Visão Computacional a partir de Imagens Estereoscópicas

Marcellus Rosa Macêdo

Visão Computacional a partir de Imagens Estereoscópicas

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida pelo Sr. Marcellus Rosa Macêdo e aprovada pela Comissão Julgadora.

Campinas, 06 de Outubro de 1995.

Jorge Stolfi
Orientador

Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Visão Computacional a partir de Imagens Estereoscópicas[1]

Marcellus Rosa Macêdo[2]

Departamento de Ciência da Computação
IMECC – UNICAMP

Banca Examinadora:

- Jorge Stolfi(Orientador)[3]
- Sdnei de Brito Alves[4]
- Neucimar Jeronimo Leite[3]
- Pedro Jussieu de Rezende (Suplente)[3]

---

[1]Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação da UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.
[2]O autor é Engenheiro Eletricista pela Escola Federal de Engenharia de Itajubá
[3]Professor do Departamento de Ciência da Computação – IMECC – UNICAMP.
[4]Professor da Escola Federal de Engenharia de Itajubá –EFEI

Dedicatória

Dedico este trabalho ao meu pai e à minha mãe pela confiança que em mim depositaram.

Agradecimentos

Ao Prof. Jorge Stolfi pelo estímulo, orientação, colaboração e, sobretudo, paciência com que me acompanhou durante o desenvolvimento deste trabalho.

Aos colegas de Mestrado pelo apoio e amizade, em particular ao amigo Sérgio Lage pelo grande incentivo que sempre chegou nas horas certas e a todos aqueles que, direta ou indiretamente me ajudaram.

Aos meus Pais a quem devo, de maneira particular, este momento.

Apoio financeiro

CAPES;

FAPESP - Processo 93/4079-5.

Resumo

Neste trabalho descrevemos a implementação de um protótipo de software para reconhecimento e localização espacial de um objeto poliédrico a partir de um par de imagens estereoscópicas do mesmo. Nesta implementação usamos uma abordagem original, no sentido que a segmentação de pontos em arestas é feita no espaço $\mathbf{R}^3$, *depois* de casar os pontos correspondentes nas duas imagens e calcular a profundidade dos mesmos.

Abstract

We describe here the implementation of a software prototype for the identification and location of a polyedral object in three-space, given two stereoscopic images of the same. An original feature of this implementation is that the segmentation of edges is performed in three-space, *after* matching corresponding pairs of points in the two images and computing their depth.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

O problema básico da área de *visão computacional* é extrair informações úteis de imagens de modo automático e eficaz. As aplicações de visão computacional são inúmeras e crescentes, incluindo por exemplo, controle de robôs industriais, inspeção automática, controle de veículos autônomos, sensoriamento remoto, engenharia reversa, etc.

Em muitas aplicações industriais de visão computacional, imagens bidimensionais são suficientes. Entretanto, em várias aplicações, é necessário conhecer a estrutura tridimensional da cena analisada incluindo profundidade e orientação das superfícies.

A habilidade humana de perceber profundidade é ao mesmo tempo trivial e enigmática. Percebemos relações espaciais em três dimensões sem nenhum esforço, mas os meios pelos quais o fazemos são, em grande parte, desconhecidos.

Atualmente, existem vários métodos para obter profundidade e orientação de uma cena. Entre eles estão o sistema detector de profundidade a laser (baseado na medida do intervalo de tempo entre a emissão e a recepção da luz refletida na superfície do objeto), estereoscopia fotométrica (que estima a forma do objeto a partir da distribuição de luz e sombra sobre a sua superfície), métodos de iluminação controlada, e a estereoscopia multiocular (que se baseia na comparação de várias imagens simultâneas obtidas de posições diferentes) [20].

De modo geral métodos que dependem de fonte de iluminação controlada são caros, lentos, e nem sempre podem ser usados. Por exemplo, métodos à laser são potencialmente perigosos e apresentam problemas com superfícies brilhantes refletivas. Técnicas fotométricas exigem conhecimento detalhado da iluminação da cena, e das propriedades refletivas dos objetos, e fornecem apenas informações grosseiras sobre a forma dos mesmos. Em contraste com as outras técnicas, a estereoscopia multiocular tem baixo custo e apresenta poucas restrições quanto a iluminação e natureza dos objetos.

1.1 Visão estereoscópica binocular

No método de estereoscopia multiocular, várias imagens da mesma cena são amostradas a partir de diferentes pontos de observação. A informação sobre a terceira dimensão é deduzida a partir das diferenças entre essas imagens. Tipicamente empregam-se duas imagens obtidas de pontos relativamente próximos entre si (comparado com as dimensões da cena). Desta forma as imagens obtidas assemelham-se às vistas proporcionadas pelos dois olhos de um observador humano.

1.1.1 Modelo geométrico de uma câmera

De modo geral o processo de aquisição de uma imagem pode ser descrito pelo seguinte modelo: a posição da câmera é representada por um ponto F do espaço, o *ponto focal*. A imagem a ser adquirida é modelada por um retângulo I, contido em um *plano de projeção* π situado em frente à câmera. O sistema ótico da câmera projeta cada ponto P da cena no ponto P' da imagem, determinado pelo encontro da reta PF com o plano π. Veja a figura 1.1.

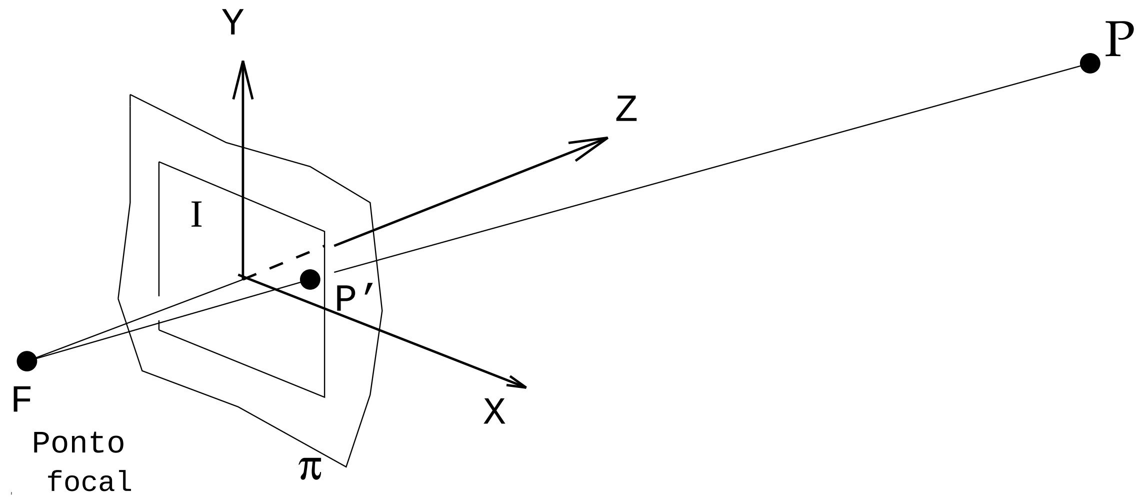

Figura 1.1: Modelo de uma câmera geométrica

Na verdade, em câmeras reais o retângulo I (que corresponde fisicamente ao filme ou sensor da câmera) fica atrás do ponto F. Entretanto a única conseqüência deste detalhe é que a imagem projetada no sensor está invertida em relação ao modelo. Esta inversão é trivialmente corrigida quando a imagem é armazenada.

1.1.2 Geometria da visão estereoscópica binocular

Imagine duas câmeras em diferentes posições como as da figura 1.2.

