, Seitz e Szeliski (2007) na criação do sistema *Photo Tourism*. Este sistema permite a um usuário navegar em um conjunto de fotos de um mesmo ambiente, visualizando as fotos capturadas de diferentes posições conforme ele se movimenta por um espaço virtual. O modelo do ambiente em si é de interesse menor durante o uso do sistema pelo usuário final, e os pontos aparecem apenas como forma de indicar a localização aproximada dos objetos no espaço. Uma vez que a estrutura do ambiente já foi determinada a partir de um conjunto de imagens, uma nova imagem pode ser inserida na base de dados realizando-se apenas a estimativa de seus parâmetros, sem modificações ao mapa. Os parâmetros intrínsecos podem ser ajustados para cada imagem, mas a precisão desta estimativa não é crítica para o propósito desta aplicação.

Outro exemplo interessante de aplicação baseada em pontos desenvolvida nos últimos anos é a localização de câmera dentro de um ambiente desconhecido, em tempo real. Um dos trabalhos pioneiros da área foi o desenvolvido por Davison et al. (2007). O objetivo principal da aplicação é rastrear a câmera ao longo do tempo, mas como o ambiente não é completamente conhecido torna-se necessário estimar também a posição de novos pontos de referência ao longo do tempo. Para isto são extraídos de cada imagem um pequeno número de pontos que se tenta rastrear por alguns instantes, enquanto a própria câmera é rastreada a partir de referências já conhecidas. Ao longo do tempo estes novos pontos são inseridos no mapa, permitindo continuar o processo até que um mapa minimamente satisfatório é obtido, e o sistema passa então a apenas rastrear a câmera. Este sistema requer o conhecimento prévio dos parâmetros intrínsecos, além de um modelo inicial de parte do ambiente para permitir iniciar o processo.

O problema da localização em tempo real se diferencia por poder contar com boas estimativas iniciais para a posição da câmera em cada novo quadro capturado. Estas estimativas são obtidas através de um modelo físico da dinâmica da câmera ao longo do tempo. A busca por pontos correspondentes entre os diferentes quadros também é simplificada pelo fato de que a distância percorrida é pequena, criando pequenas disparidades nas coordenadas dos pontos e assim limitando as regiões em que os pontos correspondentes precisam ser buscados. Muitas variações já surgiram desde o trabalho de Davison et al. (2007), mudando a forma como se realiza a extração de amostras, associação, ou a estimativa de parâmetros (EADE; DRUMMOND, 2006; KLEIN; MURRAY, 2007; MOURAGNON et al., 2006; STRASDAT; MONTIEL; DAVISON, 2010).

O estudo do rastreamento de câmera em tempo real tem se mostrado oportuno para promover a integração das comunidades de pesquisadores oriundos da computação e da robótica (STRASDAT; MONTIEL; DAVISON, 2010). Dentro da computação este caso específico de BA é geralmente entendido como um caso do problema de estrutura a partir

do movimento, que recebe a sigla SFM do inglês *shape from motion*. A ênfase maior dentro da computação era inicialmente na obtenção da estrutura, que é o problema que se encontra analisando imagens de armações estereoscópicas, por exemplo, porém o caso em que também se deseja obter a localização de câmera acaba sendo englobado quando cenários mais gerais são considerados. Já na robótica este problema é entendido como um caso específico do problema de *localização e mapeamento simultâneos*, bastante conhecido pela sigla SLAM do inglês *simultaneous localization and mapping*. Este problema é estudado dentro da robótica utilizando-se sensores de diferentes tipos, incluindo câmeras, que se destacam nesta área por serem sensores que não são capazes de realizar leituras diretas da distância de objetos, mas apenas de direções. Como na robótica móvel as câmeras colocadas a bordo dos robôs encontram-se sempre em movimento o problema da realização de localização de câmera recebeu um pouco mais de atenção desta comunidade no passado, enquanto na computação foi possível dar um pouco mais de atenção para problemas de reconstrução. Hoje, porém, há um interesse generalizado na análise de vídeos e de grandes conjuntos de imagens, e assim os interesses destas comunidades estão se sobrepondo cada vez mais.

Algumas destas técnicas de rastreamento em tempo real de câmera em ambientes desconhecidos empregam para a estimativa de parâmetros algoritmos relativamente simples se comparados à resolução de um problema completo de BA. É o caso do uso de algoritmos como o filtro de Kalman utilizado por Davison et al. (2007), por exemplo. Pesquisas recentes têm demonstrado, entretanto, que esta simplificação não é necessariamente vantajosa, e processos mais sofisticados de estimativa de parâmetros podem ser empregados em tempo real produzindo melhores resultados (STRASDAT; MONTIEL; DAVISON, 2010; KAESZ; RANGANATHAN, 2008; MEI et al., 2009). Técnicas como o filtro de Kalman também podem se tornar pouco eficientes quando o número de parâmetros do ambiente cresce demais, tornando necessária a aplicação de técnicas mais sofisticadas.

Este conjunto de técnicas listadas até agora nesta seção ilustram bem a diversidade de problemas que se pode abordar com base no mesmo princípio de *bundle adjustment*, e utilizando-se apenas observações pontuais. A estrutura do ambiente é modelada por um conjunto de pontos esparsos, formando uma nuvem de pontos, e as câmeras definem uma outra nuvem de referenciais com diferentes posições e orientações espalhados pelo mesmo espaço. Os pontos a serem localizados são primeiramente obtidos através de algum processo de extração (TUYTELAARS; MIKOLAJCZYK, 2007), e associados de alguma maneira entre as diferentes imagens. Um erro de reprojeção para cada observação de cada marco pode ser então calculado a partir das coordenadas tridimensionais do marco e dos parâmetros da câmera. Ao minimizar este erro, mantendo diferentes conjuntos de parâmetros fixos, é possível realizar procedimentos de rastreamento de câmera, reconstrução da estrutura do ambiente, calibração, ou até mesmo tudo de uma só vez.

A otimização nestes problemas é usualmente realizada através da aplicação de alguma

versão algoritmo de Gauss-Newton, mais especificamente o algoritmo de Levenberg-Marquardt (TRIGGS et al., 2000; FLETCHER; LEYFFER, 2002), mas existem alternativas com aproximações numéricas que podem apresentar bons desempenhos, tal como o uso do algoritmo da perna-de-cachorro de Powell (LOURAKIS; ARGYROS, 2005), ou de otimização pelo algoritmo do gradiente conjugado com uso de precondicionadores (BYROD; ASTROM, 2009). Existem também métodos baseados em reordenação de variáveis e separação do sistema por blocos, como o apresentado por Jeong, Nister e Steedly (2010). Em problemas de rastreamento podem ser aplicadas técnicas clássicas como o filtro de Kalman (DAVISON et al., 2007), ou ainda técnicas mais sofisticadas como a utilizada por Kaess e Ranganathan (2008), que envolvem tanto melhorias no resultado da estimativa pela realização de suavização, quanto propostas de maneiras eficientes de realizar os cálculos.

Estas técnicas de visão baseada em pontos permitem portanto resolver uma grande variedade de problemas, e podem apresentar desempenhos muito bons. Porém ainda assim as nuvens de pontos resultantes destas técnicas possuem um potencial limitado para aplicação. Quanto mais esparsos os pontos, mais o mapa produzido serve apenas para realizar a tarefa de rastreamento de câmeras ou de calibração, mas tornam os métodos inúteis para tarefas como modelagem e reconhecimento de objetos, ou localização de obstáculos para robótica móvel ou realidade aumentada. Uma forma de tentar contornar isto para criar modelos do ambiente com maior potencial de aplicação seria o uso de técnicas de reconstrução densa, em que se busca aproveitar todos pixels das imagens ao invés de apenas poucos pontos de interesse.

Um exemplo notório de um sistema de reconstrução densa é o detector de obstáculos empregado nas sondas marcianas MER da NASA (MAIMONE; LEGER, 2007). Um par de câmeras localizado à frente das sondas constitui um arranjo estereoscópico calibrado, e as imagens obtidas pelas câmeras são devidamente *retificadas*, ou seja, transformadas de maneira a simplificar os processos de análise seguintes, e então um grande número de pontos correspondentes em ambas imagens é produzido. A nuvem de pontos resultante é utilizada para calcular um modelo de elevação do terreno à frente da sonda, que é utilizado para planejar trajetórias seguras. A Figura 1.1 mostra um exemplo de uma foto obtida pela sonda, e o modelo de elevação de terreno correspondente. Este sistema descende da pesquisa de Moravec (1980). Naquele trabalho os obstáculos eram qualquer ponto percebido nas imagens, e o solo transitável era liso e portanto não produzia observações. No caso das sondas marcianas o solo é bastante acidentado e texturizado. A riqueza em detalhes visuais é uma condição necessária para se conseguir gerar modelos tão detalhados, e a complexidade do terreno faz com que a transitabilidade precise ser determinada de uma forma mais complexa se comparada ao trabalho de Moravec.

Alguns outros exemplos atuais e relevantes de técnicas para reconstrução densa são primeiro o trabalho de Furukawa e Ponce (2010), que é atualmente um das técnicas do

tipo que alcançam maior precisão. Este sistema inicia a análise dos dados realizando localização e mapeamento a partir de características pontuais em um primeiro estágio, alcançando então a reconstrução densa através de um segundo estágio iniciado com os parâmetros obtidos no primeiro. Outro exemplo é a técnica de rastreamento de câmera com reconstrução densa em tempo real apresentada por Newcombe, Lovegrove e Davison (2011). Esta segunda técnica possui duas grandes virtudes. A primeira é o fato de criar um modelo bem mais útil do que uma mera nuvem de pontos, e que pode ser aproveitado em aplicações de realidade aumentada ou robótica. A segunda virtude é que a reconstrução densa contribui para o próprio processo de rastreamento, tornando o sistema robusto a movimentos bastante bruscos, o que costuma ser um problema para as técnicas de rastreamento de câmera em tempo real baseadas em extração de características geométricas.

### 3.2 Técnicas de visão baseadas em bordas

O interesse pela análise de curvas presentes em imagens sempre existiu ao lado de outras entidades como pontos ou até outras entidades mais complexas. De fato, pontos de interesse de imagens são chamados de *cantos* por alguns autores, pois não indicariam quaisquer tipos de pontos de características interessantes, mas especificamente pontos finais de curvas, ou pontos de junção entre duas ou mais curvas diferentes. O popular detector proposto por Harris e Stephens (1988) pretendia originalmente servir tanto para detecção de curvas quanto de cantos, apesar de pesquisas similares terem se focado apenas em pontos (MORAVEC, 1980; SHI; TOMASI, 1994). A ideia de que existem curvas com extremidades e junções na imagem deu origem a muitas pesquisas na área de extração de pontos de interesse, que no entanto visavam efetivamente extrair estas curvas, e encontrar apenas estes pontos.

É relativamente difícil definir um ponto de interesse sem se valer da ideias relacionadas à existência de curvas. Isto se reflete no fato de que as funções auxiliares que são utilizadas na análise das imagens para extraír pontos de interesse são relativamente complexas. É preciso valer-se de conceitos como a autocorrelação local, ou cálculos de momentos que buscam caracterizar toda a região ao redor de cada ponto. Já bordas são relativamente mais simples de serem definidas. Uma região de uma imagem próxima a uma borda são caracterizadas pela existência de uma direção em que não há variação na cor da imagem, ou em alguma outra grandeza ligada à textura por exemplo, enquanto a variação na direção ortogonal é intensa. O uso da própria intensidade de imagem ao invés de alguma outra função auxiliar é geralmente bem-sucedido, o que não costuma ser o caso para pontos.

Uma prova da superioridade de bordas como entidade fundamental para a análise de imagens é a forma como são construídos alvos para a calibração de câmeras. Um padrão muito utilizado é o de um tabuleiro de xadrez, com quadrados brancos e pretos alternados. Os cantos destes quadrados são os marcos utilizados para o processo de calibração, e

para a sua extração são geralmente utilizadas técnicas como as discutidas anteriormente. Este cenário de aplicação permite entretanto o desenvolvimento de detectores mais especializados, o que pode ser bastante benéfico (SUN et al., 2008; DAO; SUGIMOTO, 2010). Isto é apenas um exemplo de como um extrator de cantos criado com base em uma melhor modelagem das bordas que originam estes pontos pode funcionar bem melhor do que extractores genéricos. No caso genérico as prioridades são velocidade de cálculo, simplicidade de implementação, e capacidade de produzir uma boa quantidade de pontos. Qualquer aplicação baseada em características geométricas pode se beneficiar da utilização de modelos mais precisos e especializados quando possível. E bordas podem oferecer um melhor caminho para a criação destes modelos mais precisos.

O modelo mais comum para uma borda ideal em uma imagem é uma descontinuidade no valor de intensidade da função encontrada durante um deslocamento na direção ortogonal à borda, criando uma função degrau, como foi discutido na Seção 2.2. Um dos aspectos mais importantes do estudo da localização de bordas é compreender como melhor detectar estas descontinuidades em uma dimensão, e é possível considerar modelos além do de degrau, tais como descontinuidades apenas derivada da função — borda do tipo “telhado” — ou ainda linhas finas, criadas por duas descontinuidades muito próximas e com sentidos opostos. O uso deste tipo de modelagem mais sofisticada é mais simples em análises com bordas do que com pontos devido à natureza unidimensional das bordas. Modelos mais complexos para pontos precisam ser basear em funções de duas dimensões.

Os métodos para detecção de bordas e extração de linhas discutidos na Seção 2.2 são utilizados na prática para se resolver diversos problemas de visão envolvendo retas, curvas e entidades similares. Boa parte das pesquisas neste tipo de problema utilizam alguma técnica como o seguimento de contorno, ou a transformada de Hough ou o algoritmo RANSAC para extraír retas ou segmentos de reta das imagens, que são então analisadas posteriormente. Aplicações que demandam menor tempo de processamento podem utilizar extractores de reta como o proposto por Matas, Galambos e Kittler (2000), por exemplo, que realiza alguma forma de sub-amostragem. Diferentes versões do algoritmo RANSAC também podem oferecer, por exemplo, maior precisão ou menor tempo de processamento para realizar a tarefa de extração de retas.

Depois que estas características são extraídas, elas devem então ser analisadas para realizar a tarefa desejada, assim como ocorre com pontos nos métodos discutidos na Seção 3.1. Assim como os pontos extraídos são considerados como sendo a projeção de marcos pontuais, retas ou curvas extraídas das imagens podem ser considerados como a projeção de retas que existem no ambiente. Através de uma definição de um erro de ajuste que compara o modelo extraído com uma projeção predita, é possível então realizar uma otimização que encontra os parâmetros desejados para os modelos de cena e de câmera. Apesar destas entidades não darem origem aos grupos de linhas de onde originou-se o nome *bundle adjustment*, o princípio de funcionamento da técnica com linhas ao invés

de pontos é o mesmo.

Assim como nas técnicas baseadas em pontos, as retas extraídas de diferentes imagens devem ser agrupadas de acordo com o objeto do ambiente que as produziu, e quando necessário é possível empregar algoritmos como o RANSAC para encontrar estas associações, e também é possível realizar processos de rastreamento em que se encontra facilmente novas observações correspondentes conforme novas imagens são analisadas pelo sistema. Uma diferença importante do BA com retas comparado ao uso de pontos é que não há uma forma mais simples de se definir o erro de reprojeção. A simples subtração das coordenadas de dois pontos cria um vetor residual cuja norma quase sempre serve para definir uma função de erro. No caso de retas ou segmentos de reta há diferentes maneiras de proceder, e esta escolha pode ter um impacto significativo nos resultados alcançados.

É interessante notar que o problema de *bundle adjustment* é bastante similar ao próprio problema de extração de linhas. Ambos dependem de uma associação de dados que pode ou não ser disponibilizada previamente, e os parâmetros dos modelos são encontrados a partir destes dados através de alguma técnica de otimização, e a associação também pode ser determinada ao longo da otimização se necessário. A grande semelhança entre estes problemas significa que todas técnicas discutidas na Seção 2.3 ser aplicadas em ambos problemas. Na prática é mais comum ver o RANSAC aplicado a problemas de BA, e a transformada de Hough para extração de linhas. Mas é possível, por exemplo, abordar o problema de reconstrução de cena em duas dimensões a partir de câmeras com parâmetros conhecidos utilizando para isto técnicas mais tradicionalmente empregadas para a extração de linhas. Isto foi demonstrado em um par de artigos publicados durante a pesquisa que levou ao desenvolvimento do Corisco (WERNECK; TRUZZI; COSTA, 2009; WERNECK; COSTA, 2010, 2011a).

### 3.2.1 Extração de ponto de fuga

O problema da extração de pontos de fuga, introduzido na Seção 2.2.3, é um terceiro problema de estimativa de parâmetros de modelos geométricos que depende de uma associação de dados, assim como o BA e a extração de retas. A projeção de um grupo de linhas que são paralelas no espaço produz um conjunto de linhas no plano da imagem que se unem em um ponto em comum que é o ponto de fuga. O ponto de fuga permite encontrar a direção no referencial da câmera das linhas do ambiente associadas a ele. Para estimar as coordenadas de um ponto de fuga, ou de sua direção associada através do modelo de câmera, é preciso determinar quais linhas da imagem são projeções desta direção, ou seja, quais linhas cruzam o ponto de fuga, e então realizar um processo de minimização de erros a partir destas entidades.

Esta estimativa das direções das retas do ambiente no referencial de câmera através da extração de pontos de fuga permite realizar duas tarefas importantes, que são a estimativa da orientação da câmera em relação ao ambiente e a estimativa de parâmetros intrínsecos

da câmera. Isto requer o conhecimento das direções possíveis das retas do ambiente, e para isto pode ser possível utilizar a chamada restrição de ambiente antrópico mencionada na Seção 1.1, que é uma condição bastante comum. Esta hipótese diz que as retas se encontram direcionadas de acordo com os três eixos de um certo referencial, denominado referencial natural do ambiente, ou canônico.

A principal diferença da estimação de orientação de câmera através de pontos de fuga comparado com as técnicas baseadas em pontos é o fato de que o processo depende de uma única imagem, enquanto no caso do uso de pontos é preciso ao menos uma segunda câmera, além da criação de um modelo do ambiente. Algumas aplicações possíveis para este tipo de estimação de orientação são a condução de robôs móveis, reconhecimento de objetos e ainda o desenvolvimento de sistemas de Realidade Aumentada. Os maiores atrativos da técnica são a possibilidade de criar métodos de alto desempenho, a facilidade de funcionar em ambientes desconhecidos, e o fato de que se leva em consideração características do ambiente que podem ter significado dentro da aplicação, que são as direções do referencial natural do ambiente. A estimação de orientação através de uma única imagem também pode ser útil para o próprio problema de BA, servindo de estimativa inicial da orientação em um sistema como o de Antone e Teller (2002) ou o de Sinha, Steedly e Szeliski (2010). A extração de pontos de fuga não está necessariamente ligada a aplicações como estas, no entanto, e o problema pode ser abordado de um ponto de vista completamente abstrato, sem levar em consideração a existência de algum modelo de câmera.

Uma das pesquisas pioneiras em estimação de ponto de fuga foi a de Barnard (1983). A técnica proposta em seu artigo inicia com uma extração de segmentos de reta. Para cada segmento calcula-se o vetor normal do chamado *plano de interpretação*, que é um plano que tangencia a reta e a origem do referencial da câmera, que é o ponto focal do sistema ótico. Estes planos dão origem a círculos máximos sobre a esfera Gaussiana centrada na origem. Sobre a esfera Gaussiana são também definidos acumuladores referentes a pequenas regiões, que são incrementados para cada círculo que as tangencia. Os acumuladores com grandes valores indicam as direções das retas correspondentes no ambiente, e consequentemente a direção do seu ponto de fuga. É interessante notar que não é realmente necessário conhecer os parâmetros do modelo de câmera para que esta técnica funcione. O modelo pode ser completamente fictício, e servir apenas para criar um espaço de parâmetros mais adequado para lidar com as retas sobre o plano.

Esta técnica dos acumuladores é na realidade apenas mais um exemplo da transformada de Hough, e pesquisas subsequentes logo buscaram aplicar as inovações obtidas para esta ferramenta para o problema específico de extração de ponto de fuga. Por exemplo, Quan e Mohr (1989) utiliza uma transformada de Hough hierárquica na esfera, a partir de linhas extraídas com a transformada comum. Já Tuytelaars et al. (1998) buscou explorar melhor o fato de que há diversas transformadas de Hough sendo aplicadas em

cascata nesta abordagem para o problema. A primeira transformada para a extração de linhas, uma segunda para localizar os pontos de fuga, e até mesmo uma terceira transformada pode ser utilizada para extrair, por exemplo, a linha do horizonte da imagem.

O método apresentado por Cantoni et al. (2001) também se baseia na transformada de Hough, mas ao invés da esfera gaussiana ele utiliza a própria área da imagem como espaço de parâmetros onde o ponto de fuga é procurado. Como resultado há a desvantagem de não se poder considerar pontos de fuga localizados fora da área da imagem, mas trabalhar no plano traz algumas vantagens práticas para a implementação. Este método é interessante porque é mais um exemplo de um uso de edgels para a análise de imagens. O processo de análise deste método inicia com uma detecção de bordas, e para cada ponto detectado preserva-se sua localização e direção do gradiente, o que constitui um edgel. Ao invés de seguir este passo com alguma forma de agrupamento das observações para extração de linhas, estes autores simplesmente instanciaram linhas independentes a partir de cada um destes edgels. Os acumuladores são atualizados para cada uma destas linhas, e o ponto de fuga será encontrado sobre o acumulador com maior valor no final do processo.

Uma forma alternativa de realizar a extração dos pontos de fuga é associar as linhas em pares, supondo elas serem de um mesmo grupo, e então calcular uma localização hipotética do ponto de fuga a partir delas (LUTTON; MAHE; LOPEZ-KRAHE, 1994). Esta seria uma abordagem semelhante ao funcionamento do RANSAC. Outra abordagem possível, alternativa à transformada de Hough, é o uso do algoritmo J-linkage seguido por uma estimação EM (TARDIF, 2009).

Enquanto estes métodos discutidos realizam inicialmente uma extração desinformada de retas seguida pela estimativa de um ponto de fuga a partir delas, é possível abordar o problema desde o princípio como uma extração de retas que obedece a restrição imposta pelo ponto de fuga. Isto é realizado por exemplo no método proposto por McLean e Kotturi (1995). Um outro exemplo de método que encontra pontos de fuga sem realizar uma extração desinformada de linhas é o apresentado por Kogan, Maurer e Keshet (2009), onde pares de linhas e colunas da imagem são analisados para encontrar hipóteses da localização de um ponto de fuga diretamente.

Estes dois últimos exemplos mostram que é possível lidar com a questão da estimativa de ponto de fuga evitando a extração de retas. A seção a seguir traz alguns métodos que vão ainda além, e incluem até mesmo restrições ligadas ao problema da estimativa de orientação de câmera no processo. O *Corisco* segue esta mesma estratégia, assim como outros métodos para a estimativa de orientação a partir de edgels que buscam explorar as restrições impostas pelo problema tão cedo quanto possível na análise dos dados. Porém o *Corisco* não realiza uma extração simultânea de retas, trabalhando apenas com edgels que permanecem desassociados, apesar de serem classificados de acordo com seu ponto de fuga associado.

Este conjunto de métodos para a extração de pontos de fuga citados aqui ilustra bem a variedade funções de erro que se pode definir envolvendo retas. As formas mais comuns de definir a distância entre uma reta e um ponto de fuga são:

- *Euclideana* — A simples distância entre o ponto e a reta sobre o plano da imagem (CANTONI et al., 2001).
- *Alinhamento na imagem* — A distância entre as extremidades de um segmento de reta até a reta que liga o ponto mediano do segmento ao ponto de fuga (MCLEAN; KOTTURI, 1995; ROTHER, 2002; SCHINDLER; KRISHNAMURTHY; DELLAERT, 2006; TARDIF, 2009).
- *Alinhamento espacial* — A projeção do vetor normal do plano definido a partir da reta e do ponto focal sobre o vetor da direção do ponto de fuga (BARNARD, 1983; COLLINS; WEISS, 1990).

A extração de pontos de fuga constitui portanto um problema na fronteira entre análises de imagem de mais baixo nível e problemas de visão de mais alto nível. As mesmas questões relativas a como proceder para extrair observações, associá-las e então realizar uma estimativa de parâmetros existem para a extração de retas, para problemas de nível um pouco mais alto como a extração de pontos de fuga, que não envolvem necessariamente modelos de câmera, e problemas ainda superiores de visão, como o BA. E enquanto é possível resolver um problema de mais alto nível em vários estágios intermediários onde se encontram parâmetros para estruturas cada vez mais complexas, também é possível em certos casos realizar uma bordagem mais direta, resolvendo o problema com um menor afastamento dos dados originais de entrada.

A seção a seguir irá finalmente abordar o uso de retas em técnicas de visão de maior grau de abstração, e não simplesmente para resolver o problema intermediário de extração de ponto de fuga como discutido nesta seção.

### 3.2.2 Aplicações baseadas em linhas

O uso de linhas extraídas de imagens para problemas de localização de câmera ou de objetos é algo bastante antigo. Um trabalho pioneiro na área foi o de Duda (1970), onde o rodapé das paredes de um ambiente antrópico conhecido era utilizado para realizar localização. Esta técnica foi aplicada em um dos primeiros robôs móveis, Shakey (NILSSON, 1984). Além da localização, este sistema também era capaz de identificar objetos pela câmera, o que é necessário para que o robô possa realizar tarefas.

O rastreamento de câmeras através de linhas no solo é um problema específico com bastante potencial para aplicação, e algumas variações continuam sendo investigadas. Uma técnica de SLAM monocular baseada em linhas no solo foi apresentada por Santana e Medeiros (2010). As linhas são extraídas das imagens utilizando a transformada

de Hough. Alguns outras trabalhos interessantes onde se utiliza a restrição de que as bordas detectadas estão sobre o plano do solo foram apresentados por Howard e Kitchen (1997), Ulrich e Nourbakhsh (2000) e Lenseir e Veloso (2003). Nestas pesquisas não se realiza extração de retas, entretanto. Bordas são encontradas em cada coluna da imagem, varrendo-a de baixo para cima, e cada uma destas linhas cria uma leitura diferente de direção e distância até um obstáculo. Este conjunto de leituras é analisado a seguir como se fosse a saída de um sensor de distâncias, como os baseados em laser ou sonar.

Outra forma restrita de se analisar linhas extraídas de imagens para aplicações de visão é a utilização apenas de retas verticais encontradas no ambiente. A geometria deste problema foi estudada inicialmente por Nakamura e Ueda (1982), mas a técnica só foi tornada prática por Sugihara (1988) e Krotkov (1989), que consideraram o problema das associações entre as observações e marcos para realizar a localização do robô. O ambiente neste problema é bidimensional, constituído pelo plano do solo ortogonal às linhas verticais que são observadas. O sistema apresentado por Kriegman, Triendl e Binford (1989) também trabalha com linhas verticais, porém este é um exemplo de um método onde ocorre um mapeamento destes marcos, e não apenas o rastreamento da câmera. Este sistema também utiliza imagens de uma armação estereoscópica, e não de uma única câmera.

Algumas outras aplicações de visão que envolvem retas verticais ou no plano horizontal que podem ser citadas foram demonstradas por Ohya, Kosaka e Kak (1998), Arras e Tomatis (1999), Aider, Hoppenot e Colle (2005), Hile e Borriello (2008) e Barra, Ribeiro e Costa (2009). Um exemplo relativamente recente que destacamos é o sistema apresentado por Kim e Oh (2008). Trata-se de um robô móvel que realiza SLAM a partir de dois sensores: um medidor de distâncias a laser que realiza leituras em um plano paralelo ao solo, e ainda uma câmera unidirecional. A câmera é utilizada apenas para se observar retas verticais do ambiente, que na imagem são aproximadamente retas que emanam do centro da imagem, e são portanto simples de analisar apesar da forte distorção existente nas imagens. Os dados obtidos com o laser complementam a análise de imagens produzindo as distâncias das paredes que se assume ligarem as linhas verticais. O sistema exemplifica assim as dificuldades existentes na análise de imagens distorcidas. Uma extração de retas pode ser facilmente empregada para a obtenção alguns dados, mas a análise das linhas horizontais do ambiente, por exemplo, foi descartada pela sua maior complexidade. Outro exemplo recente de uma aplicação de visão monocular com um arranjo catadióptrico que utiliza apenas linhas verticais foi apresentado por Wongphati, Niparnan e Sudsang (2008).

Linhos tem sido empregadas também em sistemas de SLAM monocular em tempo real mais genéricos, com modelos e movimentos de câmera irrestritos no espaço tridimensional. Este é o caso dos métodos apresentados por Eade e Drummond (2006), Gee e Mayol-Cuevas (2006), Smith, Reid e Davison (2006) e Chao et al. (2008). Linhas apresen-

tam uma alternativa interessante para aplicações de tempo real pela possibilidade de se rastrear uma linha através de buscas unidimensionais por bordas em direções ortogonais às direções aparentes estimadas dos marcos. O método criado por Eade e Drummond (2006) se destaca pelo fato de que os marcos utilizados não são linhas de fato, mas edgels sem comprimento. Aqueles autores denominam *edgelet* cada um destes marcos contidos no mapa, e *edgel* as suas observações retiradas da imagem. Outros métodos de SLAM em tempo real baseados em linhas do ambiente utilizam modelos constituídos por segmentos de linha, linhas infinitas, ou ainda linhas limitadas por cantos que são detectados com técnicas de visão através de pontos.

Uma outra técnica de SLAM visual que se pode destacar pela relação com o tema desta tese foi desenvolvida por Lebegue e Aggarwal (1993). Trata-se de um robô móvel que realiza SLAM visual, porém auxiliado ainda por um inclinômetro e um hodômetro. Como o robô sempre possui uma boa estimativa inicial de sua orientação no ambiente, é possível explorar isto durante a análise da imagem. Em primeiro lugar os pixels são descartados de acordo com sua intensidade de gradiente. A seguir calcula-se para cada ponto a direção predita para uma reta que apontasse para cada uma das três possíveis direções no ambiente. Se o gradiente de um pixel aponta a uma direção aproximadamente ortogonal à direção de algum dos pontos de fuga, o pixel é associado àquela direção, e é então ativado em uma imagem que contém as bordas detectadas para aquela classe específica. Desta forma realiza-se uma detecção de bordas com classificação de direções. O processo segue com uma extração de segmentos de reta, porém cada direção é analisada separadamente.

Depois das retas serem extraídas, a posição estimada dos pontos de fuga encontrados de cada grupo é utilizada então para refinar a estimativa da orientação do robô com relação ao referencial natural do ambiente. Para isto calculam-se explicitamente as coordenadas dos pontos de fuga sobre a imagem, e depois os ângulos da orientação com relação a eles. Este passo de processamento é bastante rudimentar se comparado às alternativas, porém o que é mais interessante na técnica é o fato de que a informação de orientação da câmera existente *a priori* é utilizada durante o processo de análise da imagem e extração de evidências.

Existem vários exemplos de sistemas em que a orientação da câmera com relação ao referencial natural do ambiente é encontrada para realizar alguma tarefa. A condução de veículos ou robôs móveis é uma aplicação com bastante potencial (SIMOND; RIVES, 2003; SCHINDLER; DELLAERT, 2004; NIETO; SALGADO, 2007). Outra importante aplicação é utilizar a estimativa de orientação para tornar mais simples um problema de BA que contenha muitas câmeras e marcos (ANTONE; TELLER, 2002; SINHA; STEEDLY; SZE-LISKI, 2010).

Todas as aplicações de linhas para visão citadas até aqui envolvem apenas a localização de câmera ou a reconstrução do ambiente, mas linhas retas também são muito

importantes para a calibração, especialmente no caso em que uma única imagem está disponível. A determinação de parâmetros da distorção de lente em um modelo de distorção radial pode ser realizada com base no fato de que linhas retas projetadas na imagem também deveriam ser retas no caso do modelo *pinhole*, e qualquer curvatura encontrada deve ser efeito da distorção (THORMAHLEN; BROSZIO; WASSERMANN, 2003; ROSTEN; LOVELAND, 2009). Este fato é conhecido como *restrição de fio-de-prumo* na área de calibração. Enquanto a princípio esta restrição basta para encontrar os parâmetros de distorção, isolado de qualquer análise envolvendo a geometria espacial, a determinação de pontos de fuga da imagem pode auxiliar no processo (GONZALEZ-AGUILERA; GOMEZ-LAHOZ; RODRÍGUEZ-GONZÁLVEZ, 2011).

O método de Rosten e Loveland (2009) apresenta uma alternativa interessante ao problema da calibração de distorção pela condição de fio-de-prumo por ser baseado em edgels, e portanto evitar a extração de linhas. Não existe entretanto nenhuma exploração de características do ambiente neste método. O valor minimizado é uma entropia relacionada ao alinhamento entre os diversos edgels dentro da imagem. É bom notar que este método também calcula o erro no espaço retificado, sem distorções. O Corisco compartilha com este método o uso de edgels e a capacidade de lidar com distorções, mas se diferencia por calcular o erro no espaço da imagem original, distorcida.

A necessidade extrair curvas da imagem, e não retas, pode ser uma grande inconveniência para estas técnicas de calibração que utilizam a restrição de fio-de-prumo, e que não são baseadas em edgels. A projeção distorcida das retas do ambiente produz curvas nas imagens, e não retas, e por isso não é de se esperar que técnicas tradicionais de extração de retas possam fornecer o mesmo desempenho nestas imagens do que o habitual. Mas como consequência da dificuldade de um processo de extração de curvas mais adequado e genérico, muitos autores acabam optando por empregar métodos tradicionais de extração de retas. O resultado é geralmente a extração de pequenos segmentos de reta ao longo de cada curva, e o erro no alinhamento entre as bordas detectadas inicialmente e os modelos de segmento de reta criados a partir delas deve ser minimizado durante a calibração. Estes segmentos de reta também podem ser ligados entre si para a formação de curvas mais extensas.

Enquanto a restrição de fio-de-prumo permite lidar com distorções, a extração de pontos de fuga junto da restrição de ortogonalidade entre as direções das retas do ambiente permite realizar ainda a calibração de parâmetros intrínsecos da câmera como a distância focal e ponto principal. A realização de calibração a partir de retas e pontos de fuga conhecidos foi demonstrada inicialmente nas pesquisas de Caprile e Torre (1990) e Cipolla, Drummond e Robertson (1999).

Um processo mais automatizado foi demonstrado por Grammatikopoulos, Karras e Petsa (2007). Este último método inicia com uma extração de linhas e estimativa de distorção utilizando um método como os citados anteriormente (THORMAHLEN; BROSZIO;

WASSERMANN, 2003). Os pontos de fuga mutuamente ortogonais são então encontrados com a técnica de Rother (2002). O processo de estimativa dos pontos de fuga resulta também na classificação dos segmentos de reta. O processo continua então com uma otimização para a minimização do erro quadrático entre as bordas detectadas e modelos de reta ajustados a cada grupo de bordas. As retas são restringidas pelo ponto de fuga associado. A cada passo da otimização as bordas são analisadas novamente e re-agrupadas, e novas retas são ajustadas. Estas retas são sempre encontradas mantendo-se a restrição de que devem passar pelo seu ponto de fuga. O processo completo é interessante pela qualidade que busca alcançar, porém é bastante intrincado, e depende de repetidas execuções de algoritmos como o RANSAC para o re-agrupamento de bordas, e isto fatalmente resulta em um processo bastante custoso e demorado.

É importante perceber que em todos os métodos citados aqui até agora as distorções da imagem ou são ignoradas, ou são removidas através de algum processo de retificação. Ou seja, os métodos assumem que as imagens possuem uma projeção pontual perfeita, ou então realizam uma retificação em que se utiliza um modelo da distorção para mapear os pontos detectados de suas coordenadas na imagem original para as coordenadas correspondentes de uma projeção pontual ideal. O sistema de SLAM visual desenvolvido por Bosse et al. (2002) se diferencia um pouco disto porque as bordas detectadas não são mapeadas para o plano, mas sim utilizadas para calcular a direção de raios, que são então utilizados para realizar uma extração de linhas parametrizadas por um vetor normal ao seu plano.

Apesar de pouco comuns, podemos citar ao menos dois exemplos de métodos propostos para calibração através do ajuste de modelos de curvas às posições das bordas detectadas na imagem original distorcida. O primeiro seria o método proposto por Strand e Hayman (2005), onde utiliza-se para o modelo de divisão da Equação 2.5. Esta distorção faz com que as linhas retas sejam projetadas em círculos, e os autores realizam então um ajuste destas entidades geométricas nas bordas detectadas sobre a imagem para extrair as projeções das retas. Os círculos extraídos são então utilizados para calcular o parâmetro  $\kappa$  do modelo de distorção. Um método similar de calibração foi aplicado a imagens com projeção equiretangular por Hughes et al. (2010).

O segundo método com análises de curvas que destacamos é o proposto por Barreto e Araujo (2005) para a calibração de arranjos catadióptricos. Os modelos de câmera considerados naquela pesquisa fazem com que retas produzam curvas cônicas genéricas sobre a imagem, e os autores propõem portanto a extração destas curvas, valendo-se de algumas restrições para melhorar a qualidade dos resultados. Tanto esta pesquisa quanto a citada anteriormente apresentam resultados em que tarefas conduzidas a partir de análises nas imagens originais distorcidas superaram métodos baseados em imagens retificadas.

O Corisco compartilha com estes dois últimos métodos citados o desejo em evitar realizar retificações de imagem ou mesmo qualquer outro tipo de mapeamento das carac-

terísticas geométricas extraídas das imagens para espaços considerados ideais como o de um modelo de câmera *pinhole* ou a esfera Gaussiana. Tanto nestes métodos quanto no *Corisco* as análises ocorrem todas no próprio espaço da imagem original, distorcida. Isto geralmente beneficia a precisão dos resultados alcançados por qualquer método, e permite ainda trabalhar com imagens com projeções como a equidistante e a equiretangular, como já foi dito anteriormente.

### 3.2.3 A busca por estrutura

Como já foi mencionado anteriormente, o uso de linhas em aplicações de visão não é vantajoso apenas por trazer benefícios relacionados ao desempenho dos métodos, por exemplo. Esta abordagem pode ser necessária para viabilizar diversos tipos de aplicação, porque é mais fácil atribuir significado a entidades de mapas construídos com linhas do que com pontos, já que elas podem representar contornos de objetos ou regiões de interesse do espaço com mais facilidade.

O desejo em produzir mapas mais ricos em conteúdo semântico do que os oferecidos pelas técnicas baseadas em pontos é bem representado pelo trabalho desenvolvido por alguns pesquisadores. Gee et al. (2008) demonstra como ajustar estruturas de maior grau de abstração como linhas e planos a pontos que são produzidos de forma inicialmente independente por um sistema de SLAM monocular em tempo real mais convencional. Conforme os pontos são associados a estas estruturas, a sua forma de representação é modificada, assim como a forma com que os parâmetros do modelo do ambiente são atualizados ao longo do tempo. A técnica de percepção, contudo, permanece estritamente baseada em pontos, e o sistema continua dependendo da existência de regiões altamente texturizadas ao longo das paredes e linhas para que elas possam ser percebidas. Ou seja, deseja-se encontrar linhas e planos no espaço, porém não se busca detectar este tipo de estrutura nas imagens diretamente.

Um exemplo em que há modificação na forma de análise das imagens é o trabalho de Flint et al. (2010). O sistema funciona a princípio como um SLAM monocular baseado em pontos, convencional. Porém, ao longo do tempo realiza-se uma extração de retas seguida de extração dos pontos de fuga, e consequente estimativa da orientação do referencial canônico do ambiente com relação à orientação estimada da câmera. O sistema trabalha com seu próprio referencial, e apenas calcula a orientação do referencial natural do ambiente com relação a ele ao longo do tempo.

O *Corisco* busca contribuir para o desenvolvimento de sistemas de visão direcionados a ambientes que possuem uma certa estrutura, como ambientes antrópicos. O método busca oferecer um bom desempenho tanto em termos de custo computacional quanto precisão, com uma possibilidade de priorizar um ou outro. A análise que é feita inicialmente da imagem também é relativamente simples, não requerendo por exemplo a extração de retas. Desta forma o *Corisco* é capaz de entregar uma informação de alto nível a respeito

do ambiente, que é a orientação de seu referencial natural em relação à câmera, sem realizar nenhuma tarefa secundária mais complexa tal como o rastreamento de câmera, a reconstrução do ambiente seguida da extração de planos do mapa resultante ou mesmo uma extração de linhas, como é o caso nestes outros métodos citados nesta seção. O *Corisco* seria portanto um método bastante atraente, devido a seu funcionamento relativamente simples, para ser utilizado nos primeiros estágios de uma análise de imagens em um sistema de visão maior que explore a existência de um sistema referencial natural em um ambiente para realizar tarefas mais complexas.

A Seção 3.3 a seguir discute toda a família de métodos similares ao *Corisco* que o precederam, e que assim como ele se baseiam em uma única imagem e dispensam a extração de retas ou a reconstrução do ambiente para encontrar a orientação da câmera.

### **3.3 Estimação de orientação a partir de edgels e comparações com o Corisco**

O *Corisco* pertence a uma família de outros métodos que foram desenvolvidos nos últimos 10 anos que se baseiam em uma mesma técnica mais geral para a estimação de orientação de câmera em ambientes antrópicos a partir de edgels, dispensando a extração de linhas. O trabalho pioneiro nesta área foi o realizado por Coughlan e Yuille (1999, 2003). O problema é encarado do ponto de vista da teoria de probabilidades, com a definição de uma expressão que calcula a verossimilhança de um conjunto de observações em função dos parâmetros de um modelo. Nesta técnica as observações são obtidas dos vetores gradiente em cada pixel da imagem, e os parâmetros são os que modelam a orientação da câmera com relação ao referencial canônico, além dos parâmetros intrínsecos da câmera. No caso da estimação de orientação, apenas os parâmetros da orientação são procurados, enquanto no caso da calibração os parâmetros do modelo de câmera também são variados junto dos da orientação durante a busca por um ponto mínimo da função objetivo.

A partir da definição deste modelo probabilístico é possível obter uma expressão para um estimador MAP, como descrito na Seção 2.4.1, que permite estimar a orientação de câmera a partir do gradiente da imagem. Um processo de otimização é então empregado para encontrar a orientação com maior probabilidade *a posteriori*. O método de otimização empregado por Coughlan e Yuille (2003) é bastante simples. A função é amostrada em regiões onde se acredita estar a solução e o resultado é encontrado por um simples processo de busca determinística que percorre o espaço de soluções possíveis.

A expressão de verossimilhança utilizada no método apresentado por Coughlan e Yuille (2003) é

$$p(\mathbf{u}_n | \Psi, \mathbf{p}_n) = \sum_k p(\mathbf{u}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k), \quad (3.1)$$

onde o vetor  $\mathbf{u}_n$  é o gradiente da intensidade luminosa da imagem sobre o pixel de índice  $n$ , localizado na posição  $\mathbf{p}_n$  da imagem. Este vetor possui norma  $|\mathbf{u}_n|$  e direção, ou ângulo,  $\angle \mathbf{u}_n$ . O vetor  $\Psi$  contém os parâmetros da orientação da câmera, que no trabalho daqueles

autores eram os ângulos de Euler, a serem descritos na Seção 4.3. A variável  $c_n$  diz qual é a classe de cada um dos pixels, e esta é a variável desconhecida sobre a qual é realizada uma marginalização para produzir a função de esperança da verossimilhança em que se baseia a função objetivo que deve ser minimizada. Cada pixel pode pertencer a uma de cinco classes: ou o pixel não é uma borda, ou é uma borda alinhada a uma das três direções do referencial natural do ambiente, ou o pixel é uma borda que não está alinhada a alguma destas direções.

No caso dos pixels que não são uma borda, e dos que são bordas não-alinhadas, a verossimilhança depende apenas de  $|\mathbf{u}|$ . Os valores da função  $p(|\mathbf{u}| | c)$  para as classes borda e não-borda utilizados por Coughlan e Yuille (2003) foram amostrados de imagens reais, em um experimento anterior a respeito de extração de bordas. Já para as probabilidades de  $p(\angle \mathbf{u} | \Psi, \mathbf{p}, c)$  foi utilizado um modelo mais simples, de uma função caixote. Para calcular esta função é preciso primeiro utilizar  $\Psi$  e  $\mathbf{p}_n$  para calcular uma direção predita para aquela borda  $n$  específica, além da direção no ambiente ditada pela classe  $c_n$ . O ângulo desta direção predita é então subtraído de  $\angle \mathbf{u}$ , e a distribuição de probabilidades é definida ao redor deste resíduo. Assim a Equação 3.2 pode ser reescrita como

$$p(\mathbf{u}_n | \Psi, \mathbf{p}_n) = \sum_k p(|\mathbf{u}| | c_n = k) p(\angle \mathbf{u}_n - f(\Psi, \mathbf{p}_n) | c_n = k) p(c_n = k), \quad (3.2)$$

onde a função  $f(\Psi, \mathbf{p}, k)$  realiza o cálculo do ângulo da projeção na posição  $\mathbf{p}$  da imagem da direção  $k$  do ambiente, com a câmera orientada segundo  $\Psi$ . Esta função  $f$  utiliza o modelo de câmera, e é baseada na Equação 2.18. É importante lembrar que o vetor gradiente medido,  $\mathbf{u}$ , é ortogonal ao vetor  $\mathbf{v}$  calculado pela Equação 2.18.

O Capítulo 4 discute como esta função de verossimilhança é utilizada para definir a função objetivo que é minimizada neste e nos outros métodos que seguem esta mesma técnica para o problema da estimativa de orientação. A forma como a função de verossimilhança é calculada para cada observação é uma das principais maneiras como estes diferentes métodos se diferenciam.

Este trabalho pioneiro de Coughlan e Yuille (1999, 2003) foi seguido pela pesquisa de Deutscher, Isard e MacCormick (2002), que demonstraram a possibilidade de realizar calibração de câmera utilizando a mesma forma de estimativa por MAP. Além da orientação da câmera, aquele método também estima a distância focal de imagens individuais. Outra modificação relevante naquela proposta é a utilização de uma busca estocástica como método de otimização para a função objetivo. Também foram utilizados quaternions no lugar de ângulos de Euler para parametrizar as orientações.

A próxima evolução ocorrida no desenvolvimento desta técnica foi a pesquisa de Schindler e Dellaert (2004). Estes autores propuseram o uso do algoritmo EM para realizar a estimativa de parâmetros, substituindo a estimativa por MAP. Naquele método proposto também foram utilizadas funções de densidade de probabilidade Gaussiana para

tentar simplificar o cálculo da função objetivo. A otimização daquele método é realizada através do algoritmo de Levenberg-Marquardt, utilizando ainda um algoritmo de diferenciação automática para calcular todas as derivadas necessárias. Uma das características mais interessantes do método de Schindler e Dellaert (2004) é a possibilidade de se poder levar em conta múltiplas direções possíveis para as retas do ambiente. A existência de três direções principais mutuamente ortogonais é utilizada para iniciar o processo, e novas direções são detectadas e estimadas ao longo do tempo.

A troca da técnica de estimação da MAP para a EM significa que a mesma fórmula da Equação 3.2 é utilizada no processo de estimação, porém os valores de  $p(c_n = k)$  não são mais constantes, e sim variáveis modificadas durante o funcionamento do algoritmo EM. A aplicação do EM também seria um passo inicial para uma futura adaptação da técnica para a utilização de um modelo probabilístico baseado em campos aleatórios de Markov.

Os autores daquele trabalho demonstraram também a possibilidade de se determinar a distância focal com seu método, como no trabalho de Deutscher, Isard e MacCormick (2002). Também foi demonstrada a possibilidade de utilizar o método para estimar a orientação de um robô móvel de forma sequencial, atualizando uma estimativa da orientação ao longo do tempo. O artigo ainda menciona a possibilidade de se levar em conta distorções de imagem, o que porém não foi demonstrado.

O trabalho mais recente nesta área, anterior a esta tese, é o de Denis, Elder e Estrada (2008). A modificação mais importante introduzida naquela proposta é que não se utiliza de forma direta todos os pontos do gradiente da imagem como observação. Um método sofisticado de detecção de bordas com precisão de sub-pixel é utilizado primeiro para determinar quais pixels são parte de linhas existentes na imagem. Estes pixels são os únicos considerados no restante do processo. Como todas as observações consideradas são assumidas como sendo bordas de fato o modelo empregado pode se simplificado, passando a depender apenas da direção do gradiente em cada observação. Ou seja, a verossimilhança da orientação através de uma única observação é calculada a partir da fórmula

$$p(\angle \mathbf{u}_n | \Psi, \mathbf{p}_n) = \sum_k p(\angle \mathbf{u}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k). \quad (3.3)$$

Como a informação da intensidade do gradiente é descartada neste método, e apenas algumas poucas observações medidas de forma mais precisa são utilizadas, este processo pode ser facilmente enxergado como uma forma de extração de edgels.

Outra diferença deste método comparado com as primeiras propostas é que foi empregado um modelo mais sofisticado para a função de densidade de probabilidade das direções medidas dos edgels em função das direções corretas das projeções das retas do ambiente. Foi empregada uma mistura das funções Laplaciana generalizada e uniforme para modelar os efeitos de outliers. Os modelos utilizados anteriormente foram a distribuição Gaussiana (SCHINDLER; DELLAERT, 2004) e uma simples função caixote, uni-

forme (DEUTSCHER; ISARD; MACCORMICK, 2002; COUGHLAN; YUILLE, 2003). Uma outra modificação relevante de Denis, Elder e Estrada (2008) é que foram testadas várias formas de otimização, em especial utilizando o algoritmo de otimização BFGS, da classe de algoritmos Quasi-Newton.

Um último detalhe relevante do trabalho de Denis, Elder e Estrada (2008) é que a única forma de sub-amostragem empregada é a detecção de bordas, já que todas as bordas detectadas são utilizadas. Apesar de Coughlan e Yuille (2003) proporem a análise de imagens completas, pesquisas subsequentes sempre contaram com alguma forma de sub-amostragem para reduzir o volume de dados e acelerar os cálculos, buscando sempre manter a qualidade das estimativas. Deutscher, Isard e MacCormick (2002) realizam a sub-amostragem sorteando observações dentro de ladrilhos da imagem, buscando assim realizar uma amostragem mais homogênea ao longo da imagem. Esta técnica de amostragem aleatória combina adequadamente com o método de otimização por busca estocástica que foi empregado por aqueles autores. Já Schindler e Dellaert (2004) não deixam muito claro a forma como foi realizada a sub-amostragem, se houve um ladrilhamento ou se novas observações eram sorteadas a cada nova iteração da otimização, mas certamente foi realizada uma sub-amostragem. Naquele artigo também se demonstra a possibilidade de realizar uma limiarização inicial das observações pela magnitude do gradiente, introduzindo assim uma forma primitiva de detecção de bordas no método.

Um último aspecto relevante ao problema da estimação de orientação a partir de edgels, mas pouco explorado nestas quatro métodos propostos é a questão da inicialização do processo de otimização. Enquanto técnicas para a extração de pontos de fuga baseadas em linhas e em algoritmos como o RANSAC funcionam de forma direta, sem depender de estimativas iniciais, os processos de otimização utilizados por Denis, Elder e Estrada (2008) e por Schindler e Dellaert (2004) dependem bastante da existência de uma boa estimativa inicial da orientação. Isto se deve às características da função objetivo utilizada, que fatalmente contará com mínimos locais e outras características que podem prejudicar o desempenho do processo de otimização. Uma implementação mais cuidadosa deste tipo de estimação deve necessariamente contar com alguma forma de cálculo de soluções iniciais aproximadas para serem a seguir aperfeiçoadas através dos processos de otimização contínua.

Como será detalhado na próxima seção e no Capítulo 4, o Corisco se diferencia destes quatro métodos propostos das seguintes maneiras:

- Em primeiro lugar, as observações utilizadas são edgels extraídos com uma máscara em forma de grade, enquanto os métodos anteriores utilizaram ou todo o gradiente da imagem, ou todas as bordas detectadas, ou amostragens aleatórias.
- A função objetivo definida para a estimação é baseada na técnica de M-estimação ao invés das técnicas MAP ou EM. A mesma função objetivo pode também ser

facilmente adaptada para trabalhar com qualquer modelo de câmera. O *Corisco* também evita o uso de funções cujo cálculo é muito custoso, como as funções trigonométricas e exponenciais.

- E por fim, a minimização da função objetivo é realizada no *Corisco* por um processo que inicia com uma busca aleatória baseada no algoritmo RANSAC e termina com uma otimização contínua baseada no algoritmo FilterSQP, utilizando derivadas calculadas com fórmulas fechadas e ainda caminhando pelo espaço quadridimensional de quaternions, que são utilizados para parametrizar a orientação da câmera. Os outros métodos utilizaram ou formas mais rudimentares de otimização por buscas estocásticas ou determinísticas, ou ainda algoritmos de otimização contínua baseadas em equações com cálculo relativamente custoso.

### 3.3.1 *O Corisco comparado a técnicas diretamente relacionadas*

Os métodos de Coughlan e Yuille (2003) e de Deutscher, Isard e MacCormick (2002) deram origem à técnica de estimação de orientação utilizada no *Corisco*, baseada em inferência probabilística, e demonstram seu potencial. Porém estes primeiros métodos demonstrados foram baseados em processos muito rudimentares de otimização. Isto foi melhorado a seguir por Schindler e Dellaert (2004) e depois por Denis, Elder e Estrada (2008). Como foi dito na seção anterior, o *Corisco* se diferencia dos métodos propostos anteriormente de três maneiras principais.

A primeira diferença é com relação à forma como são extraídas as observações utilizadas no *Corisco*. Os primeiro método proposto por Coughlan e Yuille (2003) utiliza simplesmente todos os vetores do gradiente sobre cada pixel da imagem. Isto resulta em uma quantidade considerável de dados que precisam ser reiteradamente analisados durante o processo de otimização, e portanto um método de sub-amostragem aleatória dos dados foi introduzido por Deutscher, Isard e MacCormick (2002). Neste segundo método argumenta-se que a sub-amostragem também ajuda a suavizar a função objetivo.

Na pesquisa de Schindler e Dellaert (2004) também foi utilizada alguma forma de sub-amostragem, apesar do procedimento exato não ter sido detalhado por estes autores, mas eles ainda argumentam que além do benefício da redução de dados a sub-amostragem também ajuda a evitar a existência de variáveis com fortes correlações. Em geral é benéfico para um processo de estimação probabilística que a independência que se assume existir entre as variáveis do problema seja realmente constatada na prática. Estes mesmos pesquisadores também realizaram uma remoção dos dados de bordas detectadas com gradientes muito baixos, mas ainda assim mantiveram a intensidade do gradiente dentro do modelo probabilístico.

Na proposta original de Coughlan e Yuille (2003) a intenção era poder resolver o problema da estimação de orientação e da detecção de bordas simultaneamente. Apesar

da ideia ser bastante interessante, o problema da detecção de bordas não é na realidade muito desafiador, e na prática acaba sendo melhor realizar portanto algum tipo de limiarização como feito por Schindler e Dellaert (2004), já que isto ajuda muito a reduzir o volume de dados que precisam ser analisados, mas sem causar impacto na qualidade das soluções encontradas. Ao invés de realizar apenas detecção de bordas simultaneamente à estimativa de orientação, seria mais interessante tentar criar um processo em que ocorre uma extração de retas junto da estimativa de orientação, que seria justamente o objetivo da sugestão de Schindler e Dellaert (2004) de aplicar o modelo de campos aleatórios de Markov para este problema.

No método proposto por Denis, Elder e Estrada (2008) foi finalmente aplicada uma técnica mais sofisticada de detecção de bordas para o problema, substituindo tanto este uso da limiarização quanto a própria modelagem da norma do gradiente dentro do processo de estimativa, utilizando assim a Equação 3.3 ao invés da Equação 3.2 como base para definir a função de erro da estimativa. Não só este método remove observações que não são bordas dos dados, como o cálculo da posição sobre a imagem destes pontos foi realizado com precisão de sub-pixel. O benefício desta melhoria na medição não é necessariamente notável, mas é importante por demonstrar claramente uma mudança de paradigma comparado com a ideia de simplesmente considerar o valor do gradiente amostrado sobre cada pixel da imagem como alvo de análise. Porque os pontos podem agora se localizados em qualquer lugar da imagem além das coordenadas dos pixels, em posições que são calculadas por um algoritmo de extração, este método não se baseia mais em uma simples detecção de bordas ou amostragem do gradiente da função. Trata-se de uma extração de características geométricas.

No *Corisco* esta proposta de Denis, Elder e Estrada (2008) foi levada adiante, com a utilização de um método que efetivamente realiza uma extração de características geométricas, que são os edgels. Esta extração de edgels utilizada no *Corisco* foi inspirada em processos existentes em alguns trabalhos recentes, principalmente no sistema de SLAM em tempo real de Eade e Drummond (2006), e é importante notar que são processos diferentes de uma simples detecção de bordas em toda a imagem, porque não há uma preocupação em manter-se conjuntos contínuos de bordas detectadas na saída do processo, por exemplo, para que estas observações sejam eventualmente analisada por um seguidor de contornos, por exemplo. Nestes processos de extração de edgels estas entidades são realmente o que se deseja extrair.

No extrator de edgels do *Corisco*, descrito na Seção 4.1 à frente, a imagem é subamostrada desde o princípio pelo uso de uma máscara em forma de grade que seleciona um conjunto restrito de linhas e colunas que serão analisadas. A varredura das linhas e colunas visa causar uma amostragem regular das linhas e curvas contidas na imagem, mesmo que com um período de amostragem que varia conforme a direção destas linhas. A máscara em forma de grade também garante uma amostragem uniforme sobre a área da

imagem, ao contrário do que ocorre com uma amostragem aleatória. Também garante-se que não haverá amostras vizinhas nos dados, atendendo à preocupação levantada por Schindler e Dellaert (2004) de remover da análise conjuntos de variáveis fortemente correlacionadas.

Amostragens aleatórias para a sub-amostragem neste problema não são a forma mais adequada de lidar com linhas devido à baixa probabilidade de se conseguir sortear um pixel localizado sobre uma borda. Esta probabilidade depende da área ao redor de uma linha da imagem onde estas detecções são bem-sucedidas, e teoricamente esta área deveria ser extremamente reduzida, já que uma linha ideal é uma entidade geométrica que não possui espessura. Seria de se esperar, portanto um desempenho cada vez pior desta técnica em imagens de resoluções cada vez maiores. A varredura de uma linha de uma grade na busca por pontos de intersecção com retas e curvas da imagem, por outro lado, é algo perfeitamente aceitável deste ponto de vista teórico já que este processo não depende de nenhuma forma de uma área definida ao redor das linhas para que as detecções sejam bem-sucedidas.

A ideia de substituir uma análise diretamente baseada nos valores do gradiente pela realização de uma extração de edgels pode parecer uma modificação estrutural significativa, mas é na realidade muito fácil passar de uma abordagem para a outra neste problema. Os métodos baseados no gradiente podem ser simplesmente interpretados como métodos que utilizam uma forma sub-ótima de extração de edgels, enquanto do outro ponto de vista a extração de edgels pode simplesmente ser interpretada como uma forma de sub-amostragem. A técnica permite facilmente, desde a sua introdução por Coughlan e Yuille (2003), considerar pontos amostrados de forma completamente desconectada e irregular sobre a área da imagem, apenas o conhecimento das coordenadas  $p$  e do vetor  $u$  para cada observação são necessários e estes valores ainda aparecem sempre de forma explícita em todas as equações.

A segunda diferença principal do *Corisco* comparado aos métodos existentes similares de estimação de orientação está na função objetivo que é utilizada no processo de estimação de parâmetros. Em primeiro lugar, o *Corisco* não utiliza o estimador MAP dos dois primeiros métodos propostos, ou o algoritmo EM dos dois últimos. Ao invés destes métodos o *Corisco* emprega a técnica de M-estimação, discutida na Seção 2.4.2. Para compreender porque esta mudança pode ser feita sem problemas é preciso primeiro suportar a troca do EM pelo chamado EM generalizado, ou GEM. No GEM não existem as etapas separadas  $E$  e  $M$ , e pode ser possível realizar uma estimação das probabilidades de classe ao longo da própria otimização, ou seja, o passo  $E$  ocorre dentro de cada iteração de otimização que constituiria o passo  $M$ .

Ao estudar a aplicação de GEM para funções de erro como as que serão discutidas nas Seções 4.3.1 e 4.3.2, obtidas a partir da Equação 3.3, é possível notar que o processo pode ser interpretado como a minimização de uma função de erro com características

peculiares, que poderia ser substituída por um M-estimador. Do ponto de vista oposto, o algoritmo EM também pode ser visto neste cenário de aplicação como sendo equivalente ao algoritmo IRLS, que é um algoritmo clássico dentro da Estatística Robusta. Esta mudança do uso de EM para o uso de um M-estimador consegue no mínimo simplificar a implementação do método, e permite a experimentação com o cálculo de expressões mais simples para a função de erro, na busca por um melhor desempenho. A aproximação desta área de estudos com a teoria de Estatística Robusta também permite libertar a técnica da preocupação em se conhecer detalhadamente o formato das funções de densidade de probabilidade envolvidas no problema, e trazer o foco para a implementação, permitindo a busca por métodos com melhor desempenho.

Esta troca do EM para a M-estimação afeta como o modelo probabilístico é utilizado na definição da função objetivo, e uma consequência positiva desta mudança é que o uso da M-estimação permite ao *Corisco* evitar de realizar cálculos das funções logaritmo e exponencial, que são utilizadas nos métodos anteriores e que são bastante custosos.

Existe ainda uma segunda modificação no *Corisco* relacionada à função objetivo, que ocorre no cálculo do resíduo entre as direções  $\angle \mathbf{u}$  medidas e preditas pelo modelo geométrico. Este cálculo é algo que não depende nem da escolha do método de estimação nem do próprio modelo probabilístico. Enquanto nos métodos anteriores este resíduo foi sempre calculado com uso da função arco-tangente, o *Corisco* utiliza apenas operações vetoriais, evitando o cálculo desta função bastante custosa. Além deste benefício, esta forma de calcular os resíduos em cada ponto permite ao *Corisco* utilizar qualquer modelo de câmera desejado, bastando para isto fornecer um procedimento que calcula a matriz Jacobiana necessária em cada ponto, utilizada na Equação 2.18. Portanto não só qualquer modelo de câmera pode ser utilizado como o procedimento ainda é exatamente o mesmo para qualquer caso.

A terceira forma como o *Corisco* se diferencia dos métodos similares é na escolha do algoritmo de otimização. O *Corisco* utiliza o algoritmo FilterSQP (FLETCHER; LEYFFER, 2002), que se trata de um algoritmo de otimização com restrições. A função objetivo no *Corisco* é definida utilizando-se todos os quatro valores de um quaternion utilizado para modelar a rotação entre os referenciais da câmera e o canônico, e este algoritmo cuida para que a norma do quaternion seja unitária ao final do processo, o que é uma condição necessária para que o quaternion represente de fato uma rotação no espaço tridimensional.

Fórmulas fechadas para o cálculo do gradiente e da derivada foram deduzidas para serem utilizadas pelo algoritmo de otimização. As expressões necessárias para estes cálculos são relativamente simples de serem implementadas, e envolvem basicamente polinômios, com uma utilização da função raiz quadrada em alguns momentos. As expressões correspondentes seriam bem mais complexas nos casos dos outros métodos existentes, e de fato até hoje foram utilizados apenas métodos de diferenciação numérica (DENIS; ELDER;

ESTRADA, 2008) ou diferenciação automática (SCHINDLER; DELLAERT, 2004) para realizar estes cálculos. As expressões utilizadas no *Corisco* são menos complexas devido ao uso de operações vetoriais no cálculo do resíduo, e também à utilização de quaternions irrestritos, em todo o espaço quadridimensional, para parametrizar a orientação. Um comprometimento necessário é a necessidade de se utilizar um algoritmo de otimização com restrições para resolver o problema, porém a restrição de norma unitária do quaternion é bastante simples de tratar, e o algoritmo FilterSQP apresentou um desempenho bastante satisfatório.

O FilterSQP é um método de otimização contínua, e no *Corisco* ele produz a estimativa final. Este é o correlato da aplicação dos algoritmos de Levenberg-Marquadt (SCHINDLER; DELLAERT, 2004) e BFGS (DENIS; ELDER; ESTRADA, 2008) em métodos anteriores. Porém o *Corisco* também utiliza opcionalmente um outro método de otimização baseado em uma busca aleatória como a que é realizada no algoritmo RANSAC.

As funções objetivo utilizadas em todos estes métodos apresentam um certo desafio para os algoritmos de otimização contínua, porque elas não permitem uma boa convergência global, principalmente devido à existência de mínimos locais. Desta forma é bastante desejável que se utilize um método inicial de estimação da orientação que produza uma solução mais próxima da ideal, permitindo que os algoritmos de otimização contínua funcionem de maneira adequada. Em todos estes métodos foram empregadas técnicas relativamente rústicas para isto, e nenhum deles apresenta uma solução realmente completa pra o problema da estimação de orientação. Por exemplo, Denis, Elder e Estrada (2008) assumem que a câmera está aproximadamente alinhada com o eixo vertical, e então a função objetivo é amostrada em rotações ao redor deste eixo para encontrar uma estimativa inicial da solução.

No *Corisco* foi utilizada uma busca aleatória que funciona como o algoritmo RANSAC para procurar por uma estimativa inicial da solução. Grupos de três observações são sorteados aleatoriamente para produzir uma hipótese de orientação, e o valor da função objetivo é calculado para cada uma destas hipóteses. A hipótese que produz o menor valor é mantida como a estimativa inicial. Este procedimento permite analisar qualquer imagem de entrada sem nenhum tipo de assunção inicial sobre a orientação da câmera, dotando o processo total de boas características de convergência global, complementando assim o passo final de otimização contínua. O número de hipóteses analisadas nesta busca aleatória cria um comprometimento natural entre o custo computacional e a qualidade obtida para esta estimativa inicial.

### 3.3.2 *O Corisco considerado em um contexto mais amplo*

A Tabela 3.1 contém uma lista de métodos mais relevantes citados nesta tese, além de uma breve descrição de algumas de suas principais características para auxiliar na comparação. Vamos a seguir ressaltar as principais semelhanças e diferenças entre cada método com o

*Corisco*, proposto nesta tese. O símbolo  $\circ$  indica que o método daquela linha apresenta a propriedade da coluna correspondente. O símbolo  $\times$  indica que o método não apresenta a propriedade.

Os primeiros métodos listados na Tabela 3.1 são os métodos discutidos na seção anterior, similares ao *Corisco* (COUGHLAN; YUILLE, 2003; DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008). Todos exploram a restrição de ambiente antrópico para estimar a orientação da câmera, o que é indicado pela coluna *Amb. Antrópico*. Os métodos ainda se diferenciam nas técnicas de otimização e de sub-amostragem empregadas, mas compartilham o fato de que o erro que é medido e minimizado se baseia na direção do vetor gradiente, como indica a coluna *Variável testada*. No caso do método pesquisado por Denis, Elder e Estrada (2008) foram testadas diferentes abordagens, inclusive a substituição desta medição de erro pelo que denominamos erro de alinhamento espacial anteriormente, que é a projeção do vetor normal  $\mathbf{n}$  associado a cada edgel sobre o vetor unitário na direção  $\mathbf{r}$  das retas do ambiente no referencial da câmera, ou direção do ponto de fuga.

A coluna *Com distorção* indica se foi demonstrado o funcionamento do método com modelos de câmera diferentes do de *pinhole*. Isto não foi o caso com nenhum destes quatro métodos apesar da possibilidade ter sido reconhecida. Já a coluna *Estima  $f$*  indica se foi demonstrada a estimativa de distância focal em cada método. No caso destes quatro métodos antecessores do *Corisco* esta possibilidade foi demonstrada por Deutscher, Isard e MacCormick (2002) e por Schindler e Dellaert (2004), o que não significa que há limitações nas outras propostas que impeçam a realização de calibração, mas apenas se houve alguma demonstração desta possibilidade. Qualquer um destes métodos pode a princípio ser modificado para fazer com que sua otimização não realize apenas a estimativa da orientação mas também de parâmetros intrínsecos. É apenas a estimativa por erro de alinhamento espacial testada por Denis, Elder e Estrada (2008) que pode resultar em uma verdadeira impossibilidade de modificar o método para realizar calibração.

A última linha da Tabela 3.1 traz as características do *Corisco*. Suas principais diferenças comparado a estes seus antecessores são as técnicas de otimização e de sub-amostragem empregadas, além do fato de que o método pode lidar com distorções de lente, ou com qualquer modelo de câmera possível. A natureza das medições de onde se origina o erro que é minimizado é a mesma em todos estes métodos. São direções de vetores localizados sobre a imagem, tal como o gradiente  $\mathbf{u}$ . Porém nos métodos existentes utiliza-se uma diferença entre valores de ângulos de vetores localizados sobre a imagem, enquanto no *Corisco* utiliza-se o seno desta diferença de ângulos.

O quinto e quarto métodos listados na tabela são baseados em técnicas de visão com pontos. O método de Snavely, Seitz e Szeliski (2007) é um exemplo de uma técnica puramente baseada em pontos. Neste método a análise de todas as imagens se inicia com um processo de extração de pontos. Quando há um mapa de pontos disponível ocorre

| Técnica                                    | Com distorção | Estima $f$ | Não extrai retas | Amb. Antrópico | Classificação         | Otimização | Variável testada | Subamostragem   |
|--|---------------|------------|------------------|----------------|-----------------------|------------|------------------|-----------------|
| Coughlan e Yuille (2003)                   | ×             | ×          | ○                | ○              | MAP                   | Busca      | u                | ×               |
| Deutscher, Isard e MacCormick (2002)       | ×             | ○          | ○                | ○              | MAP                   | Est.       | u                | Ladri.<br>Alea. |
| Schindler e Dellaert (2004)                | ×             | ○          | ○                | ○              | EM                    | LM         | u                | Alea.           |
| Denis, Elder e Estrada (2008)              | ×             | ×          | ○                | ○              | EM                    | LM         | u                | ×               |
|  | ×             | ×          | ○                | ○              | EM                    | BFGS       | u                | ×               |
|  | ×             | ×          | ○                | ○              | EM                    | BFGS       | n,r              | ×               |
| Snavely, Seitz e Szeliski (2007)           | ○             | ○          | ○                | ×              | RANSAC                | LM         | p                | -               |
| Sinha, Steedly e Szeliski (2010)           | ×             | ○          | ×                | ○              | RANSAC                | LM         | n,r              | -               |
| Schindler, Krishnamurthy e Dellaert (2006) | ×             | ○          | ×                | ○              | RANSAC                | LM         | p,n              | -               |
| Eade e Drummond (2006)                     | ×             | ×          | ○                | ×              | Rastr.                | FastSLAM   | p,n              | Ladri.          |
| Flint et al. (2010)                        | ×             | ×          | ×                | ○              | EM                    | Grad.      | n,r              | ×               |
| Thormahlen, Broszio e Wassermann (2003)    | ○             | ×          | ×                | ×              | B. estoc.<br>+ RANSAC | LM         | p,n              | ×               |
| Rosten e Loveland (2009)                   | ○             | ×          | ○                | ×              | 1-D Hough             | Est. + NM  | u                | Ladri.          |
| Rother (2002)                              | ×             | ○          | ×                | ○              | ~RANSAC               | LM         | p,n              | ×               |
| Grammatikopoulos, Karras e Petsa (2007)    | ○             | ○          | ×                | ○              | ~RANSAC               | LM         | p,n              | ×               |
| Tardif (2009)                              | ×             | ○          | ×                | ○              | J-link. EM            | misc       | p,n              | ×               |
| Gonzalez-Aguilera e Gomez-Lahoz (2008)     | ○             | ×          | ×                | ×              | RANSAC<br>M-Estim.    | Lsqr       | p,n              | ×               |
| <i>Corisco</i> (método proposto)           | ○             | ○          | ○                | ○              | RANSAC<br>M-Estim.    | FilterSQP  | u                | Grade           |

Tabela 3.1: Tabela comparativa do método proposto, o *Corisco*, e alternativas existentes. O símbolo ○ indica que o método possui a propriedade da coluna correspondente, e o símbolo × indica a ausência da propriedade.

para cada nova imagem analisada apenas uma identificação das observações e uma localização da câmera, além de uma calibração se necessária. Se o mapa não possui pontos o suficiente o sistema resolve o problema completo de mapeamento, localização e calibração a partir de um grande conjunto de imagens. O sistema de Snavely, Seitz e Szeliski (2007) utiliza o RANSAC onde se faz necessário, e ainda é capaz de utilizar modelos de câmera com distorções. Porém é bom deixar claro que, assim como em muitos outros métodos baseados em pontos, há vários processos onde não há uma verdadeira modelagem das distorções. Estes sistemas apenas contam com a robustez de processos tal como a extração de pontos para que não sejam muito afetados pelas distorções.

O método de Snavely, Seitz e Szeliski (2007) é puramente baseado em pontos, por isso a coluna da tabela rotulada *Não extrai retas* traz o símbolo  $\circ$ . Esta é uma semelhança entre este método e os anteriores, porém trata-se de uma técnica de natureza bastante diferente. Aqueles métodos são baseados em edgels ao invés de pontos, e apesar de não realizarem extração de linhas das imagens ainda dependem de um ambiente que contém retas. Já o erro minimizado neste método é o erro de reprojeção dos pontos, portanto sua natureza é a de uma posição sobre a imagem, o que é indicado pelo vetor  $p$  na coluna *Variável testada* da tabela. Este método também depende da análise de múltiplas imagens simultaneamente, e portanto não é monocular como o *Corisco* e outros métodos similares baseados em linhas ou edgels.

O método de Sinha, Steedly e Szeliski (2010) traz algumas similaridades com o de Snavely, Seitz e Szeliski (2007). Ambos métodos analisam grandes conjuntos de imagens para resolver um problema de reconstrução do ambiente, calibração e localização de câmeras simultaneamente, e ambos produzem um modelo final do ambiente que é baseado em pontos. Porém no caso de Sinha, Steedly e Szeliski (2010) foi empregada uma extração de linhas e de pontos de fuga para realizar uma estimativa inicial de orientação da câmera, deixando apenas a translação de cada câmera para ser estimada ao final, além da estrutura do ambiente. Para lidar com lentes com distorção radial o sistema realiza uma retificação da imagem inicialmente, assumindo parâmetros conhecidos e produzindo uma imagem de projeção pontual. Os autores ainda sugerem que poderia ser realizada a calibração das imagens a partir das linhas e pontos de fuga, porém naquele artigo especificamente foi assumido o conhecimento de todos os parâmetros intrínsecos, incluindo parâmetros de distorção.

Como no método de Sinha, Steedly e Szeliski (2010) não há mapeamento de linhas no ambiente, o *Corisco* poderia representar um método mais interessante para fornecer as estimativas iniciais de orientação das câmeras. Não só o *Corisco* é capaz de estimar a orientação sem realizar extração de retas, como ele dispensa a retificação das imagens. Ainda assim o fato de que o método de Sinha, Steedly e Szeliski (2010) não produz um mapa contendo linhas constitui uma forma de desperdício. O sistema apresentado por Schindler, Krishnamurthy e Dellaert (2006) dá um exemplo de como um método de esti-

mação de orientação através de edgels (SCHINDLER; DELLAERT, 2004) pode ser utilizado como base em um sistema maior, e com as mesmas observações sendo utilizadas em todos os níveis. Naquela proposta ocorre inicialmente uma detecção de bordas, que são então classificadas de acordo com a direção de sua reta correspondente no espaço ao mesmo tempo em que ocorre a estimativa da orientação da câmera. São então extraídas retas a partir destas bordas, e o sistema realiza um mapeamento das retas aproveitando a informação de suas orientações, e impondo as restrições de ambiente antrópico quando possível. Na etapa final de localização e reconstrução estes pesquisadores utilizaram uma versão quadrática do que denominamos erro de alinhamento na imagem, baseado na distância entre pontos extremos de segmentos de reta a uma linha que corta o segmento e também o ponto de fuga. Na tabela este erro é indicado pelos vetores  $p, n$ , onde  $p$  indica a dependência nas posições dos pontos extremos, e  $n$  é o modelo da linha obtida para cada segmento para realizar este cálculo do erro.

A Figura 3.1 ilustra as diferentes abordagens para a medição de erros encontradas na Tabela 3.1. As técnicas baseadas em pontos constituem o caso  $p$ , onde utilizam-se distâncias entre pontos localizados sobre a imagem. As técnicas baseadas em edgels constituem o caso  $u$ , onde são analisadas direções medidas sobre o plano da imagem. No erro do tipo  $p, n$  há linhas que modelam diferentes conjuntos de pontos, e o erro se baseia nas distâncias destes pontos até as retas. Um caso especial de interesse é quando os pontos são apenas as extremidades de segmentos de reta. O último caso,  $n, r$ , é o denominado alinhamento espacial, e o mais difícil de ilustrar. Neste caso o erro é baseado a partir dos vetores normais dos planos que definem cada reta, e o plano ortogonal à direção do ponto de fuga sobre o qual todas as normais deveriam se localizar. É interessante observar que  $u$  pode ser aproximado como um erro do tipo  $p, n$  através da definição de pequenos segmentos de reta de tamanhos iguais a partir dos edgels, e também que segmentos de reta mais longos afetam mais uma medição de erros do tipo  $p, n$ . Deve ser notado também que tanto erros do tipo  $u$ ,  $p, n$  quanto  $n, r$  tornariam-se nulos se não houvessem erros de medição.

O oitavo método listado na Tabela 3.1 é o desenvolvido por Eade e Drummond (2006) para SLAM visual. Este método é baseado apenas na extração e rastreamento de edgels, e portanto possui uma relação interessante com o *Corisco* e métodos similares. Porém neste método de Eade e Drummond (2006) não se considera nenhuma restrição nas direções das retas do ambiente. Seria possível unir estes dois métodos para criar um sistema de SLAM visual onde as informações obtidas pelo *Corisco* auxiliariam na estimativa da orientação da câmera e também facilitariam o mapeamento devido à classificação das observações de acordo com suas orientações no ambiente. A restrição dos edgels no ambiente a apenas três direções possíveis beneficiaria ainda o processo por reduzir a dimensionalidade do problema, enquanto a classificação também facilita a associação de dados, como ocorre no método de Schindler, Krishnamurthy e Dellaert (2006).

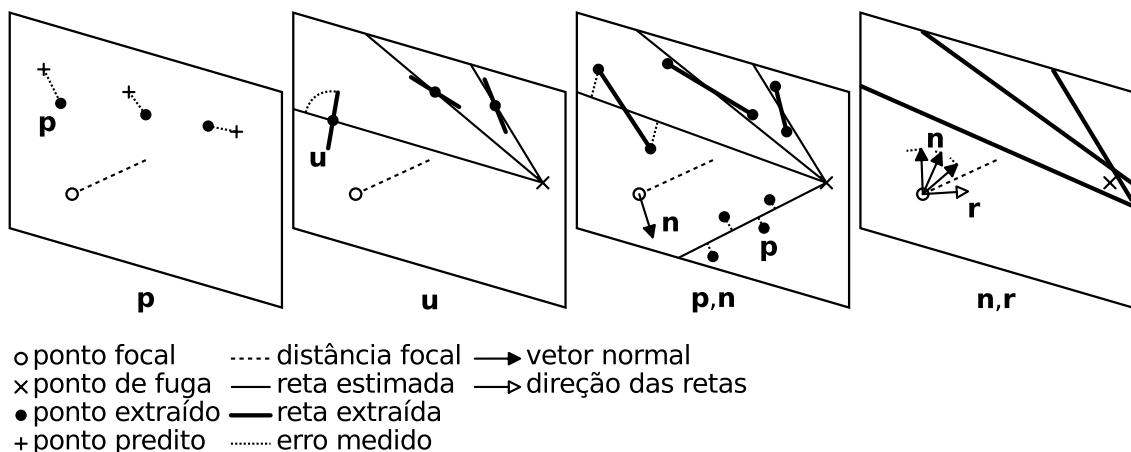


Figura 3.1: Diferentes abordagens para a medição de erros. No primeiro caso os erros são distâncias entre pontos sobre a imagem. No segundo caso os erros são angulares, relacionados a diferenças entre direções medidas sobre diferentes pontos da imagem. No terceiro caso os erros são distâncias entre pontos e retas, incluindo o caso em que há segmentos de retas parametrizados apenas por um par de pontos. O último caso é o alinhamento espacial, que leva em consideração o afastamento dos vetores normais de cada reta do plano tridimensional ao qual deveriam se restringir.

Um exemplo de sistema de SLAM em tempo real, mas que explora a condição de ambiente antrópico, foi apresentado por Flint et al. (2010). Este sistema realiza uma estimativa de orientação das imagens para extrair linhas e criar modelos do ambiente baseados no seu referencial natural. Porém nesta proposta ocorre um processo de rastreamento separado da estimativa de orientação e extração de linhas, e a aplicação da estimativa de orientação não está portanto tão integrada ao restante do sistema quanto poderia ser possível. Como o método proposto por Eade e Drummond (2006) resolve o problema de SLAM apenas com edgels, sem extrair retas a princípio, este talvez o sistema em que se poderia integrar mais facilmente um método como o *Corisco* para produzir um sistema de SLAM em tempo real utilizando apenas de edgels que explore a condição de ambiente antrópico.

Com exceção do trabalho de Snavely, Seitz e Szeliski (2007), que é baseado em pontos, todos os métodos discutidos nesta seção até agora foram desenvolvidos para câmeras com o modelo de pinhole, sem considerar a possibilidade de outros modelos de câmera. O método de calibração apresentado por Thormahlen, Broszio e Wassermann (2003) demonstra como a chamada condição de fio-de-prumo pode ser utilizada junto de um processo de extração de linhas para a estimativa de distorção de câmera no caso de modelos de câmera com distorção radial. Os segmentos de linha que são extraídos são na realidade trechos de curvas, devido à distorção, e a calibração é realizada através de uma otimização que busca retificar as bordas detectadas subjacentes e tornar os segmentos extraídos mais retos. O método depende portanto de um processo de extração de linhas que opera em condições claramente inadequadas, já que as projeções das retas do ambiente são curvas na imagem devido à distorção. É importante notar também que o processo proposto por

Thormahlen, Broszio e Wassermann (2003) mapeia as bordas da imagem para um espaço não-distorcido para calcular o erro, enquanto a princípio seria mais desejável restringir as análises ao espaço da imagem original (TORR, 2000). Este método também não considera a geometria da cena em seu funcionamento, baseando-se portanto estritamente na condição de fio-de-prumo.

Rosten e Loveland (2009) apresentam uma alternativa interessante ao problema da calibração de distorção pela condição de fio-de-prumo por basearem as análises em edgels, e portando evitar o problemas com a extração de linhas ou de curvas da imagem distorcida. Porém este também se trata de um método estritamente baseado na condição de fio-de-prumo, sem nenhuma exploração de características do ambiente. O valor minimizado para a estimação dos parâmetros de distorção é uma entropia relacionada ao alinhamento entre os diversos edgels da imagem. É importante notar que este método também calcula o erro no espaço retificado, sem distorções, assim como no método de Snavely, Seitz e Szeliski (2007), o que não é considerado uma tática ideal.

Este método de Rosten e Loveland (2009) constitui, junto do *Corisco*, os únicos exemplos de métodos baseados em edgels citados nesta tese que levam em consideração imagens distorcidas. Porém o *Corisco* trabalha no próprio espaço da imagem distorcida, e ainda é um método baseado na geometria espacial de um problema de visão mais complexo, enquanto na proposta de Rosten e Loveland (2009) as observações são reiteradamente mapeadas para o espaço da imagem ideal com projeção retilínea, e não se consideram fatores espaciais. O *Corisco* pode ser adaptado para realizar calibração de parâmetros de distorção da câmera, assim como é feito para a estimação da distância focal. Porém assim como acontece com esta outra grandeza a precisão alcançada ainda não muito boa, e para problemas de calibração seria aconselhável utilizar o *Corisco* apenas como um método complementar, junto a alguma outra técnica baseada na extração de linhas, por exemplo, para que resultados de melhor precisão fossem encontrados.

Grammatikopoulos, Karras e Petsa (2007) apresentam um método para calibração que é baseado na extração de retas, seguida pela estimação de pontos de fuga utilizando o método de Rother (2002). Uma vez que as bordas associadas em linhas e classificadas de acordo com seus pontos de fuga estão disponíveis, é realizada uma otimização que estima todos os parâmetros desejados através da minimização do erro de reprojeção das bordas em relação às retas ajustadas. O processo possui uma forma iterativa, com as bordas sendo re-agrupadas a cada nova interação para permitir corrigir erros causados pela distorção que vai sendo gradualmente removida conforme a solução vai sendo aproximada. Apesar de ser um processo que busca estimar todos os parâmetros ao mesmo tempo, as iterações fatalmente o tornam ineficiente, dependendo de reiteradas extrações de linhas e de pontos de fuga. Para lidar com distorções de câmera Grammatikopoulos, Karras e Petsa (2007) simplesmente propõem que se utilize um método como o de Thormahlen, Broszio e Wassermann (2003), criando mais um estágio no processo completo de calibração. Ou

seja, é questionável se o método merece o símbolo  $\circ$  na coluna Com distorção da tabela.

Na prática, o método anterior assume portanto que não há distorções na imagem, e se necessário as distorções existentes devem ser eliminadas previamente. Outro método relativamente moderno de calibração baseada em extração de linhas, e que assume imagens sem distorções, é o proposto por Tardif (2009). Sua característica mais marcante é o uso do algoritmo J-linkage como forma de agrupar as linhas em pontos de fuga e encontrar uma estimativa inicial da solução, que é aperfeiçoada pelo algoritmo EM. A função de erro utilizada é baseada no alinhamento da reta com os pontos de fuga, seguindo a abordagem  $p,n$ . Deve ser observado, no entanto, que nesta técnica ocorre uma extração de segmentos de reta, e as extremidades dos seguimentos são utilizadas nas contas, enquanto que na proposta de Grammatikopoulos, Karras e Petsa (2007) são mantidas todas as bordas detectadas no começo da análise.

Destacamos por fim o método de estimação de parâmetros de distorção através de extração de linhas desenvolvido por Gonzalez-Aguilera e Gomez-Lahoz (2008), o último da Tabela 3.1 antes do *Corisco*. Este método realiza uma extração de pequenos segmentos de reta a partir das curvas da imagem, como os anteriores, que são a seguir agrupados em curvas a partir de suas extremidades. Estes autores exploram tanto a restrição de fio-de-prumo quanto a de ambiente antrópico para realizar calibração e estimar a orientação da câmera. A estimação de parâmetros final é feita através da minimização de um erro do tipo  $p,n$ , porém a extração de pontos de fuga é realizada por um erro do tipo  $n,r$ . Os autores do trabalho não deixam claro, porém, de que forma a estimação dos parâmetros de distorção é efetuada a partir da função de erro definida, se foi empregado algum método de regressão não-linear, ou se houve apenas uma estimação por regressão linear, linearizando-se a função de erro ao redor de uma estimativa inicial onde não há distorção.

Uma característica interessante deste método é o fato de ser empregada uma técnica de estimação robusta para esta extração de pontos de fuga, o que é realizado após a aplicação do algoritmo RANSAC. Esta abordagem com RANSAC seguido de estimação robusta é semelhante à do *Corisco*. O *Corisco* porém se diferencia bastante do trabalho de Gonzalez-Aguilera e Gomez-Lahoz (2008) por não requerer a extração de curvas, e por ser baseado na otimização de uma única função objetivo em todo o seu processo.

### 3.3.3 Retrospectiva

Há portanto dois tipos mais populares de técnicas de visão baseada em linhas no ambiente que levam em consideração distorções de imagem. O primeiro tipo simplesmente utiliza algum processo prévio e separado de retificação da imagem ou mapeamento de bordas detectadas para remover distorções. Ou seja, na realidade estas são técnicas de visão desenvolvidas apenas para o caso do modelo de *pinhole*, e como já foi discutido, isto traz limitações relacionadas ao desempenho dos processos e precisão dos resultados. E ainda por cima é bastante inconveniente ou até impossível utilizar este tipo de abordagem com

projeções tal como a polar equidistante e a equiretangular.

A segunda abordagem mais popular aplica técnicas de extração de retas diretamente às imagens distorcidas (THORMAHLEN; BROSZIO; WASSERMANN, 2003; GONZALEZ-AGUILERA; GOMEZ-LAHOZ, 2008). Esta prática possui algo de contraditório, já que é sabido que as projeções das retas do ambiente na imagem são curvas. Como resultado desta técnica são produzidos pequenos segmentos de reta próximos a trechos das curvas da imagem, que são ou simplesmente utilizados para definir conjuntos de pontos observações que são a seguir retificadas, ou então são agrupados para formar modelos de curvas.

Uma outra abordagem menos utilizada é extrair curvas com técnicas mais apropriadas, como discutido na Seção 3.2.2. A menor popularidade desta abordagem decorre provavelmente da maior complexidade em se implementar extratores de curvas, além do fato de que câmeras com distorções mais intensas são utilizadas apenas por uma minoria de aplicações mais especializadas.

O *Corisco* se destaca neste cenário portanto por ser um método que trabalha diretamente com qualquer modelo de câmera, inclusive com distorções bastante intensas, porém evitando realizar extrações de curvas ou de retas. Isto é uma decorrência do fato do *Corisco* basear suas análises apenas em edgels, e um mérito portando deste tipo de técnica. Mas apesar da possibilidade de se poder utilizar edgels para resolver problemas tal como a estimação de orientação de câmera em ambiente antrópicos levando em consideração modelos de câmera genéricos já ter sido reconhecida no passado, o *Corisco* é a primeira demonstração efetiva disto. Além disso o *Corisco* trabalha no espaço da imagem distorcida, o que apesar de ser uma característica reconhecida por muitos pesquisadores como bastante desejável, é algo raramente realizado, principalmente em técnicas baseadas em linhas ou edgels.

A característica mais peculiar do *Corisco* é o método de extração de edgels com uma máscara em forma de grade, que é algo que foi desenvolvido durante esta pesquisa. Do ponto de vista dos métodos antecessores ao *Corisco* para a estimação de orientação a partir de edgels, esta forma de extração de edgels pode ser vista como uma evolução natural, constituindo apenas uma sub-amostragem realizada de forma determinística dos mesmos dados que são utilizados no método de Denis, Elder e Estrada (2008), que são todas as bordas detectadas na imagem. Ao realizar uma extração mais esparsa destas observações sobre a imagem, o *Corisco* aproxima a técnica da estimação de orientação através de bordas de seus antecessores a outros métodos baseados em edgels que vem sendo desenvolvidos para resolver outros problemas, mais notadamente o sistema de SLAM visual através de edgels apresentado por Eade e Drummond (2006), mas também qualquer outro sistema que realiza rastreamento de retas através de uma pequena quantidade de pontos localizados sobre elas, o que constitui uma forma de extração de observações similar à do *Corisco*. A extração de edgels através da máscara em forma de grade é mais adequada para aplicações de alto desempenho.

Uma vez que os edgels são extraídos, eles podem servir como entrada para o processo de estimação de orientação. Este processo ocorre através da minimização de uma determinada função objetivo. Nos métodos antecessores do *Corisco* esta função foi definida através da técnica estimação MAP (COUGHLAN; YUILLE, 2003; DEUTSCHER; ISARD; MACCORMICK, 2002), ou EM (SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008). No *Corisco* foi utilizada a técnica de M-estimação para definir esta função objetivo, o que aproxima mais a teoria à área de estudos da estatística robusta. Em todos estes casos a função objetivo possui a forma de um cálculo de uma soma de erros que são medidos para cada edgel. Estes erros provém da comparação entre a direção de um edgel e uma direção predita de acordo com o modelo da câmera e uma orientação hipotética. A orientação que é encontrada é a que maximiza o alinhamento entre os edgels e as direções preditas, minimizando o erro. Um detalhe relevante do *Corisco* é que o cálculo desta diferença de direções não utiliza operações trigonométricas, mas apenas uma multiplicação de vetores normalizados, tornando novamente o método mais atraente para aplicações de alto desempenho.

Além das operações mais simples, as fórmulas utilizadas no cálculo da função de erro no *Corisco* permite utilizar com maior facilidade o algoritmo de otimização FilterSQP, de Fletcher e Leyffer (2002), o que representa uma evolução comparado aos algoritmos de otimização contínua utilizados anteriormente que não utilizam fórmulas fechadas para o cálculo das derivadas (SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008). A otimização também é beneficiada pela parametrização da orientação a partir de quaternions, que é o motivo da necessidade do uso de um algoritmo de otimização com restrições, como será explicado no Capítulo 4.

Além do algoritmo de otimização contínua, que apenas produz uma estimativa final, o *Corisco* também utiliza uma forma de busca aleatória que melhora a convergência global do processo para os casos onde não há uma boa estimativa inicial da solução. Isto já foi utilizado anteriormente em técnicas baseadas em linhas, mas não com edgels. Assim o *Corisco* emprega este tipo de estimação pela primeira vez entre as técnicas baseadas em edgels, inspirando-se em outras abordagens possíveis para o problema.

Outra diferença importante do *Corisco* com relação aos métodos existentes para estimação de orientação a partir de edgels, e mesmo comparado a muitas outras técnicas de visão baseada em linhas, é que no *Corisco* foi demonstrado pela primeira vez a possibilidade de se trabalhar com edgels em imagens distorcidas, inclusive em imagens de projeção equiretangular. E não só é possível trabalhar com modelos de câmera genéricos, como as análises são realizadas sempre no espaço da imagem original, sem retificação. O procedimento também é unificado, não há diferenças no algoritmo para se utilizar diferentes modelos, basta alterar uma sub-rotina utilizada durante a estimação que calcula os coeficientes da matriz Jacobiana da projeção para cada ponto da imagem.

Além da estimação de orientação, o *Corisco* pode ser facilmente modificado para rea-

lizar também a estimação de parâmetros internos da câmera, tal como a distância focal, de forma semelhante ao que foi realizado por Deutscher, Isard e MacCormick (2002) ou por Schindler e Dellaert (2004). Esta possibilidade também foi investigada durante o desenvolvimento do *Corisco*. Por evitar extrair linhas ou curvas das imagens, e explorar a condição de ambiente antrópico, o *Corisco* representaria uma alternativa bastante interessante a outros métodos de calibração baseados em ambientes com linhas (GRAMMATIKOPOULOS; KARRAS; PETSA, 2007; GONZALEZ-AGUILERA; GOMEZ-LAHOZ, 2008; ROSTEN; LOVELAND, 2009; THORMAHLEN; BROZIO; WASSERMANN, 2003). O *Corisco* ainda não oferece uma precisão muito boa para a calibração, mas pode servir como um método complementar para este problema, auxiliando em uma extração e classificação inicial de curvas, por exemplo.

Este capítulo discutiu diversas técnicas de visão baseada em características geométricas extraídas de imagens, incluindo técnicas que consideram múltiplas imagens e técnicas monoculares, além de técnicas baseadas em pontos e em retas no ambiente. Foi demonstrada também como diferentes métodos tentam explorar a existência de linhas paralelas no ambiente e lidar com modelos de câmera diferentes do de *pinhole*. O desenvolvimento do *Corisco* visou atender algumas deficiências que foram observadas neste cenário de técnicas existentes para realizar tarefas fundamentais, produzindo um método monocular para a estimação de orientação notavelmente flexível com relação a modelos de câmera, e com um grande apelo para aplicações de tempo real ou de alto desempenho. O *Corisco* é especialmente adequado para cenários em que não se deseja trabalhar com características pontuais, e em que também se deseja evitar realizar extrações de linhas nos primeiros estágios de análise, postergando esta tarefa para depois que mais informações como a orientação da câmera tenham sido obtidas.

No Capítulo 4 a seguir é apresentado em detalhes o funcionamento do *Corisco*. O funcionamento de alguns aspectos mais fundamentais de outros métodos anteriores também serão discutidos, permitindo uma melhor compreensão de como o *Corisco* foi desenvolvido a partir deles.



## 4 Detalhamento do Corisco

*Se entrega, Corisco. —Perseguição,*  
Glauber Rocha e Sérgio Ricardo

Este capítulo descreve o *Corisco*, que é o método proposto nesta tese para a estimativa de orientação da câmera em ambientes antrópicos a partir de edgels. O *Corisco* é caracterizado principalmente pela forma como as imagens são analisadas, produzindo uma extração esparsa de edgels, e pela função de erro calculada a partir destes edgels que é utilizada para a estimativa de parâmetros. Esta função de erro pode ser utilizada de diferentes maneiras. Por exemplo, ela pode servir como base para uma aplicação de rastreamento de câmera ou ainda para estimar os parâmetros intrínsecos da câmera. Estas aplicações alternativas também foram consideradas durante o desenvolvimento do *Corisco*, porém o foco principal do seu desenvolvimento foi a tarefa mais fundamental representada pela análise de uma única imagem para determinação da orientação da câmera em relação ao referencial natural do ambiente com conhecimento do modelo de câmera.

A Figura 4.1 apresenta um diagrama de blocos do processo que constitui o *Corisco*. O bloco *Extrator de edgels* é o único que tem acesso à imagem de entrada, e ele produz uma lista de edgels que são os únicos dados remanescentes utilizados no restante do processo, além dos parâmetros *intrínsecos* da câmera. Estes parâmetros, são a única outra entrada do processo além da imagem, e controlam os cálculos do bloco *modelo de câmera*, que é utilizado para produzir dois conjuntos de dados. Estas saídas do *modelo de câmera* são listas cujos elementos são calculados a partir de cada um dos edgels. A lista  $\mathbf{J}$  da figura contém as matrizes jacobianas da projeção calculadas na posição de cada edgel, e  $\mathbf{n}$  representa os vetores normais dos planos de interpretação calculados para a cada edgel. Cada normal pode ser calculada através de um produto vetorial entre a direção no espaço calculada a partir da posição do edgel na imagem e um segundo vetor calculado a partir da direção do edgel na imagem.

Uma vez que os edgels, Jacobianas e normais estejam disponíveis tem início o processo de otimização para a estimativa da orientação. O bloco *Função objetivo* representa o procedimento que realiza o cálculo de um erro de modelagem dos dados fornecidos para cada conjunto de parâmetros de orientação  $\Psi$  testado. Este cálculo utiliza as Jacobianas  $\mathbf{J}$  para calcular direções preditas para cada edgel, que são comparadas com eles, e por isso tantos os edgels quanto  $\mathbf{J}$  são entradas do bloco *Função objetivo*. Apenas  $\Psi$  varia durante a otimização para a estimativa de orientação, enquanto as outras entradas não são alteradas.

A otimização começa com uma busca aleatória pelo algoritmo RANSAC, representado pelo bloco *RANSAC*, onde grupos de vetores normais são sorteados e utilizados para

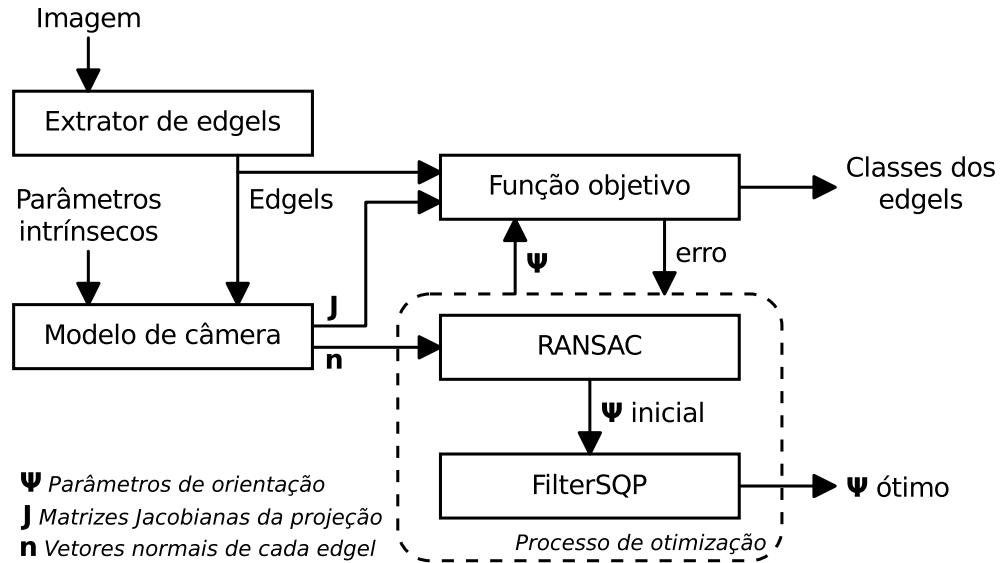


Figura 4.1: Diagrama de blocos do processo desenvolvido que constitui o *Corisco*. Os parâmetros de orientação  $\Psi$  ótimos são encontrados por um processo de otimização iterativo e de dois estágios. A função de erro que é otimizada utiliza parâmetros obtidos a partir do modelo de câmera e dos edgels extraídos da imagem de entrada. A classificação dos edgels também pode ser obtida como resultado do processo.

produzir hipóteses de orientação. O erro de cada hipótese é calculado, e a hipótese com o menor erro é utilizada como  $\Psi$  inicial para o FilterSQP, o que está representado na Figura 4.1 pela seta ligando os blocos RANSAC e FilterSQP. Enquanto o RANSAC precisa de acessar a lista de normais  $n$ , o FilterSQP precisa apenas de acessar a função objetivo. O FilterSQP necessita ainda do cálculo das derivadas da função, o que não está representado explicitamente na Figura 4.1, mas pode ser considerado como parte do bloco *Função objetivo*.