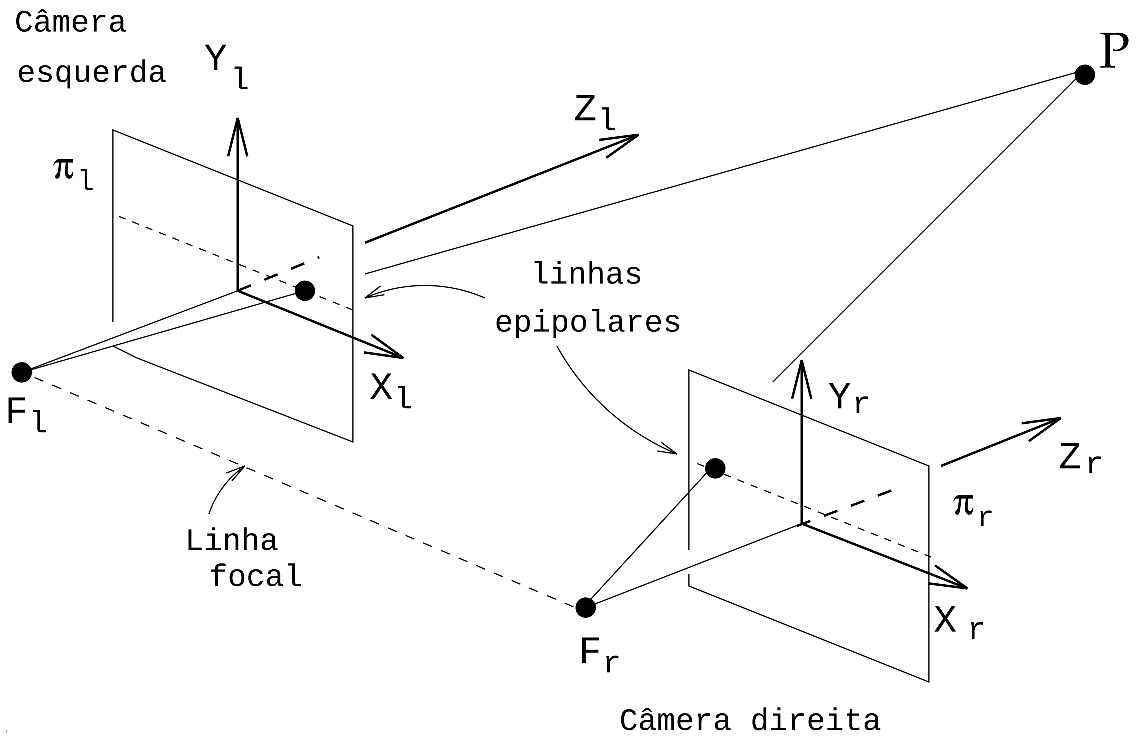

Figura 1.2: Modelo de duas câmeras deslocadas entre si para obtenção de imagens estereoscópicas

Sejam F_l e F_r os pontos focais das duas câmeras e π_l e π_r os respectivos planos de projeção. Um plano que contém a linha imaginária entre os dois pontos focais (*linha focal*) é chamado de *plano epipolar*. As retas em que o mesmo plano epipolar intercepta os planos π_l e π_r são as *linhas epipolares correspondentes*. A importância das linhas epipolares é que as duas imagens de um ponto P do espaço pertencem às duas linhas epipolares correspondentes. Como veremos adiante, isto permite reduzir o problema bidimensional de casar detalhes correspondentes nas duas imagens, ao problema unidimensional de casar tais detalhes ao longo de um par de linhas epipolares correspondentes.

1.2 Visão estereoscópica computacional

Os passos envolvidos no processo estereoscópico são basicamente os seguintes [4]:

1. Aquisição das imagens;
2. Calibração das câmeras;
3. Determinação e casamento de detalhes correspondentes;
4. Determinação de profundidade;
5. Interpolação e segmentação;
6. Comparação com modelo abstrato.

Estas etapas não são efetuadas necessariamente na ordem acima.

1.2.1 Aquisição das imagens

A maneira óbvia para aquisição das duas imagens é obtê-las simultaneamente usando câmeras montadas lado a lado numa mesma plataforma, como mostrado anteriormente (figura 1.2). As imagens também podem ser obtidas por uma única câmera, deslocando-a de posição para obter as imagens. Este método tem a desvantagem de que as duas imagens não são obtidas simultaneamente e, conseqüentemente só pode ser utilizado em ambientes estáticos. Se a cena se modificar durante o deslocamento da câmera, a informação de disparidade dos pontos correspondentes não se relacionará com a profundidade dos pontos em três dimensões.

O artigo de Gruver e Goshtasby [6] enumera as principais propriedades que um sistema de aquisição de imagens estereoscópicas deve possuir.

1.2.2 Calibração das câmeras e retificação das imagens

Para que seja possível o cálculo da profundidade a partir de imagens estereoscópicas, é necessário se conhecer os parâmetros da câmera, isto é, as posições de F, π, I no espaço. Estes parâmetros nem sempre são conhecidos com a precisão necessária. Portanto é necessário calcular estes parâmetros a partir das próprias imagens (ou de imagens especiais para calibração).

Esta etapa de *calibração das câmeras* é crucial, especialmente em aplicações que envolvem medidas quantitativas como engenharia reversa e controle de robôs.

As câmeras também não são perfeitas e podem apresentar uma variedade de aberrações provocadas por defeitos intrínsecos da câmera (ex. defeito das lentes) ou por defeitos de montagem (ex. alinhamento). Nestes casos, a etapa de calibração deve incluir o ajuste da imagem de modo a cancelar os efeitos destas deformações.

A partir do conhecimento dos parâmetros das câmeras é possível determinar os pares de linhas epipolares correspondentes, o que é essencial para o casamento dos detalhes correspondentes nas duas imagens. Um raciocínio geométrico elementar permite concluir que as imagens sempre podem ser retificadas de modo que pares de linhas epipolares correspondentes se tornem linhas de varredura horizontais de mesma ordenada nas imagens, facilitando a etapa de determinação de detalhes correspondentes e possibilitando o cálculo de sua posição no espaço [2].

1.2.3 Determinação e casamento de detalhes correspondentes

Esta etapa consiste em identificar *pares de pontos conjugados*: pontos projetados nas duas imagens que correspondem ao mesmo ponto na cena observada. Seria ideal se pudéssemos encontrar tais correspondências para todos os pixels das duas imagens. Infelizmente, é óbvio que apenas a informação de intensidade do nível de cinza contida em único pixel é insuficiente para permitir um casamento sem ambigüidade. Portanto, os elementos que compõem cada par conjugado devem ser coleções de pixels que apresentam certo padrão característico de variação de intensidade. Por exemplo, um salto brusco de intensidade numa imagem provavelmente corresponde a outro salto semelhante na outra, pois ambos provavelmente são imagens de uma mesma aresta do objeto.

Vários tipos de padrões são encontrados na literatura, incluindo padrões de baixo nível (como degraus de intensidade, texturas e cores dos pixels), padrões de nível intermediário (como contornos, segmentos de retas, curvas, esquinas e regiões), e padrões de alto nível (como formas estruturadas) [9, 21, 14, 25, 2, 1, 10]. Os padrões de baixo nível podem ser extraídas mais facilmente do que as de alto nível, mas são mais difíceis de se casar e vice-versa. Por exemplo, o casamento de regiões inteiras seria ideal para a recuperação de superfícies, mas só a decomposição da imagem em regiões já é um problema de grande complexidade.

1.2.4 Determinação de profundidade

Uma vez determinados os parâmetros das câmeras, a posição tridimensional de um ponto na cena, identificado e casado nas imagens estereoscópicas na fase anterior, pode ser determinada através de simples triangulação.

Para descrever este cálculo, suporemos que as câmeras estão montadas de modo que seus eixos óticos estejam paralelos e separados por uma distância b, e que a linha focal é paralela aos planos das imagens, e especificamente às linhas de varredura. Suporemos também que as projeções ortogonais dos focos nos planos das imagens têm coordenadas (x_0, y_0), e que f (a *distância focal*) é a distância do centro da lente ao plano da imagem das duas câmeras. Veja a figura 1.3.