Quando o FilterSQP atinge sua condição de parada o  $\Psi$  ótimo encontrado para minimizar o erro se torna uma das saídas do processo realizado pelo *Corisco*. A classificação dos edgels de acordo com a direção de sua reta originária no ambiente também pode ser obtida como uma saída do processo. Esta classificação é realizada dentro do bloco *Função objetivo*, e portanto basta executar o cálculo da função utilizando o  $\Psi$  ótimo encontrado como entrada e tomar as classificações resultantes para criar a outra saída do processo.

A Figura 4.2 apresenta um exemplo de aplicação do *Corisco*. Primeiro o gradiente de cada canal da imagem de entrada é calculado, e então são extraídos edgels da imagem. O gráfico acima e à direita da figura mostra a intensidade do gradiente da imagem, e o resultado do extrator de edgels se encontra abaixo e à esquerda. Os edgels são os dados produzidos pelo bloco *Extrator de edgels* na Figura 4.1. As linhas contínuas no gráfico dos edgels são apenas uma impressão causada pela alta densidade de edgels comparada com a resolução da imagem, o gráfico contém na realidade apenas pequenos segmentos de reta desenhados sobre cada edgel e orientados de acordo com sua direção. A Figura 4.3

mostra um destaque deste mesmo gráfico, onde é possível ver com clareza que se tratam de pequenos segmentos de reta separados.

O gráfico abaixo e à direita na Figura 4.2 possui 77 pontos dispostos de forma regular sobre a imagem. Sobre cada ponto existe uma tripla de segmentos de reta, que são as direções preditas em cada ponto para as projeções das retas nas três direções possíveis do ambiente. Esta predição leva em consideração o modelo de câmera, e foi calculada nesta figura a partir da orientação estimada pelo *Corisco* para esta imagem de entrada. É possível observar que uma das direções de cada ponto aponta sempre para um ponto de fuga próximo ao centro da imagem. Esta é a direção do corredor deste ambiente. Neste exemplo foi levado em consideração uma leve distorção radial na imagem, o que pode ser percebido como uma curvatura ao seguir, por exemplo, as direções aproximadamente verticais nas extremidades direita e esquerda do gráfico.

Durante o cálculo da função de erro são calculadas direções preditas para cada edgel a partir da sua posição na imagem, do modelo de câmera e de uma orientação hipotética. Como há três direções possíveis para as retas do ambiente, são calculadas tripas de direções possíveis para cada edgel, assim como no gráfico da Figura 4.2 onde há três direções desenhadas para cada ponto. Estas três direções preditas são comparadas com a direção do edgel, e a direção mais próxima é tomada como sendo direção correspondente, determinando assim a classe do edgel pra aquela orientação hipotética. O erro relativo àquele edgel é então obtido através da comparação da sua direção medida com esta direção predita selecionada.

As Figuras 4.4 e 4.5 contém mais alguns exemplos de resultados do *Corisco* para a base de dados YorkUrbanDB (DENIS; ELDER; ESTRADA, 2008). Os gráficos inferiores da Figura 4.5 mostram um exemplo de aplicação do método para uma imagem com projeção polar equiretangular, obtida com uma lente olho-de-peixe. Enquanto nos outros exemplos a distorção da lente é relativamente fraca, neste último caso a distorção é bastante intensa. Quando não há distorção, as direções preditas apontam sempre para um mesmo ponto, o que não ocorre neste último caso, exceto na região próxima dos pontos de fuga.

As mesmas imagens da Figura 4.5 foram utilizadas para criar Figura 4.6. A primeira e a terceira linha de gráficos contém simplesmente os edgels extraídos das imagens. Cada coluna de gráficos da Figura 4.6 contém dados relativos a uma das direções dos ambiente, e portanto os edgels estão separados de acordo com a classe atribuída a cada um deles. Esta classificação foi realizada através do bloco *Função objetivo* da Figura 4.1, utilizando como entrada a orientação estimada pelo processo de otimização. A segunda e quarta linha de gráficos trazes conjuntos de direções preditas sobre pontos amostrados regularmente sobre a imagem, assim como nos exemplos anteriores, porém com gráficos distintos para cada direção possível. No caso da projeção polar equidistante pode ser possível visualizar os dois pontos de fuga sobre a imagem relativos a uma mesma direção do espaço, o que pode ser visto claramente nos gráficos mais abaixo e à esquerda da Figura 4.6.

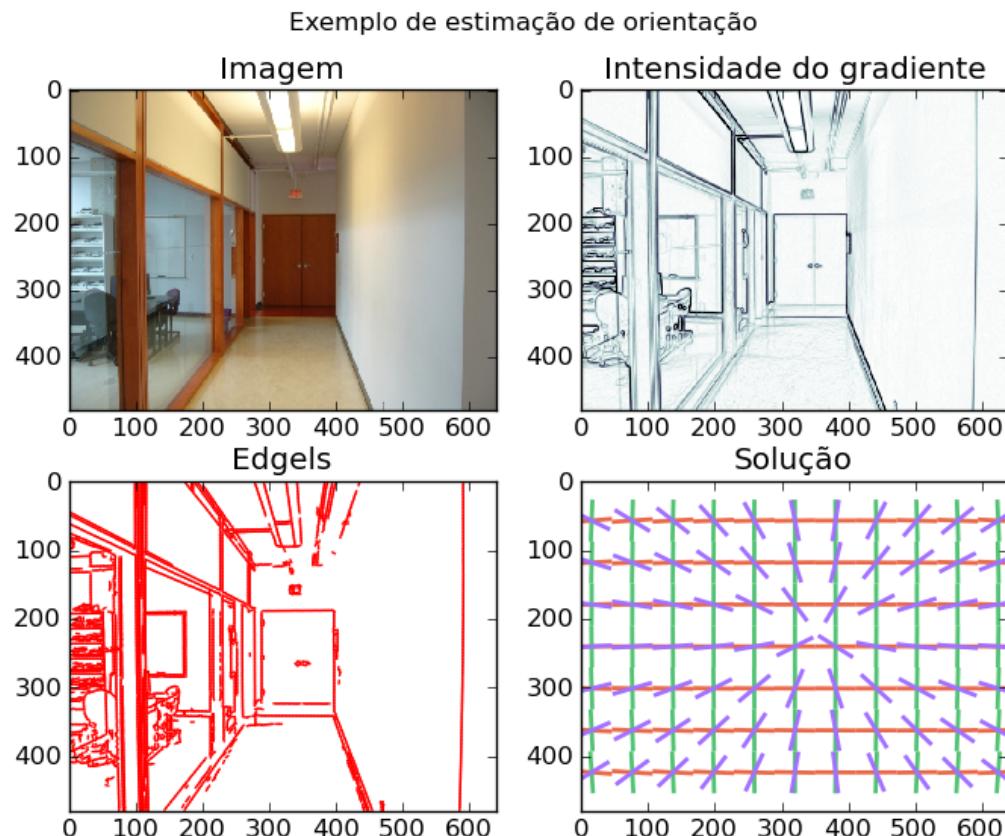


Figura 4.2: Exemplo da análise da imagem e do resultado do processo de estimativa de orientação. No topo e à esquerda encontra-se a imagem de entrada, e à direita a intensidade do gradiente. Em baixo e à esquerda encontra-se a saída do extrator de edgels, e à direita algumas direções preditas a partir da orientação estimada.

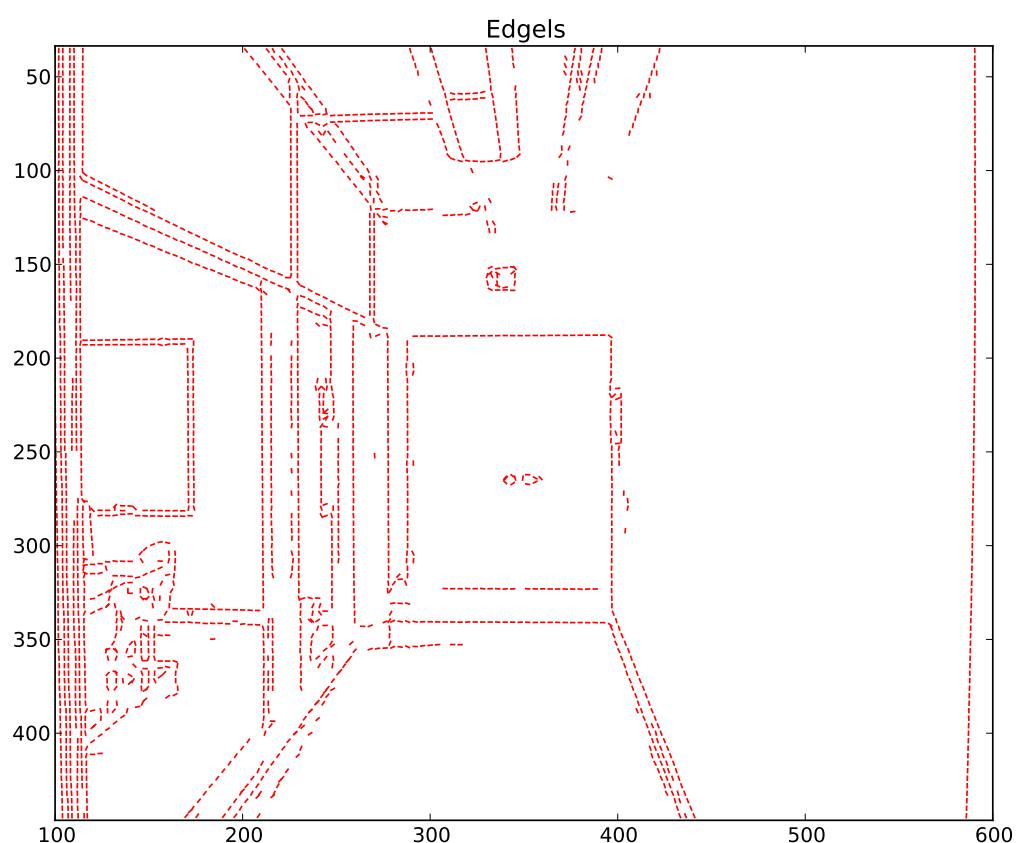


Figura 4.3: Detalhe dos edgels extraídos da Figura 4.2.

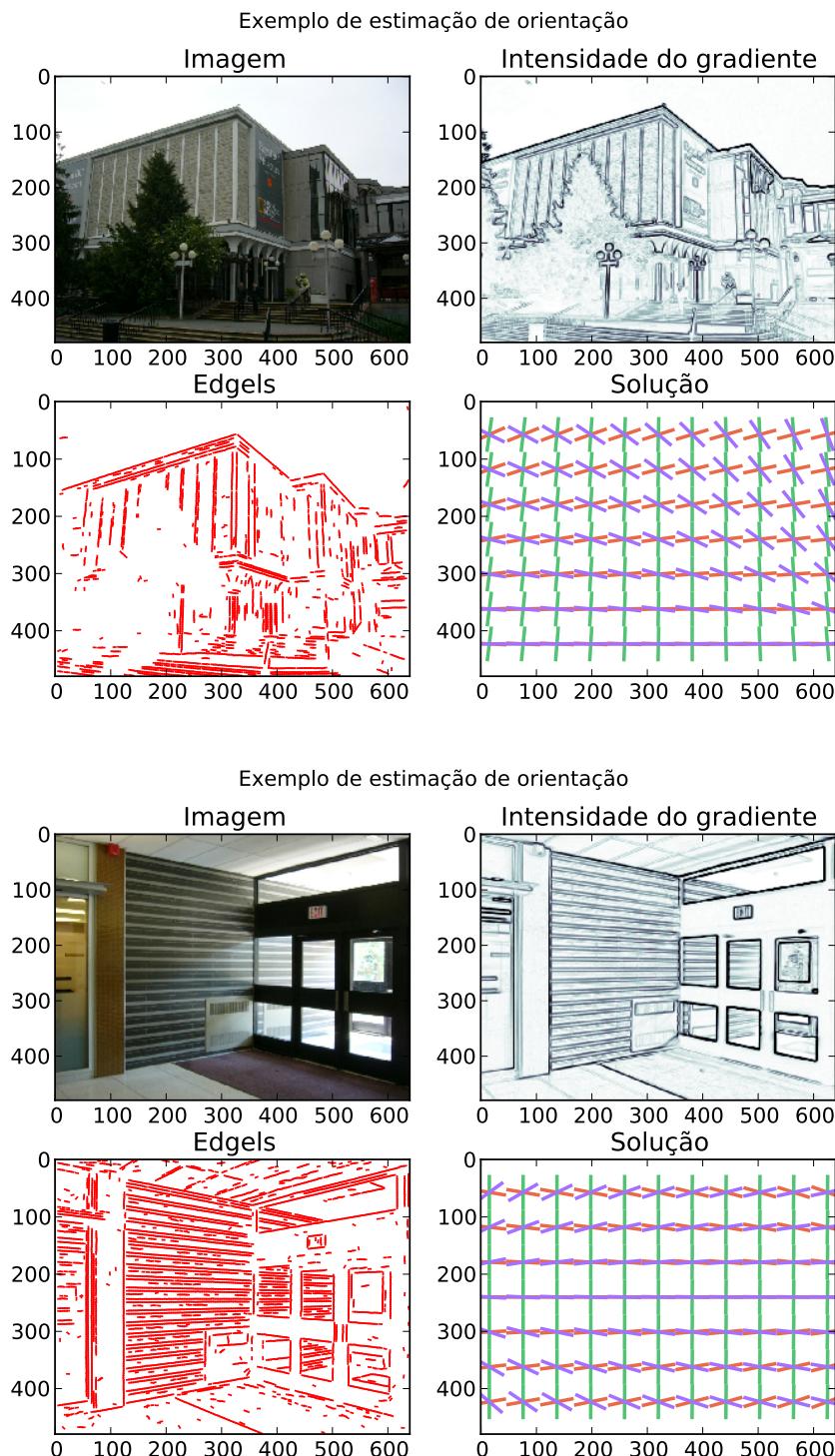


Figura 4.4: Exemplos do resultado da estimativa de orientação pelo *Corisco*.

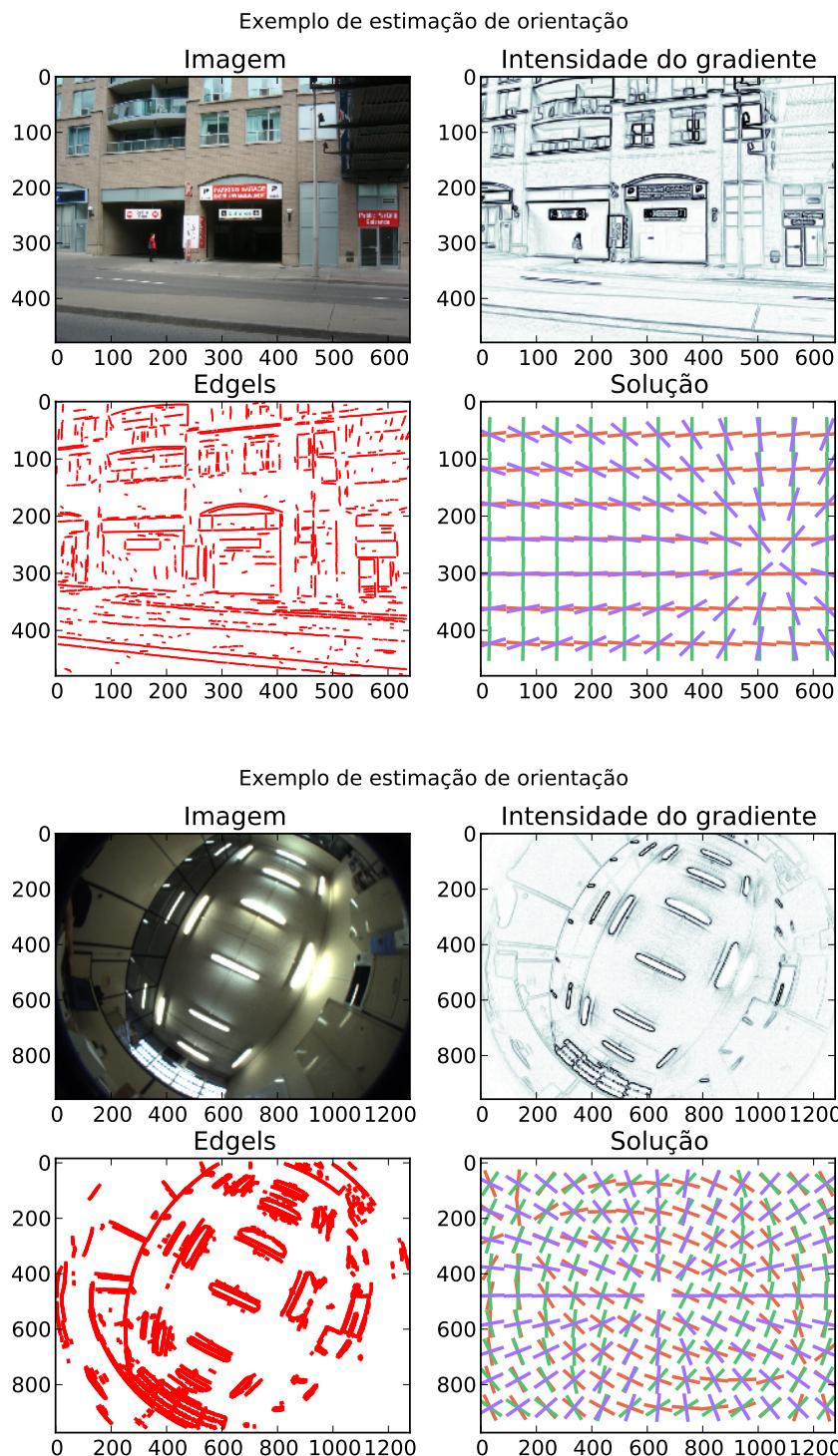


Figura 4.5: Exemplos do resultado da estimativa de orientação pelo *Corisco* para mais uma imagem da base de dados YorkUrbanDB, e um exemplo com uma imagem de projeção polar equidistante, obtida com uma lente olho-de-peixe.

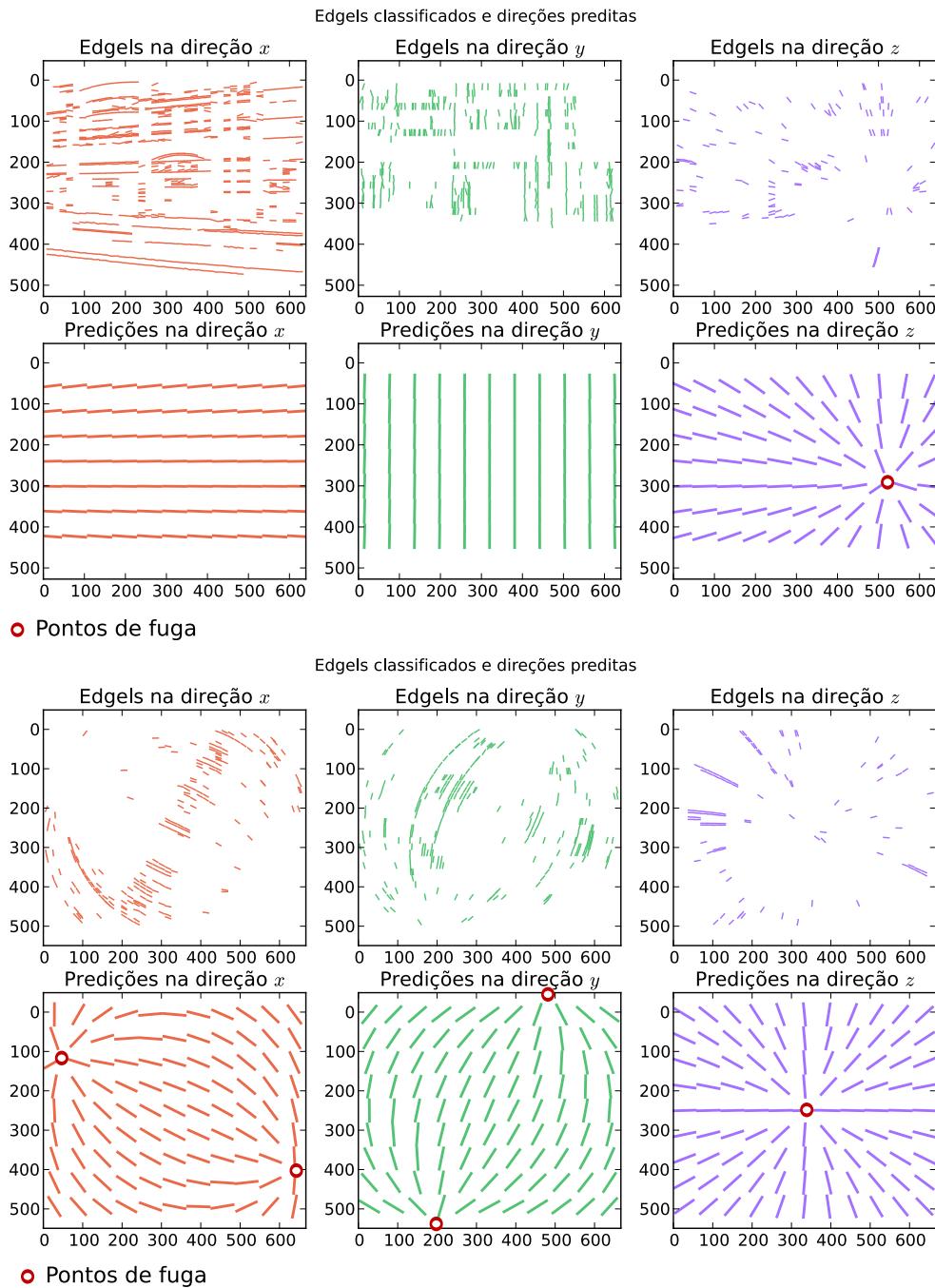


Figura 4.6: Classificação de edgels e direções preditas, calculadas pelo *Corisco* a partir da solução encontrada para as imagens da Figura 4.5. Cada gráfico contém dados relativos a apenas uma das direções possíveis do ambiente, permitindo uma visualização melhor. A localização aproximada dos pontos de fuga foi assinalada nos gráficos das direções preditas. No caso da imagem de projeção aproximadamente há duas direções aproximadamente paralelas ao plano da imagem, e apenas um ponto de fuga pode ser visto. No caso da imagem da lente olho-de-peixe é possível observar ambos pontos de fuga para as duas primeiras direções nos dois gráficos mais abaixo e à esquerda, enquanto para a terceira direção o ponto de fuga se localiza aproximadamente no centro da imagem.

O restante deste capítulo descreve em maiores detalhes cada passo do processo de análise e estimação de parâmetros realizado pelo *Corisco*. A Seção 4.1 a seguir descreve o funcionamento do extrator de edgels. A Seção 4.2 descreve como a direção de um edgel é predita a partir da orientação da câmera, e como a orientação é parametrizada no *Corisco*, a partir de quaternions. Esta é uma aplicação direta da teoria discutida anteriormente na seção 2.1.2.

Estas direções preditas dos edgels são comparadas com as medidas para realizar a estimação de orientação. O cálculo da função objetivo utilizada na estimação de parâmetros é descrito na Seção 4.3, e o processo de otimização empregado na estimação de orientação é descrito nas Seções 4.4 e 4.5, que descrevem separadamente cada um dos dois passos do processo de otimização completo. Isto conclui a exposição do processo ilustrado na Figura 4.1, que é o método proposto para a estimação de orientação de câmera, que denominamos *Corisco*.

Como foi discutido anteriormente, o *Corisco* pode ser utilizado na solução de outros problemas mais complexos do que a estimação de orientação, o que porém pode envolver pequenas modificações ao procedimento proposto originalmente. Estas aplicações possíveis são discutidas no final deste capítulo. A Seção 4.6.2 discute como é possível estimar os parâmetros internos através de um processo de otimização realizado por cima da estimação de orientação, porém com a mesma função objetivo do *Corisco*. Já a Seção 4.6.3 descreve um processo que pode ser utilizado após a extração de edgels para tentar remover observações que não sejam parte de linhas retas do ambiente, e ainda uma forma de extração de linhas que pode ser realizada através do agrupamento de edgels após a estimação da orientação da câmera.

## 4.1 Extração de edgels

O funcionamento do *Corisco* não depende de um processo específico de extração de edgels, e diferentes procedimentos poderiam ser adotados para a obtenção destas entidades, substituindo o bloco *Extrator de edgels* da Figura 4.1. Porém o método específico de extração proposto foi desenvolvido com os mesmos propósitos de projeto como um todo, de buscar criar procedimentos mais determinísticos, mais adequados para eventuais aplicações de alto desempenho, e que explorassem melhor a geometria do problema.

A extração de edgels do *Corisco* se baseia em uma varredura da imagem sobre um conjunto de linhas e colunas que constituem uma máscara em forma de grade. A utilização desta grade foi investigada inicialmente sem a realização da extração de bordas (WERNECK; COSTA, 2011b), porém na proposta final cada um dos pixels visitados nesta varredura da grade passa por um detector de borda e por um teste da sua direção, e caso seja aceito, a posição onde esta linha ou coluna da grade sendo varrida corta a borda detectada é calculada com precisão de sub-pixel a partir dos valores de intensidade do gradiente. Esta posição calculada é armazenada junto da direção do gradiente naquele

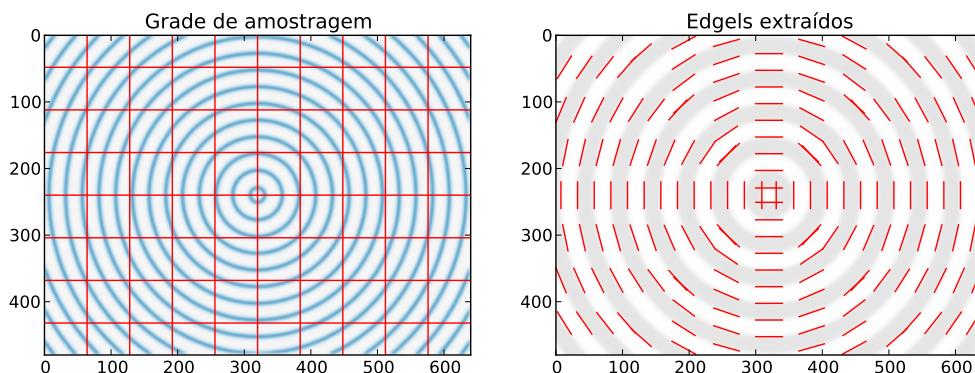


Figura 4.7: Máscara em forma de grade e edgels extraídos da imagem da Figura 2.11.

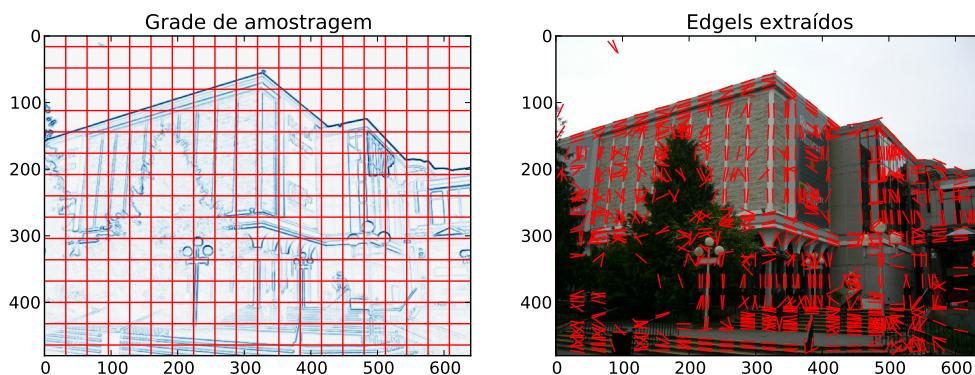


Figura 4.8: Máscara em forma de grade e edgels extraídos da imagem da Figura 2.12.

ponto, originando um novo edgel extraído. A variação do tamanho da grade permite controlar de uma forma natural a quantidade de observações que são produzidas, resultando em uma amostragem de linhas longas da imagem com uma resolução mínima garantida. Os edgels também são descartados se a intensidade do gradiente for menor do que um valor mínimo. A detecção de bordas e o cálculo do gradiente da imagem foram discutidos na Seção 2.2

Os gráficos à esquerda nas Figuras 4.7 e 4.8 mostram a intensidade do gradiente para duas imagens diferentes, e superposto a isto estão linhas verticais e horizontais que indicam as linhas e colunas da imagem que foram varridas durante o processo de extração de edgels do *Corisco*. Os gráficos à direita nestas figuras mostram os edgels que foram extraídos. Cada edgel é representado através de um pequeno segmento de reta cujo centro é o ponto em que as bordas foram localizadas sobre a grade. Estes segmentos de reta também estão orientados na direção da borda sobre estes pontos, que é simplesmente a direção ortogonal ao gradiente naquele ponto.

A Figura 4.7 foi gerada artificialmente, e contém um número de anéis pretos e brancos. É possível observar que os edgels extraídos da imagem seguem a direção tangencial das

bordas destes anéis, que são círculos concêntricos, e portanto as direções dos edgels são ortogonais às direções de suas posições em relação ao centro da imagem.

A Figura 4.9 mostra os valores das componentes horizontal e vertical do gradiente sobre uma das linhas da mesma imagem mostrada na Figura 4.7. Durante a varredura é realizada uma detecção de borda seguida de uma localização com precisão de sub-pixel da localização da borda sobre a trajetória analisada. Este processo pode ser entendido como uma busca dos extremos da curva desenhada neste gráfico da Figura 4.9, sendo que, como será explicado a seguir, um cuidado especial ainda deve ser tomado para lidar com imagens coloridas, e apenas bordas aproximadamente ortogonais às trajetórias analisadas são extraídas.

#### 4.1.1 Procedimento para a extração de edgels

O procedimento conduzido para cada pixel durante a varredura de uma linha da grade é realizado através dos seguintes passos:

1. *Obtenção da direção corrigida:* O gradiente sobre o novo pixel é consultado. Se o sinal da componente horizontal do gradiente em algum canal for negativo, as duas componentes do gradiente deste canal tem o seu sinal trocado. Estes gradientes com sentido corrigido de cada canal são então somados para gerar um único vetor que é tomado como o gradiente daquele ponto. O termo *gradiente* se refere a seguir a este vetor calculado a partir da soma dos gradientes com sentido corrigido.
2. *Teste de intensidade:* Se o valor da norma quadrática do gradiente for menor do que um limiar, o ponto atual é descartado e o pixel seguinte é analisado.
3. *Teste de direção:* Se o valor absoluto da componente vertical do gradiente for maior do que o da horizontal, isto significa que o gradiente naquele ponto é aproximadamente vertical, e a borda é aproximadamente horizontal. Assim o ponto atual é descartado e o pixel seguinte é analisado.
4. *Teste de superioridade local:* Se o gradiente for forte o bastante, e aproximadamente horizontal, a norma do gradiente é comparada com os valores no pixel anterior e posterior sobre a trajetória analisada. Caso o valor da norma do gradiente do pixel atual não seja maior do que ambos ele é descartado e o pixel seguinte é analisado.
5. *Instanciação do edgel:* Se a norma do gradiente local for maior do que o limiar, se a sua direção for aproximadamente horizontal, e o pixel atual ainda for um máximo local da norma do gradiente, então houve uma detecção de borda bem-sucedida sobre este pixel. Um edgel deve agora ser extraído a partir das características locais da imagem.

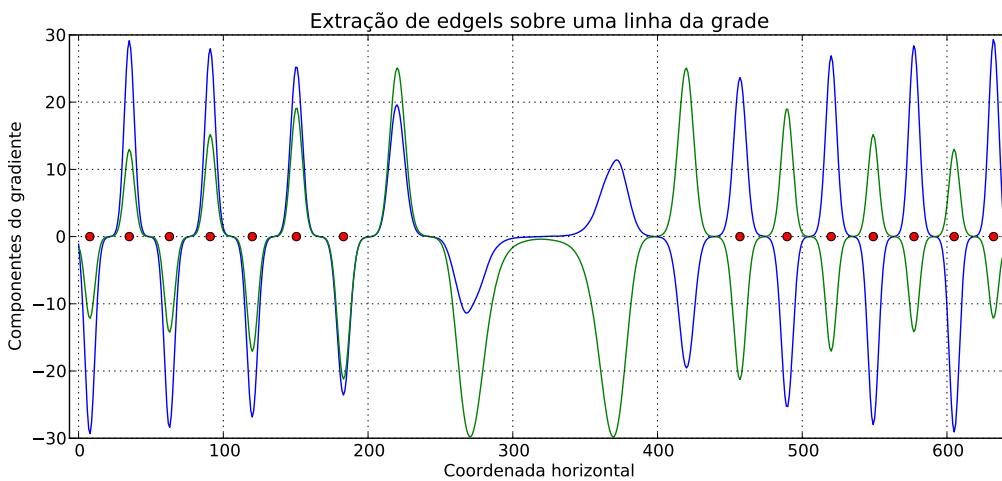


Figura 4.9: Valores das componentes do gradiente e posições onde edgels foram detectados na segunda linha varrida na imagem da Figura 4.7.

- A posição do edgel é o ponto onde a borda próxima ao pixel atual corta a linha sendo percorrida, e é calculada com precisão de sub-pixel a partir das normas do gradiente no pixel atual e seus dois vizinhos, com o ajuste de uma parábola a estes valores. A posição da borda é a posição do vértice desta parábola, ou seja, seu ponto máximo.
- A direção atribuída ao edgel é simplesmente uma direção ortogonal à do gradiente do pixel atual.

6. Após a instanciação do novo edgel, o próximo pixel da linha é analisado.

A extração de edgels nas colunas da imagem é similar ao caso das linhas, com a diferença de que os sentidos dos gradientes em cada canal são corrigidos de acordo com a componente vertical e são pontos com o gradiente aproximadamente verticais que são retidos. Os pixels vizinhos também se tornam os pixels acima e abaixo do atual, e não à esquerda e direita. Uma forma de implementar a varredura nas colunas seria simplesmente fazer uma varredura por linhas na imagem transposta.

A Figura 4.9 mostra os valores das componentes do gradiente para a segunda linha da grade da Figura 4.7. Cada ponto vermelho do gráfico indica a posição de um edgel extraído. É possível notar que após o sétimo edgel há um pico no gradiente onde não foi detectada uma borda. Isto se deve ao fato de que a direção do gradiente naquele ponto já não era mais vertical, assim como nos três picos seguintes, depois dos quais voltam a haver edgels sendo extraídos. É interessante observar ainda nestas curvas que os sinais das componentes do gradiente passam de iguais para opostos após o centro da imagem, o que é uma decorrência da simetria espelhada do círculo em relação ao eixo vertical.

A Figura 4.10 mostra um gráfico semelhante ao da Figura 4.9 para a quarta linha da grade mostrada na Figura 4.8. Como esta imagem não é artificial, as curvas não são tão

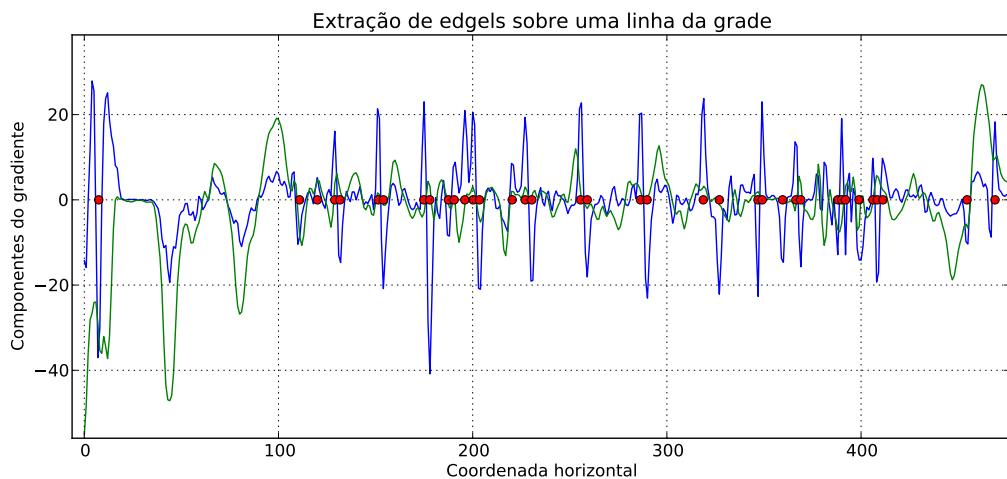


Figura 4.10: Valores das componentes do gradiente e posições onde edgels foram detectados na quarta linha varrida na imagem da Figura 4.8.

suaves como no gráfico anterior. Mas ainda é possível notar como os edgels extraídos se localizam sob os picos na intensidade do gradiente, e como os picos são ignorados quando a direção do gradiente não é horizontal.

Um edgel extraído se localiza portanto em uma posição sobre a grade onde ela é cortada por uma borda da imagem, e a direção do edgel é a direção da borda naquele ponto. A direção é determinada a partir do vetor calculado durante a varredura que é a soma dos gradientes de cada canal com o sentido corrigido para evitar um cancelamento de vetores com a mesma direção porém sentidos opostos, que o problema ilustrado na Figura 2.14. É interessante observar que um dos valores das coordenadas de cada edgel sempre será um número inteiro, que é a posição da linha ou coluna da grade em que este edgel foi encontrado. Porém isto não é relevante no restante do processo, e o fato de um edgel ter sido observado em uma linha ou coluna é desprezado.

Na prática, a direção que é armazenada para cada edgel é a direção do próprio vetor gradiente, ortogonal à borda. E esta direção é representada através de um vetor bidimensional de norma unitária. O motivo é porque é esta direção que é originalmente calculada a partir do gradiente, e também é esta direção que será utilizada a seguir no cálculo da função erro. Mas não seria difícil trabalhar com a direção ortogonal, da borda. Para calcular esta direção ortogonal a partir do vetor normalizado na direção do gradiente basta trocar o valor de um componente pelo outro, e inverter o sinal de um dos dois valores. Assim qualquer expressão calculada com estes valores propostos, do vetor na direção do gradiente, pode ser facilmente adaptada para utilizar os valores do vetor ortogonal, e a escolha entre armazenar uma direção ou a outra cabe apenas à preferência do programador.

#### 4.1.2 Consideração sobre o espaçamento da grade

A escolha do espaçamento da grade, ou seja, das distâncias entre cada linha e coluna, é um problema que não foi estudado a fundo durante esta pesquisa, assim como métodos similares existentes não possuem uma indicação específica de como determinar os parâmetros que são utilizados em seus processos de sub-amostragem dos dados (DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004; EADE; DRUMMOND, 2006). Este é portanto um parâmetro de entrada para o *Corisco* que deve ser selecionado em cada aplicação para obter o comportamento desejado. Se o espaçamento da grade for ajustado para 1, todos os pixels da imagem serão analisados, e extrator de edgels proposto se torna um processo muito similar a detectores de borda tradicionais. Conforme o espaçamento da grade aumenta, o tempo de processamento cai proporcionalmente, mas sem afetar muito a princípio a qualidade dos resultados. Este efeito foi observado em experimentos, relatados no Capítulo 5.

Apesar do comprometimento entre qualidade e desempenho com a variação do espaçamento da grade ser facilmente observado em diferentes casos, não é fácil encontrar uma recomendação geral para qual espaçamento deve ser utilizado para acelerar o processo o máximo possível sem afetar a precisão dos resultados. O mesmo vale ainda para outros procedimentos de pré-processamento que podem ser utilizados tanto neste problema quanto em outros, tal como o uso de filtros passa-baixas para eliminar ruídos e suavizar a imagem, ou a mudança de escala das imagens para a redução de dados. Imagens de entrada ruidosas sempre poderiam a princípio ser filtradas e reduzidas por algum fator de escala máximo sem afetar significativamente o resultado de qualquer método, e isto depende das condições específicas das imagens e dos dados contidos nelas.

Em uma aplicação qualquer de visão não costuma ser óbvio de que forma uma imagem pode receber um pré-processamento simples como uma redução de escala sem afetar os resultados, ou ainda qual filtro específico deve ser utilizado em um problema para se calcular o gradiente de uma imagem, por exemplo. Da mesma forma também não é óbvio qual parâmetro deve ser selecionado no *Corisco* para controlar a máscara em forma de grade durante a extração de edgels.

O *Corisco* também foi implementado com um método de detecção de bordas um pouco mais sofisticado do que o descrito aqui nesta seção, utilizando o método dos momentos de Zernike, introduzido por Ghosal e Mehrotra (1993). Há poucas diferenças entre o uso deste método e o método mais simples baseado apenas no gradiente. Em especial, o uso da máscara em forma de grade não foi modificado. Métodos baseados em momentos tem o potencial para apresentar melhores resultados, como demonstrado ao menos desde o trabalho de Lyvers et al. (1989), e por isso houve o interesse em se tentar aplicar algum destes métodos ao problema da estimação de orientação, o que é uma possibilidade de melhoria que ainda não havia sido demonstrada anteriormente dentro desta área. O uso de momentos na análise de bordas também serve como um primeiro passo antes de uma

eventual aplicação de métodos de análise de bordas ainda mais sofisticados, baseados por exemplo em filtros direcionais tal como proposto por Perona (1995), e já demonstrado para o problema de extração de um único ponto de fuga em imagens de estradas por Audibert e Ponce (2009).

A seção a seguir discute como os edgels extraídos segundo o procedimento descrito nesta seção são utilizados para o cálculo da função objetivo que é utilizada pelo *Corisco* para estimar a orientação da câmera.

## 4.2 Função da direção de um edgel

Para realizar a estimativa de orientação é preciso calcular uma função de erro que depende da comparação da orientação medida de cada edgel com uma direção predita. Esta direção predita é a direção tangencial de uma curva que cortaria aquele ponto da imagem onde se localiza o edgel. Esta curva seria a projeção de uma reta do ambiente, e apenas no caso do modelo de câmera *pinhole* esta curva da projeção é sempre uma reta sobre a imagem.

O cálculo desta direção local da projeção de uma reta sobre um ponto específico, que é a predição da direção de um edgel, depende das coordenadas do ponto  $p$  sobre a imagem, do modelo de câmera, e da direção  $r$  em relação ao referencial da câmera da reta que teria originado a projeção. Esta direção da reta é calculada a partir da orientação da câmera em relação ao referencial do ambiente dada pelos parâmetros  $\Psi$  e de um modelo do ambiente que diz quais são as direções possíveis para as retas. No caso em que se considera um ambiente antrópico, para se calcular uma direção  $r$  a partir de uma orientação  $\Psi$  basta obter os coeficientes de uma matriz de rotação tridimensional correspondentes a  $\Psi$ , e escolher uma das três direções possíveis, entre os três eixos do referencial do ambiente.

Em outras palavras, a orientação codifica as três direções possíveis para as retas. Escolhendo-se uma das três direções do ambiente, que representam três classes possíveis para as observações, encontra-se sua direção  $r$  correspondente a partir de  $\Psi$ . A direção predita para esta classe pode ser então encontrada em um ponto  $p$  da imagem através do modelo de câmera. É possível estender este problema para levar em consideração retas em orientações genéricas no ambiente, porém esta possibilidade não foi estudada nesta pesquisa.

Como foi discutido na Seção 2.1, um modelo de câmera é constituído pelo mapeamento entre os pontos do espaço e da imagem. Trata-se de um par de funções que calculam as coordenadas  $(p^x, p^y)$  de um ponto  $p$  da imagem a partir de um ponto  $q$  do ambiente com coordenadas  $(q^x, q^y, q^z)$ . Estas três coordenadas constituem os argumentos de entrada para o cálculo de uma projeção, que depende ainda dos parâmetros intrínsecos e extrínsecos da câmera. Os parâmetros intrínsecos fazem parte do modelo de câmera, e para o problema da estimativa de orientação se assume serem conhecidos. Eles incluem

por exemplo a distância focal e as coordenadas do ponto principal centro no caso do modelo *pinhole*. Já os parâmetros extrínsecos são a posição e orientação da câmera com relação ao referencial do ambiente. No problema da estimativa de orientação estudado aqui não se conhece a posição da câmera no espaço, e podemos assumir que ela se encontra na origem. Resta portanto encontrar apenas a orientação da câmera, que é dada por uma matriz de rotação em três dimensões. A Figura 2.15 ilustra a relação que existe entre as direções das projeções das retas sobre a imagem e a orientação da câmera em relação a um ambiente antrópico. A estimativa da orientação é feita através de uma busca por uma orientação que faça com que as direções preditas em diferentes pontos para as projeções de retas, calculadas a partir do modelo geométrico do problema, se aproximem das direções encontradas no processo de extração de edgels discutido na Seção 4.1.

O restante desta seção mostra primeiro de que forma a orientação da câmera é parametrizada no *Corisco*, utilizando quaternions, e então como os modelos de câmera e o cálculo das projeções de retas discutidos na Seção 2.1.2 são utilizados para calcular as direções preditas para cada edgel, que são comparadas com as direções medidas através da função objetivo.

#### 4.2.1 Parametrização da orientação da câmera

Uma matriz de rotação tridimensional pode ser parametrizada de diferentes formas. Uma delas é uma matriz completa de rotação, dada por 9 valores. Uma matriz é um modelo muito geral, no entanto, e existem restrições que devem ser impostas a estes valores para que esta seja estritamente uma matriz de rotação tridimensional. Apenas três valores são necessários e suficientes para modelar uma rotação tridimensional, ou seja, uma rotação possui apenas três graus de liberdade, e portanto uma matriz completa conteria seis valores que não poderiam ser escolhidos de forma arbitrária.

Uma forma mais adequada e bastante comum de representar uma rotação tridimensional é através de três ângulos,  $\theta_1$ ,  $\theta_2$  e  $\theta_3$ , denominados ângulos de Euler. Cada ângulo especifica uma rotação ao redor de um eixo específico, produzindo três matrizes que devem ser aplicadas, por exemplo, da seguinte maneira

$$\mathbf{R} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 \\ -\sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_2) & \sin(\theta_2) \\ 0 & -\sin(\theta_2) & \cos(\theta_2) \end{bmatrix} \begin{bmatrix} \cos(\theta_3) & \sin(\theta_3) & 0 \\ -\sin(\theta_3) & \cos(\theta_3) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (4.1)$$

Esta expressão produz qualquer matriz de rotação possível, e apenas isto. As expressões para cada valor da matriz final podem se encontradas a partir desta equação, e cada uma destas expressões pode depender ou de um, dois, ou até de todos os três parâmetros.

No *Corisco* foi utilizada uma outra forma de representar rotações, através de quaternions normalizados. Esta não é uma parametrização incomum em aplicações de visão,

e foi utilizada no trabalho de (DEUTSCHER; ISARD; MACCORMICK, 2002), para citar apenas um exemplo. Um *quaternion* é uma quádrupla de valores  $(\Psi^a, \Psi^b, \Psi^c, \Psi^d)$  que são uma generalização dos números complexos, constituindo toda uma álgebra própria. Quaternions podem ser utilizados para definir matrizes de rotação, assim como ângulos de Euler. Em primeiro lugar é preciso que este quaternion seja normalizado, ou seja,  $\Psi^{a2} + \Psi^{b2} + \Psi^{c2} + \Psi^{d2} = 1$ . Os coeficientes da matriz de rotação são dados por

$$\mathbf{R}(\Psi) = \begin{bmatrix} \Psi^{a2} + \Psi^{b2} - \Psi^{c2} - \Psi^{d2} & 2\Psi^b\Psi^c + 2\Psi^a\Psi^d & 2\Psi^b\Psi^d - 2\Psi^a\Psi^c \\ 2\Psi^b\Psi^c - 2\Psi^a\Psi^d & \Psi^{a2} - \Psi^{b2} + \Psi^{c2} - \Psi^{d2} & 2\Psi^c\Psi^d + 2\Psi^a\Psi^b \\ 2\Psi^b\Psi^d + 2\Psi^a\Psi^c & 2\Psi^c\Psi^d - 2\Psi^a\Psi^b & \Psi^{a2} - \Psi^{b2} - \Psi^{c2} + \Psi^{d2} \end{bmatrix}. \quad (4.2)$$

Se o quaternion não for normalizado a matriz sofrerá uma mudança de escala pelo fator  $|\Psi|^2$ , onde  $|\Psi| = \sqrt{\Psi^{a2} + \Psi^{b2} + \Psi^{c2} + \Psi^{d2}}$ . É possível interpretar os componentes de um quaternion como sendo uma direção definida pelo vetor  $(\Psi^b, \Psi^c, \Psi^d)$  que define o eixo de rotação, e o ângulo da rotação é obtido a partir de  $\Psi^a$ .

A álgebra de quaternions permite multiplicá-los de forma que o quaternion resultante produz exatamente a matriz de rotação que seria encontrada multiplicando-se as matrizes dos quaternions de entrada, na mesma ordem. Ou seja, não só os quaternions representam de forma biunívoca as rotações, como a operação de multiplicação de quaternions representa a multiplicação de matrizes de rotação com exatidão. Mas a maior vantagem no uso de quaternions reside no fato de que as fórmulas para os coeficientes são todas relativamente simples, apenas polinômios de segunda ordem nos parâmetros do quaternion, e não funções trigonométricas como no caso dos ângulos de Euler.