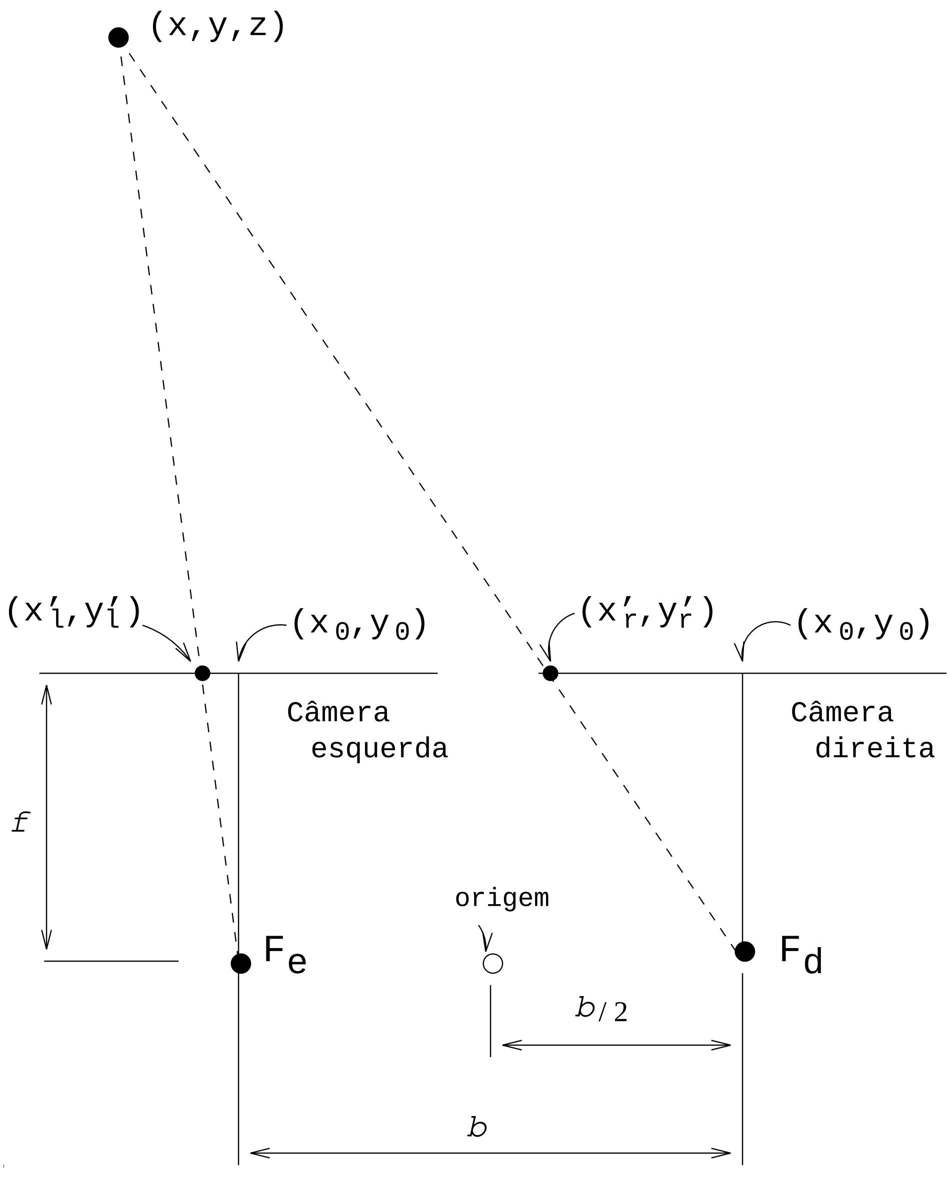

Figura 1.3: Geometria básica da visão estereoscópica

Sejam x'_l e x'_r as posições horizontais de dois pontos, um em cada imagem, que foram casados na fase anterior. Seja y' a ordenada comum aos dois. Supondo-se que os dois pontos correspondem a um mesmo ponto do objeto, as coordenadas espaciais do mesmo podem ser determinadas pelas fórmulas padrão de estereoscopia [8]. Da figura 1.3, temos:

$$\frac{x'_l - x_0}{f} = \frac{x + b/2}{z} \qquad \frac{x'_r - x_0}{f} = \frac{x - b/2}{z} \qquad \frac{y'_l - y_0}{f} = \frac{y'_r - y_0}{f} = \frac{y}{z} \tag{1.1}$$

Rearranjando obtemos as seguintes fórmulas:

$$x = \frac{b}{2}\,\frac{x'_l + x'_r - 2x_0}{x'_l - x'_r} \qquad y = b\frac{y' - y_0}{x'_l - x'_r} \qquad z = b\frac{f}{x'_l - x'_r} \tag{1.2}$$

As coordenadas (x, y, z) calculadas segundo as fórmulas 1.2 são medidas relativamente a um sistema de coordenadas com eixos paralelos aos eixos das imagens, e cuja origem é situada no ponto médio da linha focal. Veja a figura 1.3.

O conjunto de pontos no espaço tridimensional obtidos no final desta etapa é chamado de *mapa de profundidade* da cena.

1.2.5 Interpolação e segmentação

Aplicações envolvendo estereoscopismo muitas vezes necessitam de um mapa de profundidade denso. A etapa anterior devolve um mapa de profundidade esparso pelo fato dos detalhes presentes em cada imagem serem esparsos. Nesta etapa os *espaços* entre os detalhes são preenchidos através de interpolação, resultando em estruturas de nível mais alto como superfícies, segmentos, etc.

1.2.6 Comparação com modelo

Nesta etapa, o objetivo é estabelecer uma correspondência entre os pontos ou estruturas de nível mais alto obtidas na etapa anterior e o modelo particular de um objeto conhecido. Um subproduto desta identificação é a determinação de sua posição e orientação na cena. A maioria das técnicas de reconhecimento casam modelos de objetos conhecidos com objetos observados através da comparação de elementos primitivos (ex. arestas, vértices). Entretanto é extremamente improvável que os elementos extraídos das imagens sejam idênticos aos do modelo conhecido. Em geral há vários casamentos imperfeitos, alguns corretos e outros não. Portanto, o módulo de casamento deve ser capaz de avaliar numericamente a plausibilidade de cada casamento, e escolher os mais prováveis.

1.3 Resumo deste trabalho

O objetivo deste trabalho é a implementação de um programa para identificar e determinar a posição no espaço de um objeto de forma e dimensões conhecidas usando a técnica de visão estereoscópica binocular.

No protótipo que implementamos, temos como entrada duas imagens estereoscópicas previamente retificadas e um modelo matemático do objeto procurado. O objetivo é obtermos como saída a identificação e a posição tridimensional deste objeto.

Os objetos utilizados para a identificação estão restritos a poliedros: cubos, paralelepípedos, prismas, etc.

Nosso protótipo não inclui a calibração e retificação das imagens. As imagens utilizadas nos testes (capítulo 5) já estavam suficientemente calibradas. Vários artigos da bibliografia tratam extensivamente deste problema [24, 7, 11, 22].

A ordem das etapas de detecção, casamento e segmentação no nosso protótipo, difere da ordem seguida na maioria dos sistemas descritos na literatura [9, 21, 14, 25]. Nesses trabalhos, os detalhes usados para o casamento são geralmente elementos mais complexos como arestas, e sua segmentação em duas dimensões é realizada em cada imagem separadamente, antes do casamento. No nosso sistema, os detalhes são apenas descontinuidades dos níveis de cinza ao longo de cada linha de varredura. O casamento destes detalhes resulta num conjunto de pontos do $\mathbf{R}^3$. Só neste momento é que fazemos a segmentação desses pontos, agrupando-os em segmentos de retas.

Na etapa de casamento com o modelo usamos os vértices como elementos característicos para localizar e posicionar o modelo. Estes vértices utilizados são essencialmente os extremos dos segmentos encontrados na etapa anterior e modificados nesta etapa.

No capítulo 2 descrevemos a detecção de pontos de descontinuidade nas duas imagens, o casamento dos mesmos e o cálculo da posição espacial dos pontos casados. No capítulo 3, descrevemos um algoritmo para agrupar esses pontos em segmentos de reta. No capítulo 4, descrevemos um algoritmo para unir os extremos dos segmentos em vértices e um algoritmo para identificar na cena o objeto procurado a partir de comparação com modelo, utilizando como primitivas os vértices encontrados. No capítulo 5, apresentamos os resultados desses algoritmos aplicados a alguns pares de imagens simuladas e reais.

Capítulo 2

Detecção e casamento de descontinuidades

2.1 Introdução

Nesta fase identificamos em cada imagem, separadamente, certos detalhes ou primitivas que podem ser usadas para determinar os pares de pontos correspondentes nas imagens.

Para o nosso problema resolvemos utilizar como primitivas descontinuidades de intensidades de níveis de cinza. Essas primitivas correspondem a eventos físicos nas imagens como contornos de objetos ou fronteiras entre superfícies de cores distintas. A seguir mostramos como fazemos a identificação e casamento dessas primitivas.

2.2 Varredura unidimensional

A abordagem que adotamos consiste em procurar descontinuidades ao longo de cada linha epipolar, independentemente. Como observamos na introdução, estamos supondo que estas linhas coincidem com as linhas de varredura da imagem.

A idéia do algoritmo é dividir cada linha de varredura da imagem em *carreiras*, que são seqüências de pixels consecutivos cujas intensidades estão próximas entre si. Cada intervalo entre duas carreiras seguidas — que chamaremos de *degrau* — é então interpretado como a imagem de um ponto de uma aresta do objeto. A figura 2.1 ilustra a presença de carreiras e degraus ao longo de uma linha de varredura da imagem de um cubo.

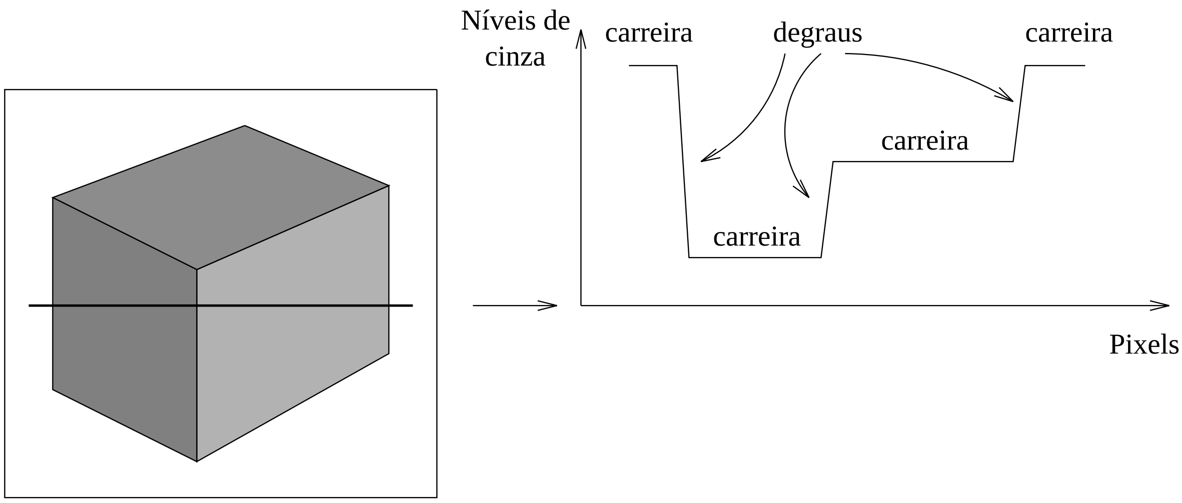

Figura 2.1: Um cubo e a variação de níveis de cinza ao longo de uma linha da imagem.

Esta abordagem é obviamente pouco adequada para detectar uma aresta cuja projeção na imagem é aproximadamente horizontal. Em muitas aplicações, este defeito seria inaceitável, e portanto teríamos que usar algoritmos de detecção mais complexos, como os de Sobel [3, 18], que examinam simultaneamente várias linhas de varredura consecutivas (veja seção 2.8).

Entretanto, no nosso caso este defeito pode ser ignorado. Acontece que as arestas horizontais das imagens são praticamente inúteis para a determinação de profundidade, pois em geral cada ponto de uma tal aresta pode ser casado com infinitos pontos da aresta correspondente na outra imagem. Apenas os extremos de tais arestas podem ser casados com confiança; mas somente quando esses extremos pertencem também a arestas não horizontais.

Esta afirmação se aplica em parte também a arestas que formam um ângulo pequeno com a horizontal: devido a erros de medida, quantização, e calibração das câmeras, cada ponto de uma tal aresta pode ser casado com um pedaço relativamente grande da aresta correspondente na outra imagem. Assim, a posição no espaço de tais arestas está sujeita a erros bastante grandes. Portanto, mesmo que o algoritmo de detecção de descontinuidades fosse sensível a arestas quase-horizontais, teríamos que descartar essas arestas antes de tentar a reconstrução estereoscópica.

Na figura 2.2 temos a imagem de um cubo e os degraus detectados pelo nosso algoritmo (descrito mais adiante). Podemos notar que as arestas que formam um ângulo pequeno com a horizontal não são totalmente detectadas, enquanto que as arestas aproximadamente verticais são totalmente detectadas.

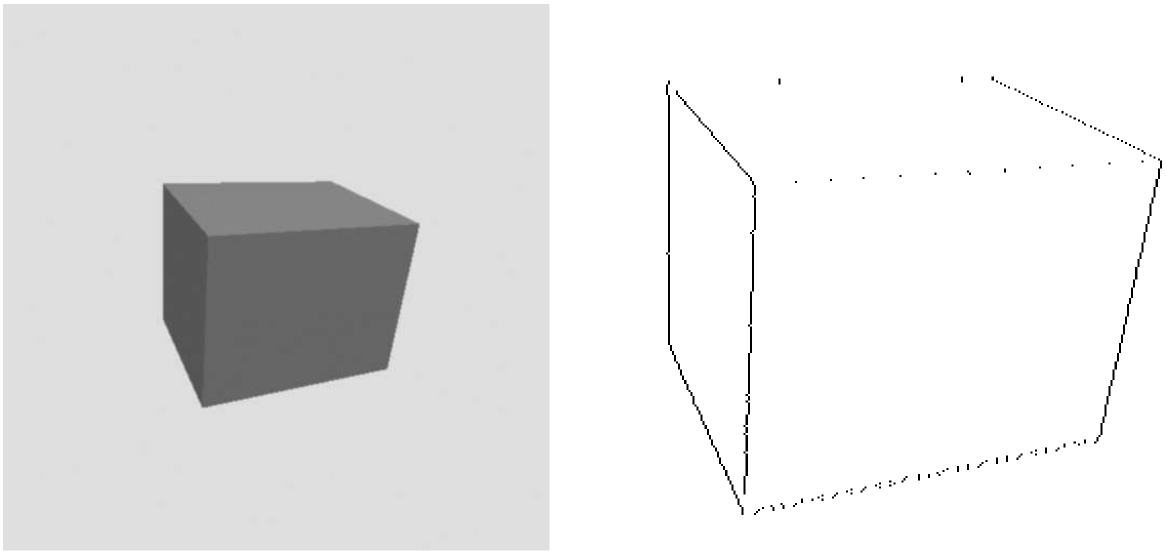

Figura 2.2: Imagem de um cubo e a respectiva detecção de descontinuidades.

É importante observar que uma aresta do objeto pode cobrir vários pixels consecutivos numa mesma linha da imagem; quer por focalização imperfeita da câmera, quer devido à inclinação da aresta. Por exemplo, se uma aresta do objeto, perfeitamente em foco, faz um ângulo de 0.1 radianos com a horizontal, ela aparecerá ao longo de cada linha como uma rampa linear de 10 pixels de largura. Veja figura 2.3. Para que o casamento funcione, é importante agrupar esses pixels, e devolver o ponto médio do grupo como sendo a posição nominal da aresta.

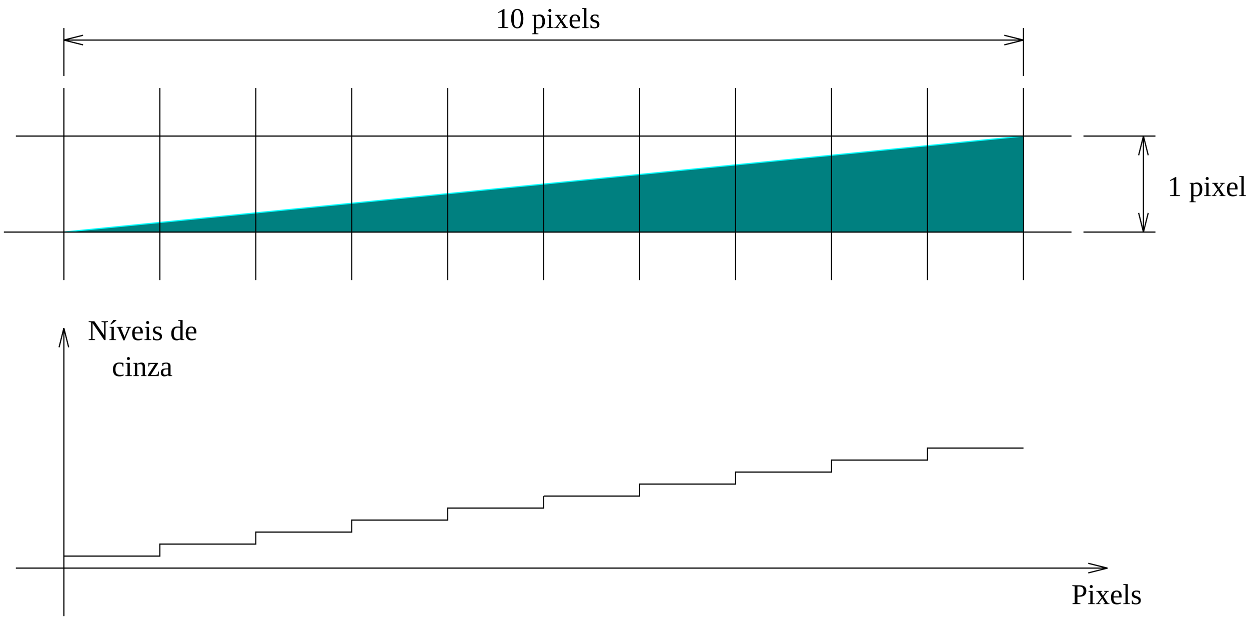

Figura 2.3: Uma aresta formando um ângulo pequeno com a horizontal e a quantização da variação de níveis de cinza em níveis discretos através da aresta.

2.3 Determinação de carreiras

O objetivo na determinação de carreiras é agrupar pixels que pertençam a uma mesma face do objeto observado.

Definimos uma carreira como sendo a seqüência maximal p_r, p_{r+1},..., p_s de pelo menos K pixels consecutivos que satisfaz

$$|p_i - p_{i+1}| \;\leq\; 2\delta \tag{2.1}$$

para todo i com $r \leq i < s$.