O propósito do *Corisco* é estimar a orientação da câmera com relação ao ambiente. Esta orientação é dada por um quaternion  $\Psi$ . As direções  $\mathbf{r}$  das linhas do ambiente com relação ao referencial da câmera são obtidas a partir de cada linha da matriz de rotação calculada a partir de  $\Psi$ , utilizando a Equação 4.2, ou seja

$$\mathbf{R}(\Psi) = \begin{bmatrix} \mathbf{r}_x \\ \mathbf{r}_y \\ \mathbf{r}_z \end{bmatrix}. \quad (4.3)$$

Assim são calculadas durante a estimação de orientação, para um dado conjunto de parâmetros hipotéticos  $\Psi$ , as direções em relação à câmera  $\mathbf{r}_x$ ,  $\mathbf{r}_y$  e  $\mathbf{r}_z$  a que deveriam se orientar as retas do ambiente. Estas direções no ambiente são utilizadas para calcular as direções preditas dos edgels utilizando o modelo de câmera, como será discutido na Seção 4.2.2.

Quaternions também foram utilizados no método de estimação de orientação desenvolvido por Deutscher, Isard e MacCormick (2002), mas naquele método foi empregada apenas uma técnica de busca estocástica para a estimação, e as vantagens da represen-

tação da orientação através de quaternions não foram tão exploradas quanto no *Corisco*. Nos outros métodos similares existentes (COUGHLAN; YUILLE, 2003; SCHINDLER; DELLAERT, 2004; DENIS; ELDER; ESTRADA, 2008) foram empregadas parametrizações baseadas em ângulos, similares à da Equação 4.1. A Seção 4.5 irá demonstrar de que forma a representação através de quaternions ao invés de ângulos é explorada no *Corisco* durante o processo de estimação, permitindo que sejam utilizadas fórmulas fechadas para o cálculo das derivadas da função objetivo. Antes, porém, discutiremos brevemente o cálculo das direções preditas para cada edgel, e como estas direções preditas são comparadas com as medidas pela função objetivo.

#### 4.2.2 Cálculo das direções preditas dos edgels

A Seção 2.1 discutiu diferentes modelos de câmera, e como é possível calcular a posição  $p$  da projeção na imagem de um ponto  $q$  do ambiente, e como calcular a direção do raio incidente associado a um ponto  $p$ . O modelo de câmera também pode ser utilizado para calcular a direção  $v$  sobre a imagem da projeção em um ponto  $p$  de uma reta do ambiente orientada à direção  $r$ , o que é feito através da Equação 2.18, onde a matriz Jacobiana  $J$  depende de  $p$  e do modelo de câmera.

O *Corisco* se baseia na Equação 2.18 para calcular as direções preditas dos edgels extraídos. A matriz  $J$  é calculada para cada edgel, a partir de sua posição  $p$ , e então para uma dada orientação  $\Psi$  calculam-se as direções  $r$  segundo as Equações 4.2 e 4.3. Assim são obtidas direções  $v$  preditas para cada edgel. São calculadas três direções  $v$  para cada edgel, uma para cada direção  $r$  calculada. Estes três vetores  $v$  calculados em cada ponto precisam ser normalizados de forma independente para produzir vetores unitários. A função objetivo escolhe depois uma única destas direções para utilizar com referência, classificando cada edgel e encontrando um resíduo para a orientação especificada no início dos cálculos.

O *Corisco* pode ser portanto aplicado a qualquer modelo de câmera possível, já que o modelo afeta apenas o cálculo de  $J$ . A implementação do *Corisco* seguiu uma forma bastante modular, onde a troca do modelo de câmera implica apenas na substituição de um procedimento que calcula a matriz Jacobiana da projeção em função de uma posição sobre a imagem.

Um detalhe interessante desta forma de calcular as direções preditas para os edgels é que se o quaternion  $\Psi$  não for normalizado os vetores  $r$  sofrerão uma mudança de escala, mas como o cálculo de  $v$  envolve apenas operações lineares, e termina com uma normalização, isto não é relevante. Assim não é estritamente necessário normalizar  $\Psi$  para executar os cálculos. Isto simplifica bastante as expressões envolvidas, e permite encontrar expressões fechadas não muito complexas para as derivadas da função objetivo que será calculada.

O funcionamento do *Corisco* se baseia portanto em um modelo de câmera que é

fundamentalmente apenas o mapeamento do ambiente para o espaço de imagem, de  $q$  para  $p$ . A partir deste mapeamento é possível definir um outro mapeamento de direções  $r$  no espaço para direções  $v$  no plano de imagem, sendo que uma tripla de direções  $r$  são definidas a partir de um conjunto de parâmetros de orientação  $\Psi$ , e as direções  $v$  são calculadas para cada direção  $r$  sobre diversos pontos da imagem. O que o *Corisco* utiliza ultimamente são mapeamentos do espaço quadridimensional dos parâmetros  $\Psi$  para triplas de direções  $v$  para diversos pontos  $p$  da imagem. Conforme uma estimativa  $\Psi$  é variada estas triplas de direções sobre a imagem também são.

O processo de otimização busca encontrar um  $\Psi$  para o qual haverá o melhor alinhamento entre direções medidas e direções  $v$  calculadas e selecionadas de cada tripla para cada edgel. Esta seleção e avaliação de alinhamento são realizadas pela função objetivo, discutida na Seção 4.3 a seguir. O processo de otimização é discutido nas Seções 4.4 e 4.5.

### 4.3 Cálculo da função objetivo

O *Corisco* é baseado em um processo de otimização de uma função objetivo que depende da orientação da câmera e dos parâmetros dos edgels extraídos, além de um modelo de câmera conhecido. Os edgels variam de acordo com a imagem, e a função objetivo compara os seus valores com valores preditos a partir de uma orientação hipotética. Os parâmetros da orientação são alterados por um procedimento de otimização até que se encontre uma combinação ótima que representa um ponto mínimo do valor da função objetivo para aquela imagem. Para definir esta função objetivo é necessário empregar uma das técnicas discutidas na Seção 2.4. No problema da estimação de orientação a partir de edgels a técnica utilizada inicialmente foi a MAP, que foi depois substituída pelo algoritmo EM. No *Corisco* a técnica utilizada no lugar destas para criar a função objetivo é a M-estimação.

Nos casos da estimação por MAP ou por EM o desenvolvimento do algoritmo de estimação começa com a seleção das funções de densidade de probabilidade do modelo gerativo dos dados que serão analisados. A seleção dos modelos mais apropriados para o problema da estimação de orientação é uma questão que foi discutida por todos os pesquisadores da área. Por exemplo, Coughlan e Yuille (2003) utilizaram modelos de probabilidade bastante detalhados, baseados em histogramas, para modelar a intensidade do gradiente. Já Denis, Elder e Estrada (2008) estudaram mais detalhadamente a modelagem do resíduo da orientação dos edgels, que é a grandeza mais relevante para o problema, e aplicaram um modelo de mistura de distribuições de Laplace generalizadas. Já Schindler e Dellaert (2004) utilizaram a distribuição normal, visando transformar o passo  $M$  da sua implementação do algoritmo EM em um problema de quadrados mínimos, e assim permitir a aplicação da técnica de otimização de Gauss-Newton. A Seção 4.3.1 discute em mais detalhes como as técnicas MAP e EM foram empregadas nos métodos existentes de

estimação de orientação de câmera.

Assim como é preciso escolher as distribuições de probabilidade nas técnicas MAP e EM, a M-estimação exige a escolha de uma função  $\rho$ . No *Corisco* a função biquadrada de Tukey foi a principal empregada. A Seção 2.4.2 discutiu alguns procedimentos iterativos que podem ser utilizados com a M-estimação para minimizar a função objetivo, tal como o algoritmo IRLS. Esta possibilidade é interessante do ponto de vista teórico, porque mostra como a M-estimação possui ligações com o algoritmo EM. No *Corisco*, entretanto, a abordagem utilizada para a minimização da função objetivo é a da aplicação direta de algoritmos de otimização. Do ponto de vista da técnica EM, portanto, a abordagem do *Corisco* poderia ser interpretada como sendo mais uma espécie de estimação por GEM do que uma forma de EM mais tradicional.

Do ponto de vista da otimização, o que as técnicas como o IRLS ou outras implementações do EM oferecem são formas diferentes de encontrar o ponto mínimo da função objetivo lidando com as complexidades oferecidas por diferentes funções  $\rho$ , e lidando também com o desconhecimento das classes das observações nos problemas em que isto é relevante. No caso do *Corisco*, determinamos que seria possível aplicar uma técnica de otimização diretamente sobre a função objetivo que foi definida, cuidado da questão da classificação de forma implícita no cálculo da função, e portanto alternativas como o IRLS ou o EM tradicional não seriam estritamente necessárias e poderiam inclusive resultar em um pior desempenho computacional.

É importante ressaltar a diferença que há entre a técnica EM e técnicas como a MAP ou o algoritmo IRLS. Nestas duas últimas define-se claramente a função objetivo a ser minimizada. No caso de uma estimação por EM pode haver uma estimação simultânea dos parâmetros principais que se deseja estimar, e dos chamados parâmetros incompletos, que neste caso estudado são as classes das observações. Assim o funcionamento do EM constitui uma forma de otimização particular, que pode não ser possível de interpretar como uma simples otimização de uma função objetivo. Mas ainda assim o conceito deste tipo de otimização tradicional faz parte do EM, e no momento em que o algoritmo converge é possível enxergá-lo como uma otimização comum. Há casos também em que se pode interpretar uma otimização simples como uma forma de EM, ou de GEM, e é isto que ocorre no *Corisco*.

A Seção 4.3.2 descreve o procedimento adotado para o cálculo da função objetivo do *Corisco*, e o processo de otimização é discutido nas Seções 4.4 e 4.5.

#### 4.3.1 Aplicações de MAP e EM à estimação de orientação a partir de edgels

A Seção 2.4 discutiu o fato de que técnicas de estimação como MAP e EM resultam em uma minimização de erros calculados como na Equação 2.36. Mesmo que de forma implícita, estas técnicas definem de alguma maneira uma função de erro  $\rho$ , e a ligação entre estas técnicas e este fato é evidenciada pela ligação entre o algoritmo IRLS e a

técnica EM. Esta função de erro é aplicada para cada observação em análise, produzindo uma função de erro total que é utilizada como a função objetivo de um processo de otimização que resulta na estimativa dos parâmetros desejados.

Todos os métodos propostos no passado para a estimação de orientação a partir de edgels começam realizando uma extração de observações a partir do gradiente da imagem de entrada que produz um conjunto de edgels com posições  $\mathbf{p}_n$  e direções  $\mathbf{v}_n$ . Todos estes métodos também realizam a seguir alguma forma de minimização de uma função objetivo que é sempre uma soma de erros calculados para cada edgel. E este erro calculado para cada edgel leva em consideração as direções  $\mathbf{v}_{nk}$  preditas sobre cada  $\mathbf{p}_n$  para as projeções de cada direção  $\mathbf{r}_k$  possível das retas do ambiente, obtidas a partir dos parâmetros  $\Psi$  da orientação da câmera. Esta orientação pode ser parametrizada de diferentes formas, e  $\Psi$  pode ser ou um quaternion ou um conjunto de ângulos.

Como todos os métodos anteriores ao *Corisco* utilizaram apenas o modelo de câmera *pinhole*, este cálculo de  $\mathbf{v}_{nk}$  é realizado pela Equação 2.20. E em todos estes métodos anteriores é utilizada a função arco-tangente para realizar um cálculo explícito dos ângulos destes vetores  $\mathbf{v}_{nk}$ , representados aqui pelo símbolo  $\angle \mathbf{v}_{nk}$ .

Nestes métodos existentes os modelos gerativos para a direção de um edgel são misturas de funções de densidade de probabilidade de acordo com as direções  $k$  do ambiente. Cada direção constitui uma classe, e cada classe produz uma observação a partir de uma distribuição probabilística ao redor do valor  $\mathbf{v}_{rk}$  predito. Como resultado temos a fórmula

$$p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n) = \sum_k p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k) \quad (4.4)$$

onde  $c_n$  é a classe do edgel de índice  $n$ , ou a direção no ambiente da linha que produziu este edgel. Estas classes são não apenas as três direções possíveis,  $x$ ,  $y$  e  $z$ , há ainda uma quarta possibilidade de uma direção inválida, não-alinhada a estas três direções. Os valores de  $p(c_n)$  são determinados previamente. As distribuições condicionais de  $\angle \mathbf{v}_n$  dependem do cálculo das direções preditas  $\mathbf{v}_{nk}$ , explicado na Seção 2.1.2, e  $\Psi$  e  $\mathbf{p}_n$  surgem como variáveis condicionais justamente pela presença destas grandezas nas fórmulas para o cálculo de  $\mathbf{v}_{nk}$ .

A função  $p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k)$  pode ser escrita como uma simples distribuição de  $p(\angle \mathbf{v}_n - \angle \mathbf{v}_{nk}(\Psi))$ . O modelo de observação apenas especifica que ocorre uma distribuição probabilística dos valores observados  $\angle \mathbf{v}_n$  ao redor do valor  $\angle \mathbf{v}_{nk}(\Psi)$ , que é calculado para um dado  $\Psi$ , o  $p_n$  de cada amostra e para cada classe  $k$ . Esta distribuição segue uma função específica conhecida, e igual para todas as observações, exceto pelo centro da distribuição definido por  $\angle \mathbf{v}_{nk}(\Psi)$ . O problema é portanto bastante semelhante à determinação de um parâmetro de translação de uma PDF, como no caso dos  $\mu_{1\dots K}$  da Equação 2.43, porém no caso da estimação de orientação estes parâmetros utilizados para definir as distribuições são funções do conjunto de parâmetros  $\Psi$ , que é o que se deseja

estimar.

Em outras palavras, as diferentes funções  $p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k)$  são apenas uma mesma função de distribuição para todas as observações, porém centrada em  $\angle \mathbf{v}_{nk}$ , e por isso podemos pensar apenas em distribuições dos resíduos  $\angle \mathbf{v}_n - \angle \mathbf{v}_{nk}(\Psi)$ . Isto contrasta com modelos mais complexos onde as variâncias destas distribuições poderiam mudar de acordo com  $p_n$  ou  $c_n$ , por exemplo. Um problema tradicional semelhante a este, ao contrário do caso da Equação 2.32, é o do ajuste de uma curva a um conjunto de pontos no espaço. Neste problema, para cada valor observado existe uma distribuição cujo centro varia de acordo com o modelo da curva. A estimativa busca minimizar os resíduos das diferentes medições, sendo que cada uma delas possui um valor de referência diferente, obtido através de uma função dos parâmetros do modelo.

A Equação 4.4 nos primeiros métodos propostos (COUGHLAN; YUILLE, 2003; DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004) incorporava também, na realidade, uma modelagem probabilística do problema da detecção de bordas. Assim o modelo original nestes métodos leva em consideração tanto a direção quanto a intensidade do gradiente da imagem em cada ponto, simultaneamente. Isto requer a existência de uma quinta classe no modelo, que são os pontos que não são uma borda. Apesar de interessante do ponto de vista teórico, esta abordagem acaba não oferecendo uma vantagem significativa ao procedimento de selecionar pixels pela simples limiarização da sua intensidade de gradiente. Assim foi adotado no *Corisco* o mesmo princípio do trabalho de Denis, Elder e Estrada (2008), em que a detecção de bordas é realizada antes do processo de estimativa de orientação. Esta aplicação dos métodos MAP e EM descrita aqui é portanto uma variação da original, que leva em consideração apenas os ângulos das observações.

O modelo gerativo para o conjunto completo dos dados é criado a partir da Equação 4.4. Para isto assume-se que todas as observações são independentes, e então a probabilidade do conjunto de observações é simplesmente o produto da probabilidade de cada observação individual, ou seja, para um conjunto de  $N$  observações temos

$$p(\angle \mathbf{v}_1, \angle \mathbf{v}_2, \dots, \angle \mathbf{v}_N | \Psi, \mathbf{p}_1, \dots, \mathbf{p}_N) = \prod_n \sum_k p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k). \quad (4.5)$$

É interessante observar aqui que esta assunção de independência não é muito exata. Há dependências entre estas variáveis aleatórias na realidade, especialmente entre pontos localizados muito próximos na imagem. Na pesquisa de (SCHINDLER; DELLAERT, 2004) a subamostragem das entradas é justificada não apenas pelo desejo de acelerar os cálculos, mas também para tornar as observações mais independentes. No *Corisco* a subamostragem pela máscara em forma de grade descrita na Seção 4.1 permite especificamente impor uma distância mínima entre as observações, o que alcança de forma ainda melhor o propósito de selecionar conjuntos de pontos relativamente afastados entre si.

A Equação 4.5 modela portanto a probabilidade do conjunto completo de observações selecionadas, sejam elas todos os pixels da imagem (COUGHLAN; YUILLE, 2003), sub-amostragens aleatórias (DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; DELLAERT, 2004) ou pontos encontrados por um detector de bordas (DENIS; ELDER; ESTRADA, 2008), ou ainda o resultado do processo de extração de edgels do *Corisco*.. A partir deste modelo define-se uma função objetivo com o uso do logaritmo da esperança da verossimilhança de  $\Psi$ , como descrito na Seção 2.4.1. Assim temos

$$F(\Psi) = - \sum_n \log \left( \sum_k p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k) \right). \quad (4.6)$$

A fórmula da Equação 4.6 é a função objetivo que é minimizada nos trabalhos de Coughlan e Yuille (2003) e Deutscher, Isard e MacCormick (2002). Mas a definição da expressão exata depende ainda de escolher um modelo para  $p(\angle \mathbf{v}_{nk} | \dots)$ . Duas alternativas utilizadas foram um simples modelo de função “caixote”, uniforme entre dois limiares (COUGHLAN; YUILLE, 2003), e triangular (DEUTSCHER; ISARD; MACCORMICK, 2002). A largura total do suporte destas distribuições tende a ser algo próximo de  $10^\circ$  a  $20^\circ$ , ou seja, um desvio absoluto máximo do centro da distribuição de aproximadamente 0.15 graus radianos.

Os valores pré-estabelecidos de  $p(c_n = k)$  tendem a ser algo como uma mesma probabilidade para cada direção, e uma probabilidade duas vezes maior para a classe de bordas não-alinhadas. Ou seja, 1/5 para cada direção válida, e 2/5 para a classe de bordas não-alinhadas aos eixos do referencial do ambiente. É claro que idealmente a classe de bordas não-alinhadas deveria possuir probabilidade nula, e este valor parece até desafiar a própria validade da assunção de ambiente antrópico. Esta introdução da quarta classe é de fato necessária para tornar a estimativa insensível aos *outliers* criados por direções não-alinhadas, mas sem interferir muito no processo de outra forma.

A introdução desta quarta classe no modelo acaba tendo o efeito de tornar a estimação mais robusta ao efetivamente criar uma função de erro redescendente. A Figura 4.11 mostra a curva da log-verossimilhança da distribuição Gaussiana de uma única variável, que é simplesmente uma parábola. Em comparação, abaixo dela encontra-se a curva encontrada em um modelo de mistura da Gaussiana com uma distribuição uniforme, que é exatamente a forma como a quarta classe atua na Equação 4.6. Podemos ver claramente que esta abordagem produz uma função de erro redescendente, notando que a função permanece aproximadamente quadrática perto do centro, mas eventualmente se torna constante.

As técnicas de otimização empregadas para encontrar o  $\Psi$  que minimiza a Equação 4.6 nestas pesquisas pioneiras da área foram relativamente rústicas. Os autores empregaram simples buscas determinísticas ou aleatórias, o que permite sem dúvida avaliar

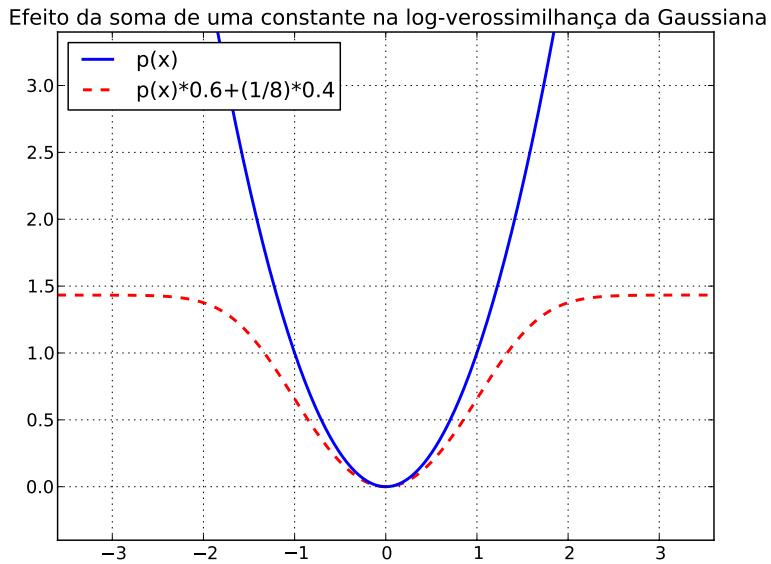


Figura 4.11: Criação de uma função de erro robusta em um modelo de mistura com Gaussiana e uma distribuição uniforme. A curva contínua é o erro quadrático convencional obtido de uma distribuição Gaussiana pura. A curva tracejada é obtida de um modelo de mistura de uma Gaussiana e uma distribuição uniforme.

o funcionamento da técnica, mas resulta fatalmente em processos relativamente pouco eficientes ou precisos. A necessidade do cálculo da função logaritmo por si só já constitui um aspecto negativo desta abordagem do ponto de vista de velocidade de computação, o que também vale para a função arco-tangente. Modificações da expressão seriam necessárias tanto para acelerar os cálculos quanto para viabilizar o uso de técnicas de otimização contínua.

No método proposto por Schindler e Dellaert (2004) a expressão da função objetivo é modificada de forma que este logaritmo é eliminado. Em primeiro lugar este autores aplicaram a chamada desigualdade de Jensen, que diz que

$$\log(E[x]) \leq E[\log(x)], \quad (4.7)$$

onde  $E[\dots]$  representa o operador esperança. Esta modificação pode ser realizada na expressão porque preserva a localização do ponto extremo sendo buscado. Desta forma os logaritmos de somatórios na Equação 4.6 dão lugar a somatórios de logaritmos, e a expressão passa a ser

$$-\sum_n \sum_k p(c_n = k) \log(p(\angle \mathbf{v}_n | \boldsymbol{\Psi}, \mathbf{p}_n, c_n = k)). \quad (4.8)$$

Outra modificação introduzida por estes pesquisadores foi o uso de um modelo probabilístico baseado na distribuição Gaussiana, o que finalmente fez com que esta expressão

se tornasse uma minimização de erros quadráticos, produzindo a nova expressão para a função objetivo

$$F(\Psi) = \sum_n \sum_k p(c_n = k) (\angle \mathbf{v}_n - \angle \mathbf{v}_{nk}(\Psi))^2. \quad (4.9)$$

O símbolo  $\angle \mathbf{v}_n$  é o ângulo da direção medida do edgel  $n$ , calculado através da função arco-tangente. Já  $\angle \mathbf{v}_{nk}(\Psi)$  representa o ângulo do vetor predito  $\mathbf{v}_{nk}$ , calculado a partir da orientação hipotética  $\Psi$ , das coordenadas do edgel  $\mathbf{p}_n$  e para cada direção  $\mathbf{r}_k$  possível no ambiente. Esta fórmula ignora a classe de retas não-alinhadas, porque ela possui uma PDF uniforme, independente da orientação da câmera, e portanto não contribui para a estimativa neste momento do processo.

Além destas modificações da função objetivo, Schindler e Dellaert (2004) introduziram ainda outra grande modificação ao processo de estimativa, que é o que marca a troca do método MAP pelo EM. Trata-se da atualização dos valores de  $p(c_n)$  que ocorre no passo  $E$ , de forma similar ao apresentado na Equação 2.46. Os novos valores de  $p(c_n)$  são obtidos pela fórmula

$$p(\angle \mathbf{v}_n | \Psi, \mathbf{p}_n, c_n = k) p(c_n = k), \quad (4.10)$$

que produz um conjunto de valores que devem ser posteriormente normalizados para que somem 1.

Se houvessem no problema apenas duas classes, de dados *inliers* e *outliers*, este processo de recalcular  $p(c_n)$  a partir de uma verossimilhança seguido da solução do problema de quadrados mínimos ponderados da Equação 4.9 tornaria o processo muito similar ao algoritmo IRLS, realizando uma minimização de erros calculados pela função  $\rho$  apresentada na Figura 2.18. Isto não ocorre, porém, porque existem as múltiplas classes. É importante também deixar claro que apenas Schindler e Dellaert (2004) efetivamente transformaram a função objetivo em uma clara soma de erros quadráticos ponderados, enquanto os métodos anteriores calculavam a função de verossimilhança. Esta porém não é a diferença mais significativa deste método, mas sim o fato de que ocorrem as repetidas atualizações dos valores de  $p(c_n)$ . Esta é a diferença mais significativa entre o processo resultante da técnica EM e da técnica MAP, que foi empregada por Coughlan e Yuille (2003) e por Deutscher, Isard e MacCormick (2002).

Uma característica interessante desta atualização dos valores de  $p(c_n)$  é que, como já foi mencionado anteriormente, conforme o valor da probabilidade para uma das classes vai crescendo, ele tende a se tornar preponderante. Após a convergência do processo, é comum que o valor de uma única classe tenha se tornado 1, enquanto o valor para as outras classes se torna nulo. Isto apenas não acontece para observações que se encontram sobre a chamada *linha de fuga*, que liga dois pontos de fuga. Nesta situação podem haver duas direções do ambiente que produzem edgels com direções muito próximas, e não é raro que isto seja verificado em uma fotografia. Mas ainda assim as observações deste tipo são sempre poucas em uma amostragem, e para a maioria das observações de

bordas alinhadas aos eixos do ambiente, o valor de  $p(c_n)$  será bastante próximo de 1 para uma das direções.

É interessante notar também que neste cálculo de  $p(c_n)$  do passo  $E$  leva-se em consideração a classe de bordas não-alinhadas, que foi desprezada no passo  $M$ . O resultado disto, considerando que a verossimilhança para esta classe é um valor constante, é que se nenhum valor de verossimilhança para as outras classes for comparável a este valor constante da quarta classe, ela irá concentrar toda a probabilidade  $p(c_n)$  para aquela amostra. Como consequência, esta amostra irá parar de influir no cálculo da Equação 4.9, já que os pesos para esta amostra serão todos próximos de 0. Este caso corresponde a um resíduo de alta magnitude passando pela função redescendente apresentada na Figura 4.11.

Este mecanismo de classificação das observações é algo que pode ser observado em diversas aplicações da técnica EM. Mas no caso específico da estimação de orientação há um outro mecanismo que contribui para este fenômeno, que não é explorado diretamente. Acontece que as direções  $r_k$  não são independentes, como poderia ser em um problema onde direções genéricas no ambiente são consideradas. Existe a condição de ortogonalidade entre as direções. Assim a alteração dos parâmetros de orientação  $\Psi$  para alinhar um certa direção a um certo conjunto de pontos associados irá tender a corrigir também as outras direções. Ou seja, a atualização dos parâmetros devido a uma única classe tende a corrigir também os parâmetros relativos às outras classes.

Como consequência destas relações de dependência entre as classes, não é de se esperar que este processo de estimação apresente muitos desafios para a sua convergência. Comparado a um problema da estimação dos parâmetros de um modelo probabilístico de mistura de Gaussianas, que é um exemplo notório de aplicação da técnica EM, o problema da estimação de orientação deveria representar um desafio menor. No caso da mistura de Gaussianas os parâmetros das diferentes classes, que são por exemplo os centros de cada Gaussiana subjacente do modelo, são totalmente independentes. Já no caso da estimação de orientação os parâmetros de cada classe seriam os parâmetros de cada direção  $r_k$ , mas existem na realidade menos parâmetros livres sendo controlados, que são apenas os três parâmetros livres de  $\Psi$ .

Tendo em vista estas fortes restrições que existem no problema específico da estimação de orientação, a aplicação da técnica EM de forma tradicional oferece uma flexibilidade desnecessária. Isto incentivou o uso no *Corisco* de uma forma mais simples de realizar a classificação das observações. Ao invés de manter uma tabela de valores contínuos entre 0 e 1 para cada par de classe e observação, uma única classe é eleita a cada momento como a classe correta para a observação, e utilizada na conta da função objetivo. O procedimento adotado possui portanto a peculiaridade oferecida pela técnica EM de realizar a classificação dos dados e de levar isto em consideração nas contas, ao contrário da técnica MAP onde utiliza-se apenas uma distribuição constante de probabilidades de classe *a priori*. Porém no *Corisco* os cálculos realizados são mais simples do que na aplicação

de EM, e como será visto a seguir, evitam a operação exponencial que é necessária para o cálculo da verossimilhança na Equação 4.10. Além disso a classificação passa a ser o resultado de um processo não-iterativo, enquanto no EM é iterativo. Esta modificação da estimação por EM com a realização de classificações absolutas pode ser vista como algo similar ao que é feito no algoritmo das  $k$ -médias, cuja ligação com a estimação por EM é bem-conhecida.

Antes de concluir esta seção, devemos mencionar a pesquisa de Denis, Elder e Estrada (2008). Estes autores testaram várias pequenas modificações aos métodos existentes, e testaram em especial um novo modelo gerativo, com funções de Laplace generalizadas ao invés de Gaussianas. Eles também aplicaram uma técnica de otimização contínua mais sofisticada para o problema, o algoritmo BFGS, tanto para a estimação por MAP quanto EM. Estas duas modificações, além da sub-amostragem por detecção de borda, levaram a um desempenho superior ao do método de Coughlan e Yuille (2003) tanto em precisão quanto em velocidade. Já o uso da técnica EM resultou naquela pesquisa em um desempenho mais lento mas com uma precisão apenas comparável à obtida com a técnica MAP.

A modificação do modelo gerativo torna as expressões utilizadas por Denis, Elder e Estrada (2008) menos convenientes do que no método de Schindler e Dellaert (2004), já que o passo  $M$  deixa de ser um problema de mínimos quadrados. Do ponto de vista da teoria de estimação probabilística mais básica, é interessante que os resíduos do problema tenham sido modelados de forma mais cuidadosa. Buscar utilizar modelos precisos para as distribuições de probabilidade de qualquer problema é certamente uma atitude louvável. Porém mantemos o argumento de que neste problema o uso de uma função de erro redescendente, além da realização da classificação das observações, são as características mais relevantes que devem ser incorporadas ao processo de estimação, enquanto o formato exato da função de distribuição possui uma importância secundária. Esta foi a abordagem utilizada na criação do *Corisco*.

#### 4.3.2 Função objetivo utilizada no *Corisco*

O cálculo da função objetivo utilizada no *Corisco* começa com os cálculos das três direções preditas possíveis  $v_{nk}$  para cada edgel, assim como ocorre nos métodos anteriores. Este cálculo é realizado como descrito na Seção 2.1.2, permitindo o uso de qualquer modelo de câmera, inclusive com fortes distorções. A seguir são calculados resíduos para cada uma destas direções preditas.

Aqui surge uma diferença importante do *Corisco*. Enquanto todos os métodos anteriores utilizam a função arco-tangente para encontrar ângulos, que são então subtraídos dos ângulos medidos, no *Corisco* são utilizados apenas vetores normalizados. Tradicio-

nalmente os resíduos são calculados por

$$\angle \mathbf{v}_n - \angle \mathbf{v}_{nk}. \quad (4.11)$$

A proposta é utilizar o produto vetorial de cada direção predita  $\mathbf{v}_{nk}$  pela direção  $\mathbf{u}_n$  do gradiente do edgel, que é ortogonal à direção  $\mathbf{v}_n$  do edgel, ou seja

$$\mathbf{u}_n \mathbf{v}_{nk} = (\mathbf{v}_n^\perp) \mathbf{v}_{nk} = \sin(\angle \mathbf{v}_n - \angle \mathbf{v}_{nk}) \simeq \angle \mathbf{v}_n - \angle \mathbf{v}_{nk}. \quad (4.12)$$

O uso da multiplicação vetorial elimina o cálculo da função arco-tangente. No lugar disto surge a necessidade de se calcular o recíproco de uma raiz quadrada para se encontrar fatores de normalização para os vetores. Isto representa uma problema menor, porque esta operação é bastante comum em aplicações de Computação Gráfica e outras áreas, e muitos processadores acessíveis possuem hoje instruções especializadas para realizar este cálculo, inclusive de forma vetorizada, tornando esta alternativa bem mais atraente.

É interessante notar que esta modificação modifica o formato da PDF que modela os resíduos do problema, porque o valor calculado é o seno da diferença dos ângulos ao invés da própria diferença. Para ser extremamente rigoroso com relação à escolha do modelo da PDF seria necessário escolher uma das duas grandezas para modelar, ou o erro angular ou o seno dele que é calculado através da multiplicação vetorial. Mas em primeiro lugar, a aproximação  $\sin(x) \simeq x$  é notoriamente boa para a faixa de valores esperados, em torno de 0.10 ou 0.15 graus radianos, e portanto a distorção é muito pequena. Em segundo lugar, como está sendo adotada a técnica de M-estimação, a função de erro já está sendo escolhida de forma quase arbitrária, e não guiada pela PDF, então mesmo que a faixa dos valores de resíduo fosse grande o bastante para que a curvatura da função seno fosse relevante, a preocupação com o formato exato da PDF já se tornou um fator de menor prioridade no projeto do procedimento de estimação. Em outras palavras, apesar da introdução desta função seno na medição de resíduo que é analisada, a aplicação da M-estimação prossegue sem alterações porque a função seno está apenas distorcendo uma PDF cujo formato exato já não está sendo modelado.

Uma vez que os resíduos tenham sido calculados para cada edgel, utiliza-se então uma função  $\rho$  para encontrar valores de erro. No *Corisco* foi utilizada a função biquadrada de Tukey. O parâmetro de escala utilizado na função é em torno de 0.10 a 0.15 graus radianos, consistente com as pesquisas anteriores. Outras funções foram implementadas também, e podem ser facilmente selecionadas para uso no programa desenvolvido, mas nesta pesquisa foi dada maior atenção apenas à função de Tukey, e não foi observada alguma vantagem significativa no uso das alternativas.

Para cada edgel  $n$  e pra cada direção  $k$  calcula-se portanto um erro

$$\rho(\mathbf{u}_n \mathbf{v}_{nk}). \quad (4.13)$$

Estando em posse destes erros escolhe-se então para cada edgel a direção que resulta no menor erro, e esta é eleita como a classe daquele edgel para aquele  $\Psi$  utilizado para calcular as direções preditas  $\mathbf{v}_{nk}$ . O valor do erro para esta classe e para cada observação é então acumulado para produzir enfim o valor de erro total da função objetivo. Podemos portanto escrever uma fórmula fechada para a função objetivo, que é dada por

$$F(\Psi) = \sum_n \min_k \rho(\mathbf{u}_n \mathbf{v}_{nk}). \quad (4.14)$$

A operação  $\min_k$  retorna o menor dos 3 valores de erro calculados. Os erros são dados pela função  $\rho$ , tendo como argumento os resíduos encontrados para cada classe  $k$ . Os resíduos são obtidos através da multiplicação vetorial de cada vetor observado  $\mathbf{u}_n$  com cada uma das três direções preditas  $\mathbf{v}_{nk}$ . Estas direções preditas são calculadas para cada  $\mathbf{p}_n$  específico a partir das direções  $\mathbf{r}_k$ , que por sua vez são obtidas da orientação hipotética  $\Psi$ .

Esta função objetivo possui muitas semelhanças com as apresentadas nas Equações 4.6 e 4.9. Em primeiro lugar, todas estas funções são somatórios de parcelas calculadas para cada uma das observações. Em segundo lugar, estas parcelas dependem de resíduos calculados a partir dos vetores preditos  $\mathbf{v}_{nk}$ . As diferenças residem na expressão para o cálculo das parcelas de erro e no fato de haver ou não algum processo de classificação das observações.

Como foi visto na Seção 4.3.1, a introdução da quarta classe de bordas não-alinhadas e com distribuição uniforme nas aplicações de MAP e EM para o problema da estimativa de orientação tem o efeito de fazer com que as estimativas tenham uma natureza de minimização de uma função de erro redescendente. No caso da técnica MAP, a parcela de erro relativa a uma observação  $n$  tem a forma de um logaritmo de uma função de verossimilhança dada pela mistura de uma distribuição de suporte limitado com uma distribuição uniforme. Isto causa um efeito de limitação do valor do erro, como demonstra o gráfico da Figura 4.11. No caso da técnica EM, a ponderação do erro quadrático pelos valores  $p(c_n)$ , que são obtidos através de uma função de verossimilhança Gaussiana, produz a função de erro demonstrada na Figura 2.18. No método proposto este mesmo efeito é causado de forma mais simples, pelo uso direto de uma função de erro redescendente, a função biquadrada de Tukey.

Como apenas as três classes relativas às três direções possíveis são levadas em consideração na Equação 4.14, não havendo uma quarta classe para retas não-alinhadas, e como é obrigatório escolher uma das três classes para uma certa observação para calcular a sua parcela do erro total na função objetivo do Corisco, haverá fatalmente ocasiões em que serão escolhidas classes erradas. Existirão também bordas não-alinhadas que sempre estarão contribuindo uma parcela de erro. É o uso da função redescendente que permite evitar estes problemas, pois estes erros de classificação implicarão em resíduos de grande

magnitude, que são assim jogados na região constante da função  $\rho$ , fazendo com que aquela observação interfira na estimativa já que a derivada do erro desta amostra com relação a  $\Psi$  passará a ser nula.

Com relação ao processo de classificação, no caso da técnica MAP isto não é realizado. A princípio isto poderia não ser um problema, sendo que para a grande parte das observações a verossimilhança da classe correta terá um valor alto, enquanto as verossimilhanças das classes incorretas teriam valores próximos de zero. Entretanto, há ao menos um problema que se pode apontar de forma bastante clara no uso do método MAP. Existe a possibilidade da estimativa tender a convergir para configurações que maximizam o número de amostras que possam ser classificadas em duas classes simultaneamente. Ou seja, a estimativa pode tender a escolher orientações que maximizam a quantidade de observações sobre uma linha de fuga. Este é um efeito que foi observado na prática durante esta pesquisa, e que é evitado pela realização de classificação, seja pelo método EM ou seja pelo método de classificação utilizado no *Corisco*.

Outros motivos apontados por Schindler e Dellaert (2004) para recorrer ao uso de EM estão ligados na realidade a fatores como a possibilidade de se estender a técnica no futuro para problemas maiores, ou ainda à possibilidade de aplicação de um modelo como o de campos aleatórios de Markov para a classificação das observações. Excetuando-se estas possibilidade de extensão, a justificativa para utilizar o EM se torna portanto apenas o fato do que ele realiza alguma forma de classificação das observações durante a estimativa, ao contrário do MAP. O desenvolvimento do *Corisco* buscou oferecer uma forma alternativa de realizar a classificação das observações neste problema, através de um processo mais simples do que a implementação de EM apresentada por Schindler e Dellaert (2004).

O *Corisco* se diferencia portanto dos outros métodos existentes para a estimativa de orientação a partir de edgels em primeiro lugar pelo avanço no esforço em simplificar as expressões. Foi realizada uma eliminação da função log, como já realizada anteriormente, mas no *Corisco* também foi eliminado qualquer uso da função exponencial, que ainda era utilizada nos passos  $E$  dos métodos existentes. Além disso abandonou-se o uso da função arco-tangente, e a única operação relativamente complexa remanescente é a que calcula o recíproco de uma raiz quadrada, que é uma operação para a qual existem boas técnicas de cálculo para aplicações de alto desempenho.

A função objetivo proposta apresenta características similares às anteriores, mas sua definição se baseia diretamente na teoria de Estatística Robusta. Desta forma o *Corisco* dispensa toda a modelagem probabilística apresentada nas pesquisas anteriores, que apesar de relevantes do ponto de vista teórico, acabam sendo um impedimento para a evolução desta técnica de estimativa de orientação. A abordagem tradicional desvia o foco das pesquisas para questões como a escolha dos modelos probabilísticos, e como modificar as expressões obtidas pela teoria para tornar o seu cálculo mais simples. Porém esta escolha do modelo exato dos resíduos pode não ser algo prioritário, no sentido de que é

bastante possível, e até esperado, que o uso de modelos diferentes possam resultar em desempenhos igualmente satisfatórios. Isto é comprovado pelo fato de que tantos modelos diferentes já foram utilizados no passado, sempre permitindo realizar a estimativa desejada mesmo que com diferentes níveis de precisão.

Além desta escolha do modelo probabilístico não ser algo crítico, é impossível garantir que os dados obtidos de toda e qualquer imagem jamais analisada por uma aplicação deste método exibam exatamente as mesmas características do modelo probabilístico selecionado durante o desenvolvimento da aplicação. É salutar, portanto, levar em consideração esta possibilidade quase certa de que na prática será necessário realizar a estimativa a partir de dados que não foram produzidos com exatamente o mesmo modelo utilizado na implementação do método. A abordagem do problema pela Estatística Robusta pretende permitir que isto seja levado em consideração.

A Estatística Robusta permite ainda desenvolver o processo de estimativa na direção oposta da abordagem tradicional, permitindo que sejam mais facilmente criados processos de estimativa com características desejáveis específicas. Em técnicas como a EM é feita uma modelagem “de baixo para cima”, em que se desenvolve um modelo probabilístico, e a seguir são feitas manobras como a aplicação da função logaritmo e da desigualdade de Jensen para se tentar criar uma fórmula que não só realize a estimativa desejada, mas que também ofereça vantagens do ponto de vista de complexidade computacional. Já na abordagem utilizada no *Corisco* o processo é desenvolvido “de cima para baixo”. A função objetivo tem por princípio várias características convenientes do ponto de vista computacional, e além disso pode ser definida de forma a também apresentar as características necessárias para permitir estimativas bem-sucedidas, ou seja, as características existentes nas funções objetivo encontradas através das técnicas mais tradicionais tal como o fato da função de erro ser redescendente.

A seleção da função de erro é onde se busca atender os requisitos do modelo probabilístico, mas apenas aproximando as funções de distribuição ideais subjacentes, sem muita preocupação em implementá-las com exatidão. Como dito anteriormente, isto é aceitável pois nem é mesmo possível garantir que os dados analisados na prática sempre seguem uma determinada distribuição ideal. A escolha da função de erro também permite buscar vantagens computacionais, ligadas por exemplo ao fato de se poder aplicar técnicas de otimização contínua, ou utilizar exclusivamente operações com baixo custo computacional.

As Seções 4.4 e 4.5 a seguir discutem enfim como realizar o processo de otimização da função objetivo do *Corisco*, descrita nesta seção. Mas deve ficar claro que esta mesma função objetivo poderia ser utilizada de outras formas em diferentes aplicações. Estimativas iniciais dos parâmetros desejados podem ser conhecidas, por exemplo, e podem haver restrições extras que podem ser incorporadas no processo de estimativa, restringindo o domínio sobre o qual se realiza a otimização. Portanto a função objetivo descrita nesta seção

poderia também ser utilizada com processos de otimização diferentes do que será explicado a seguir. Esta forma de calcular a função objetivo poderia até mesmo ser utilizada também com formas de extração de características geométricas diferentes das utilizadas no método proposto aqui. Por exemplo, seria possível adaptar esta função objetivo para outras formas de cálculo do resíduo, por exemplo, com base nos vetores normais das bordas (ANTONE; TELLER, 2000), ou nas distâncias das extremidades de segmentos de reta a uma reta predita (ROTHER, 2002), ou até mesmo a partir de filtros direcionais (AUDIBERT; PONCE, 2009).

#### 4.4 Estimação de parâmetros por busca aleatória

O *Corisco* estima a orientação de câmera em ambientes antrópicos a partir de edgels, e para isto se baseia na otimização da função objetivo definida na Seção 4.3.2, que utiliza um conjunto de edgels extraídos da imagem de entrada conforme descrito na Seção 4.1. Existem dois estágios neste processo de otimização utilizado no *Corisco*, como mostra a Figura 4.1. A primeira etapa, apresentada nesta seção, busca enfrentar o problema da convergência global e permitir que a otimização seja realizada sem qualquer estimativa inicial da solução. Ela é baseada em uma espécie de busca aleatória, similar a o que é realizado no algoritmo RANSAC discutido na Seção 2.3.1. O segundo passo, explicado na Seção 4.5, é onde finalmente ocorre uma otimização contínua. Este passo é baseado no algoritmo FilterSQP (FLETCHER; LEYFFER, 2002), que é uma adaptação para otimização com restrições do tradicional algoritmo de otimização baseado no método de Newton-Raphson.

Estas duas etapas da otimização são complementares. A otimização contínua da segunda etapa costuma apresentar dificuldades para convergir se for inicializada muito distante da solução. Já a otimização por busca aleatória não depende de uma estimativa inicial, porém sua convergência fica mais lenta ao longo do tempo. Desta forma este método do primeiro passo eventualmente torna-se ineficiente, mas mais do que isso, este método não é capaz de representar qualquer solução possível, e assim é inherentemente impreciso. A busca aleatória é portanto ideal para iniciar o processo, por dispensar uma estimativa inicial, e o seu resultado impreciso serve para inicializar a otimização contínua.

Existem aplicações onde a precisão obtida com poucas iterações do primeiro passo poderia bastar. Há casos também onde pode haver uma estimativa inicial, como em um problema de rastreamento de câmera em um robô móvel, e assim seria possível dispensar o primeiro passo. O *Corisco* pode ser portanto adaptado para diferentes aplicações, mas no caso estudado durante esta pesquisa, para o qual o *Corisco* foi desenvolvido, ambos passos são necessários.

A forma ideal de realizar esta fusão dos dois métodos de otimização, que é feita aqui através da simples alternação para o segundo método após um número fixo de iterações do primeiro, é algo que ainda pode ser melhor pesquisado. O problema da otimiza-

ção global é reconhecidamente complexo, e o *Corisco* demonstra uma forma de resolver este problema neste cenário específico, onde existem características particulares a serem exploradas. Por exemplo, é possível explorar o fato de que o domínio onde ocorre a otimização é limitado, e que a função objetivo é diferenciável.

Esta tática de otimização baseada em uma busca aleatória seguida de outra forma de otimização já foi aplicada em outras aplicações de visão (CHUM; MATAS; OBDRZALEK, 2004; ROSTEN; LOVELAND, 2009; TARDIF, 2009), embora nenhum método baseado em edgels tenha utilizado algo similar. Coughlan e Yuille (2003) utilizou uma busca determinística, Deutscher, Isard e MacCormick (2002) utilizou apenas uma busca aleatória pelo espaço de parâmetros, não guiada pelos dados. Já Schindler e Dellaert (2004) e Denis, Elder e Estrada (2008) investigaram também o uso de otimização contínua a partir de boas estimativas iniciais da solução.

No método de Deutscher, Isard e MacCormick (2002) a busca aleatória empregada oferece boas características de convergência global. Já nos outros casos, os autores reconhecem que é preciso inicializar os procedimentos com boas estimativas iniciais para obter bons resultados, e são exploradas condições tal como o fato de que as rotações de câmera nos testes realizados nestas pesquisas foram predominantemente ao redor do eixo vertical. No desenvolvimento do *Corisco* buscou-se alcançar uma maior flexibilidade. O *Corisco* não depende de estimativas iniciais, e não assume qualquer restrição às soluções possíveis.

O método de otimização por busca aleatória proposto no *Corisco* faz parte da família do algoritmo RANSAC, discutido na Seção 2.3.1 (CHOI; KIM; YU, 2009; OLSON, 2001). O RANSAC permite lidar com o problema da classificação de dados simultaneamente à estimação de parâmetros de modelos geométricos. Um exemplo clássico deste tipo de problema é um conjunto de pontos ao qual é preciso ajustar um conjunto de linhas retas. As observações são os pontos, e é preciso determinar quais conjuntos de pontos fazem parte de uma mesma reta, e quais pontos devem ser simplesmente ignorados. Se estas classificações fossem conhecidas, seria possível apenas estimar os parâmetros de cada reta a partir de cada conjunto de observações considerado separadamente. É a necessidade de classificar os dados que torna necessário o uso do RANSAC, tanto para determinar quais pontos fazem parte de quais retas, quanto quais pontos são *outliers* que devem ser descartados.

Como explicado na Seção 2.3.1, no problema do ajuste de retas seriam sorteados pares de pontos para produzir modelos geométricos hipotéticos, e a qualidade de cada hipótese é avaliada através de um procedimento que compara cada modelo com todo o conjunto de dados observados. Para o problema do ajuste de retas o procedimento é geralmente um cálculo de uma função de erro total baseada nas distâncias de cada ponto até a reta hipotética testada. Uma reta que apresenta um erro baixo é mantida como uma boa estimativa de solução. O procedimento utilizado para avaliar um modelo pode ser

sempre aproveitado para realizar uma classificação dos pontos. Por exemplo, um *outlier* pode ser definido como um ponto localizado a uma determinada distância mínima da reta estimada.

Em problemas mais convencionais, como na extração de retas, as observações são classificadas ou como sendo um *outlier*, ou como pertencendo a algum dos modelos geométricos cujos parâmetros devem ser estimados. Este tipo de classificação também é a que ocorre no problema da extração de pontos de fuga, discutido na Seção 2.2.3. Cada observação, que podem ser uma linha reta, ou um segmento de reta ou um edgel, é classificada de acordo com o seu ponto de fuga, ou seja, de acordo com a direção da reta no ambiente que a originou, e cada conjunto destas observações permitiria estimar esta direção assim como um conjunto de pontos permite estimar os parâmetros de uma reta no problema mencionado anteriormente.

A condição de ambiente antrópico, de que as direções das retas no ambiente são mutuamente ortogonais, modifica o problema de forma significativa. Há agora um vínculo entre os parâmetros dos modelos, e ao invés de estimar os parâmetros de cada direção separadamente, deseja-se estimar a orientação da câmera, de onde é possível calcular os parâmetros de todos os modelos geométricos individuais. Isto significa que apesar das observações ainda serem classificadas de acordo com as diferentes direções possíveis, há na realidade um único modelo geométrico cujos parâmetros são procurados.

No caso da extração de retas uma variação correspondente seria a extração de um conjunto de retas de características específicas a partir de um conjunto de pontos. Por exemplo, suponha que se sabe que os pontos observados fazem parte de um quadrado de dimensões conhecidas. Neste caso há quatro retas que devem ser extraídas, com distâncias e orientações relativas determinadas pela restrição de que as retas formam os lados de um quadrado. Os pontos ainda são classificados de acordo com a reta a que pertencem, que são cada lado do quadrado, mas o modelo geométrico a ser extraído é o quadrado completo, e seus parâmetros são dados por uma translação e uma rotação. Assim temos um único modelo geométrico sendo extraído, mas composto por um conjunto de entidades de nível mais baixo que produzem as observações.

No problema da estimação de orientação em ambientes antrópicos as classes são as três direções possíveis das retas no ambiente. Um par de edgels de uma mesma classe pode ser utilizado para calcular a direção  $r$  das retas originais no ambiente. Para isto basta encontrar, para cada edgel, o vetor normal do plano constituídos a partir dele e pelo ponto focal da câmera, o chamado *plano de interpretação* de uma reta sobre a imagem. A direção  $r$  é obtida a partir de um par de edgels da mesma classe através do produto vetorial de suas normais correspondentes.

Todos os vetores normais dos edgel de uma mesma classe são ortogonais à direção  $r$  desta classe. Portanto o conjunto de todos os vetores normais possíveis, das três classes existentes, formam sempre três círculos máximos mutuamente ortogonais na esfera

Gaussiana definida ao redor do ponto focal, sendo que a posição destes círculos varia de acordo com a orientação da câmera. O conjunto de todos os edgels da imagem forma um conjunto de pontos sobre a esfera Gaussiana, e um par de normais selecionado ao acaso define um círculo máximo sobre a esfera. O problema é bastante similar ao da extração de linha retas, portanto porém os pontos se encontram sobre uma esfera, e ao invés de retas os modelos são círculos máximos sobre a esfera.

O cálculo do vetor normal de um edgel depende do modelo de câmera. O vetor normal  $\mathbf{n}_k$  de um edgel localizado na posição  $\mathbf{p}_k$  da imagem e com direção de gradiente  $\mathbf{u}_k$ , ortogonal à direção da projeção da reta naquele ponto, é dada pela fórmula

$$u_k^x \mathbf{J}_k^x + u_k^y \mathbf{J}_k^y, \quad (4.15)$$

onde  $\mathbf{J}_k^x$  é a linha da Jacobiana que possui as derivadas da coordenada  $x$ , e  $\mathbf{J}_k^y$  são as derivadas da coordenada  $y$ .

Como discutido anteriormente, em aplicações onde se deseja encontrar as direções das retas existentes no ambiente — ou talvez a localização dos pontos de fuga em uma imagem de projeção pontual — sem a imposição da restrição de ortogonalidade das direções, a estimação de cada direção é feita separadamente. Um algoritmo de busca aleatória como o descrito aqui funcionaria apenas selecionando pares de observações para produzir hipóteses de direções possíveis, que seriam então comparadas ao conjunto total de dados. O desconhecimento do número de modelos possíveis e a inexistência de qualquer restrição torna este problema muito similar ao da extração de um conjunto de retas a um conjunto de pontos sobre um plano.

Já no caso da estimação de orientação em ambientes com linhas em três direções mutuamente ortogonais há apenas um modelo a ser encontrado, que é o conjunto das três direções das retas no ambiente parametrizadas por uma matriz de rotação. O maior número de parâmetros livres neste modelo, comparado a uma única direção individual, implica que ao menos três observações precisam ser utilizadas para se encontrar uma hipótese de modelo. No método desenvolvido isto foi realizado da seguinte forma:

- Um par de observações são selecionadas, e assume-se que elas são da mesma classe. O produto vetorial dos vetores normais destas observações produz  $\mathbf{r}_x$ , e então encontra-se um quaternion qualquer que produza esta mesma direção para esta classe.
- Uma terceira observação diferente é selecionada, que se assume ser de uma segunda classe. Ela é testada para garantir que a normal deste edgel seja minimamente distante da direção  $\mathbf{r}_x$ . O quaternion produzido anteriormente é então rotacionado ao redor da direção  $\mathbf{r}_x$  até que  $\mathbf{r}_y$  se torne ortogonal à normal da terceira observação.

A busca aleatória se dá através de sucessivos sorteios de triplas de observações, produ-

zindo hipóteses de orientação como explicado anteriormente. Para cada hipótese calculase o valor da função objetivo discutida na Seção 4.3.2. Os parâmetros que produzem o menor valor encontrado para todas as orientações testadas são então simplesmente retidos como sendo a melhor estimativa. O número de testes é pré-determinado, quanto mais testes, maior é a probabilidade de se encontrar uma solução a uma certa distância mínima da solução ótima.

Este procedimento de busca é relativamente simples comparado a algumas modificações ao RANSAC já existentes. Porém é ao menos um procedimento melhor do que uma procura aleatória pelo espaço de parâmetros, como realizado por Deutscher, Isard e MacCormick (2002), porque a distribuição dos dados é explorada. O *Corisco* também utiliza uma função de erro mais sofisticada do que a utilizada na versão mais tradicional do RANSAC, onde ocorre apenas uma contagem do número de *inliers*, definidos como pontos existentes a uma distância mínima dos modelos geométricos.

Outras alternativas ao RANSAC como o algoritmo RUDR (OLSON, 2001) ou ainda o RANSAC com otimizações locais (CHUM; MATAS; OBDRZALEK, 2004) buscam em primeiro lugar lidar com o fato de que no algoritmo original apenas as hipóteses produzidas a partir dos grupos mínimos de observações podem ser uma solução. Como no método proposto este passo de estimação será seguido por uma otimização contínua, não faz muito sentido preocupar-se em implementar este tipo de técnica, a não ser que se substituam ambos os passos por um verdadeiro procedimento integrado de estimação, que possa substituir os dois passos por inteiro.

O primeiro passo de estimação do método proposto é portanto esta busca realizada a partir de conjuntos mínimos de amostras sorteadas dos dados sendo analisados. Este passo não requer nenhum tipo de estimativa inicial da solução, mas apenas o conhecimento do modelo de câmera. Cada conjunto de parâmetros sorteado é testado, de acordo com a função objetivo, e a melhor estimativa é retida. O passo seguinte irá partir desta estimativa da solução para encontrar o mínimo desta mesma função objetivo, porém utilizando um método de otimização contínua.

#### **4.5 Estimação de parâmetros por otimização não-linear**

O algoritmo de otimização contínua adotado no *Corisco* é o FilterSQP, desenvolvido por Fletcher e Leyffer (2002). A forma como a otimização é realizada neste passo da estimação possui a característica peculiar de que a função objetivo é definida sobre todo o espaço dos quaternions, incluindo quaternions não-normalizados. Apesar de todo o espaço quadridimensional poder ser explorado durante o processo de otimização, a solução encontrada deve se restringir ao conjunto dos quaternions normalizados, que representam rotações tridimensionais. Para isto é necessário que o processo empregue alguma técnica de otimização com restrições, que é justamente o que o FilterSQP oferece. Isto contrasta com técnicas mais usuais onde a otimização não contém restrições, e é realizada livremente

sobre um espaço tridimensional de parâmetros que modelam uma rotação, utilizando por exemplo ângulos de Euler.

Um problema sem restrições segue a fórmula

$$\operatorname{argmin}_{\Psi} f(\Psi), \quad (4.16)$$

enquanto um problema de otimização com restrições segue a fórmula

$$\begin{aligned} & \operatorname{argmin}_{\Psi} f(\Psi) \\ & \text{sujeito a } c_1(\Psi) = 0 \\ & \quad c_2(\Psi) = 0 \\ & \quad \dots \end{aligned} \quad (4.17)$$

onde as funções  $c_i(\Psi)$  definem hiper-superfícies no espaço de parâmetros dentro das quais a solução deve se localizar. No caso da otimização realizada pelo *Corisco* existe apenas uma função de restrição, que impõe uma norma unitária para  $\Psi$ , resultando em

$$c(\Psi) = |\Psi| - 1 = 0. \quad (4.18)$$

A Figura 4.12 mostra um exemplo de uma otimização realizada por FilterSQP. O problema de otimização deste exemplo é

$$\begin{aligned} & \operatorname{argmin}_{x,y} f(x,y) = (x+y)^2. \\ & \text{sujeito a } x^2 + y^2 = 1 \end{aligned} \quad (4.19)$$

O mínimo desta função se localiza sobre a diagonal secundária, onde  $x = -y$ . A otimização é restrita ao círculo unitário centrado na origem. Este gráfico mostra que durante o processo de otimização consideram-se pontos que não se localizam exatamente dentro do sub-espaco da restrição da otimização, mas ao longo do tempo a trajetória tende a se aproximar desta região, e a solução entregue sempre obedecerá a restrição. Outro fato interessante que este gráfico ilustra é que um problema de otimização que seria mal definido no espaço completo pode se tornar bem-definido com uma restrição. Neste caso, o círculo faz com que apenas dois pontos do espaço sejam soluções, enquanto no espaço completo toda a diagonal secundária constitui a região que minimiza a função. No problema da estimativa de orientação ocorre algo semelhante, uma solução  $\Psi$  define na realidade uma linha no espaço, cruzando a origem, de pontos ótimos da função objetivo. Porém a restrição de norma unitária na otimização faz com que apenas os pontos desta reta que cortam a hiper-superfície da restrição sejam aceitos. No caso dos quaternions a região criada pela restrição é conhecida como 3-esfera, e é o correlato em quatro dimensões da esfera em três, ou do círculo em duas dimensões. O motivo porque a função

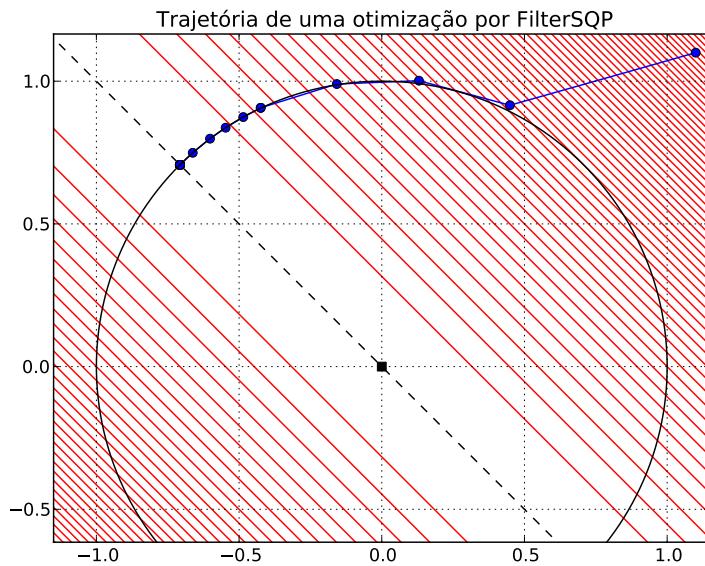


Figura 4.12: Exemplos de uma trajetória seguida em uma otimização por FilterSQP.

objetivo da Seção 4.3.2 é independente da escala de  $\Psi$  é porque mesmo que os vetores  $r$  calculados pela Equação 4.2 possam ter normas diferentes de 1 quando  $|\Psi| \neq 1$ , os vetores  $v$  utilizados nos cálculos são normalizados.

O uso de quaternions genéricos, não-normalizados, como base para o processo de otimização é vantajoso principalmente devido ao fato de que as componentes da matriz de rotação, que são as coordenadas de cada direção  $r$ , são calculadas a partir de polinômios do segundo grau dos parâmetros do quaternion  $\Psi$ , seguindo a Equação 4.2. Isto contrasta com a presença de funções trigonométricas nas abordagens alternativas, como na Equação 4.1, que não só dificultam a análise das equações como trazem uma grande carga computacional para os cálculos. No caso das expressões polinomiais as derivadas dos parâmetros de cada  $r$  em relação aos de  $\Psi$  são facilmente obtidas, e são inclusive constantes no caso das derivadas segundas. Isto é relevante porque os algoritmos de otimização contínua como o que será descrito a seguir fazem uso do cálculo das derivadas da função. Quando estas derivadas são difíceis de calcular é possível recorrer a técnicas de aproximação numérica (DENIS; ELDER; ESTRADA, 2008), ou ainda à chamada diferenciação automática (SCHINDLER; DELLAERT, 2004), como já foi realizado no passado em outros métodos para o problema da estimativa de orientação através de edgels. No *Corisco* foram utilizadas fórmulas fechadas para estas derivadas pela primeira vez para este problema.

A Seção 4.5.1 a seguir discute o método de Newton para otimização sem restrições. A Seção 4.5.2 discute o problema da otimização com restrições e o algoritmo FilterSQP, que pode ser entendido como uma extensão do método de Newton para levar em conta restrições. A Seção 4.5.3 discute por fim como exatamente é feito o cálculo das derivadas

da função objetivo do *Corisco* que são utilizadas pelo FilterSQP.

#### 4.5.1 Otimização pelo método de Newton com região de confiança

Para explicar o funcionamento do FilterSQP devemos começar com uma revisão de otimização sem restrições através do tradicional método de Newton, ou Newton-Raphson. O método de Newton-Raphson é originalmente um método iterativo utilizado para encontrar raízes de funções. Quando aplicado a este problema original, uma iteração do algoritmo atualiza a estimativa da localização de uma raiz utilizando a informação do gradiente da função em um dado ponto, modelando a função ao redor daquele ponto como sendo uma linha reta. Portanto, dada uma função  $f(x)$  com derivada  $f'(x)$ , desejamos encontrar um  $x$  que torna  $f(x) = 0$ . Se a estimativa atual da solução é  $x^t$  a atualização desta estimativa será calculada por

$$x^{t+1} = x^t - \frac{f(x^t)}{f'(x^t)}. \quad (4.20)$$

Um ponto extremo de  $f(x)$  pode ser encontrado através das raízes de  $f'(x)$ . Se aplicarmos o método de Newton-Raphson para isto, teremos a fórmula de iteração

$$x^{t+1} = x^t - \frac{f'(x^t)}{f''(x^t)}. \quad (4.21)$$

A generalização deste método de otimização para mais dimensões é facilmente realizada através da interpretação geométrica da Equação 4.21. Uma aproximação polinomial da função  $f(x)$  ao redor de  $x^t$  pode ser dada pela equação

$$f(x^t + d) \simeq f(x^t) + d f'(x^t) + d^2 f''(x^t)/2. \quad (4.22)$$

Ou seja, a função é localmente modelada como uma parábola, e os parâmetros deste modelo são retirados das derivadas de primeira e segunda ordem da função no ponto analisado  $x^t$ . O ponto seguinte da sequência  $x^{t+1}$  é o vértice desta parábola, ou o seu ponto de derivada nula, cuja posição é dada pela fórmula

$$d^* = -\frac{f'(x^t)}{2(f''(x^t)/2)} = -\frac{f'(x^t)}{f''(x^t)}, \quad (4.23)$$

e assim a fórmula da atualização é obtida através de

$$x^{t+1} = x^t + d^*. \quad (4.24)$$

Para generalizar este processo para mais dimensões, basta realizar a aproximação da função através de um modelo quadrático com o número de dimensões necessárias. O

modelo quadrático da função será dado por

$$f(\mathbf{x}^t + \mathbf{d}) \simeq f(\mathbf{x}^t) + \mathbf{d}^T \mathbf{g} + \frac{1}{2} \mathbf{d}^T \mathbf{H} \mathbf{d}, \quad (4.25)$$

onde  $\mathbf{g}$  é o gradiente da função no ponto analisado e  $\mathbf{H}$  é a matriz Hessiana, que contém todas as derivadas segundas, ou seja

$$\mathbf{H} = \nabla^2 f. \quad (4.26)$$

É importante notar que  $\mathbf{g}$  e  $\mathbf{H}$  não são constantes, seus valores são recalculados em cada passo do algoritmo em função da estimativa de solução atual  $\mathbf{x}^t$ . Se a matriz Hessiana for definida positiva, este modelo define um paraboloide elíptico, e o seu ponto mínimo pode ser encontrado pela fórmula

$$\mathbf{d}^* = -\mathbf{H}^{-1} \mathbf{g}. \quad (4.27)$$

O uso deste vetor durante um processo de otimização é conhecido como *passo de Newton*. Quando um processo se encontra próximo da solução, esta é geralmente a melhor forma de convergir para ela, pois a velocidade de convergência será quadrática. Isto significa que, dada uma solução  $\mathbf{x}^*$ , os resíduos da aproximação em cada iteração seguem a regra

$$|\mathbf{x}^{t+1} - \mathbf{x}^*| = O(|\mathbf{x}^t - \mathbf{x}^*|^2), \quad (4.28)$$

o que resulta em um processo mais rápido do que uma convergência linear, modelada pela equação

$$|\mathbf{x}^{t+1} - \mathbf{x}^*| = O(|\mathbf{x}^t - \mathbf{x}^*|). \quad (4.29)$$

A diferença na velocidade de convergência destes dois tipos de processo não deve ser subestimada. Esta pode ser a diferença entre ter uma otimização que leva apenas uma dúzia de iterações para atingir uma grande precisão, ou requerer centenas ou até milhares de iterações no caso da convergência linear.

O motivo porque o passo de Newton não pode ser utilizado em todas as iterações é, em primeiro lugar, porque a Hessiana nem sempre é positiva definida. Outro motivo é porque longe da solução este modelo quadrático pode ser uma má aproximação da função objetivo. Uma forma alternativa de construir um passo de otimização, que evita estes problemas, é apenas seguir a direção oposta do gradiente da função, ou seja, dado o vetor gradiente  $\mathbf{g}$  calculado sobre o ponto  $\mathbf{x}^t$ ,

$$\mathbf{x}^{t+1} = -\beta \mathbf{g}, \quad (4.30)$$

onde é preciso selecionar um valor adequado para o coeficiente  $\beta$  em cada instante. Este

método de otimização não é ideal porque resulta em uma convergência linear, mas é uma boa forma de complementar o método de Newton no início do processo de otimização.

Os algoritmos de otimização contínua mais sofisticados costumam portanto realizar o cálculo de cada novo deslocamento  $\mathbf{d}$  fazendo um controle para utilizar ou o passo de Newton ou um passo na direção oposta ao gradiente. No início da otimização, ou em qualquer momento em que a matriz Hessiana não seja positiva definida, ou se a aproximação quadrática se mostrar muito ruim, os passos tendem a se basear apenas no gradiente, e com valores de  $\beta$  bastante reduzidos. Se a modelagem quadrática se mostrar adequada, e se a Hessiana se tornar positiva definida, as iterações do processo de otimização começam a seguir o passo de Newton.

Um algoritmo de otimização bastante popular que realiza este tipo de controle é o *algoritmo de Levenberg-Marquardt*, ou LM (FLETCHER, 2000, Sec. 5.2). O passo em cada iteração deste algoritmo é dado pela equação

$$(\mathbf{H} + \nu \mathbf{I})\mathbf{d} = -\mathbf{g}, \quad (4.31)$$

onde o valor do coeficiente  $\nu$  é alterado ao longo do tempo para realizar o controle mencionado acima. Se  $\nu \rightarrow 0$ , temos exatamente o passo de Newton. Se  $\nu$  se torna mais elevado, ou  $\nu \rightarrow \infty$ , o resultado é um passo aproximadamente na direção oposta ao gradiente com comprimento bastante reduzido, na proporção inversa de  $\nu$ .

No LM este valor de  $\nu$  é alterado em duas ocasiões. Em primeiro lugar, ele deve ser aumentado sempre que necessário para tornar positiva definida a matriz do lado esquerdo da Equação 4.31. Após o cálculo do novo vetor  $\mathbf{d}$ , calcula-se então o valor da função no ponto  $\mathbf{x}^t + \mathbf{d}$ . Este valor é utilizado para realizar uma análise do desempenho atual da otimização, o que pode então levar a um aumento ou redução de  $\nu$ , e pode ainda levar a um eventual descarte do  $\mathbf{d}$  atual, seguido de um aumento de  $\nu$ .

Esta versão original do algoritmo LM possui algumas desvantagens, e uma abordagem alternativa que pode apresentar um melhor desempenho é a que transforma o algoritmo em um *algoritmo de região de confiança*, ou *de passo restrito*. Nos algoritmos desta família existe uma região de tamanho limitado ao redor de  $\mathbf{x}^t$  dentro de onde residirá  $\mathbf{x}^{t+1}$ . Isto é justamente o que foi utilizado no *Corisco*. Nesta versão do LM como algoritmo de região de confiança temos em cada iteração um valor de  $\delta$  que determina um raio limite para o deslocamento, e o vetor  $\mathbf{d}$  é encontrado resolvendo o problema

$$\begin{aligned} \operatorname{argmin}_{\mathbf{d}} \quad & \mathbf{d}^T \mathbf{g} + \frac{1}{2} \mathbf{d}^T \mathbf{H} \mathbf{d} \\ \text{sujeito a} \quad & |\mathbf{d}| \leq \delta \end{aligned} \quad (4.32)$$

Se a Hessiana for positiva definida, e o seu mínimo se encontrar dentro da região definida por  $\delta$ , então o resultado nesta iteração será o passo de Newton. Caso contrário, a próxima estimativa se localizará sobre o limite da região, ou seja,  $|\mathbf{d}| = \delta$ .

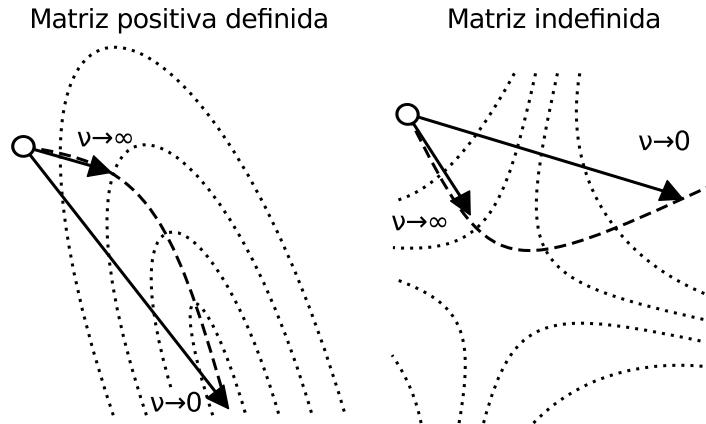


Figura 4.13: Trajetórias de passos possíveis na minimização de um modelo quadrático para dois casos diferentes de matriz Hessiana. O círculo representa a estimativa de solução atual, e a próxima estimativa se localizará em algum ponto da trajetória.

Para resolver o problema da Equação 4.32 o que se faz é encontrar em cada iteração um valor de  $\nu$  que resolva a Equação 4.31 obedecendo a restrição de que  $|d| \leq \delta$ . A variação de  $\nu$  neste cálculo para determinados  $g$  e  $H$  define uma trajetória no espaço para os possíveis  $d$  resultantes, que parte da origem quando  $\nu \rightarrow \infty$  e vai para o infinito conforme  $\nu$  decresce se  $H$  não for positiva definida. Se  $H$  for positiva definida a trajetória converge para o passo de Newton quando  $\nu \rightarrow 0$ . O que desejamos é encontrar o ponto em que esta trajetória cruza o limite da região de confiança em cada iteração. Este cálculo de  $\nu$  pode ser feito através de um método numérico de localização de raízes, explorando ainda o fato de que o valor de  $|d|$  em função de  $\nu$  pode ser encontrado rapidamente através da decomposição em auto-valores e auto-vetores de  $H$ . A Figura 4.13 mostra exemplos desta trajetória de vetores  $d$  possíveis criada através da variação de  $\nu$  na Equação 4.31.

A versão do algoritmo LM como técnica de região de confiança resulta portanto em uma sequência de problemas como o da Equação 4.32, mas ao invés de controlar o valor de  $\nu$  ao longo do tempo, controla-se o valor de  $\delta$ . Por exemplo,  $\delta$  pode ser reduzido se o erro de aproximação do modelo quadrático se mostrar elevado, e caso contrário  $\delta$  pode ser mantido igual ou até aumentado. A região de confiança também pode ser reduzida se a solução em uma certa iteração for dada pelo passo de Newton, ou seja, se  $\nu = 0$ . Em um problema como o da estimativa de orientação, como o domínio possui um tamanho restrito e bem-conhecido, é possível escolher um valor apropriado para inicializar  $\delta$  e não há necessidade de permitir que ele cresça ao longo do tempo.

#### 4.5.2 Otimização com restrições e o algoritmo FilterSQP

A Seção 4.5.1 apresentou um algoritmo de otimização sem restrições, baseado no método de Newton e utilizando regiões de confiança. No problema estudado, que é do tipo da Equação 4.17, utiliza-se uma abordagem similar denominada Programação Quadrática Sequencial, ou SQP (do Inglês Sequential Quadratic Programming). Além de aproximar

localmente a função objetivo com um modelo quadrático, a função de restrição também é aproximada com um modelo linear, tornando o cálculo do passo da Equação 4.32 algo similar a

$$\begin{aligned} \underset{\mathbf{d}}{\operatorname{argmin}} \quad & \mathbf{d}^T \mathbf{g} + \frac{1}{2} \mathbf{d}^T \mathbf{H} \mathbf{d} \\ \text{sujeito a} \quad & \mathbf{A}^T \mathbf{d} = \mathbf{b} \quad , \\ & |\mathbf{d}| \leq \delta \end{aligned} \quad (4.33)$$

onde a matriz  $\mathbf{A}$  possui tantas colunas quanto equações de restrição do problema.

Há várias abordagens para realizar otimização com restrições, uma maneira possível é utilizar funções de penalidade. Outros algoritmos, como o utilizado aqui, utilizam o método dos *multiplicadores de Lagrange*. Este método se baseia no fato de que no ponto extremo de uma função restrita a uma hiper-superfície o gradiente da função objetivo deve ser normal a esta hiper-superfície. Esta direção normal pode ser encontrada através do gradiente da função de restrição. Dada uma restrição  $c(\mathbf{x}) = 0$ , isto resulta em

$$\nabla f = \lambda \nabla c, \quad (4.34)$$

onde  $\lambda$  é um fator de escala que permite comparar o gradiente da função objetivo com o gradiente da função de restrição. Quando mais restrições existem, os seus gradientes devem ser adicionados nesta equação com diferentes multiplicadores, e o gradiente da função objetivo será então um combinação linear deles. Este não é o caso do problema estudado, entretanto, onde há apenas uma restrição. A Equação 4.34 também pode ser obtida através de uma função denominada *função Lagrangeana*, definida como

$$\Lambda(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda c(\mathbf{x}). \quad (4.35)$$

Os pontos críticos desta função, onde seu gradiente se torna zero, indicam possíveis pontos extremos de  $f$  dentro da restrição.

A aplicação do método dos multiplicadores de Lagrange ao problema da Equação 4.33, ignorando temporariamente a restrição de passo restrito, resulta na equação

$$(\mathbf{Z}^T \mathbf{H} \mathbf{Z}) \mathbf{y} = -\mathbf{Z}^T (\mathbf{g} + \mathbf{H} \mathbf{S} \mathbf{b}). \quad (4.36)$$

A matriz  $\mathbf{Z}$  representa o espaço nulo da matriz  $\mathbf{A}$ . Isto quer dizer que qualquer coluna de  $\mathbf{Z}$  deve ser ortogonal a qualquer coluna de  $\mathbf{A}$ , ou seja

$$\mathbf{A}^T \mathbf{Z} = \mathbf{0}. \quad (4.37)$$

Já a matriz  $\mathbf{S}$  é uma inversa generalizada de  $\mathbf{A}$ . Qualquer solução  $\mathbf{d}$  deve seguir a forma

$$\mathbf{d} = \mathbf{S} \mathbf{b} + \mathbf{Z} \mathbf{y}. \quad (4.38)$$

A matriz  $\mathbf{Z}^T \mathbf{H} \mathbf{Z}$  é denominada *matriz Hessiana reduzida*, e o vetor  $\mathbf{Z}^T(\mathbf{g} + \mathbf{H}\mathbf{S}\mathbf{b})$  *gradiente reduzido*.

O vetor  $\mathbf{d}$  que atende a Equação 4.33 é composto portanto por uma componente  $\mathbf{S}\mathbf{b}$  que leva a solução até o hiper-plano da restrição, e uma componente  $\mathbf{Z}\mathbf{y}$  que, por ser calculada dentro do espaço nulo de  $\mathbf{A}$ , estará sempre dentro do sub-espaco definido pela restrição linear. A Equação 4.36 pode ser entendida como uma transformação do problema para um espaço onde ele se torna um programa quadrático sem restrições, que é então resolvido da forma apresentada anteriormente. Por exemplo, o procedimento da Equação 4.31 de somar uma matriz identidade ponderada por  $\nu$  para controlar o tamanho do passo pode ser facilmente implementado com a matriz Hessiana reduzida.

Os parâmetros necessários para se calcular este passo em cada iteração são  $\mathbf{H}$ ,  $\mathbf{g}$  e  $\mathbf{A}$ , retirados das funções objetivo e de restrição, além de  $\delta$ . No caso em que a restrição do problema original sendo resolvido é realmente linear,  $\mathbf{H}$  é simplesmente a Hessiana da função objetivo. Quando a restrição é linear, é fácil encontrar algum ponto que atenda à restrição. A partir de um ponto que atende a restrição, a solução é então buscada através de uma série de passos calculados pela Equação 4.33 com matrizes  $\mathbf{A}$  e consequentemente  $\mathbf{Z}$  constantes, com alguma forma de controle de  $\delta$  tal como descrito anteriormente.

Quando a função de restrição não é linear o problema se torna um chamado *programa não-linear* em sua forma mais geral, e é preciso adotar um procedimento um pouco mais complexo. O algoritmo FilterSQP segue o chamado método de Lagrange-Newton (FLETCHER, 2000, Sec. 12.3). A matriz  $\mathbf{H}$  considerada em cada passo não é simplesmente a Hessiana da função objetivo, mas sim a Hessiana da função Lagrangeana do problema, ou seja

$$\mathbf{H} = \nabla^2 f - \lambda \nabla^2 c. \quad (4.39)$$

O gradiente  $\mathbf{g}$  utilizado no cálculo do passo  $\mathbf{d}$  é o mesmo gradiente da função objetivo, e  $\mathbf{A}$  contém as derivadas primeiras da função de restrição. A necessidade de calcular  $\mathbf{H}$  em cada iteração significa que o valor de  $\lambda$  deve ser estimado da mesma forma que  $\mathbf{x}$ . Sua atualização segue a fórmula

$$\lambda^{t+1} = \mathbf{S}\mathbf{g} \quad (4.40)$$

Em resumo, para o caso de uma única função de restrição não-linear o método SQP calcula passos para a otimização de acordo com a Equação 4.33 da seguinte forma:

- Dada uma estimativa de solução  $\mathbf{x}^t$ , calcula-se para aquele ponto a Hessiana da Lagrangeana  $\mathbf{H}$ , o gradiente da função objetivo  $\mathbf{g}$ , e o valor e o gradiente da função de restrição para montar  $\mathbf{b}$  e  $\mathbf{A}$ .
- Calcula-se para  $\mathbf{A}$  as matrizes  $\mathbf{S}$  e  $\mathbf{Z}$  através de uma decomposição SVD ou operação similar. Como  $\mathbf{A}$  possui apenas uma linha no problema estudado,  $\mathbf{S}$  também pode

ser calculada através de

$$\mathbf{S} = \frac{\mathbf{A}}{|\mathbf{A}|^2}. \quad (4.41)$$

- A Hessiana e gradiente reduzidos são calculados, e então calcula-se  $\mathbf{y}$  de acordo com

$$\begin{cases} (\mathbf{Z}^T \mathbf{H} \mathbf{Z} + \nu \mathbf{I}) \mathbf{y} = -\mathbf{Z}^T (\mathbf{g} + \mathbf{H} \mathbf{S} \mathbf{b}) \\ |\mathbf{y}| \leq \delta \end{cases} \quad (4.42)$$

- Um passo final  $\mathbf{d}$  poderia ser encontrado de acordo com Equação 4.38. Porém em problemas com restrições não-lineares é possível melhorar a convergência realizando uma *correção de segunda ordem*. Para isto calcula-se o valor da função de restrição utilizando um modelo quadrático, baseado no gradiente conhecido  $\mathbf{A}$  e na matriz Hessiana da função de restrição. O vetor  $\mathbf{Zy}$  é a entrada deste cálculo, e a saída produz um novo valor para  $\mathbf{b}$ , denominado  $\mathbf{b}'$ .
- O gradiente reduzido é recalculado com  $\mathbf{b}'$ , e um novo valor de  $\mathbf{y}$  é calculado,  $\mathbf{y}'$
- O passo final é calculado por

$$\mathbf{d}' = \mathbf{S} \mathbf{b}' + \mathbf{Z} \mathbf{y}', \quad (4.43)$$

e a nova estimativa da solução é dada por

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{d}'. \quad (4.44)$$

e  $\lambda$  é atualizado de acordo com a Equação 4.40.

- A seguir o algoritmo determina se a estimativa atual é boa o suficiente ou se o processo deve continuar, e o valor de  $\delta$  é atualizado de acordo com o procedimento que será discutido a seguir.

A Figura 4.14 ilustra o processo do cálculo de uma iteração do SQP. A curva sólida é a região da restrição dentro da qual a solução da otimização deve ser contida. O ponto mais acima representa a estimativa atual da solução, no início da iteração. O ponto imediatamente abaixo representa o deslocamento por  $\mathbf{S} \mathbf{b}$ , levando o ponto até a região da restrição, e o ponto à sua esquerda representa o deslocamento por  $\mathbf{S} \mathbf{b} + \mathbf{Z} \mathbf{y}$ . O ponto mais abaixo na figura representa a estimativa final, após a correção de segunda ordem.

Para controlar o valor de  $\delta$ , o algoritmo FilterSQP funciona mantendo uma lista de todos os valores da função objetivo encontrados em cada iteração, junto do valor absoluto correspondente da função de restrição. Esta lista é utilizada para realizar um processo de filtragem. Para que um ponto seja aceito pelo filtro, ele não pode ser *dominado* por nenhum ponto da lista. O conceito de dominação vem da teoria de otimização multiobjetivo. Um ponto é dominado por outro se todos os seus parâmetros forem menores ou

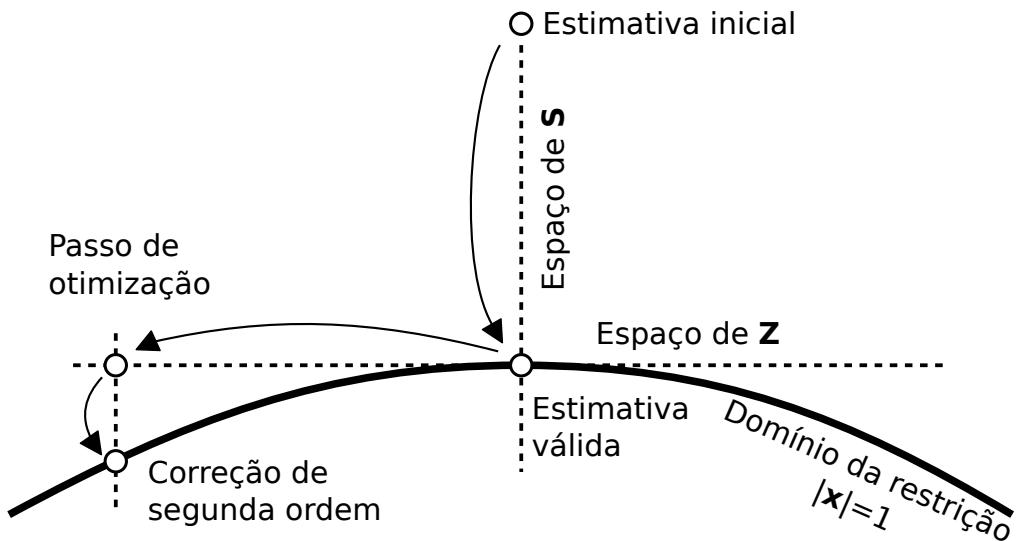


Figura 4.14: Passos de uma iteração do método SQP.

iguais do que os correspondentes de outro ponto. No caso deste algoritmo, é dito que um ponto  $\mathbf{x}^i$  domina  $\mathbf{x}$  se

$$f(\mathbf{x}^i) \leq f(\mathbf{x}) \quad \text{e} \quad |c(\mathbf{x}^i)| \leq |c(\mathbf{x})|. \quad (4.45)$$

Ou seja, para ser aceito pelo filtro um ponto analisado precisa representar uma melhoria em relação a algum ponto existente na lista, seja reduzindo o valor da função objetivo, ou seja reduzindo a distância à região da restrição.

Cada novo  $\mathbf{x}^{t+1}$  é testado pelo filtro após ser calculado. Se aceito, o ponto é então adicionado à lista de pontos do filtro, e uma nova iteração da otimização é realizada a partir deste novo ponto. Se o ponto não é aceito pelo filtro este passo é descartado, e então o valores de  $\delta$  é reduzido e um novo passo é calculado. Na implementação deste algoritmo no *Corisco*  $\delta$  é multiplicado por 0.25 nesta ocasião. O processo termina quando o tamanho de  $\mathbf{d}$  e a distância à região forem ambos menores do que um limiar em um certo passo. O limiar utilizado na prática no *Corisco* fica geralmente na faixa de  $10^{-15}$  a  $10^{-12}$ .

#### 4.5.3 Derivadas da função objetivo

Para aplicar o FilterSQP para minimizar a função objetivo do *Corisco* é preciso portanto encontrar expressões para o cálculo de suas derivadas de primeira e segunda ordem em relação a cada um dos quatro parâmetros do quaternion  $\Psi$ ,  $\Psi^a$ ,  $\Psi^b$ ,  $\Psi^c$  e  $\Psi^d$ . A expressão da função objetivo, definida na Equação 4.14, contém uma operação que retorna o menor valor de uma lista de três valores de entrada. Esta operação constitui uma função descontínua, e portanto oferece um desafio para a diferenciação. Para lidar com isto, podemos considerar que a função é na realidade calculada em dois passos, primeiro calculam-se os

valores de uma matriz  $\mathbf{K}$ , que tem um papel de função indicadora. Para cada ponto  $n$ ,  $K_{nk}$  irá valer 1 apenas para o  $k$  de sua classe, ou seja

$$K_{nk} = \begin{cases} 1 & \text{se } k = \underset{k}{\operatorname{argmin}} \rho(\mathbf{u}_n \mathbf{v}_{nk}) \\ 0 & \text{caso contrário} \end{cases}. \quad (4.46)$$

Assim a função objetivo se torna

$$F(\Psi) = \sum_{nk} K_{nk} \rho(\mathbf{u}_n \mathbf{v}_{nk}), \quad (4.47)$$

onde lembramos que os valores  $\mathbf{v}_{nk}$  dependem de  $\Psi$ . Cada ponto do conjunto de dados contribui apenas um termo para a soma devido à forma como  $\mathbf{K}$  é montada. As derivadas são então calculadas assumindo simplesmente que esta classificação está correta, mesmo que  $\mathbf{K}$  possa se modificar na próxima iteração. Ou seja, assume-se que os valores  $K_{nk}$  são todos constantes em relação a  $\Psi$ , mesmo que isto não seja estritamente verdadeiro. Na realidade  $\mathbf{K}$  se modifica apenas de forma abrupta, e é realmente constante em largas extensões do espaço de parâmetros.

É preciso deixar claro este procedimento pode não parecer muito rigoroso do ponto de vista de uma simples otimização de função, mas como discutido anteriormente, este processo pode ser interpretado como um método do tipo GEM em que os cálculos do passo  $E$  ocorrem de uma forma implícita, simultaneamente ao cálculo do valor da função.

Para calcular as derivadas da função objetivo é útil definir o vetor

$$\mathbf{w} = \mathbf{J}\mathbf{r}, \quad (4.48)$$

que então produz  $\mathbf{v}$  segundo a Equação 2.18 como

$$\mathbf{v} = \frac{\mathbf{w}}{|\mathbf{w}|}. \quad (4.49)$$

Definidas portanto as classes de cada ponto podemos obter a derivada primeira da função objetivo em relação a  $\Psi^a$  através da regra da cadeia, resultando na fórmula

$$\frac{\partial F}{\partial \Psi^a}(\Psi) = \sum_{nk} K_{nk} \rho'(\mathbf{u}_n \mathbf{v}_{nk}) \left( u_n^x \frac{\partial \mathbf{v}_{nk}^x}{\partial \Psi^a} + u_n^y \frac{\partial \mathbf{v}_{nk}^y}{\partial \Psi^a} \right). \quad (4.50)$$

Onde  $\rho'$  representa a derivada da função de erro  $\rho$ , calculada para o resíduo de cada ponto.

As derivada das componentes de cada vetor  $\mathbf{v}$  são calculadas a partir do vetor  $\mathbf{w}$

correspondente

$$\frac{\partial v^x}{\partial \Psi^a} = +w^y \left( w^y \frac{\partial w^x}{\partial \Psi^a} - w^x \frac{\partial w^y}{\partial \Psi^a} \right) |\mathbf{w}|^{-3/2}, \quad (4.51)$$

$$\frac{\partial v^y}{\partial \Psi^a} = -w^x \left( w^y \frac{\partial w^x}{\partial \Psi^a} - w^x \frac{\partial w^y}{\partial \Psi^a} \right) |\mathbf{w}|^{-3/2}. \quad (4.52)$$

As derivadas de  $\mathbf{w}$  são simples de obter, porque este vetor é apenas uma combinação linear, através da Jacobiana  $\mathbf{J}$  em cada observação, dos vetores  $\mathbf{r}$ . Suponha portanto que a classe de um ponto seja a direção  $x$  no ambiente. Assim teremos

$$\frac{\partial w^x}{\partial \Psi^a} = \mathbf{J}^x \mathbf{r}_x \quad (4.53)$$

$$\frac{\partial w^y}{\partial \Psi^a} = \mathbf{J}^y \mathbf{r}_x \quad (4.54)$$

onde  $\mathbf{J}^x$  e  $\mathbf{J}^y$  são as duas linhas da Jacobiana. As derivadas de  $\mathbf{r}$  são facilmente obtidas a partir da Equação 4.2. Por exemplo, para a direção  $\mathbf{r}^x$  e o parâmetro  $\Psi^a$  temos

$$\frac{\partial \mathbf{r}^x}{\partial \Psi^a} = 2(\Psi^a, \Psi^d, -\Psi^c), \quad (4.55)$$

e para os outros outros parâmetros temos

$$\frac{\partial \mathbf{r}^x}{\partial \Psi^b} = 2(\Psi^b, \Psi^c, \Psi^d) \quad (4.56)$$

$$\frac{\partial \mathbf{r}^x}{\partial \Psi^c} = 2(-\Psi^c, \Psi^b, -\Psi^a) \quad (4.57)$$

$$\frac{\partial \mathbf{r}^x}{\partial \Psi^d} = 2(-\Psi^d, \Psi^a, \Psi^b) \quad . \quad (4.58)$$

As derivadas segundas de  $F(\Psi)$  podem ser calculadas da mesma forma, assumindo  $\mathbf{K}$  constante. Assim temos, por exemplo, para os parâmetros  $\Psi^a$  e  $\Psi^b$

$$\begin{aligned} \frac{\partial^2 F}{\partial \Psi^a \partial \Psi^b}(\Psi) = \sum_{nk} K_{nk} & \left[ \rho''(\mathbf{u}_n \mathbf{v}_{nk}) \left( u_n^x \frac{\partial \mathbf{v}_{nk}^x}{\partial \Psi^a} + u_n^y \frac{\partial \mathbf{v}_{nk}^y}{\partial \Psi^a} \right) \left( u_n^x \frac{\partial \mathbf{v}_{nk}^x}{\partial \Psi^b} + u_n^y \frac{\partial \mathbf{v}_{nk}^y}{\partial \Psi^b} \right) \right. \\ & \left. + \rho'(\mathbf{u}_n \mathbf{v}_{nk}) \left( u_n^x \frac{\partial^2 \mathbf{v}_{nk}^x}{\partial \Psi^a \partial \Psi^b} + u_n^y \frac{\partial^2 \mathbf{v}_{nk}^y}{\partial \Psi^a \partial \Psi^b} \right) \right] . \end{aligned} \quad (4.59)$$

As derivadas segundas de  $\mathbf{v}$  são dadas por um par de expressões relativamente complexas. Utilizando a notação  $w_a^x$  para indicar a derivada de  $w^x$  em relação a  $\Psi^a$ , podemos escrever

las como

$$\begin{aligned} v_{aa}^x &= (w^x w^x w^y (2w_a^x w_b^y + 2w_b^x w_a^y + w_{ab}^x w^y) \\ &\quad - w^x w^y w^y (3w_b^x w_a^x - 2w_b^y w_a^y + w^y w_{ab}^y) \\ &\quad + w^y w^y w^y (-w_a^x w_b^y - w_b^x w_a^y + w_{ab}^x w^y) \\ &\quad - w^x w^x w^x (w_b^y w_a^y + w^y w_{ab}^y)) |\mathbf{w}|^{-5/2} \end{aligned} \quad (4.60)$$

$$\begin{aligned} v_{aa}^y &= (w^y w^y w^x (2w_a^y w_b^x + 2w_b^y w_a^x + w_{ab}^y w^x) \\ &\quad - w^y w^x w^x (3w_b^y w_a^y - 2w_b^x w_a^x + w^x w_{ab}^x) \\ &\quad + w^x w^x w^x (-w_a^y w_b^x - w_b^y w_a^x + w_{ab}^y w^x) \\ &\quad - w^y w^y w^y (w_b^x w_a^x + w^x w_{ab}^x)) |\mathbf{w}|^{-5/2} \end{aligned} \quad (4.61)$$

Estas expressões foram encontradas através de um software de manipulação simbólica, o Mathematica (INC., 2011). Elas são basicamente polinômios que dependem de  $w^x$ ,  $w^y$  e das derivadas de primeira e segunda ordem destes componentes em relação a cada um dos parâmetros. Estes polinômios são multiplicados ao final pelo fator  $|w|^{-5/2}$  para produzir os valores finais das derivadas segundas. Um fato relevante é que as derivadas segundas de  $\mathbf{w}$  são ainda mais simples do que as de primeira ordem, mostradas anteriormente, porque as derivadas segundas de  $\mathbf{r}$  em relação aos parâmetros de  $\Psi$  são constantes. Por exemplo

$$\frac{\partial^2 \mathbf{r}^x}{\partial \Psi^a \partial \Psi^a} = 2(1, 0, 0) \quad (4.62)$$

$$\frac{\partial^2 \mathbf{r}^x}{\partial \Psi^a \partial \Psi^b} = 2(0, 0, 0) \quad (4.63)$$

$$\frac{\partial^2 \mathbf{r}^x}{\partial \Psi^a \partial \Psi^c} = 2(0, 0, -1) \quad (4.64)$$

$$\frac{\partial^2 \mathbf{r}^x}{\partial \Psi^a \partial \Psi^d} = 2(0, 1, 0). \quad (4.65)$$

Com relação à função  $\rho$ , para o caso da função biquadrada de Tukey o seu valor e suas derivadas para  $x < s$  são dadas por

$$\rho(x) = 1 - (1 - (x/s)^2)^3 \quad (4.66)$$

$$\rho'(x) = 6x(s^2 - x^2)/s^6 \quad (4.67)$$

$$\rho''(x) = 6(s^4 - 6s^2x^2 + 5x^4)/s^6. \quad (4.68)$$

Isto conclui o conjunto de fórmulas necessárias para a aplicação do algoritmo FilterSQP para a otimização da função objetivo do *Corisco*.

## 4.6 Revisão e algumas aplicações imediatas

Este capítulo apresentou o funcionamento do *Corisco*, o método proposto nesta tese para a estimação de orientação de câmera a partir de uma única imagem obtida de um ambiente antrópico. O funcionamento do *Corisco* inicia com uma análise da imagem que extrai um

conjunto de edgels. Em seguida encontra-se um quaternion normalizado  $\Psi$  que modela uma rotação 3D entre o sistema referencial da câmera e o referencial natural do ambiente, cujos eixos são paralelos às suas retas constituintes. A estimativa de  $\Psi$  é feita através da minimização de uma função objetivo, que é um erro de predição das direções dos edgels. Esta predição é feita pelo cálculo das direções das projeções das retas do ambiente sobre a posição de cada edgel considerando uma orientação  $\Psi$  hipotética. Os cálculos das direções preditas fazem uso do modelo de câmera.

O *Corisco* pode ser utilizado com qualquer modelo de câmera desejado, sejam câmeras que seguem o modelo *pinhole*, de projeção pontual, ou lentes com distorções radiais, incluindo até mesmo distorções radiais muito intensas como as das lentes olho-de-peixe que produzem a projeção radial azimuthal equidistante. Projeções cilíndricas também podem ser utilizadas, como é o caso da projeção equiretangular, ou geográfica. Para utilizar qualquer um destes modelos basta fornecer ao *Corisco* uma rotina que calcula para qualquer ponto da imagem os parâmetros da matriz Jacobiana da projeção, ou seja, do mapeamento das coordenadas tridimensionais no referencial da câmera para o espaço da imagem. Estas Jacobianas precisam ser calculadas para cada edgel, e estes cálculos podem ser armazenados durante as iterações para a estimativa do  $\Psi$  ótimo.

O processo de estimativa da orientação depende da classificação de cada observação de acordo com a direção no ambiente da reta que a produziu. No *Corisco* esta classificação é feita durante o cálculo da função objetivo para qualquer  $\Psi$  testado. Assim ao final do processo é possível realizar o cálculo com o valor ótimo estimado de  $\Psi$  para obter ambas informações, tanto a orientação quanto as classes de cada observação.

Não havendo uma boa estimativa inicial da solução, o processo de estimativa inicia com uma busca aleatória que produz um número  $\Psi$  hipotéticos, e calcula o valor da função objetivo para cada um. O  $\Psi$  que produz o menor valor é escolhido como estimativa inicial. Estas hipóteses são criadas a partir de uma busca aleatória em que triplas de observações são selecionadas ao acaso, e assumem-se valores para suas classes. Cada uma destas triplas de observações com classes assumidas produz uma orientação hipotética através de um cálculo direto, que utiliza o modelo de câmera. Quanto mais testes são realizados, maior é a probabilidade de se encontrar uma hipótese a uma dada distância mínima da solução.