Os parâmetros K e δ dependem da aplicação. O valor de δ é uma estimativa da diferença máxima dos níveis de cinza entre dois pixels consecutivos de uma mesma face plana do objeto, que pode ser atribuída a ruído ou a variações de iluminação e textura na face. A função do parâmetro K é evitar que uma aresta que se estende por mais de um pixel, devido por exemplo à focalização incorreta da câmera, seja incorretamente interpretada como duas arestas separadas por uma face minúscula. Por exemplo, na figura 2.4, se $K = 4$, o patamar P não é interpretado como uma carreira por ter menos de K pixels de largura.

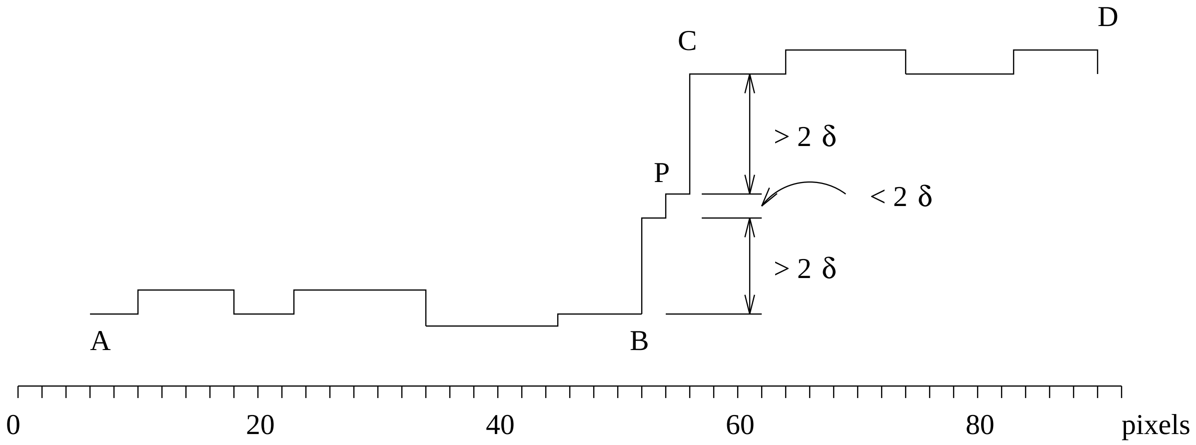

Figura 2.4: Duas carreiras ($A..B$ e $C..D$) separadas por um degrau ($B..C$).

2.3.1 Detectabilidade de uma aresta

Definimos a *amplitude* de uma aresta da imagem como sendo a diferença absoluta entre as intensidades das duas faces adjacentes à mesma.

Note que há uma relação complexa entre a amplitude Δ, o ângulo θ da aresta com a horizontal, e a detectabilidade da mesma. Supondo-se que a imagem está perfeitamente focalizada e livre de ruído, uma aresta com $\theta \geq 45°$ cruza no máximo dois pixels de cada linha de varredura. Portanto, tal aresta é sempre reconhecida quando $\Delta > 4\delta$, e nunca é reconhecida se $\Delta \leq 2\delta$. (Se Δ está entre esses dois valores, a probabilidade da aresta ser detectada varia de maneira complexa com Δ e θ; mas de qualquer maneira aumenta com os dois.)

Por outro lado, uma aresta com $\theta \leq 45°$ pode cruzar até $(1+1/\tan\theta)$ pixels da mesma linha de varredura. Portanto tal aresta é sempre detectada se $\Delta > 2\delta(1 + 1/\tan\theta)$, e nunca é detectada se $\Delta \leq 2\delta(1/\tan\theta)$.

Observe-se que qualquer borramento da imagem (por exemplo, devido a focalização imperfeita) sempre diminui a probabilidade de detecção de arestas com Δ e θ baixo.

2.3.2 Algoritmo para detecção de descontinuidades

Nosso algoritmo para detecção de carreiras começa colocando o primeiro pixel p_1 da linha de varredura numa carreira inicial, e em seguida examina os pixels $p_2, p_3, \ldots, p_n$, nessa ordem. Se a diferença $|p_i - p_{i-1}|$ for menor ou igual a 2δ, o pixel p_i é acrescentado à carreira atual; caso contrário, a carreira atual é encerrada, e p_i é considerado o início de uma nova carreira. Uma vez examinados todos os pixels, descartamos as carreiras com menos de K pixels.

Pode-se verificar que este algoritmo é simétrico, ou seja, o conjunto de carreiras resultante independe da direção em que a linha de varredura da imagem é processada.

Uma vez identificadas as carreiras, cada intervalo entre duas carreiras consecutivas é, por definição, um *degrau*. Se uma carreira termina no pixel p_r, e a seguinte começa no pixel p_s $(s > r)$, então adotamos $x' = \frac{1}{2}(r + s)$ como sendo a abscissa nominal do degrau. (A ordenada nominal do degrau é sempre a ordenada y' dos centros dos pixels da linha de varredura).

2.3.3 Definições alternativas de carreiras

Um problema da definição de carreiras acima (2.1) é que ela usa um único parâmetro δ para acomodar tanto variações pontuais devidas a ruído, quanto variações sistemáticas, devidas a mudanças graduais na iluminação ou na cor do objeto. Por exemplo, se a magnitude do ruído exige que usemos $\delta = 10$, a seqüência de pixels

$$\ldots 0, 0, 0, 20, 40, 60, 60, 60, 60 \ldots$$

será considerada uma única carreira. Entretanto essa seqüência poderia muito bem ser devida a uma aresta de inclinação moderada, ou ligeiramente fora de foco.

Numa tentativa de resolver este problema, consideramos a seguinte definição mais geral de carreira: uma seqüência maximal p_r, $p_{r+1}, \ldots, p_s$ de pelo menos K pixels consecutivos que satisfaz

$$|p_i - p_j| \;\leq\; 2\delta\,|i - j|^{\alpha} \tag{2.2}$$

para todo i, j com $r \leq i < j \leq s$, onde α é um parâmetro entre 0 e 1.

Note que para $\alpha = 1$ a condição (2.2) se reduz a condição anterior (2.1). Para $\alpha = 0$, a condição (2.2) se reduz a

$$|p_i - p_j| \;\leq\; 2\delta \tag{2.3}$$

para $r \leq i < j \leq s$.

Esta condição diz que todos os pixels que fazem parte de uma mesma carreira devem estar dentro de um intervalo de níveis de cinza de largura 2δ, ou seja, que a diferença entre o pixel de maior e o de menor valor de nível de cinza da carreira deve ser menor ou igual a 2δ.

Ao contrário da condição (2.1), a condição (2.3) tolera bastante ruído pontual sem por isso tolerar variações sistemáticas de intensidade. Por exemplo, se $\delta = 10$ a seqüência

$$\ldots 10, 20, 10, 20, 20, 10, 20, 20 \ldots$$

seria considerada uma carreira pelo critério (2.3), enquanto que

$$\ldots 0, 4, 8, 12, 16 \ldots 36 \ldots$$

não o seria.

A escolha de $\alpha = 1/2$ daria uma definição intermediária que tolera variações pontuais grandes e variações sistemáticas menores.

Uma desvantagem da definição (2.2) geral é que as carreiras não são disjuntas quando $\alpha < 1$. Por exemplo, se $\alpha = 0$ e $\delta = 10$, na seqüência de pixels

$$\ldots 0, 10, 20, 30 \ldots$$

teremos duas carreiras superpostas $(0, 10, 20)$ e $(10, 20, 30)$. Portanto, a decomposição da linha em carreiras e degraus disjuntos não é única.

Se tentarmos determinar tais carreiras por um método ganancioso análogo ao descrito na seção 2.3.2 – ou seja, acrescenta-se cada pixel à carreira corrente, enquanto a condição (2.3) for satisfeita – notaremos que o conjunto de carreiras encontrado depende da direção de processamento. Podemos visualizar este fato através do exemplo da figura 2.5. Note nessa figura que o crescimento da carreira inicial (carreira 1) termina quando é encontrado o segundo salto de 1.5δ, na direção de processamento.