Uma vez que a estimativa inicial tenha sido obtida ela é utilizada para iniciar um algoritmo de otimização contínua para minimizar a função objetivo. O algoritmo empregado é o FilterSQP, que é um algoritmo de otimização com restrições. A função objetivo é utilizada com quaternions de entrada  $\Psi$  sem restrições, ou seja, ela é definida sobre todas as 4 dimensões de um quaternion, e a restrição de norma unitária  $|\Psi| = 1$  necessária para que o quaternion produza uma matriz de rotação pela Equação 4.2 é colocada como uma restrição que deve ser atendida pelo processo de otimização. Isto difere de outros métodos onde esta restrição é atendida *a priori*, e a otimização efetivamente ocorre restrita a todo

o momento ao espaço dos conjuntos de parâmetros permitidos. O resultado disto é que as derivadas da função dadas pelas Equações 4.50 e 4.59 podem ser utilizadas durante os cálculos, que são fórmulas relativamente simples, sem o uso de funções trigonométricas ou da exponencial.

A Seção 4.6.1 a seguir traz um rascunho de um algoritmo que implementa o *Corisco*, em uma linguagem similar a Python. As Seções 4.6.2 e 4.6.3 descrevem algumas aplicações que podem ser criadas a partir de pequenas extensões do método proposto.

#### 4.6.1 Rascunho do algoritmo

```
## Procedimento que calcula o valor da função objetivo. As entradas
## são uma orientação sendo testada, os parâmetros dos edgels
## extraídos, as jacobianas calculadas através do modelo de câmera
## para cada edgel, e o parâmetro de escala para a função de
## erro. (Seção 4.3.2, Equação 4.14)
def calcula_erro(psi, edgels, jacobianas, s):
    erro_total = 0
    r = quaternion_para_matriz(psi)
    for n in range(Nedgels):
        erro_n = 1.0
        for k in range(Nclasses):
            v_nk = normaliza(dot(J, r[k])) ## (Equação 2.18)
            produto = dot(edgel[n].u, v_nk)
            erro_nk = Tukey(produto / s) ## (Tabela 2.1)
            ## Armazena apenas o menor dos erros calculados.
            erro_n = erro_nk if erro_nk < erro_n else erro_n
        erro_total += erro_n
    return erro_total

## Extrator de edgels, varre cada linha e coluna da imagem chanado uma
## mesma subrotina. (Seção 4.2)
def extrai_edgels(imagem, Lgrade):
    edgels = []
    g_imagem = gradiente(imagem)
    for k in range(0, imagem.Nlinhas, Lgrade):
        edgels.append(varre_linha(imagem, g_imagem, k))
    for k in range(0, imagem.Ncolunas, Lgrade):
        edgels.append(varre_linha(imagem.T, g_imagem.T, k))
    return edgels
```

```

## Rotina para sorteio de uma nova hipótese a ser testada dentro da
## busca aleatória. (Seção 4.4)
def sorteia_orientacao(normais):
    na,nb,nc = sorteia_arranjo(normais)
    ## Assumindo que na e nb são da mesma classe, da direção r_x.
    r_x = cross(na, nb)
    ## Produz um quaternion tal que r_x aponta na direção especificada.
    psi_ini = quaternion_alinhado(r_x)
    ## Calcula a rotação ao redor do eixo x para alinhar r_y com nc.
    v_aux = dot(n_c, psi_ini.matriz())
    ang = arctan2(vaux[1], -vaux[2])
    psi = Quaternion(sin(ang/2),0,0) * psi_ini
    return psi

## O algoritmo de otimização com restrições utilizado no Corisco. Suas
## entradas são primeiro as funções para o cálculo das funções
## objetivo e de restrição e de suas derivadas, seguidas de um ponto
## de inicialização do algoritmo, um parâmetro de controle do tamanho
## inicial dos passos, e argumentos necessários para o cálculo das
## funções. (Seções 4.5.2 e 4.5.3)
def filter_sqp(f, c, x_ini, rho_ini, args):
    convergiu = False
    filtro = Filtro()
    x = x_ini
    lam = 0
    rho = rho_ini
    while not convergiu:
        Fval, Fgrad, FHess = f(x, *args)
        Cval, Cgrad, Chess = c(x, *args)
        ## Espaço linha e espaço nulo do gradiente da função de
        ## restrição.
        Y,Z = range_null(Cgrad)
        ## Este procedimento encontra a Hessiana e gradiente reduzidos
        ## para calcular o passo no espaço tangente à função
        ## restrição. O lam é utilizado para calcular a Hessiana da
        ## função Lagrangeana do problema.
        Rgrad, Rhess = problema_reduzido(Fval, Fgrad, FHess,
                                         Cval, Cgrad, Chess, lam)
        ## Primeira aproximação.

```

```

y = passo_restrito(Rgrad, Rhess, rho)
## Correção de segunda ordem.
Cval = modelo_quadratico(Cval, Cgrad, Chess, dot(y.T,Z))
Rgrad, Rhess = problema_reduzido(Fval, Fgrad, Fhess,
                                  Cval, Cgrad, Chess, lam)
## Aproximação final.
y = passo_restrito(Rgrad, Rhess, rho)
## Passo final de deslocamento.
d = dot(y.T, Z) + dot(-Cval, Y)
x_new = x + d
## Testa se o passo deve ser aceito, e se houve convergência.
if filtro.testa(f(x_new), abs(c(x_new))):
    filtro.inclui(x_new)
    x = x_new
    ## Novo valor do multiplicador de Lagrange.
    lam = dot(Y, Fgrad)
    if abs(c(x_new)) < 1e-15 and norma(d) < d_tol:
        convergiu = True
else:
    ## Não atualiza x, e reduz tamanho do passo.
    rho = rho * 0.25
return x

## Extrai edgels da imagem.
edgels = extrai_edgels(image, Lgrade)

## Calcula direções normais de cada edgel, e as matrizes Jacobianas da
## projeção em cada ponto. O modelo de câmera é utilizado por estes
## procedimentos.
normais = calcula_normais(edgels, param_int) ## (Equação 4.15)
jacobianas = calcula_jacobianas(edgels, param_int) ## (Seção 2.1.2)

## Sorteia Npasso_1 hipóteses e guarda a melhor. (Seção 4.4)
menor_erro = Inf
for k in range(Npasso_1):
    psi_hip = sorteia_orientacao(normais)
    erro = calcula_erro(psi, edgels, jacobianas, s)
    if erro < menor_erro:
        menor_erro = erro

```

```

melhor_psi = psi_hip

## Executa a otimização. (Seção 4.5.2)
psi_opt = filter_sqp(fun_obj, fun_con, melhor_psi,
                      rho0=1e-3, args=(jacobianas,))

```

#### 4.6.2 Estimação de parâmetros intrínsecos

A mesma função objetivo proposta anteriormente neste capítulo para a estimação de orientação da câmera também pode ser utilizada para a estimação dos parâmetros intrínsecos do modelo de câmera. Para isto é necessário realizar um processo de otimização que além de estimar os parâmetros de orientação estima também os parâmetros intrínsecos. Apesar de representar uma modificação conceitualmente simples do processo, a introdução de mais graus de liberdade torna o processo de otimização mais desafiador.

Comparado ao caso da estimação de orientação, sem calibração, uma primeira dificuldade extra em considerar os parâmetros intrínsecos é que não é possível calcular os coeficientes da Jacobiana em cada ponto para serem reutilizados durante todo o processo. Qualquer que seja o método de otimização utilizado, será necessário sempre calcular novamente estas Jacobianas para cada combinação de parâmetros intrínsecos diferentes. Uma segunda dificuldade é relacionada à dificuldade em se calcular derivadas da função objetivo com relação aos parâmetros intrínsecos. Enquanto foi possível encontrar fórmulas fechadas relativamente simples para todas as derivadas em relação aos parâmetros de  $\Psi$ , não é tão fácil fazer o mesmo para a calibração, pois seria necessário encontrar fórmulas para cada modelo de câmera diferente, e alguns modelos podem ser bastante difíceis de lidar desta forma.

O método proposto para a realizar uma calibração com base no *Corisco* é um processo de otimização em dois níveis. No nível superior varia-se apenas os parâmetros intrínsecos, e então para cada combinação considerada destes parâmetros realiza-se a estimação de orientação, utilizando o *Corisco*. A mesma função objetivo é utilizada no processo de otimização subjacente do *Corisco* e nesta otimização de mais alto nível para a calibração. Neste nível superior empregam-se métodos de otimização tal como o algoritmo de Nelder-Mead, que não utiliza o cálculo de derivadas. Um detalhe interessante é que próximo da solução do problema é possível utilizar uma mesma estimativa inicial da orientação para o *Corisco*, e estimar a orientação utilizando apenas o passo de otimização contínua.

#### 4.6.3 Extração de linhas por agrupamento de edgels

O *Corisco* estima a orientação da câmera e também determina, através da função objetivo proposta, a classe dos edgels de acordo com sua direção no ambiente. O conhecimento das classes, além das direções, pode ser explorado em processos de extração de linhas. O processo proposto aqui utiliza o *Corisco* para determinar a orientação e as classes, e extrai

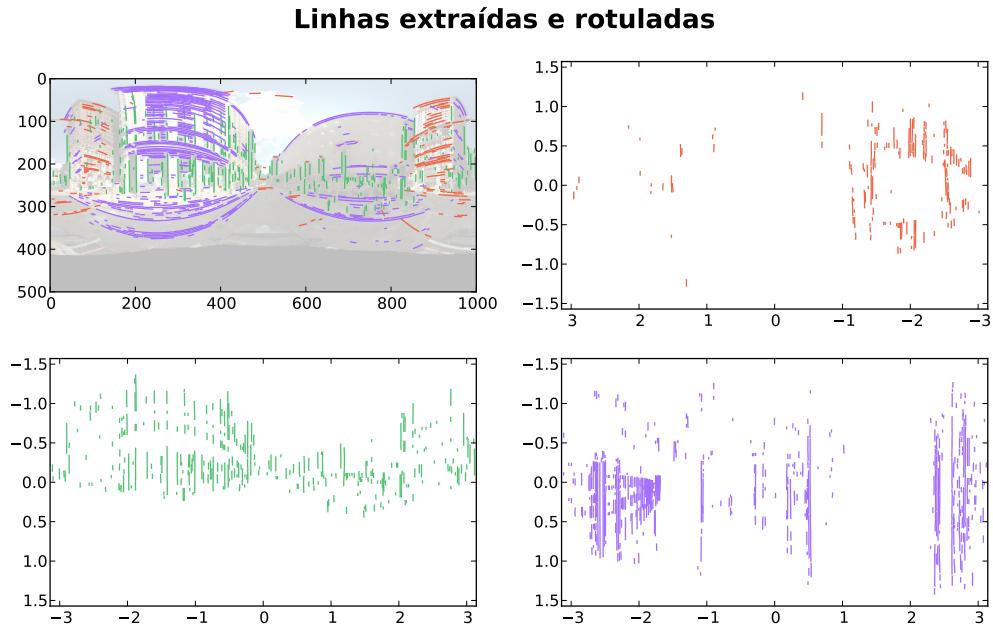


Figura 4.15: Linhos extraídos de uma imagem de projeção equiretangular utilizando o método de extração proposto, baseado no *Corisco*. As cores indicam a classe de cada linha de acordo com sua direção no ambiente.

linhas através de agrupamentos dos edgels.

O primeiro passo do processo é definir, para cada uma das três direções possíveis no ambiente, um espaço auxiliar no qual são mapeadas as coordenadas da posição de cada edgel da imagem. Os edgels são agrupados nestes espaços auxiliares. A Figura 4.15 demonstra os espaços auxiliares e as linhas encontradas sobrepostas no espaço de uma imagem original com projeção equiretangular.

Este espaço auxiliar é parametrizado através de coordenadas esféricas e a direção das retas no espaço é o zênite deste espaço, utilizado como referência para determinar o ângulo de elevação de cada ponto. Neste espaço auxiliar todas as linhas aparecem como simples linhas verticais, e portanto os pontos de uma mesma linha possuem o mesmo azimute. Por exemplo, observações na direção  $\mathbf{r}_x$  podem ser parametrizadas da seguinte forma

$$\varphi_x = \sin^{-1} \left( \frac{\mathbf{r}^x \mathbf{q}}{|\mathbf{q}|} \right) \quad (4.69)$$

$$\theta_x = \tan^{-1}(\mathbf{r}^y \mathbf{q}, \mathbf{r}^z \mathbf{q}). \quad (4.70)$$

O resultado é um espaço em que os pontos de fuga se localizam nas extremidades superior e inferior do espaço, ou seja, nos seus polos norte e sul. Na realidade, neste caso trata-se da projeção equiretangular, mas seria possível utilizar outras projeções cilíndricas para estes espaços auxiliares preservando a característica de que todas as linhas se tornam

verticais.

Uma vez que as observações tenham sido mapeadas para estes espaços auxiliares é possível realizar uma busca por pontos vizinhos para extrair apenas linhas aproximadamente verticais, e assim os edgels são agrupadas em linhas retas no ambiente. É possível aplicar um fator de escala na direção horizontal dos pontos para criar uma tendência em agrupar apenas pontos mais alinhados sobre um mesmo azimute. Qualquer método de extração de retas capaz de explorar o fato de que as retas devem ser verticais neste ambiente pode ser empregado, na realidade. De qualquer maneira, uma vez que se tenha identificado os parâmetros de uma reta em um espaço auxiliar é possível retornar à imagem original para realizar novas análises.

Este capítulo descreveu o funcionamento do *Corisco*, e de um par de aplicações possíveis. O Capítulo 5 a seguir relata experimentos realizados para avaliar o funcionamento do método.

## 5 Experimentos

*As above, so below. Be reveled, the truth be known.  
Questions, pieces strewn. Shrouded answers hewn.*

—The Emerald Law, Probot

Este capítulo relata alguns experimentos realizados com o *Corisco* para constatar sua aplicabilidade em diferentes cenários e avaliar seu desempenho. No primeiro experimento, discutido na Seção 5.1, o *Corisco* foi testado com um banco de dados organizado por Denis, Elder e Estrada (2008), o YorkUrbanDB. A precisão e velocidade do *Corisco* pode ser comparada neste caso com os resultados dos métodos estudados por aqueles pesquisadores. O modelo de câmera assumido neste experimento é o de *pinhole*.

O experimento relatado na Seção 5.2 utiliza imagens capturadas especialmente para esta pesquisa, e são apresentados testes utilizando modelos de distorção radial, comparando os resultados do *Corisco* com dados obtidos através de uma técnica baseada em pontos, mais especificamente o método desenvolvido por Snavely, Seitz e Szeliski (2006). Foi utilizado para isto o software disponibilizado pelos próprios pesquisadores, chamado *Bundler*. Além de fornecer orientações da câmera utilizadas como referência na avaliação dos resultados do *Corisco*, o *Bundler* também foi utilizado para obter os parâmetros intrínsecos, mais especificamente as distâncias focais e coeficientes de distorção radial das duas câmeras utilizadas.

A Seção 5.3 relata um experimento que demonstra a possibilidade de se aplicar o *Corisco* para estimar parâmetros intrínsecos da câmera. Este experimento utilizou os mesmos dados de referência obtidos com o *Bundler* para o experimento da Seção 5.2.

A Seção 5.4 relata a aplicação do *Corisco* para um conjunto de imagens de projeção equiretangular, obtidas do projeto StreetView da empresa Google. Assim como no caso do YorkUrbanDB, os valores dos parâmetros de orientação deste banco de dados são conhecidos, e fornecidos juntos das imagens. Porém enquanto nos dois primeiros casos os parâmetros de referência foram obtidos estritamente através de técnicas de visão, neste caso as informações foram obtidas através de equipamentos tal como acelerômetros e magnetômetros, instalados no veículo que foi utilizado para capturar as imagens.

A Seção 5.5 conclui este capítulo com uma demonstração de aplicação do *Corisco* para a análise de uma imagem obtida com uma lente do tipo olho-de-peixe, que produz imagens com projeção polar equidistante. A orientação estimada foi utilizada em um processo de retificação para produzir imagens utilizando a projeção perspectiva, com o plano da imagem ortogonal aos eixos do referencial natural do ambiente.

### 5.1 Análise do banco de dados YorkUrbanDB

O banco de dados YourkUrbanDB contém 102 imagens capturadas de ambientes antrópicos, e foi compilado por Denis, Elder e Estrada (2008). Mais informações sobre o trabalho destes autores podem ser encontradas na dissertação de mestrado de Denis (2008). Há imagens de ambientes internos, obtidas em salas e corredores no interior de edifícios, e externos, obtidas em ruas de cidades e onde é possível observar o exterior de edifícios. As imagens foram codificadas no formato JPEG e suas dimensões são 640 colunas por 480 linhas. Os parâmetros intrínsecos da câmera são fornecidos dentro do banco de dados, e foram obtidos por seus desenvolvedores através de um método baseado na extração de linhas e de pontos de fuga. Os parâmetros mais relevantes são a distância focal  $f = 674,92$  pixels e a localização do ponto principal em  $c^x = 307,55$  e  $c^y = 251,45$ .

As imagens do banco de dados possuem uma leve distorção radial, do tipo barril, porém os criadores do YorkUrbanDB não levaram isto em consideração em nenhuma análise realizada por eles, e da mesma forma este fenômeno foi ignorado no experimento realizado com o *Corisco*. Ou seja, foi assumido que o modelo *pinhole* deve ser utilizado ao trabalhar com o YorkUrbanDB, com os parâmetros intrínsecos fornecidos por seus criadores, apesar de se poder constatar que existe uma leve distorção. Uma análise mais cuidadosa destas imagens seria portanto um tema interessante para pesquisas futuras, mas como os resultados existentes para comparação foram obtidos com este modelo imperfeito, o mesmo modelo foi utilizado no experimento que será relatado nesta seção.

Para obter valores de referência para as orientações de cada imagem, o que é necessário para permitir avaliar o funcionamento de métodos como o *Corisco* e outros testados, os criadores do YorkUrbanDB utilizaram um método semi-automático, baseado em processos supervisionados de extração e classificação de linhas. Cada um dos pontos de fuga foi estimado separadamente, originando estimativas independentes para cada uma das direções dos três grupos de retas extraídos de cada imagem. Imperfeições no ambiente e na aquisição de dados fazem com que estas direções obtidas de forma independente não sejam exatamente ortogonais como exigiria a condição de ambiente antrópico. Matrizes de rotação foram obtidas a partir destas direções estimadas para fornecer uma estimativa de orientação para cada imagem, impondo assim a condição de ortogonalidade entre as direções encontradas. Estas matrizes de rotação foram convertidas em quaternions para produzir os parâmetros que foram efetivamente utilizados como referência no experimento.

Vários métodos de estimação de orientação baseados em edgels foram testados por Denis, Elder e Estrada (2008), e os resultados mais relevantes encontrados naquela pesquisa estão listados na Tabela 5.1. O primeiro e o terceiro destes métodos listados são os métodos baseados em EM testados por Denis, Elder e Estrada (2008), baseados diretamente no método apresentado por Schindler, Krishnamurthy e Dellaert (2006). O primeiro destes utiliza otimização pelo método de Newton convencional (*EM Newton*), e

Tabela 5.1: Desempenho apresentado pelo *Corisco* e outros métodos na análise do YorkUrbanDB. Diferentes combinações de tamanho de grade  $g$  e número de iterações do RANSAC  $i$  foram testadas no caso do *Corisco*. Para cada método foi medido o tempo médio de cálculo da estimativa de orientação e levantada uma distribuição do erro de predição, determinado a partir de orientações de referência contidas no banco de dados.

| Método                             | Tempo | Erro  |          |       |         |        |
|------------------------------------|-------|-------|----------|-------|---------|--------|
|                                    |       | Média | $\sigma$ | 1/4   | Mediana | 3/4    |
| EM Newton                          | 27+?  | 4,00° | 1,00°    | 1,15° | 2,61°   | 4,10°  |
| MAP Quasi-Newton                   | 6+?   | 4,00° | 1,00°    | 1,15° | 2,61°   | 4,10°  |
| EM Quasi-Newton                    | 1+?   | 9,00° | 1,00°    | 4,04° | 6,21°   | 10,33° |
| <i>Corisco</i> $i = 10^4$ $g = 1$  | 50,38 | 1,68° | 3,21°    | 0,73° | 1,27°   | 1,73°  |
| <i>Corisco</i> $i = 10^4$ $g = 4$  | 16,18 | 1,81° | 3,24°    | 0,73° | 1,11°   | 1,90°  |
| <i>Corisco</i> $i = 10^4$ $g = 32$ | 6,54  | 2,36° | 3,76°    | 0,88° | 1,48°   | 2,14°  |
| <i>Corisco</i> $i = 10^3$ $g = 1$  | 9,71  | 2,12° | 3,00°    | 0,82° | 1,52°   | 2,40°  |
| <i>Corisco</i> $i = 10^3$ $g = 4$  | 2,95  | 2,27° | 3,42°    | 0,87° | 1,46°   | 2,38°  |
| <i>Corisco</i> $i = 10^3$ $g = 32$ | 0,93  | 2,89° | 4,05°    | 1,15° | 1,86°   | 2,76°  |
| <i>Corisco</i> $i = 200$ $g = 1$   | 6,69  | 3,47° | 4,53°    | 0,88° | 1,65°   | 4,49°  |
| <i>Corisco</i> $i = 200$ $g = 4$   | 1,84  | 4,24° | 5,79°    | 1,12° | 2,06°   | 5,18°  |
| <i>Corisco</i> $i = 200$ $g = 32$  | 0,45  | 4,24° | 4,70°    | 1,50° | 2,75°   | 5,20°  |

o segundo por um método Quasi-Newton (*EM Quasi-Newton*). O modelo probabilístico utilizado nestes dois casos é o modelo novo, sugerido pelos autores. O segundo método listado na tabela (*MAP Quasi-Newton*) é uma estimativa MAP que se diferencia do trabalho de Coughlan e Yuille (1999) pelo uso de um algoritmo de detecção de bordas ao invés do simples cálculo do gradiente da imagem completa, além da otimização por um algoritmo Quasi-Newton e utilização do novo modelo probabilístico proposto naquela pesquisa. Os testes realizados por Denis, Elder e Estrada (2008) com os modelos probabilísticos antigos e outras formas de medição de erro não estão listados na Tabela 5.1.

Para realizar o experimento o *Corisco* foi utilizado com diferentes configurações de parâmetros para estimar as orientações das imagens do YorkUrbanDB. Foram variados o tamanho  $g$  da grade utilizada na extração de edgels e o número  $i$  de iterações realizadas durante o passo de busca aleatória do processo de otimização do *Corisco*. A Tabela 5.1 contém os resultados da aplicação do *Corisco* com  $i = 10.000$ ,  $1.000$  e  $200$  iterações do passo de RANSAC, e com grades de  $g = 1$ ,  $4$  e  $32$  pixels de espaçamento.

A segunda coluna da Tabela 5.1 mostra o tempo médio gasto para realizar uma estimativa para cada método e conjunto de parâmetros. No caso dos três primeiros métodos há uma interrogação somada ao valor porque os tempos apresentados por Denis, Elder e Estrada (2008) não incluem a análise inicial da imagem, e o método de detecção de bordas utilizado naquela pesquisa (ELDER; ZUCKER, 1998) é relativamente sofisticado, e não possui um custo computacional desprezível. No caso do *Corisco* o tempo para a extração dos edgels variou neste experimento de 0,06 a 0,24 segundos.

É possível observar na Tabela 5.1 que os tempos de cálculo do *Corisco* decaem tanto

com o número de iterações do RANSAC quanto com o tamanho da grade. O número de iterações influencia diretamente no tempo, já que as iterações do RANSAC possuem tempos de cálculo constantes, e quanto maior o número de passos mais o RANSAC se torna o passo mais custoso de todo o processo, e a relação entre o seu número de iterações e o tempo total se torna bastante direta. Já o tamanho da grade influencia no tempo de cálculo porque ele controla diretamente o número de edgels produzidos, e todos os cálculos envolvidos são lineares com o número de edgels.

Com uma grade de tamanho 1, que corresponde a analisar todos os pixels da imagem, foram encontrados em média cerca de 45.000 edgels nas imagens da base YorkUrbanDB. Teoricamente, o número de edgels deveria decair em proporção inversa com o tamanho da grade, já que ela realiza uma amostragem periódica das linhas das imagens. A relação de proporcionalidade inversa entre o número de observações e o tamanho da grade também existe do ponto de vista da área total da imagem que é analisada pela grade. Este efeito foi comprovado durante os experimentos. Por exemplo, para uma grade de tamanho 16 foram encontrados em média  $2.750 \simeq 45.000/16$  edgels, em concordância com este modelo.

A correlação inversa entre o tamanho da grade e o tempo de cálculo pode ser observada nos valores da Tabela 5.1, assim como a correlação direta entre o tempo e o número de iterações do RANSAC. O tempo previsto de execução do *Corisco* em função de seus parâmetros poderia ser obtido a partir destas relações, considerando ainda a existência de processos de tempo constante, um custo adicional que não é afetado pelo tamanho da grade.

A terceira e quarta colunas da Tabela 5.1 trazem os valores de média e desvio padrão dos erros de estimação. As três últimas colunas da tabela trazem os ângulos que correspondem ao primeiro quartil (1/4), mediana e terceiro quartil (3/4) de cada distribuição de erros. Esta é uma forma de analisar estas distribuições de erros melhor do que a média e desvio padrão por se tratar de uma distribuição aproximadamente exponencial, assimétrica.

O erro de estimação é encontrado comparando cada orientação obtida pelo *Corisco* com a orientação de referência correspondente. O valor é calculado em cada caso aplicando-se à orientação de referência uma rotação inversa correspondente à orientação estimada, produzindo assim para cada teste um conjunto de parâmetros de uma rotação residual. O ângulo desta rotação residual é tomado como medida de erro, independente do eixo de rotação. É interessante notar que a comparação entre as orientações é feita associando-se os eixos que se encontram mais próximos. Consequentemente o maior erro possível de se encontrar é de  $60^\circ$ .

Com relação às estatísticas dos métodos testados por Denis, Elder e Estrada (2008), deve ser notado que os valores foram retirados da análise de um gráfico de Denis (2008), já que não foram fornecidos valores em tabelas nas referências, e por isso os valores apresentados aqui não possuem uma boa precisão. Os valores foram obtidos pela inspeção

do gráfico em um programa de manipulação de imagens. Com relação aos tempos destes métodos, o processador utilizado naqueles experimentos possuía uma velocidade de relógio de 1,83GHz, enquanto nos testes do *Corisco* foi utilizado um processador Intel Core 2 quádruplo de 2,666GHz. Assim pode ser aplicado um fator de 0,7 vezes para comparar de forma mais adequada os resultados relatados por Denis, Elder e Estrada (2008) com os obtidos neste experimento.

As Figuras 5.1 e 5.2 trazem as distribuições de erro e tempos de cálculo medidos para todas as combinações de parâmetros testadas com o *Corisco* para as imagens da base YorkUrbanDB. Os gráficos à esquerda mostram as distribuições de erro das soluções obtidas comparadas aos valores de referência do banco de dados. Os gráficos à direita mostram o tempo médio de cálculo para cada combinação de parâmetros. A curva contínua mostra o tempo total gasto, e a tracejada mostra apenas o tempo gasto no passo RANSAC individualmente. O número de iterações testadas foram 10.000, 1.000 e 200, e os tamanhos de grade utilizados foram as potências de 2 no intervalo de 1 até 128.

Os gráficos de erro utilizam os chamados diagramas de caixa. Os retângulos indicam a região entre o primeiro quartil e o terceiro quartil de cada distribuição, e a linha cortando cada caixa, com uma cruz no centro, é a mediana da distribuição. Um símbolo 'x' foi utilizado para marcar ainda a localização da média de cada distribuição, que se localiza sempre além da mediana nestas distribuições em decorrência do formato aproximadamente exponencial delas. As linhas tracejadas partindo das extremidades de cada caixa indicam nestas figuras os pontos de 5% e 95% das distribuições, e as primeiras e últimas 5 amostras foram desenhadas individualmente em suas posições sobre cada linha. Isto permite ver como ocorrem ocasionalmente casos com grandes erros nestas distribuições. Estas poucas amostras com grandes erros possuem uma grande influência no cálculo do valor da média, e é por este motivo que a mediana seria mais indicada na análise deste tipo de distribuição.

É possível observar nos gráficos a forma como a precisão da estimação e o tempo de cálculo variam com o tamanho da grade. Enquanto o tempo diminui conforme a grade aumenta, tornando o processo mais rápido, os erros tendem a aumentar resultando em uma piora da precisão. Desta forma o tamanho da grade permite controlar um comprometimento entre velocidade e precisão dos resultados do *Corisco*. E enquanto o tempo de cálculo segue uma curva bastante regular, coerente com uma proporção inversa somada a um valor constante, as distribuições de erro tendem a começar com pouca variação, mas a partir de um certo ponto passam a crescer mais rapidamente. Por exemplo, em ambos gráficos da Figura 5.1 é possível notar uma variação brusca no terceiro quartil da distribuição entre os casos com  $g = 32$  e  $g = 64$ , enquanto há menos variação entre tamanhos de grade menores.

Os gráficos também demonstram que aumentar o número de iterações do passo RANSAC é benéfico para a qualidade da estimação. Porém este processo pode acabar se

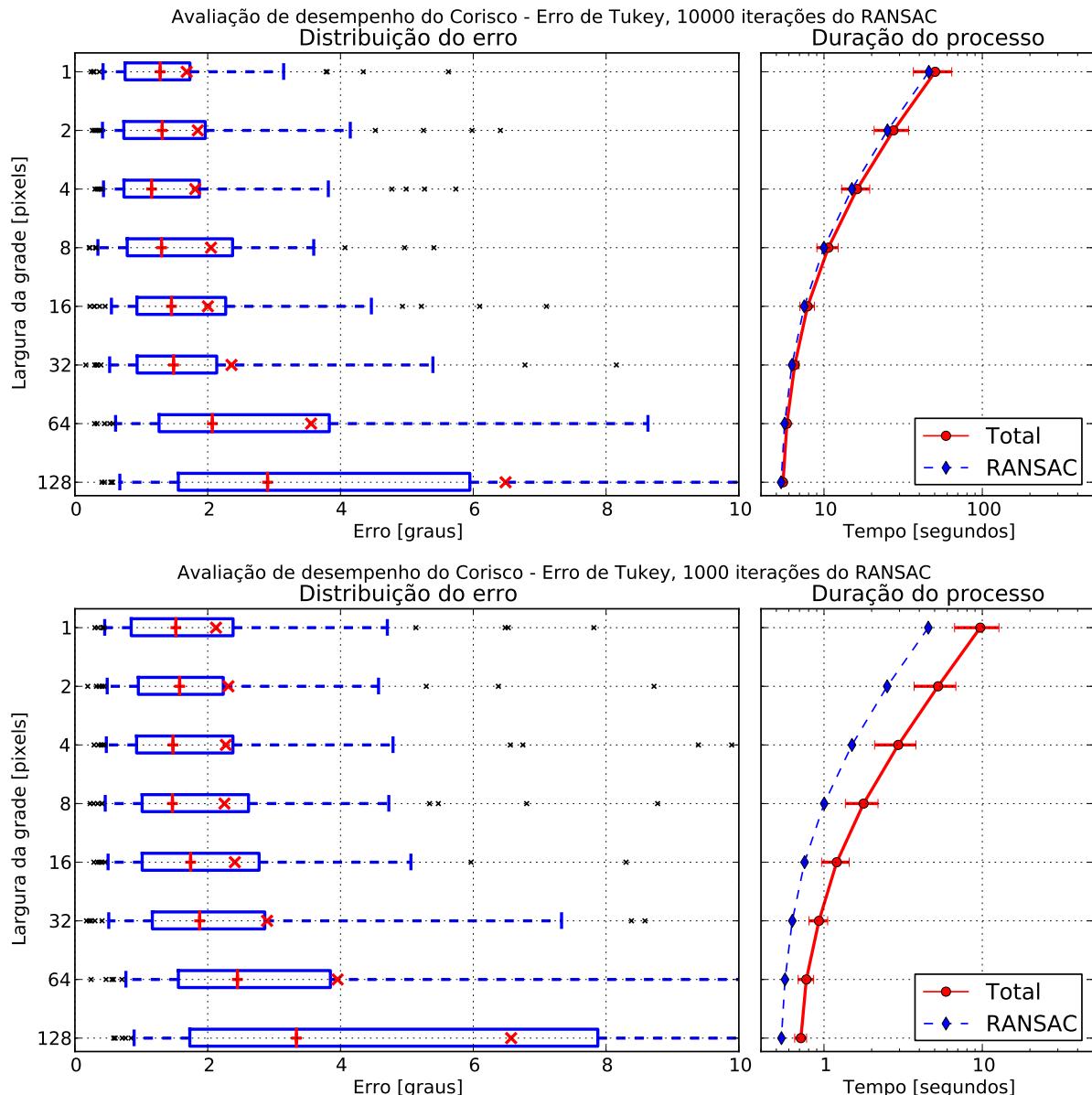


Figura 5.1: Distribuições de erro e tempo de cálculo exibidos pelo *Corisco* na análise do YorkUrbanDB com 10.000 (gráfico de cima) e 1.000 (gráfico de baixo) iterações na etapa RANSAC. As distribuições estão representadas através de diagramas de caixa, que indicam a mediana, os pontos de primeiro e terceiro quartil, e limites. Os pontos isolados em cada linha são amostras individuais que apresentaram valores extremos de erro, e foram plotadas como se fossem outliers. No gráfico de tempo as barras de erro mostram o desvio padrão das medições, e os pontos os valores médios. A curva tracejada indica o tempo gasto apenas pelo passo RANSAC.

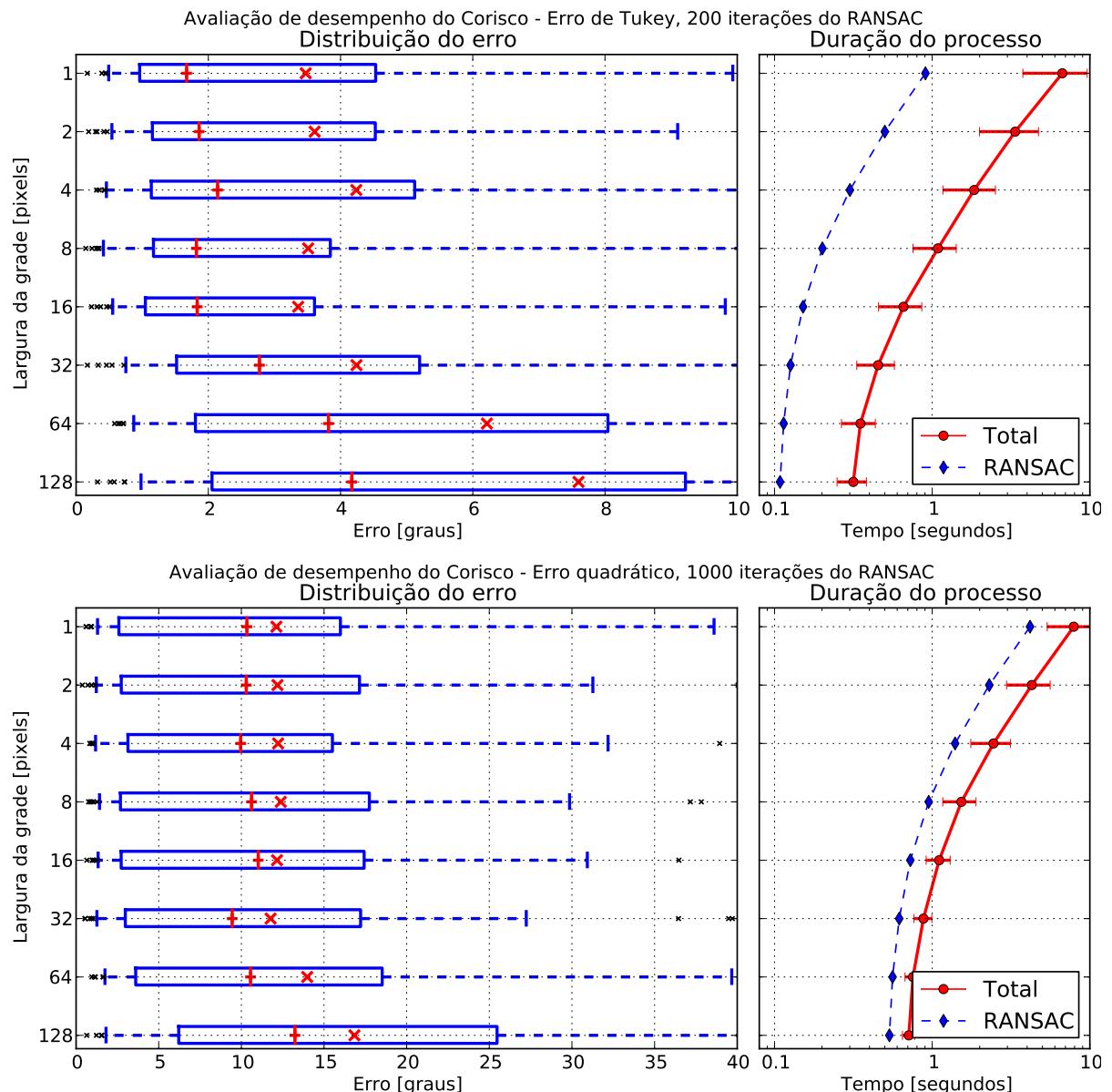


Figura 5.2: Distribuições de erro e tempo de cálculo exibidos pelo *Corisco* na análise do YorkUrbanDB com 200 iterações na etapa RANSAC (gráfico de cima), e utilizando um erro quadrático com 1000 iterações do RANSAC (gráfico de baixo).

tornando bastante custoso, e chega a dominar o tempo total de execução para  $i = 10.000$ , com as curvas no gráfico de tempo praticamente superpostas.

O par de gráficos na extremidade inferior da Figura 5.2 mostra o resultado da aplicação do *Corisco* com  $i = 1.000$ , porém utilizando uma função de erro simplesmente quadrática, e não redescendente como a função biquadrática de Tukey. O grande impacto negativo desta troca pode ser observado no crescimento do erro comparado com o outro gráfico com  $i = 1.000$  da Figura 5.1. A mediana das distribuições no caso do erro quadrático ultrapassa os  $10^\circ$ , que é o limite da região desenhada nos gráficos relativos à função de Tukey. Isto demonstra como a escolha de uma função de erro adequada é de vital importância para o bom funcionamento do *Corisco*.

Os métodos com melhor precisão investigados por Denis, Elder e Estrada (2008) são o *EM Newton* e o *MAP Quasi-Newton*, de acordo com o valor da mediana da distribuição de erro. Ambos métodos apresentaram uma distribuição de erros bastante próxima na região até o terceiro quartil, com uma diferença mais significativa apenas dentro do último quartil da distribuição, e portanto os valores na Tabela 5.1 são idênticos. O método *EM Newton* também é o método mais custoso estudado, e o *MAP Quasi-Newton* o segundo mais custoso, enquanto o método *EM Quasi-Newton* se revelou mais rápido porém menos preciso, seguindo portanto o comprometimento existente neste problema.

Os métodos alternativos mais precisos na Tabela 5.1 exibem uma mediana e ponto de terceiro quartil de aproximadamente  $2,61^\circ$  e  $4,1^\circ$ , e o tempo total de cálculo é da ordem de 6 segundos no melhor caso. O *Corisco* foi capaz de apresentar estatísticas ainda melhores do que estas em termos de precisão, atingindo um valor mínimo de mediana de  $1,11^\circ$ , mostrando que o *Corisco* é um método não apenas válido mas vantajoso comparado às alternativas para este problema.

Com relação à velocidade computacional, nos casos de maior precisão como  $i = 10^4$  e  $g = 4$  o *Corisco* apresentou uma velocidade comparável ao *EM Newton* e melhor precisão, mas variando os parâmetros é possível alcançar uma distribuição de erros comparável, mais próxima de  $2^\circ$  do que de  $1^\circ$ , com tempos de processamento menores do que 6 segundos. Este é o caso com  $i = 10^3$  e  $g = 32$ , que como pode ser visto na Tabela 5.1, apesar de manter a mediana em  $1,86^\circ$  o tempo médio de processamento cai para menos de 1 segundo, que é uma velocidade comparável aos métodos mais rápidos investigados por Denis, Elder e Estrada (2008) tal como o *EM Quasi-Newton*. Desta forma, mesmo considerando diferenças da velocidade dos computadores e ignorando o tempo de análise inicial ausente dos 6 segundos reportados para o método alternativo, a velocidade exibida pelo *Corisco* com estes parâmetros ainda é maior.

Este experimento demonstrou que o *Corisco* é um método bastante flexível e poderoso, alcançando desempenhos comparáveis senão superiores às técnicas existentes semelhantes, baseadas em edgels. A flexibilidade é causada em primeiro lugar pela subamostragem dos dados através da variação da máscara em forma de grade, e ainda pelo

ajuste possível do número de iterações da busca aleatória no início do processo de otimização.

O *Corisco* é capaz de entregar tanto uma precisão quanto eficiência seguramente comparáveis às das propostas existentes alternativas. Mas é preciso lembrar ainda que o *Corisco* pode lidar com imagens em orientações absolutamente genéricas, e não assume que a câmera esteja em uma orientação aproximadamente vertical. Além disso, enquanto os métodos alternativos discutidos nesta seção foram desenvolvidos apenas para o modelo de câmera *pinhole*, o *Corisco* pode ser facilmente aplicado para qualquer modelo de câmera, como será demonstrado nas seções a seguir.

## 5.2 Comparação com um método baseado em pontos

O segundo experimento realizado foi uma análise de um conjunto de imagens de uma mesma cena obtidas com duas câmeras diferentes. Estas imagens foram capturadas especialmente para esta pesquisa, e o conjunto foi denominado ApaSt — uma referência à Rua Apa, localizada nas imediações da cena capturada na cidade de São Paulo. As câmeras utilizadas foram as seguintes:

- Sony α230, câmera reflex digital com resolução nativa de 3880x2608 e imagens gravadas no formato raw.
- Nokia N8, um smartphone com câmera, com resolução nativa de 4000x3000 pixels e imagens gravadas em JPEG.

Estas imagens obtidas foram reduzidas para as dimensões 1000x672 e 1000x750, respectivamente. Um filtro Gaussiano com fator de suavização  $\sigma = 1,0$  também foi aplicado nestas imagens antes da extração de edgels.

As imagens foram capturadas seguindo uma trajetória aproximadamente linear, em 12 pontos diferentes para cada câmera e orientação. Cada câmera foi utilizada tanto na orientação do tipo paisagem, quanto na orientação retrato, ou seja, com uma rotação de aproximadamente 90° ao redor do eixo ótico. A existência de imagens nestas duas orientações costuma beneficiar a estimativa de alguns parâmetros, principalmente o centro ótico da câmera. Contando com as duas câmeras e as duas orientações, o conjunto ApaSt contém no total 48 imagens.

Para obter parâmetros de orientação de referência para avaliar o funcionamento do *Corisco* com estas imagens foi utilizado um método de visão multi-ocular baseado em pontos, mais especificamente o método desenvolvido e implementado por Snavely, Seitz e Szeliski (2006). Para isto foi utilizado o código disponibilizado por estes autores na Internet, um programa denominado *Bundler*.

Por se tratar de um método de *bundle adjustment*, o resultado do *Bundler* contém não apenas as orientações de cada câmera, mas também a posição de cada câmera no espaço, além de suas distâncias focais e parâmetros de distorção radial de lente, todos estimados



Figura 5.3: Duas imagens do conjunto ApaSt. A da esquerda, em orientação paisagem, foi obtida com a câmera Sony α230, e a da direita, em orientação retrato, com o Nokia N8.



Figura 5.4: Modelo tridimensional do ambiente reconstruído pelo *Bundler*. Trata-se de uma nuvem de pontos que delineiam os contornos dos edifícios observados na cena.

individualmente para cada imagem. O *Bundler* foi desenvolvido com foco em coleções heterogêneas, o que permitiu então misturar facilmente as imagens obtidas pelas duas câmeras. O processo produz também um conjunto de pontos, que constituem a estrutura do ambiente reconstruído a partir das imagens. A Figura 5.3 mostra um par de imagens do conjunto ApaSt, e a Figura 5.4 mostra duas visões do conjunto de pontos encontrados pelo *Bundler*.

Seria mais desejável poder especificar quais imagens são de qual câmera, porém o *Bundler* não permite fazer isto facilmente. As distâncias focais foram encontradas para cada câmera pela média do valores estimados para cada imagem. Os parâmetros encontrados foram

- Sony α230:  $f = 788,85$  ( $\sigma : 2,70$ ),  $\kappa = -160,08 \times 10^{-9}$  ( $\sigma : 34,58 \times 10^{-9}$ ),

- Nokia N8:  $f = 847,67 (\sigma : 1,64)$ ,  $\kappa = 111,90 \times 10^{-9} (\sigma : 9,19 \times 10^{-9})$ .

Enquanto a primeira câmera possui distorção do tipo barril ( $\kappa < 0$ ), a segunda possui distorção do tipo almofada. O *Bundler* assume que o centro ótico das câmeras se localiza no centro da área da imagem, portanto estes parâmetros não foram estimados, e para realizar o restante do experimento também foi utilizado o centro da imagem no *Corisco*, além destes outros parâmetros intrínsecos estimados pelo *Bundler*.

Após a obtenção dos parâmetros intrínsecos e das orientações de referência, o *Corisco* foi executado para estimar as orientação de cada imagem. Assim como no experimento anterior, foram utilizadas diferentes combinações de parâmetros de controle para produzir medições do tempo médio gasto pelo processo, além do erro das estimativas comparadas com as orientações de referência obtidas pelo *Bundler*.

O erro de estimativa foi obtido através do ângulo de uma rotação residual, assim como no experimento anterior. Mas há um problema neste caso, que é o fato do *Bundler* ignorar a existência de um referencial natural no ambiente. As orientações de câmera obtidas são portanto dadas de acordo com um referencial arbitrário, e para comparar estes valores com os encontrados pelo *Corisco* faz-se necessário realizar uma estimativa da rotação existente entre o referencial do *Bundler* e o referencial natural do ambiente, utilizado pelo *Corisco*.

Uma forma de encontrar esta rotação entre os dois referenciais é através da solução de um sistema linear montado a partir da expressão

$$\Psi_{\text{Bundler}} \Psi_t = \Psi_{\text{Corisco}} \quad (5.1)$$

onde o produto Hamiltoniano entre a transformação desconhecida  $\Psi_t$  e uma orientação de referência  $\Psi_{\text{Bundler}}$ , escrita na forma de uma matriz, produz os componentes da orientação estimada  $\Psi_{\text{Corisco}}$ . Utilizando-se esta expressão para cada imagem do conjunto é possível montar um sistema linear com quatro linhas para cada imagem. O resultado são os parâmetros de um quaternion que deve ser multiplicado às orientações encontradas originalmente para finalmente produzir orientações de referência que podem ser comparadas diretamente com as orientações encontradas pelo *Corisco*.

Segundo a teoria esta rotação estimada deveria bastar para transformar os parâmetros de um sistema referencial para o outro, e permitir a comparação das orientação. Porém observou-se no experimento que, para conseguir realizar a transformação com sucesso, seria necessário ainda trocar de lugar dois dos parâmetros dos quaternions de referência antes da rotação ser estimada e aplicada. Isto pode ser necessário devido a alguma diferença de representação dos dados não devidamente esclarecida. A não realização desta alternação de parâmetros resulta em resultados claramente equivocados, com a produção de curvas no espaço transformado de aspecto correto porém multiplicadas por  $-1$ , por exemplo.

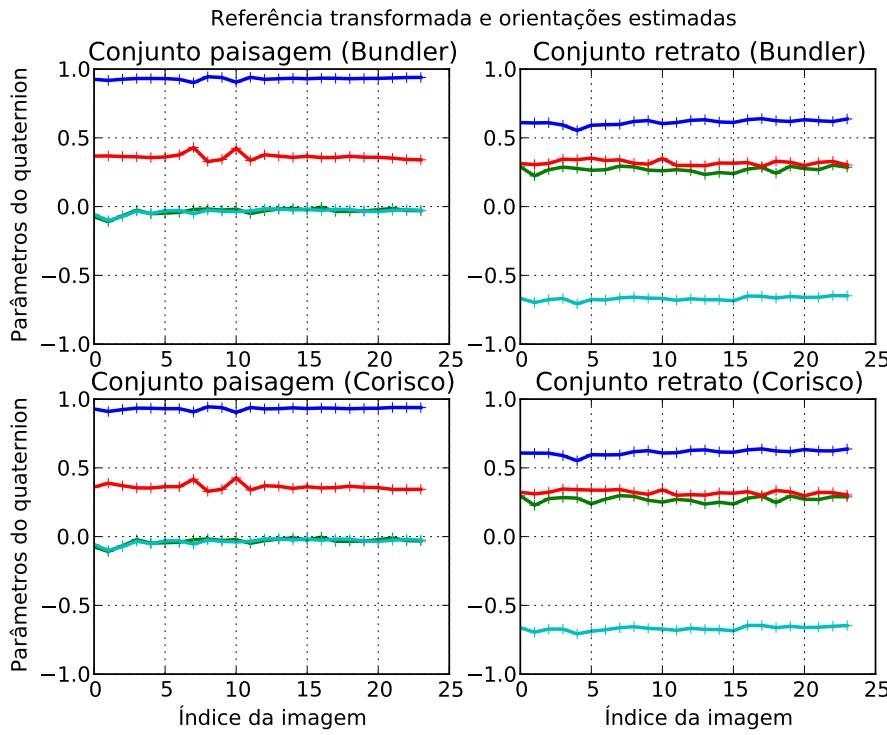


Figura 5.5: Valores das componentes dos quaternions de referência transformados, e dos quaternions estimados pelo *Corisco* para as imagens da série ApaSt.