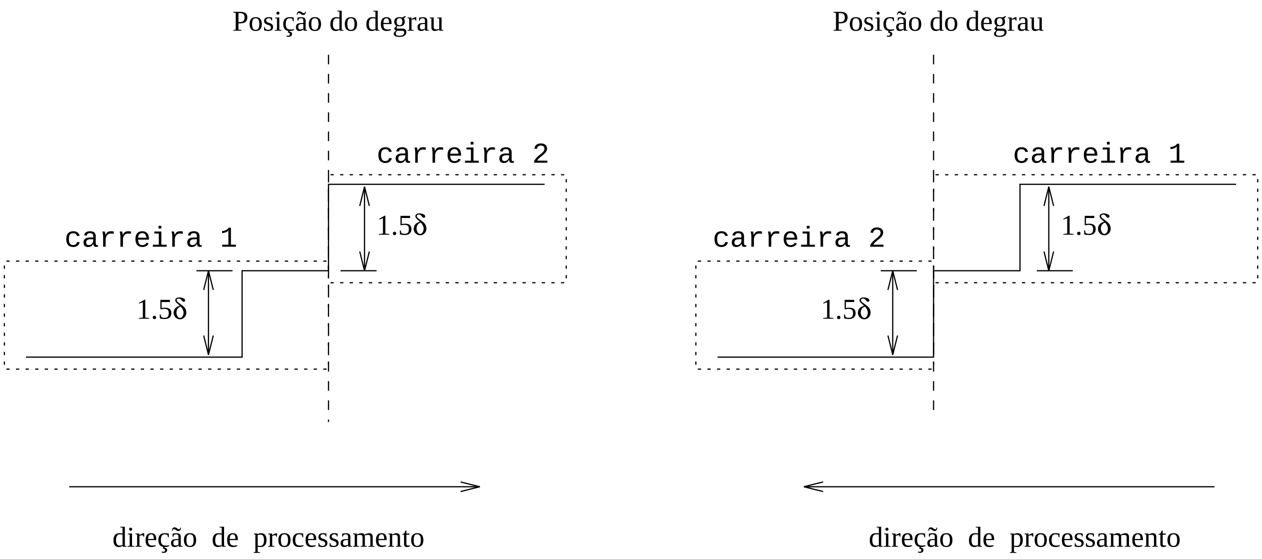

Figura 2.5: Efeito da direção de processamento na posição dos degraus.

Concluímos que esta definição mais geral, para $\alpha < 1$, introduziria erros sistemáticos na posição dos degraus, que atrapalhariam as etapas seguintes. Portanto, decidimos continuar usando a condição (2.1), apesar de suas limitações.

2.4 Seleção dos melhores degraus

Numa imagem complicada ou excessivamente ruidosa, o número de degraus encontrados por este algoritmo pode ser muito grande (proporcional ao número de pixels na linha). Para não sobrecarregar a etapa de casamento, é em geral necessário limitar o número de degraus devolvidos.

Para esse fim atribuímos a cada degrau um *índice* de *qualidade* q que tenta medir a significância do mesmo. Intuitivamente, queremos atribuir maior qualidade a degraus mais abruptos, isto é, que possuem alta taxa de variação (diferença entre níveis de intensidade, dividida pela largura do degrau).

Entretanto, devemos excluir degraus abruptos devidos à ruído; em particular degraus de largura 1 com variação de intensidade da ordem de 2δ.

Uma fórmula que parece atender a estes objetivos é

$$q = \Delta(W - w) \tag{2.4}$$

onde $\Delta = |p_r - p_s|$ é a amplitude do degrau definida na seção 2.3.1; w é a largura $s - r$ do degrau; e W é a largura máxima permitida para um degrau (degraus com $w \geq W$ são rejeitados sempre).

Dentre todos os degraus encontrados numa linha de varredura, escolhemos os D que tem maior índice de qualidade, onde D é outro parâmetro do algoritmo. De modo geral, D deve ser um pouco maior que o número máximo de arestas reais da cena que podem ocorrer ao longo de uma linha de varredura. Por exemplo, se soubermos que a cena consiste de um único cubo contra um fundo uniforme, podemos tomar $D = 5$, pois uma linha traçada sobre qualquer projeção de um cubo cruza no máximo 4 arestas visíveis do mesmo.

2.5 Casamento dos degraus

Para cada linha de varredura, a etapa anterior devolve duas listas de degraus, uma em cada imagem. Nesta etapa, procuramos formar pares de degraus, um de cada lista, que provavelmente correspondem a um mesmo ponto da cena projetado nas duas imagens.

2.5.1 Restrições geométricas

Em primeiro lugar, precisamos considerar apenas os pares de degraus – um da lista direita e um da lista esquerda, cujas posições satisfazem as restrições geométricas da visão estereoscópica binocular.

Note que as fórmulas (1.2) das coordenadas x, y, z (seção 1.2.4) pressupõem que $x_l > x_r$ ou $x_l - x_r > 0$. Caso contrário se $x_l - x_r \to 0^+$ o ponto estaria no infinito e se $x_l - x_r < 0$

o ponto estaria atrás do plano $z = 0$. Portanto só devemos considerar pares de degraus com essa propriedade.

Mais geralmente, se soubermos as distâncias mínima e máxima (z_{min} e z_{max}) entre os objetos procurados e o plano $z = 0$, devemos considerar apenas pares de degraus cujas abscissas x_l, x_r correspondem a coordenada z nesta faixa. Isto é:

$$\frac{fb}{z_{max}} \leq x_l - x_r \leq \frac{fb}{z_{min}}$$

Esta condição é conhecida na literatura como *restrição de disparidade* [2].

2.5.2 Similaridade dos degraus

Obedecida esta restrição, procuramos avaliar a probabilidade de cada par de degraus corresponder à mesma aresta real do objeto, com base nos níveis de intensidade dos pixels imediatamente anterior e posterior ao degrau.

Para avaliar esta probabilidade usamos a medida de *discrepância*

$$g \;=\; \alpha \min\{|e_1 - d_1|, |e_2 - d_2|\} + \beta \max\{|e_1 - d_1|, |e_2 - d_2|\} \tag{2.5}$$

onde (e_1, e_2) são as intensidades dos pixels imediatamente antes e depois do degrau na imagem da esquerda, e (d_1, d_2) são os mesmos valores para a imagem da direita. As constantes α e β são parâmetros do algoritmo. Valores típicos (que usamos em nossos testes) são $\alpha = 0.9$ e $\beta = 0.1$. Note que neste caso as discrepâncias variam entre 0 e a intensidade máxima do pixel (255).

A idéia por trás da fórmula (2.5) é que dois degraus são considerados semelhantes se houver semelhança entre os pixels de um mesmo lado dos dois degraus — mesmo que os pixels do outro lado sejam muito diferentes. A justificativa para este critério é que uma aresta de silhueta, vista de dois ângulos diferentes, pode ser projetada contra duas partes do fundo com cores diferentes. Ou seja, um lado do degrau pode estar ocluso ou superposto a uma outra face não pertencente ao degrau. Veja a figura 2.6

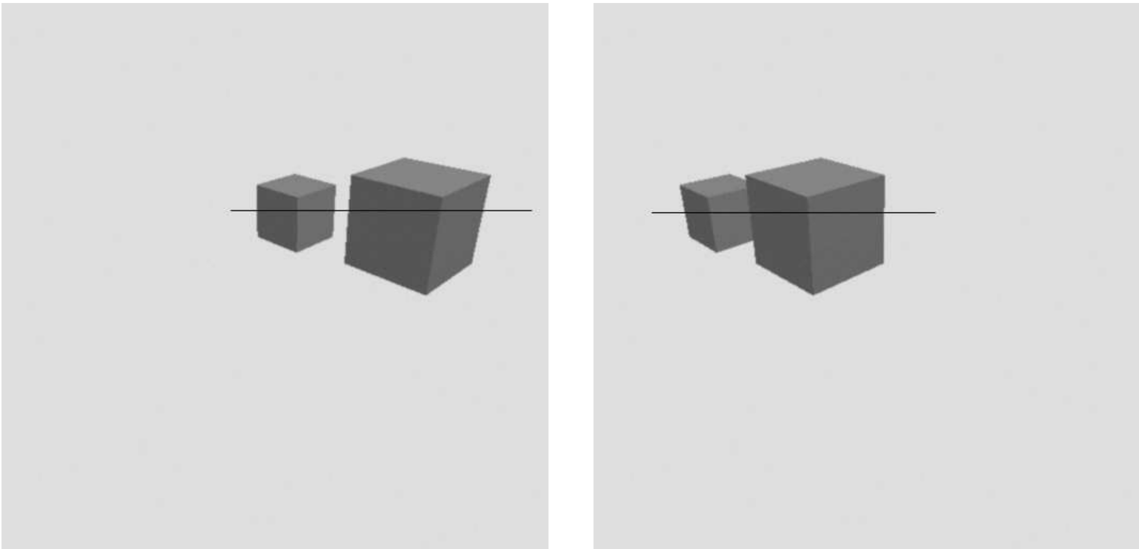

Figura 2.6: Um par estéreo sintético com 2 cubos.

Nesta figura, pode-se notar que, ao longo da linha demarcada, um lado do degrau pertencente a uma aresta do cubo maior está projetada contra fundos de cores diferentes nas duas imagens.