A Figura 5.5 contém gráficos com os valores dos parâmetros dos quaternions de referência transformados segundo este procedimento explicado, e dos quaternions encontrados pelo *Corisco* para as imagens do conjunto ApaSt com  $i = 10.000$  e  $g = 1$ . Como é possível observar, há em geral uma grande concordância entre as curvas, com apenas alguns poucos pontos apresentando maiores desvios.

A Figura 5.6 mostra os erros resultantes do mesmo caso com  $i = 10.000$  e  $g = 1$ . Os erros se limitam em geral à região abaixo de  $2^\circ$ , com uma distribuição de aspecto exponencial, exceto por um par de *outliers* com erros acima de  $10^\circ$ . As Figuras 5.7 e 5.8 trazem os resultados de testes com o *Corisco* para outras combinações dos parâmetros de controle, com  $i = 10.000, 1.000$  e  $200$ , e  $g$  de  $1$  a  $128$ . Nestes diagramas de caixa as linhas tracejadas nas extremidades esquerdas indicam toda a região do primeiro quartil, enquanto as linhas da direita mostram a posição da quarta pior amostra, enquanto as três piores amostras foram desenhadas explicitamente como *outliers*.

No caso  $i = 10.000$  e  $g = 8$  o *Corisco* apresentou um erro de estimação com mediana e terceiro quartil iguais a  $0,88^\circ$  e  $1,22^\circ$ , além de um erro máximo menor do que  $5^\circ$ . O tempo médio de análise neste caso foi de 11,02 segundos. Este é um nível de precisão ainda maior do que os resultados encontrados no experimento anterior, o que pode decorrer da modelagem mais adequada, considerando distorções radiais, ou pode ser apenas fruto de variações naturais nos dados de ambos problemas. De qualquer forma, é

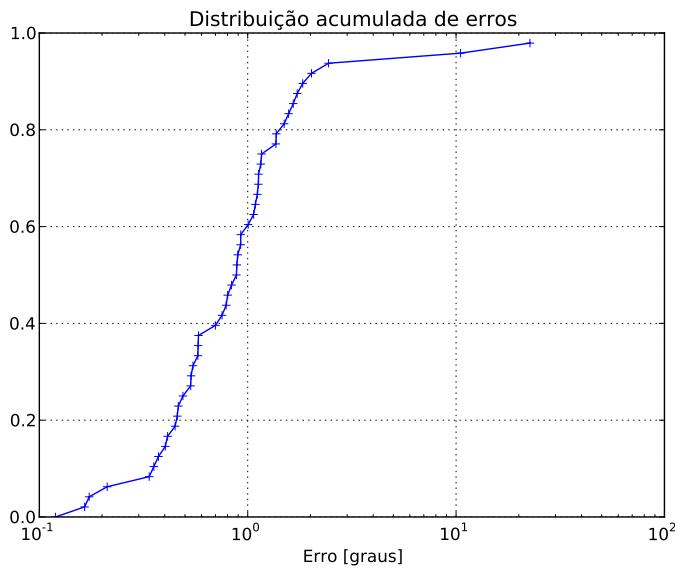


Figura 5.6: Distribuição de erros de estimação do *Corisco* para as imagens do conjunto ApaSt analisadas com  $i = 10.000$  e  $g = 1$ .

bom notar que neste experimento as referências foram obtidas através de um método de natureza bastante diferente do *Corisco*, e que sequer utiliza a condição de ambiente antrópico. Considerando isto, o nível de erro encontrado, em torno de  $1^\circ$ , parece satisfatório o bastante para validar o *Corisco* como um método capaz de adquirir boas estimativas de orientação em ambientes antrópicos.

O processo do *Bundler* é relativamente demorado, requerendo cerca de 16 minutos para produzir os resultados utilizados neste experimento. O *Bundler* depende necessariamente de analisar conjuntos de imagens simultaneamente, e além disso a estrutura do ambiente resultante é uma nuvem de pontos que não explora qualquer tipo de restrição. O *Corisco* poderia complementar esta técnica de forma interessante, explorando uma restrição do ambiente e permitindo analisar imagens individualmente para encontrar boas estimativas de alguns dos parâmetros buscados, alcançando uma boa precisão em um tempo relativamente rápido. Neste experimento o *Corisco* levou apenas cerca de um minuto para encontrar estimativas para a orientação de todas as imagens, por exemplo no caso  $i = 10^3$  e  $g = 16$ , e esta informação poderia ser utilizada pelo *Bundler* não só para realizar a otimização com uma boa estimativa inicial mas também para acelerar o processo de associação das observações. Este experimento serve tanto para mostrar mais avaliações quantitativas do *Corisco* quanto para mostrar melhor a forma como ele se compara a métodos de visão contemporâneos baseados em técnicas fundamentalmente diferentes, mas que poderiam ser utilizadas em conjunto.

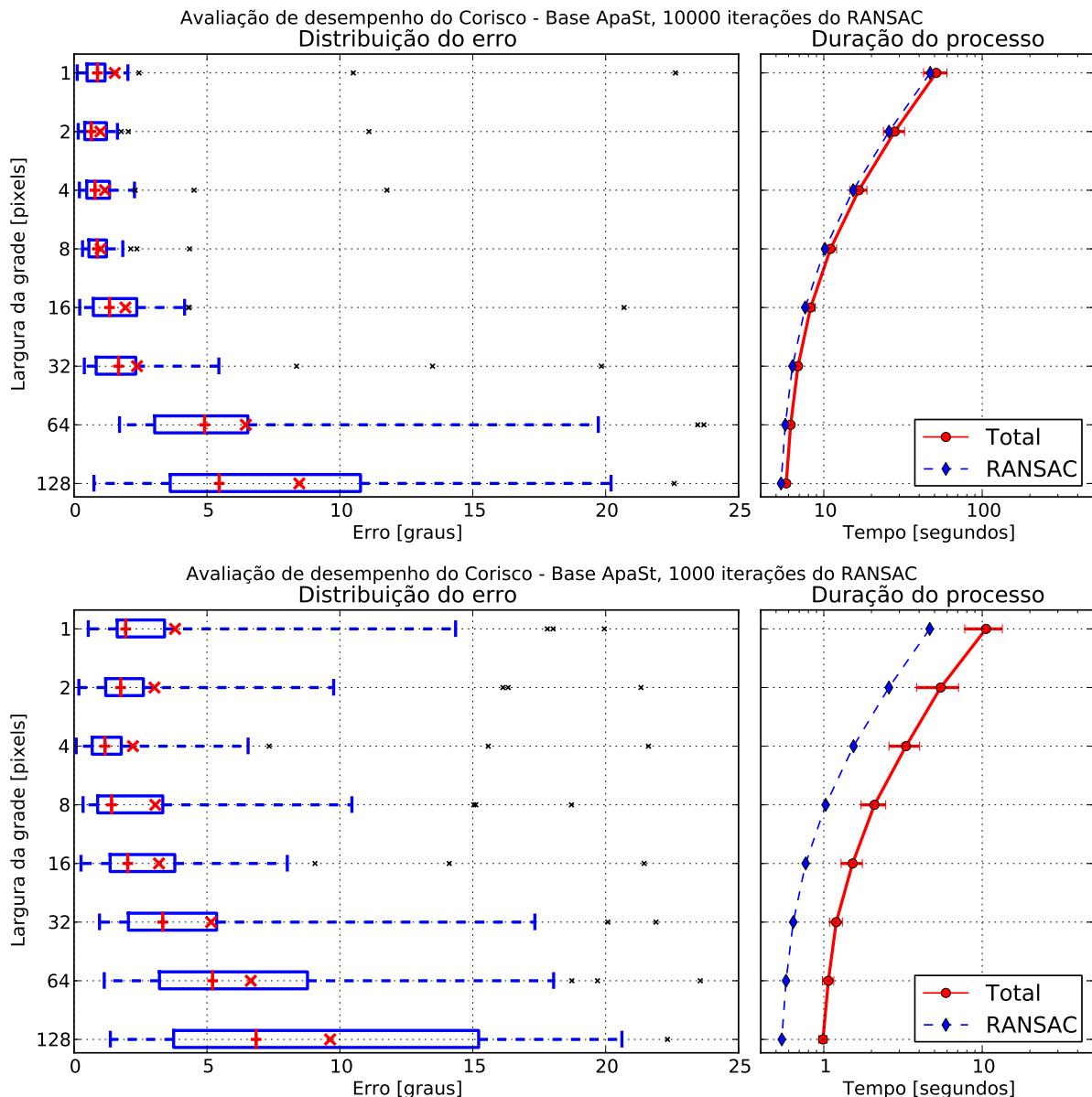


Figura 5.7: Distribuições de erro e tempo de cálculo exibidos pelo *Corisco* na análise da base ApaSt com 10.000 (gráfico de cima) e 1.000 (gráfico de baixo) iterações na etapa RANSAC. Nestes diagramas de caixotes os limites inferiores mostram a posição da amostra de menor erro, e os limites superiores a posição da quarta pior amostra.

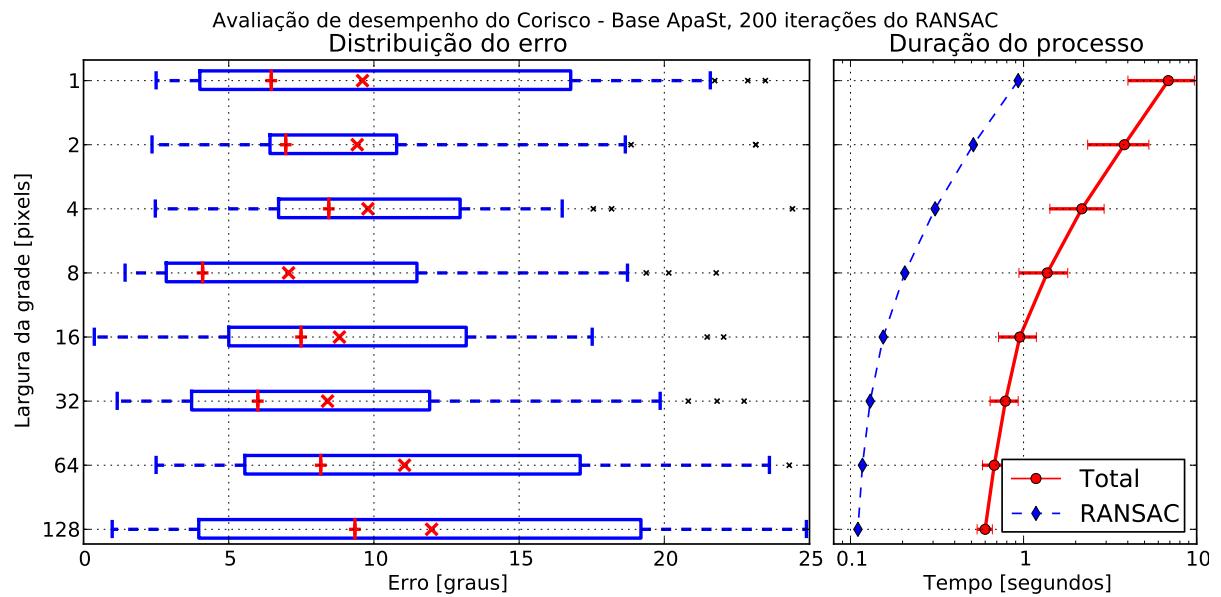


Figura 5.8: Distribuições de erro e tempo de cálculo exibidos pelo *Corisco* na análise da base ApaSt com 200 iterações na etapa RANSAC.

### 5.3 Estimação de distância focal baseada no Corisco

Como foi discutido nos Capítulos 3 e 4, existe a possibilidade de se utilizar a função objetivo do *Corisco* para realizar não somente a estimativa da orientação mas também uma estimativa de parâmetros intrínsecos da câmera, tal como a distância focal. Esta possibilidade foi investigada por pesquisadores de métodos anteriores semelhantes ao *Corisco* (DEUTSCHER; ISARD; MACCORMICK, 2002; SCHINDLER; KRISHNAMURTHY; DELLAERT, 2006), e o experimento relatado nesta seção busca demonstrar como a função objetivo utilizada para estimar a orientação no *Corisco* pode ser efetivamente explorada de forma semelhante à demonstrada nestes outros métodos existentes.

Apesar da calibração não ser o foco principal de aplicação do *Corisco*, é importante compreender que as restrições geométricas existentes no problema permitem realizar isto. O procedimento que será descrito a seguir não é necessariamente a melhor forma de estimar os parâmetros intrínsecos de uma câmera, mas como será demonstrado, os resultados produzidos podem ser úteis. E ainda pode ser possível que no futuro este procedimento seja modificado para criar uma forma mais satisfatória de calibrar câmeras com base no *Corisco*, possivelmente como um passo inicial de um processo maior de estimativa.

No experimento realizado foram analisadas as mesmas imagens do experimento da Seção 5.2, a série de imagens ApaSt. O *Corisco* foi então utilizado para encontrar as orientações de todas as imagens utilizando os parâmetros  $i = 10^4$  e  $g = 8$ , além dos valores de distância focal e distorção radial encontrados pelo *Bundler*.

Uma vez que as orientações foram encontradas com o valor de distância focal de referência, novas estimativas de orientação foram procuradas para valores diferentes da

distância focal. Estes testes foram realizados com vários valores de distância focal acima e abaixo do valor de referência. Após cada novo teste foi armazenado o valor final da função objetivo do *Corisco*, sobre a orientação ótima encontrada com aquela distância focal testada. Cada nova estimativa de orientação foi realizada utilizando como estimativa inicial a orientação encontrada nos primeiros testes, e apenas a segunda etapa do processo de otimização foi realizada. Ou seja, apenas o algoritmo FilterSQP foi empregado durante os testes com os diferentes valores de distância focal, sem novas iterações do algoritmo RANSAC. Esta é uma possibilidade de modificação do funcionamento usual do *Corisco* mencionada na Seção 1.2.2.

A função objetivo do *Corisco* é utilizada normalmente apenas para estimar a orientação. Para que ela também possa ser utilizada para a estimativa de distância focal, ou de outros parâmetros, os valores obtidos neste processo de varredura realizado devem possuir um mínimo local no valor de distância focal das câmeras, que são supostamente os valores de referência obtidos com o *Bundler*. O objetivo do experimento é avaliar esta propriedade, ou seja, constatar que a função objetivo do *Corisco* apresenta um mínimo local sobre a distância focal das câmeras quando este parâmetro é variado, sendo que para cada novo valor testado encontra-se sempre uma nova orientação ótima correspondente.

Os gráficos dos valores da função objetivo encontrados após a aplicação do *Corisco* para cada valor de distância focal testado se encontram na Figura 5.9. Os dois gráficos no topo da figura mostram curvas que são a soma dos valores da função objetivo de todas as 24 imagens obtidas com cada uma das duas câmeras. É portanto um somatório das curvas individuais obtidas para cada uma das imagens. Os gráficos da parte inferior da Figura 5.9 mostram todas as curvas individuais, superpostas. Todas as curvas nestes gráficos foram transladadas de forma a alinhar seus pontos mínimos com o valor zero, enquanto na realidade estes pontos mínimos possuem diversos valores positivos, mas irrelevantes para esta análise. Apesar de transladadas, as curvas foram desenhadas na mesma escala original, e os valores das curvas de erro total são naturalmente maiores do que os das curvas individuais, já que as primeiras são as somas das posteriores.

É possível observar claramente na Figura 5.9 que as curvas obtidas são côncavas. Além disto seus pontos mínimos, marcados com um triângulo, localizam-se de fato próximos dos valores de referência da distância focal, indicados nos gráficos através de linhas verticais tracejadas. Os pontos mínimos das curvas de erro total localizam-se bastante próximos dos valores de referência, indicando que esta talvez possa ser de fato uma boa forma de estimar este parâmetro. Já no caso das imagens individuais há erros maiores, o que indicaria que um processo de estimativa de distância focal baseado nesta função objetivo talvez não apresente uma precisão muito boa na análise de uma única imagem.

Os valores médios das posições do ponto mínimo obtidos para cada imagem separada foram 859.85 pixels para a *Sony α230* e 886.52 pixels para o *Nokia N8*. Os valores de desvio padrão do erro destas estimativas em comparação com a distância focal de

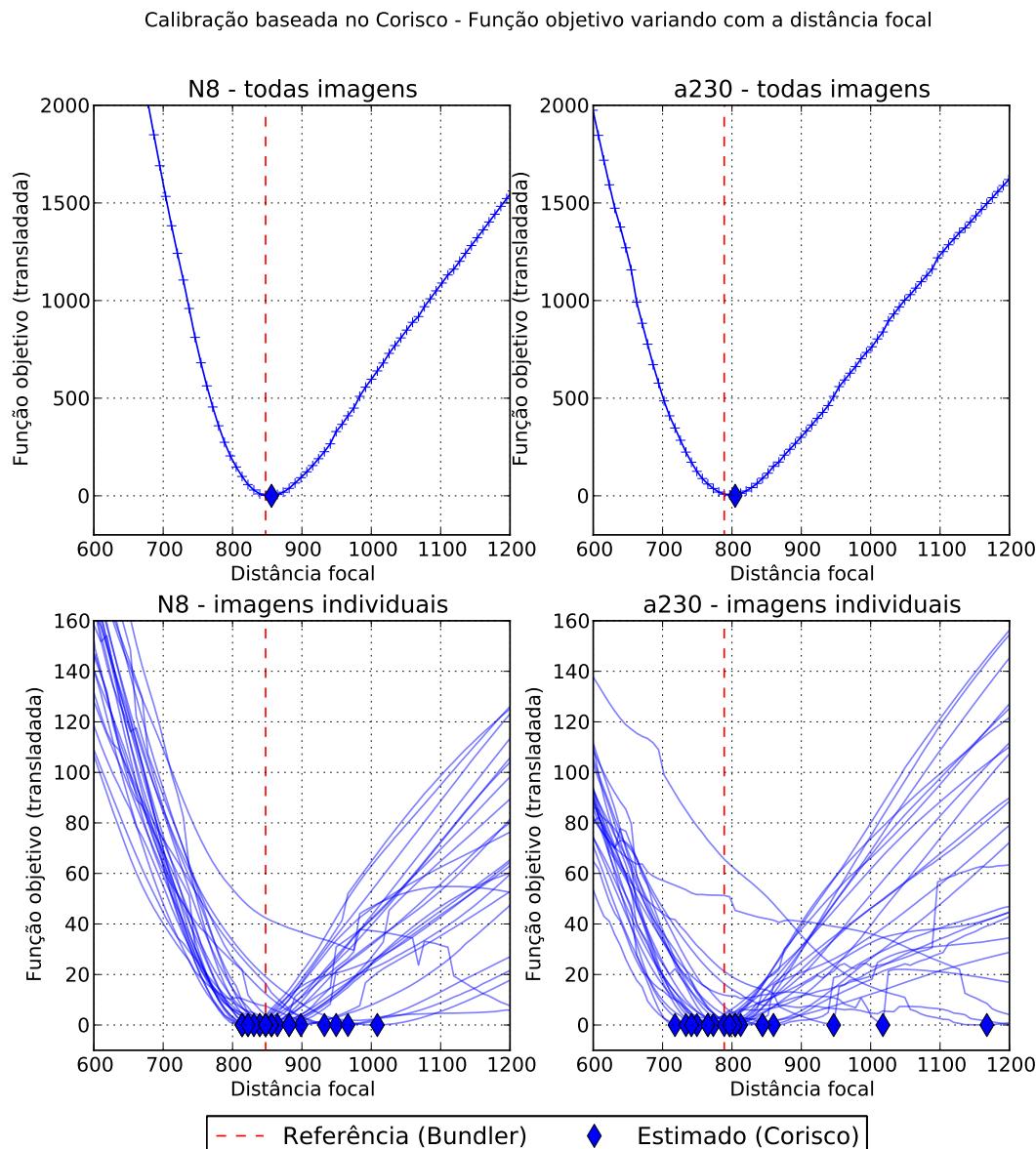


Figura 5.9: Valor da função objetivo do *Corisco* na solução encontrada para diferentes valores de distância focal. Os gráficos à esquerda se referem à câmera Nokia N8, e os da direita à câmera Sony a230. Os gráficos do topo mostram a soma dos valores para todas as imagens de cada câmera, e os gráficos abaixo mostram exemplos de curvas obtidas para uma única imagem de cada subconjunto.

referência foram 151,96 para a *Sony α230* e 106,77 para o *Nokia N8*. Os valores das médias encontradas estão a menos de 100 pixels dos valores de referência, o que não é um erro muito grande, porém os valores de desvio padrão são relativamente altos. É possível notar nos gráficos que a distribuição das estimativas individuais tende a apresentar erros maiores quando a estimativa está acima do valor de referência. Isto é coerente com o fato de que algumas das curvas possuem uma inclinação mais forte na parte esquerda do que na direita.

A existência de tantos *outliers* entre estas estimativas significa que o método não seria muito confiável para analisar imagens individuais, apesar da média total não parecer ser uma estimativa ruim. Um procedimento de calibração que possa analisar diversas imagens obtidas com uma mesma câmera minimizando uma função de erro total parece ser portanto uma alternativa que ofereceria uma grande melhoria na qualidade das estimativas.

Com relação ao tempo de cálculo deste processo de análise, foram testados 121 valores diferentes de distância focal para os dois conjuntos de 24 imagens, o que significa que o algoritmo FilterSQP foi executado 2.904 vezes para cada conjunto. O tempo total de execução foi de 1608,76 segundos para o conjunto de imagens da *Sony α230* e 1825,08 para o *Nokia N8*, o que significa uma média de 0,6 segundos para cada nova otimização.

Este experimento demonstrou que a função de erro utilizada pelo *Corisco* possui de fato um potencial para a aplicação na estimação de parâmetros intrínsecos. No testes realizados esta função apresentou mínimos locais próximos aos valores de referência da distância focal das duas câmeras testadas, lembrando ainda que a orientação ótima para cada valor testado da distância focal foi repetidamente procurada através de um processo de otimização subjacente. Além disso foram utilizados neste experimento os valores do coeficiente de distorção fornecidos pelo *Bundler*, e assumiu-se que o centro ótico se localiza no centro das imagens.

O procedimento adotado neste experimento pode ser utilizado como uma forma de estimar a distância focal onde o processo de otimização adotado é uma busca determinística no espaço de parâmetros. Um processo de otimização mais sofisticado utilizaria algum algoritmo de otimização tal como o método de Newton, ou alternativamente o algoritmo de Nelder-Mead (FLETCHER, 2000; SINGER; NELDER, 2009), que poderia ser interessante por dispensar o cálculo de derivadas. Seria também desejável que o processo fosse capaz de estimar parâmetros de distorção radial, além do centro ótico. Mas ainda que tudo isto seja realizado, é de se esperar que a qualidade dos resultados de um procedimento baseado em edgels seja sempre inferior a algum procedimento baseado em extração de curvas, e é justamente a precisão dos resultados que é geralmente o aspecto mais relevante no problema da calibração, e não a velocidade de cálculo, que é a vantagem principal geralmente oferecida por métodos baseados em edgels, incluindo o *Corisco*.

Seria interessante pesquisar no futuro de que forma métodos de calibração baseados em edgels podem ser utilizados para acelerar uma calibração em seus primeiros estágios, e facilitar então processos de extração de curvas que seriam os responsáveis por encontrar as estimativas finais e mais precisas dos parâmetros procurados. Mas é bom notar que o fenômeno demonstrado nesta seção não indica somente uma possibilidade de estimar os valores de parâmetros intrínsecos com o método proposto. O *Corisco* e sua função de erro poderiam ser utilizados em procedimentos para a identificação de câmera, por exemplo, com uma comparação do erro obtido com diferentes modelos de câmera conhecidos para selecionar o que apresenta os melhores resultados. O experimento relatado aqui demonstrou que a função de erro utilizada no *Corisco* é uma alternativa que pode ser considerada na criação de uma aplicação deste tipo, não se limitando portanto apenas ao problema da estimação de orientação.

#### 5.4 Análise de imagens de projeção equiretangular

Neste experimento o *Corisco* foi utilizado para analisar imagens de projeção equiretangular. As imagens foram obtidas com uma câmera omnidirecional profissional colocada sobre um carro que foi guiado pelas ruas de uma cidade. O conjunto de imagens do banco de dados utilizado foi produzido pela Google para o projeto StreetView, e é disponibilizado sob demanda para pesquisadores interessados. Uma sequência de 250 imagens foi selecionada para o teste e redimensionadas para 1000x500 pixels, e então a orientação de cada imagem foi estimada separadamente utilizando o *Corisco* com  $i = 10.000$  e  $g = 4$ .

A orientação de referência existente no banco de dados é representada através de quaternions, que é a mesma parametrização utilizada no *Corisco*, porém assim como no experimento da Seção 5.2 é necessário utilizar a Equação 5.1 para encontrar uma transformação dos parâmetros de referência para o espaço das estimativas do *Corisco*. E de forma semelhante ao experimento da Seção 5.2 também, foi necessário para obter sucesso realizar uma mudança de sinais dos parâmetros de referência que não poderia resultar de uma simples rotação. O uso de uma transformação linear genérica para transformar os quaternions é uma alternativa mais poderosa e que deu resultados bastante satisfatórios, porém é desejável que a transformação seja modelada de forma mais restrita, o que leva à escolha por esta estimação de uma rotação associada a esta mudança de sinais.

A Figura 5.10 mostra cada componente dos quaternions estimados pelo *Corisco*, que são as linhas contínuas vermelhas, sobrepostos aos valores da referência transformada, que são as linhas tracejadas azuis. Como as imagens foram obtidas de um carro em movimento, é possível observar uma dinâmica guiando as curvas. Estas rotações são observáveis ao assistir a sequência das imagens como um vídeo. Além das curvas ao redor dos eixos  $y$  e  $z$ , é possível notar também nestes gráficos que a câmera encontra-se a uma certa inclinação constante em relação ao solo, causada por uma rotação ao redor do eixo  $x$  que varia muito pouco ao longo do tempo.

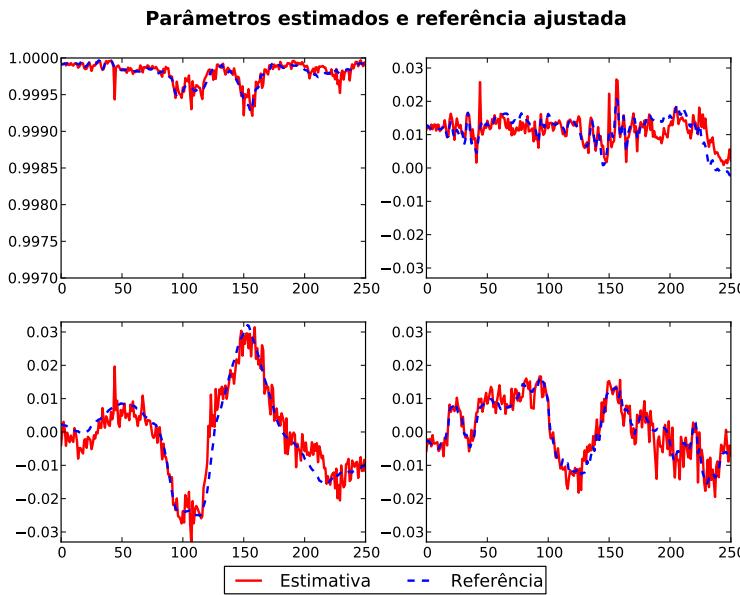


Figura 5.10: Valores das componentes dos quaternions de referência transformados, e dos quaternions estimados para as imagens da série estudada.

Apesar de haver uma dinâmica que restringe a forma dos dados, o *Corisco* foi aplicado considerando cada quadro individualmente. O resultado observado comparando as estimativas com a referência transformada é que 50% das imagens analisadas tiveram ao menos  $0,58^\circ$  de erro, e 90% apresentaram menos de  $1,02^\circ$ . O tempo médio de análise destas imagens foi de 17,35 segundos.

Como há pouca variedade nas orientações destas imagens analisadas, é possível comparar este resultado com um modelo simples onde há simplesmente uma orientação constante, calculada a partir de uma média dos valores da série. Neste caso o resultado é que 50% dos testes possuem um erro de  $1,52^\circ$  ou menos, e 90%  $3,03^\circ$  ou menos. Isto demonstra que o *Corisco* está medindo variações reais nas orientações das imagens da entrada, apesar delas serem sutis. A Figura 5.11 mostra a curva completa da distribuição de erros resultantes do *Corisco* e deste modelo com orientação constante.

Este experimento demonstrou novamente que o *Corisco* pode oferecer uma boa precisão na estimativa de orientação de câmera, porém analisando imagens individuais e sem realizar reconstrução. Neste caso as imagens possuem uma projeção cilíndrica de natureza bastante diferente das distorções radiais do experimento da Seção 5.2, mas isto não afetou a precisão alcançada pelo *Corisco*. Este experimento também é interessante porque mostra uma comparação da técnica com dados obtidos de instrumentos de natureza diferentes, não visuais. O *Corisco* mostrou um bom desempenho mesmo neste cenário bastante peculiar, demonstrando novamente sua flexibilidade e potencial para aplicação.

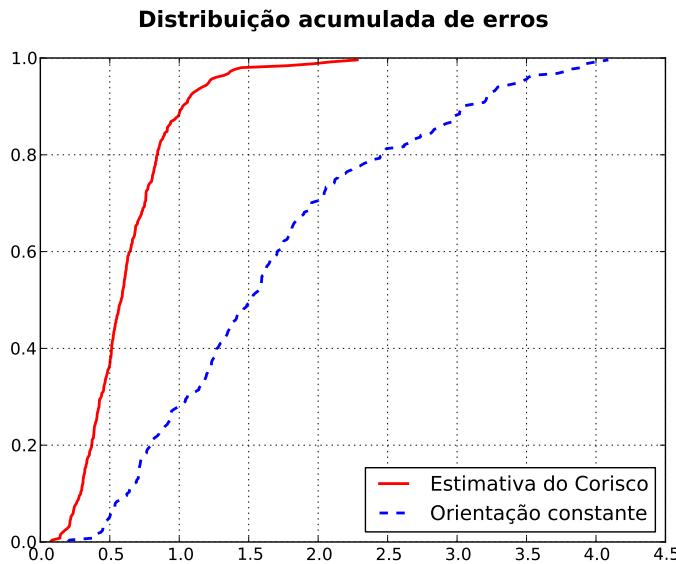


Figura 5.11: Curva de distribuição de erro para as orientações estimadas com o *Corisco* e com um modelo de orientação constante da sequência de imagens equiretangulares do Google StreetView.

## 5.5 Análise de imagens de projeção polar equidistante

No último experimento que será apresentado aqui foi analisada uma imagem obtida com uma lente olho-de-peixe. Os parâmetros intrínsecos da câmera foram obtidos inicialmente por um processo de calibração separado. Estes parâmetros foram então utilizados pelo *Corisco* para encontrar a orientação da câmera para uma imagem capturada em um ambiente interno. Para avaliar o resultado foram geradas imagens retificadas a partir da imagem original, utilizando a projeção pontual. Foram geradas cinco imagens, cada uma delas com o eixo ótico orientado a cada um dos eixos do referencial natural do ambiente. O alinhamento faz com que as projeções na imagem das linhas dos objetos do ambiente fiquem nas direções vertical, horizontal, ou apontando para um ponto de fuga localizado no próprio ponto principal da imagem. Consequentemente, os limites superior e inferior de uma parede se tornam linhas horizontais na imagem.

Uma textura, ou um desenho localizado sobre uma das paredes do ambiente pode aparecer nas imagens capturadas rotacionado de forma genérica, e ainda sofrer transformações não-rígidas de escala, ou seja, sem preservar a proporções do desenho original. Podem ocorrer até mesmo distorções não-lineares, com curvatura. Estas transformações dificultam a realização de tarefas como o reconhecimento de objetos. Uma retificação baseada apenas no modelo de câmera, sem levar em consideração a orientação da câmera com relação ao referencial natural do ambiente, é capaz apenas de eliminar as distorções de curvatura, mas a projeção retilínea ainda será capaz de deformar texturas das outras maneiras citadas anteriormente, além de causar a distorção de perspectiva se o plano da imagem não estiver alinhado com o plano do desenho. Uma retificação que alinha as

imagens sintetizadas com as direções dos planos do ambiente elimina a possibilidade de rotações e variações de escala não-rígidas, além da distorção de perspectiva, e uma textura qualquer desenhada sobre uma parede irá então aparecer nas imagens sintetizadas sofrendo apenas uma translação e variação rígida de escala, mantendo as proporções do desenho original.

Estas imagens retificadas com informação da orientação da câmera seriam portanto úteis para realizar tarefas de reconhecimento de objetos em ambientes antrópicos. Também seria possível utilizar estas imagens em um sistema de localização através de um mosaico de imagens do teto do ambiente, o que é uma técnica bem conhecida na Robótica Móvel (THRUN et al., 1999), porém a orientação seria estimada primeiro pelo *Corisco*, restando determinar apenas a sua posição a partir do mosaico que modela o ambiente, e da imagem retificada do teto. O fato da imagem estar alinhada significa que a busca da localização da imagem sintetizada sobre o mosaico pode ser realizada por algum método tal como a correlação de fase, tornando o processo mais rápido e simples do que se fosse necessário levar em consideração rotações da imagem buscada, além de distorções de perspectiva que surgem quando a câmera não está perfeitamente alinhada ao plano do teto.

O processo de calibração realizado utilizou como entrada 24 imagens capturadas de um alvo de calibração planar, contendo um padrão xadrez. Os vértices do padrão de calibração foram encontrados através do extrator de Harris e Stephens (1988), e 15 destes vértices foram identificados manualmente em cada imagem. Estes pontos extraídos e associados foram então utilizados para realizar a calibração resolvendo o problema de *bundle adjustment* considerando que a localização dos pontos do padrão de calibração eram conhecidas, e minimizando um erro quadrático da reprojeção dos pontos em cada imagem. Os parâmetros controlados neste processo de otimização foram os parâmetros de rotação e orientação de cada câmera, e os parâmetros intrínsecos do modelo de câmera, que são o ponto principal e um fator de escala, que é o coeficiente  $f$  da Equação 2.10. Este procedimento de calibração foi descrito com mais detalhes por Scatolini et al. (2012).

A imagem apresentada nas Figuras 4.5 e 4.6, que demonstram o resultado *Corisco* para imagens de lente olho-de-peixe, foi exatamente a imagem utilizada neste experimento. A Figura 5.12 mostra uma versão ampliada da imagem. A imagem foi capturada com a câmera apontando aproximadamente para o teto da sala, e com uma rotação de aproximadamente  $25^\circ$  ao redor do eixo  $z$ . A lente possui um campo de visão muito amplo, de mais de  $180^\circ$ , e portanto é possível observar todas as quatro paredes que delimitam o ambiente. As deformações da projeção fazem com que o espaçamento entre as lâmpadas do teto varie ao longo da imagem, apesar das lâmpadas estarem dispostas de maneira uniforme no ambiente. Todas as linhas horizontais do ambiente também aparecem curvadas de forma intensa, enquanto as linhas verticais do ambiente não sofrem muita distorção porque o eixo ótico da câmera, que aponta na direção do eixo  $z$  do referencial de câ-

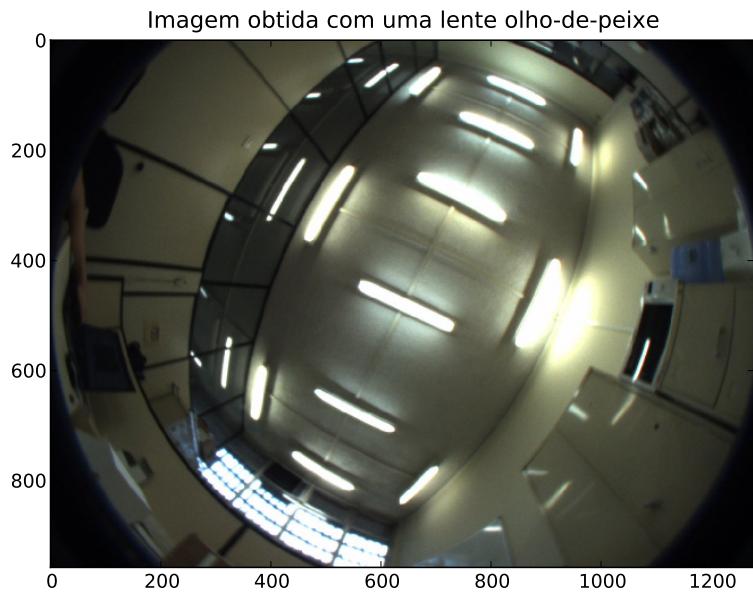


Figura 5.12: Imagem obtida com lente olho-de-peixe. Esta imagem é a fonte utilizada para a sintetização das imagens retificadas sem e com estimativa da orientação das Figuras 5.13 e 5.14.

mera, está aproximadamente alinhado com a direção vertical das retas do ambiente, que constitui a direção  $z$  do referencial do ambiente neste exemplo.

A Figura 5.13 mostra o que ocorre se for realizada uma retificação sem considerar a orientação da câmera com relação às linhas do ambiente. Nas imagens sintetizadas há muito pouca curvatura nas linhas. Idealmente haveria apenas segmentos de reta nestas imagens sintetizadas, porém erros na estimativa dos parâmetros internos e outras imperfeições no processo resultam na existência de curvaturas remanescentes nestas saídas. O não-alinhamento do referencial de câmera com o referencial natural do ambiente faz com que as linhas se encontrem em pontos de fuga localizados em posições genéricas nas imagens de saída. As regiões em branco nestas imagens sintetizadas são direções no espaço que estão fora dos limites da imagem capturada.

A imagem rotulada *Direção*  $-x$  na Figura 5.13 mostra uma projeção da região à esquerda na imagem da Figura 5.12. Nesta imagem de saída, assim como nas outras, o ponto principal foi marcado com um quadrado azul. Nesta imagem ele se localiza nas coordenadas (1000,900), e esta imagem foi criada considerando uma distância focal  $f = 500$  pixels. A distância focal de cada imagem de saída está especificada junto a ela nas figuras. É possível observar nesta imagem de saída da *Direção*  $-x$  que há um ponto de fuga para a esquerda para onde apontam a maioria das retas à direita na imagem, e um ponto de fuga à direita para onde se dirigem as retas à esquerda. As retas verticais do ambiente também estão se dirigindo para um ponto de fuga localizado para baixo da imagem. O desalinhamento da câmera com o ambiente também faz com que o espaçamento entre as lâmpadas

na imagem da *Direção z* continue desigual, assim como o tamanho das lâmpadas também varia.

Nestas primeiras imagens sintetizadas o programa FishRet (SCATOLINI et al., 2012) foi aplicado sem modificações, realizando a apenas o mapeamento da projeção polar equidistante para projeções retilíneas. A Figura 5.14 contém finalmente as imagens sintetizadas levando em consideração a orientação estimada pelo *Corisco*. Para criar estas novas imagens foi utilizada uma versão modificada do FishRet que permite considerar uma rotação qualquer entre a câmera e o ambiente, colocando os planos das imagens de saída em orientações genéricas com relação ao referencial de câmera original.

O resultado de considerar a orientação estimada durante a retificação é que agora em cada imagem sintetizada as linhas estarão orientadas ou na direção vertical, ou horizontal, ou direcionadas para o centro de projeção da imagem. É isto que se pode observar na Figura ???. No caso da imagem da *Direção z* podemos observar agora que não só as lâmpadas estão orientadas de acordo com as direções horizontal e vertical da imagem, como possuem aproximadamente o mesmo tamanho, e estão espaçadas de forma regular. Na imagem da *Direção x* é possível notar como os armários e outros objetos da sala estão representados de maneira bastante regular, enquanto na imagem original da Figura 5.12 seus formatos estão bastante distorcidos. As portas e janelas nas imagens *Direção y* e *Direção -x* também aparecem de forma regular nestas imagens, enquanto possuem distorções fortes na imagem original, e também aparecem rotacionadas e deformadas pela distorção de perspectiva nas imagens da Figura 5.13.

Ainda é possível observar imperfeições nestas imagens sintetizadas, incluindo uma curvatura nas linhas horizontais da imagem *Direção -x* da Figura 5.14. Para obter resultados ainda melhores seria necessário recorrer a algoritmos de calibração mais poderosos. Apesar das imperfeições perceptíveis, este experimento demonstra que o *Corisco* é efetivamente capaz de analisar imagens com este tipo de projeção, e demonstra também como a orientação estimada pelo *Corisco* pode ser utilizada para criar imagens com características interessantes para a realização de tarefas de mais alto nível de abstração, como o reconhecimento de objetos ou estimação da posição da câmera no espaço.

## 5.6 Discussões finais

Este capítulo relatou diversos experimentos realizados com o *Corisco* para demonstrar sua efetividade e eficiência. Os experimentos relatados nas Seções 5.1, 5.2 e 5.4 permitem avaliar o desempenho do *Corisco* na tarefa de estimação de orientação, incluindo comparações numéricas entre os resultados obtidos pelo *Corisco* e parâmetros de referência obtidos com métodos alternativos. Nestes experimentos foram utilizadas os modelos de câmera *pinhole*, o modelo de Harris de distorção radial, e a projeção equiretangular. A Seção 5.5 demonstrou por fim o funcionamento do *Corisco* com imagens de projeção polar equidistante, e demonstrou também uma utilização da orientação estimada para a pro-

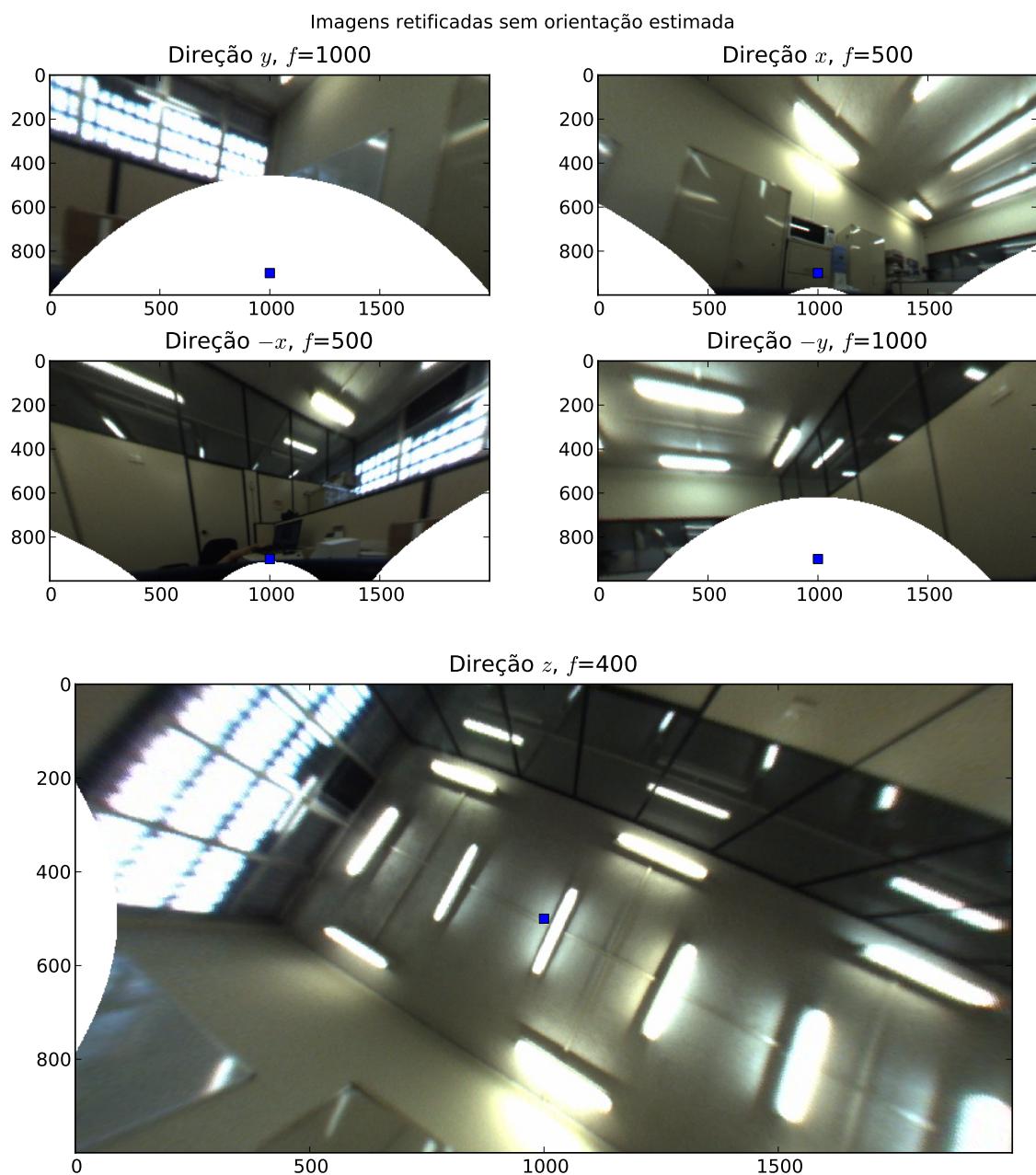


Figura 5.13: Imagens retificadas sem estimativa de orientação. Estas imagens simplesmente utilizam o modelo de câmera para produzir imagens com projeção perspectiva a partir da imagem da Figura 5.12. Os eixos a que estão alinhados cada plano destes exemplos são os próprios eixos do referencial de câmera da imagem original.

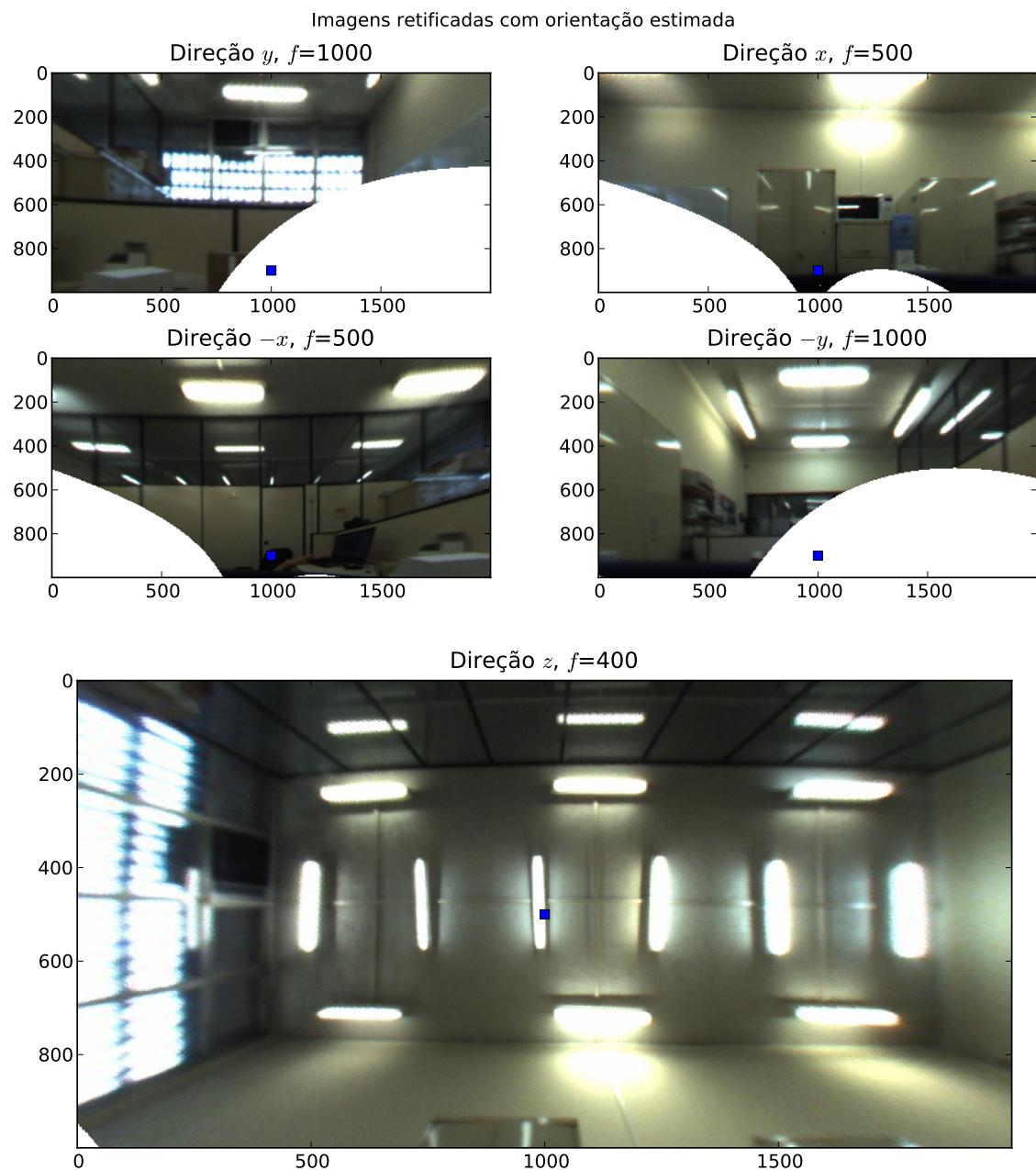


Figura 5.14: Imagens retificadas com estimação de orientação. Assim como na Figura 5.13 a imagem da Figura 5.12 foi utilizada como fonte para produzir iamgens retificadas, com projeção perspectiva, porém desta vez os planos de cada imagem de saída estão alinhados às direções do referencial natural do ambiente, estimadas pelo Corisco.

dução de imagens retificadas que podem ser aproveitadas para a realização de diferentes tarefas.

A Seção 5.3 demonstrou o potencial do *Corisco* para utilização em um processo de estimativa de distância focal, que é uma característica do *Corisco* que é compartilhada por métodos similares existentes. Esta forma de estimativa de parâmetros intrínsecos apresentada não oferece vantagens notáveis se comparada a métodos alternativos, porém existe uma importância teórica no fato de ser possível conduzir este processo da forma apresentada. No futuro, pode ser possível que algum outro processo de calibração baseado no *Corisco* apresente um desempenho mais relevante.

Os valores de referência utilizados para avaliar as estimativas obtidas com o *Corisco* possuem fontes de naturezas variadas. Em todos os casos o *Corisco* apresentou resultados de boa precisão, alcançando uma faixa de erros em torno de 1° quando foram selecionados parâmetros de controle para grande precisão. A possibilidade de explorar comprometimentos entre precisão e velocidade no *Corisco* também foi demonstrada através de testes com diferentes combinações de parâmetros de controle.