Note que o primeiro termo da fórmula 2.5 (peso 0.9) leva em conta apenas um dos lados dos dois degraus, o que tiver maior semelhança. O segundo termo da fórmula 2.5 tem a função de premiar casamentos que possuem os dois lados semelhantes. Em particular, os casamentos onde $e_1 = d_1$ e $e_2 = d_2$, que possuem alta probabilidade de corresponder ao mesmo ponto de uma aresta real do objeto, terão discrepância nula. Os casamentos onde $e_1 = d_1$ ou $e_2 = d_2$, mas não ambos, recebem uma discrepância um pouco maior (no máximo $\beta \cdot 255$).

Em particular, os degraus $(e_1, e_2) = (10, 50)$ e $(d_1, d_2) = (10, 100)$ tem discrepância $g = 0.9 \cdot 0 + 0.1 \cdot 50 = 5.0$, enquanto que os degraus $(e'_1, e'_2) = (10, 50)$ e $(d'_1, d'_2) = (20, 60)$ tem discrepância $g = 0.9 \cdot 10 + 0.1 \cdot 10 = 10.0$. O segundo casamento é penalizado pela função discrepância apesar dos degraus envolvidos terem a mesma amplitude $\Delta = 40$. A justificativa é que o primeiro casamento pode corresponder a uma face de intensidade 10 vista contra fundos de intensidade 50 e 100, respectivamente; enquanto que o segundo casamento só pode ser justificado supondo-se uma variação de 10 níveis de intensidade da iluminação do objeto, entre uma vista e outra.

Note que a medida de discrepância acima pressupõe que níveis de intensidade nas duas imagens podem ser comparados diretamente. Isto é verdade nos nossos testes, que foram realizados em imagens sintéticas ou em pares de imagens obtidas com a mesma câmera. Quando as imagens são obtidas com duas câmeras diferentes, a etapa de calibração deve incluir os ajustes nos níveis de intensidade dos pixels de forma a torná-los comparáveis.

2.5.3 Limitando casamentos múltiplos

Note que permitimos que cada degrau de uma imagem case com mais de um degrau da outra imagem, pois existem freqüentemente situações ambíguas como o da figura 2.6 que só podem ser resolvidas satisfatoriamente durante a etapa final de localização do objeto.

Uma vez calculadas as discrepâncias para todos os casamentos possíveis nas duas linhas, consideramos os n melhores casamentos. O número n é determinado por uma regra que tenta atender a dois objetivos conflitantes. Por um lado, não queremos enviar para a etapa seguinte um número excessivo de casamentos com pouca probabilidade de corresponderem a pontos de arestas reais do objeto. Por outro lado, quando há vários casamentos ambíguos com discrepâncias semelhantes, como no exemplo da figura 2.6, não queremos excluir arbitrariamente alguns deles, pois isto pode nos levar a excluir os casamentos corretos.

Portanto, o valor de n é determinado pelo seguinte algoritmo: primeiramente estipulamos um número mínimo e máximo de casamentos devolvidos dados por

$$\begin{aligned} n_{min} &= \kappa_{min} \max \{N_e, N_d\} \\ n_{max} &= \kappa_{max} \max \{N_e, N_d\} \end{aligned}$$

onde N_e, N_d são os comprimentos das duas listas, e κ_{min}, κ_{max} são parâmetros do algoritmo (tipicamente entre 1.0 e 2.0).

Os n_{min} melhores casamentos são sempre escolhidos. Além desses, continuamos aceitando casamentos adicionais, em ordem de discrepância crescente, até que a diferença entre o próximo casamento da lista e o último casamento escolhido seja maior que a diferença entre os dois últimos escolhidos; ou até escolher n_{max} casamentos.

2.6 Cálculo da posição espacial

Identificados os prováveis degraus correspondentes nas duas imagens, determinamos as coordenadas destes degraus no espaço tridimensional. Para tanto, utilizamos as fórmulas (1.2) vistas na seção 1.2.4.

A coordenada z calculada segundo estas fórmulas, terá uma imprecisão (erro esperado de medida) diferente da imprecisão das outras duas coordenadas. Na verdade, a imprecisão de z vai variar de ponto para ponto. Para não complicar o tratamento de erros na etapa de segmentação, calculamos as *coordenadas isotrópicas* do ponto, dadas por:

$$x'' = \frac{x'_l + x'_r}{2} \qquad y'' = \frac{y'}{\sqrt{2}} \qquad z'' = \frac{x'_l - x'_r}{2} \tag{2.6}$$

Neste sistema, a imprecisão em cada coordenada (x'', y'', ou z'') é igual à imprecisão da posição horizontal dos degraus (x'_l, x'_r), dividida por $\sqrt{2}$. Nota-se que as coordenadas reais (fórmulas 1.2) estão relacionadas às isotrópicas pelas transformações:

$$x = b\frac{x'' - x_0}{2z''} \qquad y = b\frac{\sqrt{2}y'' - y_0}{2z''} \qquad z = \frac{fb}{2z''} \tag{2.7}$$

A justificativa para as fórmulas 2.6 é a seguinte. Do cálculo estatístico temos que para duas variáveis A e B independentes

$$\begin{aligned} Var(A+B) &= Var(A) + Var(B) \\ Var(\alpha A) &= \alpha^2 Var(A) \end{aligned}$$

onde $Var(X)$ denota a variância da variável aleatória X.

Supondo que as coordenadas das imagens (x'_l, x'_r e y'_l) são afetadas por erros com mesma variância σ^2, temos:

$$\begin{aligned} Var(x'') &= \frac{\sigma^2 + \sigma^2}{(2)^2} = \frac{\sigma^2}{2} \\ Var(y'') &= \frac{\sigma^2}{(\sqrt{2})^2} = \frac{\sigma^2}{2} \\ Var(z'') &= \frac{\sigma^2 + \sigma^2}{(2)^2} = \frac{\sigma^2}{2} \end{aligned}$$

Portanto os erros de medida das coordenadas isotrópicas x'', y'' e z'' têm a mesma variância $\sigma^2/2$.

2.7 Análise de complexidade

Na etapa de detecção de degraus temos que o custo para detectar os degraus em uma imagem é $O(nm) = O(n_{pixels})$, onde n, m e n_{pixels} são os números de linhas , colunas e pixels da imagem respectivamente. O número de degraus detectados é menor ou igual a Dn, onde D é o parâmetro citado na seção 2.4.

Na etapa de casamento de degraus o custo é $O(nN_eN_d) = O(nD^2)$, onde N_e e N_d são os números de degraus detectados em uma linha de varredura das imagens da esquerda e direita, respectivamente. O número de pares casados é menor ou igual a $n\kappa_{max}D$.

2.8 Operador de Sobel

Mencionamos anteriormente que a detecção de degraus poderia ser feita de maneira mais robusta por meio de operadores tradicionais de detecção de arestas como os de Sobel [3, 18]. Para completar este capítulo, vamos considerar as modificações que isto acarretaria nos algoritmos acima.

Uma aresta local é uma pequena área na imagem onde os níveis de cinza estão mudando rapidamente e monotonicamente. Um operador de arestas é um operador matemático ou computacional, com um domínio de extensão espacial pequena (ex. uma janela 3×3) destinado a detectar a presença de arestas locais na imagem.

A partir das intensidades destes pixels, o operador de Sobel calcula a *magnitude* Δ da variação de intensidade da aresta (equivalente à amplitude dos nossos degraus) e a *direção* em que a variação é máxima. As fórmulas do operador de Sobel podem ser encontradas na bibliografia [18].

Em termos práticos, a diferença é que teremos de processar três linhas de varredura consecutivas de cada vez (y_l, y_{l-1}, y_{l+1}), e aplicar o operador de Sobel a cada janela 3×3 contida nessas linhas. O operador de Sobel devolve para cada abscissa x_e um valor de magnitude e direção. Escolhemos os D melhores, pela medida de qualidade (2.4); estes são os degraus.

Para nossa medida de discrepância na etapa de casamento dos degraus, temos que converter a magnitude num par de intensidades (e_1, e_2).

A informação extra da direção da aresta fornecida pelo operador de Sobel não é de grande importância para o nosso algoritmo de casamento, pois, dependendo da posição das câmeras com relação ao objeto e o objeto em questão, uma aresta do objeto pode aparecer com direções diferentes nas projeções nas duas imagens. Para usar esta informação teríamos que calcular tolerâncias para as variações de direções e talvez até fazer suposições sobre a cena.

Uma vantagem do operador de Sobel sobre o nosso operador de carreiras é que pelo fato de trabalhar com uma janela bidimensional ele é menos sensível ao ruído.

Capítulo 3

Segmentação dos pontos no espaço

3.1 Objetivo

O objetivo desta etapa é agrupar um conjunto de pontos no espaço $\mathbf{R}^3$ numa coleção de *segmentos* — subconjuntos de pontos, cada um dos quais melhor se ajusta em uma linha reta. A entrada para esta etapa é o conjunto de pontos P em $\mathbf{R}^3$ resultante do casamento dos degraus e cálculo das respectivas coordenadas isotrópicas (vide seção 2.6). Idealmente, para permitir a localização do objeto nas etapas seguintes, a coleção de segmentos resultante desta etapa deve incluir todas arestas dos objetos que são visíveis por ambas as câmeras.