Ao todo, foi possível determinar que o *Corisco* se comporta de forma comparável, senão superior, a métodos similares existentes para a estimativa de orientação a partir de edgels. Estes experimentos demonstraram ainda a capacidade do *Corisco* de trabalhar com qualquer modelo de câmera, sendo que foram realizados testes com as projeções polar equidistante, equiretangular, e com a projeção de perspectiva pura, além do modelo de Harris para distorções radiais com parâmetro de distorção positivo e negativo. As projeções polar equidistante e equiretangular, assim como outras projeções onidirecionais, são de especial interesse para áreas como a Robótica Móvel, onde são utilizados equipamentos especializados para produzir este tipo de imagem, e os experimentos demonstraram que o *Corisco* é um método que pode ser útil no desenvolvimento de sistemas dentro destas áreas.



## 6 Conclusão

*Conclude. Reform. Renew.  
Return, conquer once again.*  
—Mercury, Voivod

Esta tese apresentou o *Corisco*, um método para a estimativa de orientação de câmera em um ambiente antrópico. Apesar de depender da existência de retas de direções conhecidas no ambiente, o método não realiza a extração de linhas ou de curvas durante a análise das imagens, como é comum em técnicas alternativas para resolver este problema. Ao invés disto o *Corisco* baseia suas análises na extração de edgels da imagem de entrada. A estimativa de orientação através de edgels é uma técnica que foi desenvolvida por vários pesquisadores ao longo da última década, destacando-se os trabalhos de Coughlan e Yuille (1999, 2003), Deutscher, Isard e MacCormick (2002), Schindler, Krishnamurthy e Dellaert (2006) e Denis, Elder e Estrada (2008).

Comparado aos métodos anteriores baseados em edgels, o *Corisco* demonstrou pela primeira vez como esta técnica pode ser aplicada para analisar imagens com qualquer tipo de projeção. O *Corisco* também demonstrou uma adaptação da técnica para apresentar um melhor funcionamento em aplicações interativas, como em robôs móveis. Isto resulta de duas características importantes do processo desenvolvido, que são a possibilidade de realizar comprometimentos entre precisão e velocidade de computação, e a realização de cálculos através de expressões que possuem um custo computacional reduzido.

O comprometimento entre precisão e velocidade é realizado através de dois mecanismos. O primeiro se localiza na extração de edgels, onde uma máscara em forma de grade define a região da imagem que será analisada desde o princípio do processo. O segundo mecanismo é a definição do número de iterações utilizadas no primeiro passo do processo de otimização que efetivamente estima os parâmetros buscados, baseado no algoritmo RANSAC.

O processo de otimização que é utilizado no *Corisco* calcula uma função objetivo que é definida através da técnica de M-estimação, desenvolvida dentro da área de estudos da Estatística Robusta. Esta função objetivo se baseia em uma função de erro redescendente, mais especificamente a função biquadrada de Tukey. A escolha desta técnica para o desenvolvimento do *Corisco* representa uma abordagem de-cima-para-baixo ao problema, enquanto nos métodos existentes o uso de outras técnicas de estimativa resultam em processos de características semelhantes, porém através de abordagens de-baixo-para-cima. Deve ser notado ainda que a função objetivo do *Corisco* utiliza um processo não-iterativo de classificação dos dados, o que não é comum em aplicações de M-estimação, mas pode ser interpretado através da teoria de estimativa EM como sendo uma forma de Expectation-

Maximization generalizado (GEM).

O uso da M-estimação é o primeiro fator que contribui para simplificação do processo de estimação e das expressões utilizadas nos cálculos. Uma outra simplificação se deve ao fato de que o erro de predição calculado pelo *Corisco* é uma simples multiplicação vetorial entre pares de vetores bidimensionais que idealmente seriam ortogonais. O conjunto de vetores observados são as direções ortogonais dos edgels extraídos da imagem, e as direções preditas são calculadas a partir da hipótese de orientação sendo testada utilizando o modelo de câmera. Métodos alternativos utilizam funções trigonométricas no cálculo deste erro, porém o uso da multiplicação vetorial, associada à normalização necessária de todos os vetores envolvidos, constitui uma alternativa mais atraente com relação à complexidade computacional.

Além de evitar o cálculo de ângulos sobre a imagem, optou-se no *Corisco* pela utilização de quaternions normalizados para representar as orientações de câmera, o que se contrapõe ao uso de técnicas baseadas nos ângulos de Euler, por exemplo. Estas escolhas resultam em expressões relativamente simples e de baixo custo computacional para a função objetivo e para as suas derivadas em relação aos parâmetros de orientação. E uma segunda vantagem desta forma de calcular o erro é a facilidade com que o método pode ser adaptado para lidar com qualquer tipo de projeção, já que o cálculo das direções preditas é realizado de uma forma unificada, através de matrizes Jacobianas.

A única dificuldade associada a esta função objetivo proposta para o *Corisco* é a necessidade de tomar cuidado durante o processo de otimização para que a norma do quaternion que modela a orientação seja unitária. Isto requer a aplicação de um algoritmo de otimização com restrições, que são algoritmos mais especializados e menos disponíveis do que os algoritmos de otimização contínua sem restrições que são utilizados pelos métodos existentes. O algoritmo FilterSQP foi selecionado para uso no *Corisco* por este motivo, e apesar a dificuldade apresentada por sua implementação durante esta pesquisa, os resultados obtidos foram bastante satisfatórios. Em outras palavras, o bom funcionamento do FilterSQP junto da função objetivo proposta para o *Corisco* fez valer a pena esta escolha pouco convencional de utilizar otimização com restrições.

## 6.1 Principais contribuições alcançadas com o Corisco

Podemos destacar as seguintes contribuições resultantes da pesquisa realizada para o desenvolvimento do *Corisco*:

- Demonstração de uma forma de extração de edgels que permite sub-amostrar os dados desde o princípio da análise da imagem. Este procedimento de extração de edgels é bastante atraente do ponto de vista do desempenho computacional, e é constitui uma forma mais natural de amostrar curvas e retas de uma imagem do que a utilização de ladrilhamento. Este procedimento de extração de edgels possui potencial para ser utilizado por outros métodos além do *Corisco*.

- Eliminação de variáveis angulares e de funções trigonométricas das expressões utilizadas no processo de estimativa de parâmetros. Isto foi realizado pelo uso da multiplicação vetorial no cálculo do erro de predição de cada edgel extraído da imagem, e também pela utilização de quaternions para representar a orientação da câmera. Isto resulta em uma simplificação das expressões utilizadas durante o processo de estimativa, o que leva a um melhor desempenho computacional e também permite utilizar o *Corisco* com qualquer modelo de câmera possível, bastante apenas implementar procedimentos modulares para o cálculo de matrizes de coeficientes utilizadas durante os cálculos. Um revés desta parametrização é a necessidade de se utilizar um algoritmo de otimização com restrições para controlar a norma do quaternion, o que foi resolvido no *Corisco* pelo uso do algoritmo FilterSQP, que funcionou de forma bastante satisfatória.
- Demonstração da aplicação de M-estimação, com uma função de erro redescendente, em substituição das técnicas de estimativa MAP ou EM. Estas técnicas de estimativa possuem muitos aspectos semelhantes, e a possibilidade desta substituição não é algo incomum nas pesquisas em outros problemas. A M-estimação permitiu que o desenvolvimento do *Corisco* fosse mais focado em detalhes da sua implementação final, facilitando investigações para a melhoria do desempenho do processo, e evitando o excesso de atenção a características do problema que não são necessariamente relevantes para a qualidade final das estimativas produzidas.
- Utilização do algoritmo RANSAC como um primeiro passo do processo de otimização. Isto faz com que seja possível utilizar o *Corisco* sem qualquer restrição ou informação inicial com relação às soluções buscadas. Esta característica torna o *Corisco* uma ferramenta adequada para utilização imediata em aplicações tal como da Robótica Móvel, onde as limitações deste tipo existentes em métodos alternativos podem ser bastante inconvenientes.

O desenvolvimento do *Corisco* visou concretizar o potencial da técnica de estimativa de orientação através de edgels, produzindo um método que ofereça vantagens substanciais na utilização como ferramenta básica em sistemas mais complexos, especialmente em aplicações de Robótica Móvel e na análise de grandes conjuntos de imagens de ambientes antrópicos. Um dos principais atrativos do *Corisco* para estas aplicações é a possibilidade de poder lidar com qualquer modelo de câmera, incluindo modelos de lentes com distorções radiais, lentes olho-de-peixe e a projeção equiretangular. Outro atrativo é a possibilidade de analisar imagens sem qualquer tipo de estimativa inicial, o que pode ser bastante relevante na prática. O *Corisco* não apenas possui estas características como se baseia em processos de estimativa de parâmetros e de análise de imagens que dependem de cálculos relativamente simples. Além disso os mecanismos para o comprometimento entre velocidade e precisão podem ser muito úteis em algumas aplicações, e o nível máximo

de precisão que pode ser alcançado também pode ser bastante satisfatório, considerando que se trata de um método que não utiliza extração de curvas ou linhas retas.

## 6.2 Trabalhos futuros

Antes de mais nada seria interessante estudar o resultado da aplicação do *Corisco* em diversas aplicações possíveis que dependem diretamente da estimativa de orientação. Algumas possibilidades seriam o guiamento de robôs móveis em ambiente antrópicos a partir da orientação estimada pelo corisco, ou ainda a obtenção de estimativas iniciais de orientação para problemas de *Bundle Adjustment* com grandes conjuntos de imagens.

Também seria interessante explorar a análise de imagens utilizada no *Corisco* para realizar tarefas como a extração de linhas através do agrupamento dos edgels, como demonstrado na Seção 4.6.3, ou ainda a extração de objetos maiores como paralelepípedos, ou até mesmo o reconhecimento de objetos. Também seria interessante estudar a aplicação do *Corisco* dentro de um processo maiores de SLAM visual baseado em edgels. A aplicação do *Corisco* para a estimativa de parâmetros intrínsecos de câmera também é uma possibilidade, e deve ser possível criar um método interessante que utiliza o *Coriscopara* obter estimativas iniciais tanto da orientação da câmera nas imagens analisadas neste processo, quanto dos próprios parâmetros do modelo de câmera.

Algumas modificações no funcionamento do *Corisco* que também poderiam ser estudadas são a realização da otimização de uma forma progressiva, com uma variação controlada da quantidade de dados que é considerada ao longo do processo de estimativa. Este tipo de abordagem visaria encontrar a forma mais eficiente possível de realizar a estimativa. Com relação à análise das imagens, uma possibilidade interessante de modificação seria a utilização de filtros direcionais, que substituiriam a extração de edgels e o cálculo de erro do *Corisco*. Nesta nova forma de análise seriam também selecionados um subconjunto de pontos da imagem onde é realizada a análise, porém o erro de predição dependeria do valor de saída de um filtro direcional calculado para a direção predita em cada orientação hipotética testada.

## 6.3 Considerações finais

Em um ponto de vista mais amplo, esta tese representa antes de mais nada um bom exemplo do poder das técnicas de Reconhecimento de Padrões baseadas na Teoria de Probabilidades. Mas mais do que isto, o *Corisco* demonstra como é possível partir de métodos que seguem estritamente princípios de funcionamento tal como a maximização de verossimilhança, e desenvolver métodos que são mais atraentes do ponto de vista do desempenho computacional. No *Corisco* este comprometimento ocorreu com a aplicação da técnica de M-estimação, retirada do campo de estudos da Estatística Robusta, e durante o desenvolvimento do método foi claro a todo o momento de que maneira foram alteradas as equações que seriam obtidas pelas formas mais tradicionais de estimativa para produzir

um processo que funciona de forma mais conveniente.

O desenvolvimento do *Corisco* visou produzir um método que possa ser efetivamente aproveitado por outros pesquisadores e desenvolvedores de sistemas baseados em visão. Mas além disto espera-se que esta pesquisa possa incentivar o estudo de métodos de Reconhecimento de Padrões baseados em Teoria de Probabilidades em geral, e também especificamente na teoria de Estatística Robusta, que oferece ferramentas valiosas para combater problemas pouco triviais que surgem na prática, fora do laboratório, e que não são o foco principal das teorias mais fundamentais de estimação. Além disto, espera-se também que esta pesquisa estimule o estudo mais amplo de técnicas da área de Otimização, que são ferramentas indispensáveis para o desenvolvimento de processos mais sofisticados de estimação. A otimização com restrições em especial é uma área que recebe relativamente pouca atenção de pesquisadores de Reconhecimento de Padrões, mas o *Corisco* demonstra como este tipo de algoritmo pode ser uma alternativa interessante para a criação de novos métodos. O algoritmo FilterSQP apresentou um desempenho excelente durante esta pesquisa, e o *Corisco* pode servir como um exemplo de sua aplicabilidade em problemas deste tipo.

O *Corisco* representa portanto um exemplo de método de estimação que utiliza Estatística Robusta aliada a algoritmos relativamente modernos de otimização comum e com restrições, mas além disto o *Corisco* também é um exemplo de um método de visão baseado em edgels. Esta é uma forma de análise de imagens relativamente pouco explorada, mas com um grande potencial para a criação de métodos com características interessantes, como é o caso *Corisco*. Unindo portanto estas três técnicas — a M-estimação, a otimização por RANSAC associada à otimização contínua com restrições, e a análise de imagens baseada em edgels — o *Corisco* mostra que apesar de ainda haver grandes desafios a ser enfrentados na área de Visão Computacional, mais fortes são os poderes das técnicas de Reconhecimento de Padrões contemporâneas.



## REFERÊNCIAS

- AIDER, O. A.; HOPPENOT, P.; COLLE, E. A model-based method for indoor mobile robot localization using monocular vision and straight-line correspondences. **Robotics and Autonomous Systems**, v. 52, n. 2-3, p. 229–246, ago. 2005. ISSN 09218890. Disponível em: <<http://dx.doi.org/10.1016/j.robot.2005.03.002>>.
- AMARAL, F. R.; COSTA, A. H. R. Classificação de objetos em imagens onidirecionais com uso de retificação de imagens e de múltiplos núcleos em máquinas de vetor de suporte. In: **III Workshop on Computational Intelligence (WCI 2010)**. [S.l.: s.n.], 2010.
- ANDO, S. Consistent gradient operators. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 3, p. 252–265, mar. 2000. ISSN 01628828. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=841757](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=841757)>.
- ANTONE, M.; TELLER, S. Scalable extrinsic calibration of omni-directional image networks. **International Journal of Computer Vision**, Springer, v. 49, n. 2, p. 143–174, 2002. Disponível em: <<http://www.springerlink.com/index/u3ql77211633p861.pdf>>.
- ANTONE, M. E.; TELLER, S. Automatic recovery of relative camera rotations for urban scenes. In: **Computer Vision and Pattern Recognition**. [s.n.], 2000. p. 1–8. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=854809](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=854809)>.
- ARRAS, K.; TOMATIS, N. Improving robustness and precision in mobile robot localization by using laser range finding and monocular vision. In: **1999 Third European Workshop on Advanced Mobile Robots (Eurobot'99). Proceedings (Cat. No.99EX355)**. IEEE, 1999. p. 177–185. ISBN 0-7803-5672-1. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=827638>>.
- AUDIBERT, J.-Y.; PONCE, J. Vanishing point detection for road detection. **2009 IEEE Conference on Computer Vision and Pattern Recognition**, Ieee, n. 3, p. 96–103, jun. 2009. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5206787>>.
- BARNARD, S. T. Interpreting perspective images. **Artificial Intelligence**, v. 21, n. 4, p. 435–462, nov. 1983. ISSN 00043702. Disponível em: <[http://dx.doi.org/10.1016/S0004-3702\(83\)80021-6](http://dx.doi.org/10.1016/S0004-3702(83)80021-6)>.
- BARRA, R. J. G.; RIBEIRO, C. H.; COSTA, A. H. R. Fast vertical line correspondence between images for mobile robot localization. In: **9th International IFAC Symposium on Robot Control (SYROCO2009)**. Gifu, Japão: [s.n.], 2009. v. 9, n. 1, p. 153—158. ISBN 978-3-902661-60-9. ISSN 1474-6670. Disponível em: <<http://www.ifac-papersonline.net/Detailed/45535.html>>.
- BARRETO, J. P.; ARAUJO, H. Geometric properties of central catadioptric line images and their application in calibration. **IEEE transactions on pattern analysis and machine intelligence**, v. 27, n. 8, p. 1327–33, ago. 2005. ISSN 0162-8828. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1453519>>.
- BLAKE, A.; ISARD, M. **Active Contours**. London: Springer London, 1998. ISBN 978-1-4471-1557-1. Disponível em: <<http://www.springerlink.com/index/10.1007/978-1-4471-1555-7>>.

- BOSSE, M.; RIKOSKI, R.; LEONARD, J.; TELLER, S. Vanishing points and 3D lines from omnidirectional video. In: **Image Processing 2002 Proceedings 2002 International Conference on**. Ieee, 2002. v. 3, p. 513–516. ISBN 0780376226. ISSN 15224880. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1039020>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1039020).
- BYROD, M.; ASTROM, K. Bundle adjustment using conjugate gradients with multiscale preconditioning. In: **British Machine Vision Conference**. [S.l.: s.n.], 2009.
- CANNY, J. A computational approach to edge detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, n. 6, p. 679–698, 1986. Disponível em:  
[<http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=4767851>](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4767851).
- CANNY, J. F. **Finding Edges and Lines in Images**. [S.l.: s.n.], 1983.
- CANTONI, V.; LOMBARDI, L.; PORTA, M.; SICARD, N. Vanishing point detection: representation analysis and new approaches. In: **International Conference on Image Analysis and Processing**. [S.l.: s.n.], 2001.
- CAPRILE, B.; TORRE, V. Using vanishing points for camera calibration. **International Journal of Computer Vision**, Springer Netherlands, v. 4, n. 2, p. 127–139, mar. 1990. ISSN 0920-5691. Disponível em:  
<http://www.springerlink.com/content/k75077108473tm15/>.
- CERIANI, S.; FONTANA, G.; GIUSTI, A.; MARZORATI, D.; MATTEUCCI, M.; MIGLIORE, D.; RIZZI, D.; SORRENTI, D. G.; TADDEI, P. Rawseeds ground truth collection systems for indoor self-localization and mapping. **Autonomous Robots**, Springer Netherlands, v. 27, n. 4, p. 353–371, set. 2009. ISSN 0929-5593. Disponível em: <http://www.springerlink.com/content/k924032g72818h53/>.
- CHAO, L.; YALOU, H.; YEWEI, K.; JING, Y. Monocular SLAM using vertical straight lines with inverse-depth representation. In: **2008 7th World Congress on Intelligent Control and Automation**. IEEE, 2008. p. 3015–3020. ISBN 978-1-4244-2113-8. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4593403>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4593403).
- CHOI, S.; KIM, T.; YU, W. Performance evaluation of RANSAC family. In: **Proceedings of the British Machine Vision Conference**. [s.n.], 2009. v. 24, n. 3. Disponível em:  
[<http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf>](http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf).
- CHUM, O.; MATAS, J.; OBDRZALEK, S. Enhancing RANSAC by generalized model optimization. In: **Proc. of the Asian Conference on Computer Vision**. [s.n.], 2004. v. 2, p. 812–817. Disponível em:  
[<ftp://cmp.felk.cvut.cz/pub/cmp/articles/chum/chum-accv04.pdf>](ftp://cmp.felk.cvut.cz/pub/cmp/articles/chum/chum-accv04.pdf).
- CIPOLLA, R.; DRUMMOND, T.; ROBERTSON, D. Camera calibration from vanishing points in images of architectural scenes. In: **Proc. of the British Machine Vision Conference**. [S.l.: s.n.], 1999. v. 2, p. 382–391.
- COLLINS, R.; WEISS, R. Vanishing point calculation as a statistical inference on the unit sphere. In: **Proceedings of the Third International Conference on Computer Vision**. IEEE Comput. Soc. Press, 1990. p. 400–403. ISBN 0-8186-2057-9. Disponível em:  
[<http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=139560>](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=139560).

- COUGHLAN, J. M.; YUILLE, A. L. Manhattan world: compass direction from a single image by Bayesian inference. In: **Proceedings International Conference on Computer Vision ICCV'99**. IEEE, 1999. p. 941–947 vol.2. ISBN 0-7695-0164-8. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=790349>> [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=790349](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=790349)>.
- COUGHLAN, J. M.; YUILLE, A. L. Manhattan world: orientation and outlier detection by Bayesian inference. **Neural Computation**, MIT Press 238 Main St., Suite 500, Cambridge, MA 02142-1046 USA journals-info@mit.edu, v. 15, n. 5, p. 1063—1088, mar. 2003. Disponível em: <<http://www.mitpressjournals.org/doi/abs/10.1162/089976603765202668>>.
- DAO, V. N.; SUGIMOTO, M. A robust recognition technique for dense checkerboard patterns. In: **2010 20th International Conference on Pattern Recognition**. IEEE, 2010. p. 3081–3084. ISBN 978-1-4244-7542-1. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=5597291](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5597291)>.
- DAVISON, A. J.; REID, I. D.; MOLTON, N. D.; STASSE, O. MonoSLAM: real-time single camera SLAM. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 29, n. 6, p. 1052–67, jun. 2007. ISSN 0162-8828. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=4160954](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4160954)>.
- DENIS, P.; ELDER, J. H.; ESTRADA, F. J. Efficient edge-based methods for estimating Manhattan frames in urban imagery. In: **Proceedings of the 10th European Conference on Computer Vision**. [S.I.]: Springer-Verlag, 2008. p. 197–210.
- DENIS, P. H. **Efficient Edge-Based Methods for Estimating Manhattan Frames in Urban Imagery**. Tese (Doutorado) — York University, 2008.
- DERICHE, R. Using Canny's criteria to derive a recursively implemented optimal edge detector. **International Journal of Computer Vision**, v. 187, p. 167–187, 1987. Disponível em: <<http://www.springerlink.com/index/x82465262072776g.pdf>>.
- DEUTSCHER, J.; ISARD, M.; MACCORMICK, J. Automatic camera calibration from a single Manhattan image. In: HEYDEN, A.; SPARR, G.; NIELSEN, M.; JOHANSEN, P. (Ed.). **European Conference on Computer Vision**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. (Lecture Notes in Computer Science, v. 2353), p. 373–377. ISBN 978-3-540-43748-2. Disponível em: <<http://www.springerlink.com/content/pxm2t0e8y3t65q4m/>>.
- DUDA, R. O. **Some Current Techniques for Scene Analysis**. Menlo Park, USA, 1970.
- EADE, E.; DRUMMOND, T. Scalable monocular SLAM. In: **2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)**. IEEE, 2006. v. 1, p. 469–476. ISBN 0-7695-2597-0. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1640794>>.
- ELDER, J.; ZUCKER, S. Local scale control for edge detection and blur estimation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 20, n. 7, p. 699–716, jul. 1998. ISSN 01628828. Disponível em: <<http://portal.acm.org/citation.cfm?id=283567.283575>>.

- FLECK, M. **Perspective Projection: The Wrong Imaging Model.** [S.I.], 1995.
- FLETCHER, R. **Practical Methods of Optimization.** Wiley, 2000. 450 p. ISBN 0471494631. Disponível em:  
[<http://www.amazon.com/Practical-Methods-Optimization-R-Fletcher/dp/0471494631>](http://www.amazon.com/Practical-Methods-Optimization-R-Fletcher/dp/0471494631).
- FLETCHER, R.; LEYFFER, S. Nonlinear programming without a penalty function. **Mathematical Programming**, Springer Berlin / Heidelberg, v. 91, n. 2, p. 239–269, jan. 2002. ISSN 0025-5610. Disponível em:  
 [<http://www.springerlink.com/content/qqj37x00y79ygdl8/>](http://www.springerlink.com/content/qqj37x00y79ygdl8/).
- FLINT, A.; MEI, C.; REID, I.; MURRAY, D. Growing semantically meaningful models for visual SLAM. In: **2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.** IEEE, 2010. p. 467–474. ISBN 978-1-4244-6984-0. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5540176>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5540176).
- FORSYTH, D. A.; PONCE, J. **Computer Vision: A Modern Approach.** [S.I.]: Prentice Hall, 2003.
- FURUKAWA, Y.; PONCE, J. Accurate camera calibration from multi-view stereo and bundle adjustment. In: **2008 IEEE Conference on Computer Vision and Pattern Recognition.** IEEE, 2008. p. 1–8. ISBN 978-1-4244-2242-5. Disponível em:  
[<http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=4587681>](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4587681).
- FURUKAWA, Y.; PONCE, J. Accurate, dense, and robust multiview stereopsis. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 32, n. 8, p. 1362–76, ago. 2010. ISSN 1939-3539. Disponível em:  
[<http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=5226635>](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5226635).
- GEE, A.; CHEKHLOV, D.; CALWAY, A.; MAYOL-CUEVAS, W. Discovering higher level structure in visual SLAM. **IEEE Transactions on Robotics**, v. 24, n. 5, p. 980–990, out. 2008. ISSN 1552-3098. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4648452>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4648452).
- GEE, A.; MAYOL-CUEVAS, W. Real-time model-based SLAM using line segments. In: BEBIS, G.; BOYLE, R.; PARVIN, B.; KORACIN, D.; REMAGNINO, P.; NEFIAN, A.; MEENAKSHISUNDARAM, G.; PASCUCCI, V.; ZARA, J.; MOLINEROS, J.; THEISEL, H.; MALZBENDER, T. (Ed.). **International Symposium on Visual Computing.** Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. (Lecture Notes in Computer Science, v. 4292), p. 354–363. ISBN 978-3-540-48626-8. Disponível em:  
 [<http://www.springerlink.com/content/98462n7585016664/>](http://www.springerlink.com/content/98462n7585016664/).
- GHOSAL, S.; MEHROTRA, R. Orthogonal moment operators for subpixel edge detection. **Pattern Recognition**, v. 26, n. 2, p. 295–306, fev. 1993. ISSN 00313203. Disponível em:  
[<http://dx.doi.org/10.1016/0031-3203\(93\)90038-X>](http://dx.doi.org/10.1016/0031-3203(93)90038-X).
- GONZALEZ-AGUILERA, D.; GOMEZ-LAHOZ, J. From 2D TO 3D Through Modelling Based On A Single Image. **The Photogrammetric Record**, v. 23, n. 122, p. 208–227, jun. 2008. ISSN 0031868X. Disponível em:  
[<http://doi.wiley.com/10.1111/j.1477-9730.2008.00482.x>](http://doi.wiley.com/10.1111/j.1477-9730.2008.00482.x).

- GONZALEZ-AGUILERA, D.; GOMEZ-LAHOZ, J.; RODRÍGUEZ-GONZÁLVEZ, P. An automatic approach for radial lens distortion correction from a single image. **Sensors Journal, IEEE**, IEEE, v. 11, n. 4, p. 956–965, 2011. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5585662](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5585662)>.
- GRAMMATIKOPOULOS, L.; KARRAS, G.; PETSA, E. An automatic approach for camera calibration from vanishing points. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 62, n. 1, p. 64–76, maio 2007. ISSN 09242716. Disponível em: <<http://dx.doi.org/10.1016/j.isprsjprs.2007.02.002>>.
- HARRIS, C.; STEPHENS, M. A combined corner and edge detector. In: PLESSEY RESEARCH ROKE MANOR. **British Machine Vision Conference**. [S.I.]: Plessey Research Roke Manor, 1988. p. 147–152.
- HART, P. How the Hough transform was invented. **IEEE Signal Processing Magazine**, v. 26, n. 6, p. 18–22, nov. 2009. ISSN 1053-5888. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=5230799](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5230799)>.
- HARTLEY, R.; ZISSELMAN, A. **Multiple View Geometry in Computer Vision**. 2nd. ed. Cambridge University Press, 2003. 655 p. ISSN 05215405. ISBN 978-0-521-54051-3. Disponível em: <<http://www.robots.ox.ac.uk/vgg/hzbook/>>.
- HILE, H.; BORRIELLO, G. Positioning and orientation in indoor environments using camera phones. **IEEE Computer Graphics and Applications**, v. 28, n. 4, p. 32–39, jul. 2008. ISSN 0272-1716. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4557953>>.
- HORN, B. K. P. **Tsai's Camera Calibration Method Revisited**. [S.I.], 2000. i, n. 2000. Disponível em: <[http://people.csail.mit.edu/bkph/articles/Tsai\\_Revisited.pdf](http://people.csail.mit.edu/bkph/articles/Tsai_Revisited.pdf)>.
- HOWARD, A.; KITCHEN, L. Fast visual mapping for mobile robot navigation. In: **1997 IEEE International Conference on Intelligent Processing Systems (Cat. No.97TH8335)**. IEEE, 1997. v. 2, p. 1251–1255. ISBN 0-7803-4253-4. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=669197>>.
- HUGHES, C.; MCFEELY, R.; DENNY, P.; GLAVIN, M.; JONES, E. Equidistant fish-eye perspective with application in distortion centre estimation. **Image and Vision Computing**, Elsevier B.V., v. 28, n. 3, p. 538–551, mar. 2010. ISSN 02628856. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0262885609001863>>.
- ILLINGWORTH, J.; KITTNER, J. A survey of the Hough transform. **Computer Vision, Graphics, and Image Processing**, v. 44, n. 1, p. 87–116, out. 1988. ISSN 0734189X. Disponível em: <[http://dx.doi.org/10.1016/S0734-189X\(88\)80033-1](http://dx.doi.org/10.1016/S0734-189X(88)80033-1)>.
- INC., W. R. **Mathematica 8.0**. [S.I.]: Wolfram Research Inc., 2011.
- JEONG, Y.; NISTER, D.; STEEDLY, D. Pushing the envelope of modern methods for bundle adjustment. In: **Computer Vision and Pattern Recognition**. [s.n.], 2010. p. 1474–1481. ISBN 9781424469857. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5539795](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5539795)>.

JOSEFSSON, T.; NORDH, E.; ERIKSSON, P.-O. A flexible high-precision video system for digital recording of motor acts through lightweight reflex markers. **Computer Methods and Programs in Biomedicine**, v. 49, n. 2, p. 119–129, mar. 1996. ISSN 01692607. Disponível em: <[http://dx.doi.org/10.1016/0169-2607\(96\)01715-4](http://dx.doi.org/10.1016/0169-2607(96)01715-4)>.

KAESS, M.; RANGANATHAN, A. iSAM: Incremental smoothing and mapping. **IEEE Transactions on Robotics**, v. 24, n. 6, p. 1365–1378, 2008. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=4682731](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4682731)>.

KIM, S.; OH, S.-y. SLAM in indoor environments using omni-directional vertical and horizontal line features. **Journal of Intelligent and Robotic Systems**, v. 51, p. 31–43, 2008.

KLEIN, G.; MURRAY, D. Parallel tracking and mapping for small AR workspaces. In: **2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality**. IEEE, 2007. p. 1–10. ISBN 978-1-4244-1749-0. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=4538852](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4538852)>.

KLINE, M. **Mathematics for the Nonmathematician**. Dover Publications, 1985. 641 p. ISBN 0486248232. Disponível em: <<http://www.amazon.com/Mathematics-Nonmathematician-Dover-explaining-science/dp/0486248232>>.

KOGAN, H.; MAURER, R.; KESHET, R. Vanishing points estimation by self-similarity. In: **Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2009. p. 755–761. ISBN 9781424439911.

KRIEGMAN, D.; TRIENDL, E.; BINFORD, T. Stereo vision and navigation in buildings for mobile robots. **IEEE Transactions on Robotics and Automation**, v. 5, n. 6, p. 792–803, 1989. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=88100](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=88100)>.

KROTKOV, E. Mobile robot localization using a single image. In: **Proceedings, 1989 International Conference on Robotics and Automation**. IEEE Comput. Soc. Press, 1989. p. 978–983. ISBN 0-8186-1938-4. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=100108](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=100108)>.

LEBEGUE, X.; AGGARWAL, J. Significant line segments for an indoor mobile robot. **IEEE Transactions on Robotics and Automation**, v. 9, n. 6, p. 801–815, 1993. ISSN 1042296X. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=265923>>.

LENSEIR, S.; VELOSO, M. Visual sonar: fast obstacle avoidance using monocular vision. In: **Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)**. IEEE, 2003. v. 1, p. 886–891. ISBN 0-7803-7860-1. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=1250741](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1250741)>.

LENZ, R.; TSAI, R. Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 10, n. 5, p. 713–720, 1988. ISSN 01628828. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=6781](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6781)>.

- LOURAKIS, M.; ARGYROS, A. Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment? In: **Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1**. IEEE, 2005. p. 1526–1531 Vol. 2. ISBN 0-7695-2334-X. Disponível em: <[http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1544898&contentType=Conference\\_Paper](http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1544898&contentType=Conference_Paper)>.
- LUTTON, E.; MAHE, H.; LOPEZ-KRAHE, J. Contribution to the determination of vanishing points using Hough transform. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 16, n. 4, p. 430–438, 1994.
- LYVERS, E.; MITCHELL, O.; AKEY, M.; REEVES, A. Subpixel measurements using a moment-based edge operator. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 11, n. 12, p. 1293–1309, 1989. ISSN 01628828. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=41367](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=41367)>.
- MAIMONE, M.; LEGER, P. Overview of the mars exploration rovers' autonomous mobility and vision capabilities. In: **IEEE International Conference on Robotics and Automation (ICRA) Space Robotics Workshop**. [s.n.], 2007. Disponível em: <[http://www-robotics.jpl.nasa.gov/publications/Mark\\_Maimone/mer\\_autonomy\\_icra\\_2007.pdf](http://www-robotics.jpl.nasa.gov/publications/Mark_Maimone/mer_autonomy_icra_2007.pdf)>.
- MARONNA, R. A.; MARTIN, D. R.; YOHAI, V. J. **Robust Statistics: Theory and Methods (Wiley Series in Probability and Statistics)**. Wiley, 2006. 436 p. ISBN 0470010924. Disponível em: <<http://www.amazon.com/Robust-Statistics-Theory-Methods-Probability/dp/0470010924>>.
- MATAS, J.; GALAMBOS, C.; KITTNER, J. Robust detection of lines using the progressive probabilistic Hough transform. **Computer Vision and Image Understanding**, v. 78, n. 1, p. 119–137, abr. 2000. ISSN 10773142. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S1077314299908317>>.
- MCLACHLAN, G. J.; KRISHNAN, T. **The EM Algorithm and Extensions**. [S.l.]: John Wiley & Sons, 2007. ISBN 978-0-471-20170-0.
- MCLEAN, G.; KOTTURI, D. Vanishing point detection by line clustering. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 17, n. 11, p. 1090–1095, 1995. ISSN 01628828.
- MEI, C.; SIBLEY, G.; CUMMINS, M.; NEWMAN, P.; REID, I. A Constant-Time Efficient Stereo SLAM System. **Proceedings of the British Machine Vision Conference 2009**, British Machine Vision Association, p. 54.1–54.11, 2009. Disponível em: <<http://www.bmva.org/bmvc/2009/Papers/Paper056/Paper056.html>>.
- MIKOLAJCZYK, K.; SCHMID, C. Performance evaluation of local descriptors. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 27, n. 10, p. 1615–30, out. 2005. ISSN 0162-8828. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=1498756](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1498756)>.
- MORAVEC, H. **Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover**. Tese (Doutorado) — Stanford University, 1980. Disponível em: <[http://www.ri.cmu.edu/publication\\_view.html?pub\\_id=22](http://www.ri.cmu.edu/publication_view.html?pub_id=22)>.

MOURAGNON, E.; LHUILLIER, M.; DHOME, M.; DEKEYSER, F.; SAYD, P. Real-time localization and 3D reconstruction. In: **2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)**. IEEE, 2006. p. 363–370. ISBN 0769525970. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1640781>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1640781).

NAKAMURA, T.; UEDA, M. Matching method of vertical edge patterns for locating a mobile robot. **Transactions of the Society of Instrument and Control Engineers**, v. 18, n. 6, p. 576–582, 1982.

NEWCOMBE, R. A.; LOVEGROVE, S. J.; DAVISON, A. J. DTAM: Dense tracking and mapping in real-time. In: **2011 International Conference on Computer Vision**. IEEE, 2011. p. 2320–2327. ISBN 978-1-4577-1102-2. Disponível em:  
[<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126513>](http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126513).

NIETO, M.; SALGADO, L. Real-time vanishing point estimation in road sequences using adaptive steerable filter banks. In: **ACIVS'07 Proceedings of the 9th international conference on Advanced concepts for intelligent vision systems**. [S.l.: s.n.], 2007. p. 840–848.

NILSSON, N. J. **Shakey the Robot**. SRI International, 1984. Disponível em:  
[<http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA458918>](http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA458918).

OHYA, I.; KOSAKA, A.; KAK, A. Vision-based navigation by a mobile robot with obstacle avoidance using single-camera vision and ultrasonic sensing. **IEEE Transactions on Robotics and Automation**, v. 14, n. 6, p. 969–978, 1998. ISSN 1042296X. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=736780>>.

OLSON, C. F. A general method for geometric feature matching and model extraction. **International Journal of Computer Vision**, Springer Netherlands, v. 45, n. 1, p. 39–54, out. 2001. ISSN 0920-5691. Disponível em:  
[<http://www.springerlink.com/content/h410214304031u38/>](http://www.springerlink.com/content/h410214304031u38/).

PAPARI, G.; PETKOV, N. Edge and line oriented contour detection: state of the art. **Image and Vision Computing**, Elsevier B.V., v. 29, n. 2-3, p. 79–103, fev. 2011. ISSN 02628856. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0262885610001253>>.

PERONA, P. Deformable kernels for early vision. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 17, n. 5, p. 488–499, 1995. Disponível em:  
[<http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=391394>](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=391394).

QUAN, L.; MOHR, R. Determining perspective structures using hierarchical Hough transform. **Pattern Recognition Letters**, v. 9, n. May, p. 279–286, 1989.

ROSTEN, E.; LOVELAND, R. Camera distortion self-calibration using the plumb-line constraint and minimal Hough entropy. **Machine Vision and Applications**, Springer Berlin / Heidelberg, v. 22, n. 1, p. 77–85, abr. 2009. ISSN 0932-8092. Disponível em:  
[<http://www.springerlink.com/content/dl72369q405n8327/>](http://www.springerlink.com/content/dl72369q405n8327/).

ROTHER, C. A new approach to vanishing point detection in architectural environments. **Image and Vision Computing**, v. 20, n. 9-10, p. 647–655, 2002. Disponível em:  
[<http://dx.doi.org/10.1016/S0262-8856\(02\)00054-9>](http://dx.doi.org/10.1016/S0262-8856(02)00054-9).

- SACHT, L. K.; CARVALHO, P. C.; VELHO, L.; CASTRINA, E. D.; GATTASS, M. Face and straight line detection in equirectangular images. In: **VI Workshop de Visão Computacional (WVC)**. [S.l.: s.n.], 2010. p. 101–106.
- SANTANA, A. M.; MEDEIROS, A. A. Monocular-SLAM using floor lines. In: **2010 Latin American Robotics Symposium and Intelligent Robotics Meeting**. IEEE, 2010. p. 97–102. ISBN 978-1-4244-8639-7. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5702188>>.
- SCATOLINI, C. D. A.; WERNECK, N. L.; TRUZZI, F. S.; COSTA, A. H. R. FishRet: calibração e retificação de imagens onidirecionais. In: **Congresso Brasileiro de Automática (CBA)**. [S.l.: s.n.], 2012.
- SCHINDLER, G.; DELLAERT, F. Atlanta world: an expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In: **Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004**. IEEE, 2004. p. 203–209. ISBN 0-7695-2158-4. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=1315033](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1315033)>.
- SCHINDLER, G.; KRISHNAMURTHY, P.; DELLAERT, F. Line-based structure from motion for urban environments. In: **Third International Symposium on 3D Data Processing, Visualization, and Transmission**. [s.n.], 2006. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=4155810](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4155810)>.
- SE, S.; LOWE, D. Vision-based mobile robot localization and mapping using scale-invariant features. In: **2001 IEEE International Conference on Robotics and Automation**. [s.n.], 2001. Disponível em: <[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=932909](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=932909)>.
- SHI, J.; TOMASI, C. Good features to track. In: **Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94**. IEEE Comput. Soc. Press, 1994. p. 593–600. ISBN 0-8186-5825-8. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=323794](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=323794)>.
- SIMOND, N.; RIVES, P. Homography from a vanishing point in urban scenes. In: **Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on**. [S.l.: s.n.], 2003. p. 1005–1010.
- SINGER, S.; NELDER, J. **Nelder-Mead algorithm**. 2009. Disponível em: <[http://www.scholarpedia.org/article/Nelder-Mead\\_algorithm](http://www.scholarpedia.org/article/Nelder-Mead_algorithm)>.
- SINHA, S.; STEEDLY, D.; SZELISKI, R. A multi-stage linear approach to structure from motion. In: **ECCV Workshop on Reconstruction and Modeling of Large Scale 3D Virtual Environments**. [s.n.], 2010. v. 51, p. 1–14. Disponível em: <<http://research.microsoft.com/pubs/138039/Sinha-RMLE10.pdf>>.
- SMITH, P.; REID, I.; DAVISON, A. Real-time monocular SLAM with straight lines. In: **British Machine Vision Conference**. [s.n.], 2006. v. 1. ISSN 1-904410-14-6. Disponível em: <<http://www.citeulike.org/user/serebryakov/article/4828883>>.

- SMITH, R. An overview of the tesseract OCR engine. In: **Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2.** IEEE, 2007. p. 629–633. ISBN 0-7695-2822-8. ISSN 1520-5363. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=4376991](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4376991)>.
- SNAVELY, N.; SEITZ, S. M.; SZELISKI, R. Photo tourism: exploring photo collections in 3D. In: **ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006).** [s.n.], 2006. p. 835–846. Disponível em: <<http://dl.acm.org/citation.cfm?id=1141964>>.
- SNAVELY, N.; SEITZ, S. M.; SZELISKI, R. Modeling the world from Internet photo collections. **International Journal of Computer Vision**, Springer Netherlands, v. 80, n. 2, p. 189–210, dez. 2007. ISSN 0920-5691. Disponível em: <<http://www.springerlink.com/content/55m8034t78051qt8/>>.
- STEPHENS, R. Probabilistic approach to the Hough transform. **Image and Vision Computing**, v. 9, n. 1, p. 66–71, fev. 1991. ISSN 02628856. Disponível em: <[http://dx.doi.org/10.1016/0262-8856\(91\)90051-P](http://dx.doi.org/10.1016/0262-8856(91)90051-P)>.
- STRAND, R.; HAYMAN, E. Correcting radial distortion by circle fitting. In: **British Machine Vision Conference.** British Machine Vision Association, 2005. ISBN 1-901725-29-4. Disponível em: <<http://www.bmva.org/bmvc/2005/papers/paper-138.html> [http://www-dsp.elet.polimi.it/VA-TLC/Elaborati/Correzione di distorsioni radiali tramite fitting di circonference/bmvc\\_final\\_138.pdf](http://www-dsp.elet.polimi.it/VA-TLC/Elaborati/Correzione di distorsioni radiali tramite fitting di circonference/bmvc_final_138.pdf)>.
- STRASDAT, H.; MONTIEL, J. M. M.; DAVISON, A. J. Real-time monocular SLAM: Why filter? In: **2010 IEEE International Conference on Robotics and Automation.** IEEE, 2010. p. 2657–2664. ISBN 978-1-4244-5038-1. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=5509636](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5509636)>.
- SUGIHARA, K. Some location problems for robot navigation using a single camera. **Computer Vision, Graphics, and Image Processing**, v. 42, n. 1, p. 112–129, abr. 1988. ISSN 0734189X. Disponível em: <[http://dx.doi.org/10.1016/0734-189X\(88\)90145-4](http://dx.doi.org/10.1016/0734-189X(88)90145-4)>.
- SUN, W.; YANG, X.; XIAO, S.; HU, W. Robust checkerboard recognition for efficient nonplanar geometry registration in projector-camera systems. In: **Proceedings of the 5th ACM/IEEE International Workshop on Projector camera systems - PROCAMS '08.** New York, New York, USA: ACM Press, 2008. p. 1. ISBN 9781605582726. Disponível em: <<http://dl.acm.org/citation.cfm?id=1394622.1394625>>.
- SWART, D.; TORRENCE, B. The viewable sphere. **Math Horizons**, n. September, p. 14–17, 2011.
- TARDIF, J.-P. Non-iterative approach for fast and accurate vanishing point detection. In: **IEEE 12th International Conference on Computer Vision (ICCV).** [S.l.: s.n.], 2009. ISBN 9781424444199.
- THORMAHLEN, T.; BROSZIO, H.; WASSERMANN, I. Robust line-based calibration of lens distortion from a single view. In: **Proceedings of Mirage.** [S.l.: s.n.], 2003.

- THRUN, S.; BENNEWITZ, M.; BURGARD, W.; CREMERS, A.; DELLAERT, F.; FOX, D.; HAHNEL, D.; ROSENBERG, C.; ROY, N.; SCHULTE, J.; SCHULZ, D. MINERVA: A second-generation museum tour-guide robot. In: **Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)**. IEEE, 1999. v. 3, p. 1999–2005. ISBN 0-7803-5180-0. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=770401>>.
- TOLDO, R.; FUSIELLO, A. Robust multiple structures estimation with J-linkage. In: **European Conference on Computer Vision**. Springer Berlin / Heidelberg, 2008. v. 5302, p. 537–547. ISBN 978-3-540-88681-5. Disponível em: <<http://www.springerlink.com/content/f0310906x67884qv/>>.
- TORDOFF, B.; MURRAY, D. The impact of radial distortion on the self-calibration of rotating cameras. **Computer Vision and Image Understanding**, Elsevier, v. 96, n. 1, p. 17–34, out. 2004. ISSN 10773142. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1077314204000906>>.
- TORDOFF, B. J.; MURRAY, D. W. Guided-MLESAC: Faster image transform estimation by using matching priors. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 27, n. 10, p. 1523–35, out. 2005. ISSN 0162-8828. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/16237989>>.
- TORII, A.; HAVLENA, M.; PAJDLA, T. From Google Street View to 3D city models. In: **2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops**. IEEE, 2009. p. 2188–2195. ISBN 978-1-4244-4442-7. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5457551>>.
- TORR, P. MLESAC: A new robust estimator with application to estimating image geometry. **Computer Vision and Image Understanding**, v. 78, n. 1, p. 138–156, abr. 2000. ISSN 10773142. Disponível em: <<http://dx.doi.org/10.1006/cviu.1999.0832>>.
- TRIGGS, B.; MCLAUCHLAN, P.; HARTLEY, R.; FITZGIBBON, A. W. Bundle adjustment — A modern synthesis. In: **ICCV '99 Proceedings of the International Workshop on Vision Algorithms: Theory and Practice**. Springer-Verlag, 2000. v. 34099, p. 298–372. Disponível em: <<http://www.springerlink.com/index/PLVCRQ5BX753A2TN.pdf>>.
- TRUCCO, E.; VERRI, A. **Introductory Techniques for 3-D Computer Vision**. [S.l.]: Prentice-Hall, 1998. ISBN 9780132611084.
- TUYTELAARS, T.; MIKOLAJCZYK, K. Local invariant feature detectors: A survey. **Foundations and Trends in Computer Graphics and Vision**, v. 3, n. 3, p. 177–280, jan. 2007. ISSN 1572-2740. Disponível em: <<http://dl.acm.org/citation.cfm?id=1391081.1391082>>.
- TUYTELAARS, T.; PROESMANS, M.; Van Gool, L.; MI, E. The cascaded Hough transform. In: **International Conference on Image Processing**. [S.l.: s.n.], 1998. v. 2, p. 736–739.
- ULRICH, I.; NOURBAKHSH, I. R. Appearance-based obstacle detection with monocular color vision. In: **Proceedings of the AAAI National Conference on Artificial Intelligence**. [s.n.], 2000. p. 866–871. ISBN 0-262-51112-6. Disponível em: <<http://dl.acm.org/citation.cfm?id=647288.721755>>.

WERNECK, N. L.; COSTA, A. H. R. Monocular visual mapping with the Fast Hough Transform. In: **Anais do VI Workshop de Visão Computacional**. [s.n.], 2010. Disponível em: <[http://iris.sel.eesc.usp.br/wvc/anais\\_WVC2010/artigos/oral/72827.pdf](http://iris.sel.eesc.usp.br/wvc/anais_WVC2010/artigos/oral/72827.pdf)>.

WERNECK, N. L.; COSTA, A. H. R. Mapping with monocular vision in two dimensions. **International Journal of Natural Computing Research**, v. 1, n. 4, 2011.

WERNECK, N. L.; COSTA, A. H. R. Speeding up probabilistic inference of camera orientation by function approximation and grid masking. In: BARANOSKI, G.; SKALA, V. (Ed.). **WSCG 2011 Communication Papers**. Plzen, Czech Republic: UNION Agency, 2011. p. 127–134. Disponível em: <<http://nwerneck.sdf.org/almoxarifado/nic-wscg2011.pdf>>.

WERNECK, N. L.; TRUZZI, F. S.; COSTA, A. H. R. Medição de distância e altura de bordas horizontais com visão monocular linear para robôs móveis. In: **Anais do V Workshop de Visão Computacional**. São Paulo: [s.n.], 2009. Disponível em: <[http://www.lti.pcs.usp.br/nwerneck/almoxarifado/59760\\_WerneckTruzziCosta.pdf](http://www.lti.pcs.usp.br/nwerneck/almoxarifado/59760_WerneckTruzziCosta.pdf)>.

WIKIPEDIA. **Expectation–maximization algorithm**. 2012. Acessado em: 03/07/2012.

WONGPHATI, M.; NIPARNAN, N.; SUDSANG, A. Bearing only FastSLAM using vertical line information from an omnidirectional camera. In: **Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on**. [s.n.], 2008. p. 1188–1193. Disponível em: <[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=4913169](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4913169)>.

ZHANG, Z. A flexible new technique for camera calibration. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 22, n. 11, p. 1330–1334, 2000. ISSN 01628828. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=888718>>.

ZIOU, D.; TABBONE, S. Edge detection techniques - An overview. **International Journal of Pattern Recognition and Image Analysis**, v. 8, n. 4, p. 1–41, 1998. ISSN 10546618.