3.2 Idéia do algoritmo

A idéia do algoritmo consiste em gerar inicialmente um grande número de *candidatos* a segmentos, cada um contendo uns poucos pontos vizinhos e aproximadamente colineares; e em seguida distribuir os demais pontos de P entre esses candidatos.

Mais precisamente, na fase de *geração inicial*, calculamos para cada ponto dado p seu ponto mais próximo p'. Os n pares $\{p, p'\}$, assim obtidos, são a coleção inicial de candidatos a segmentos.

Em seguida, na etapa de *amadurecimento*, escolhemos repetidamente o candidato R mais promissor e acrescentamos ao mesmo o ponto de $P \setminus R$ que está mais próximo a R e aproximadamente alinhado com R; até obter um candidato com um certo número mínimo de pontos. Este candidato é então submetido a um processo de *crescimento*, e se acrescenta todos os pontos que estão aproximadamente alinhados com o mesmo, em ordem de distância crescente. Cada um dos conjuntos, assim obtido, é considerado um *segmento completo*.

As etapas de amadurecimento e crescimento são repetidas enquanto possíveis. Os resultados dessas etapas resultam no conjunto de segmentos completos.

A seguir, descrevemos estas etapas com mais detalhes. Para tanto, precisamos primeiro definir alguns conceitos.

3.3 Direções principais e momentos de inércia do candidato

Um *candidato* a segmento é um conjunto de pontos do $\mathbf{R}^3$. A posição, direção e extensão aproximadas de um candidato podem ser descritas pelo *baricentro*, *eixos principais* e *momentos de inércia* desse conjunto de pontos, definidos a seguir [3]. (Estes nomes são emprestados da física, mas os mesmos conceitos são também usados em estatística, controle, modelagem, etc.)

O *baricentro* b de um candidato é simplesmente a média aritmética das posições dos pontos; ou seja, o centro de massa dos mesmos, supondo-se que cada ponto seja uma partícula de massa unitária.

Para calcular os demais parâmetros do candidato, transladamos todos os seus pontos de forma a trazer o baricentro para a origem. Feito isso, definimos então a *matriz de dispersão* dos mesmos como sendo a matriz simétrica de dimensão 3×3

$$M = X^T X$$

onde X é a matriz de coordenadas dos pontos transladados,

$$X_i = (x_{i1}, x_{i2}, x_{i3}) \qquad i = 1, 2, \ldots, n$$

Pode-se mostrar que a matriz M tem três autovalores reais não-negativos $\lambda_1, \lambda_2, \lambda_3$, correspondentes a três autovetores ortogonais V_1, V_2, V_3. No que segue, vamos supor que $\lambda_1 \geq \lambda_2 \geq \lambda_3$, e $|V_1| = |V_2| = |V_3| = 1$.

Por definição, os vetores V_i são os *eixos principais* do candidato. O eixo V_1 define a direção de máxima dispersão, na qual a nuvem de pontos é mais alongada. Se os pontos são amostras de um segmento de reta, V_1 é a direção mais provável deste segmento, enquanto que V_2 e V_3 definem direções perpendiculares ao mesmo.

Os autovalores λ_i são os *momentos de inércia* em relação aos planos perpendiculares a esses eixos. Se projetarmos todos os pontos do candidato sobre o eixo V_i, a variância dessas projeções será λ_i/n. Portanto, as dimensões aproximadas da nuvem de pontos são proporcionais aos números

$$D_i = \sqrt{\frac{\lambda_i}{n-1} + \epsilon^2} \tag{3.1}$$

Nesta fórmula, o parâmetro ϵ é a magnitude dos erros de medida, quantização e aproximação, obtidos na etapa de detecção de descontinuidades (seção 2.3.2).

Supondo-se que os pontos são amostras de um segmento de reta, D_1 é aproximadamente proporcional ao comprimento do segmento. As quantidades D_2 e D_3 medem a dispersão dos pontos nas direções perpendiculares à reta, e devem idealmente ser bem pequenas — da ordem de ϵ. (A justificativa para definir a dispersão D_i pela fórmula (3.1), em vez de simplesmente $\sqrt{\lambda_i/n}$, é que a primeira é mais confiável quando o número de pontos é pequeno.)

Para calcular os autovetores e autovalores da matriz M, usamos o algoritmo de Jacobi [19, páginas 335–380].

3.3.1 Sistema de coordenadas do candidato

Todo candidato R define um sistema de coordenadas tridimensionais próprio, cuja origem é o baricentro b, cujos eixos são paralelos aos vetores principais V_i, e cujas unidades de medida em cada eixo são proporcionais às dispersões D_i dos pontos.

Mais precisamente, as coordenadas $y = (y_1, y_2, y_3)$ de um ponto, no sistema do candidato, são obtidas a partir de suas coordenadas globais $x = (x_1, x_2, x_3)$ pela fórmula

$$y_i = \frac{(x - b) \cdot V_i}{D_i}$$

onde $\cdot$ denota produto escalar.

3.3.2 Métrica definida por um candidato

Cada candidato também define uma norma $|x|_R$ para vetores do $\mathbf{R}^3$, que em princípio é simplesmente a norma euclidiana no sistema local de coordenadas:

$$|x|_R = \sqrt{y_1^2 + y_2^2 + y_3^2}$$

Levando em conta erros de medidas nas coordenadas x (da ordem de ϵ), uma estimativa mais correta para esta norma é:

$$|x|_R = \sqrt{y_1^2 + (\frac{\epsilon}{D_1})^2 + y_2^2 + (\frac{\epsilon}{D_2})^2 + y_3^2 + (\frac{\epsilon}{D_3})^2}$$

Esta norma é usada para comparar distâncias entre pontos dados e o candidato, dando preferência a pontos alinhados com o mesmo. Os pontos que estão a uma distância fixa r do baricentro, nesta métrica, estão na superfície de um elipsóide cujos semi-diâmetros principais são paralelos aos vetores V_i e têm comprimento aproximadamente igual a rD_i. A figura 3.1 ilustra a seção transversal deste elipsóide. Os pontos que estão na elipse menor estarão a distância 1 do baricentro, enquanto que os pontos que estão na elipse maior estão a uma distância 2 do baricentro definido pela nuvem de pontos.

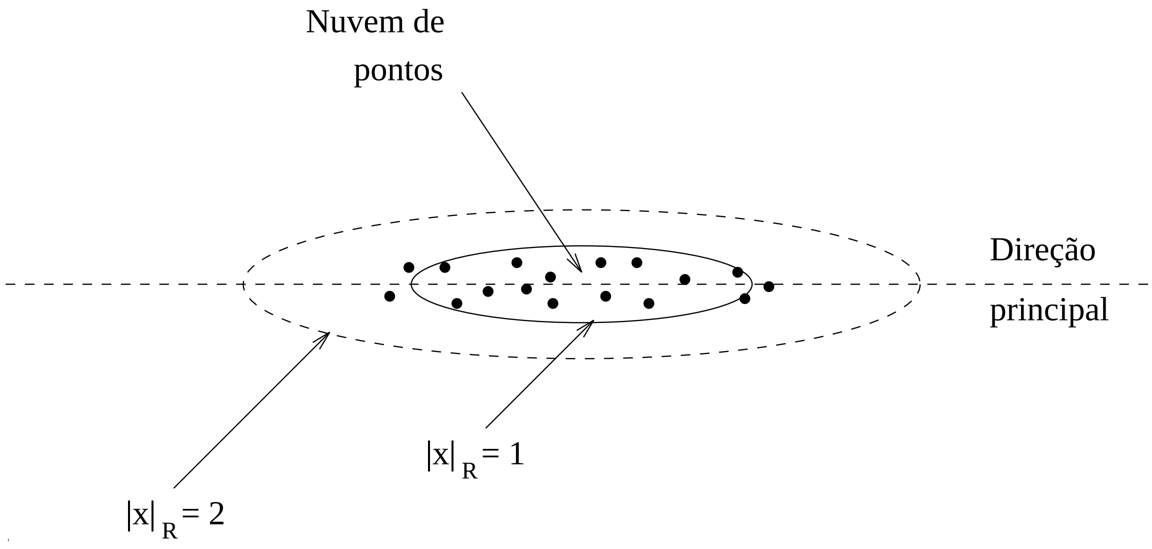

Figura 3.1: Métrica definida por um candidato.

3.3.3 Qualidade

Definimos a *qualidade* de um candidato como sendo a razão entre a dispersão máxima dos pontos, e a dispersão lateral dos mesmos, isto é

$$Q = \frac{2D_1^2}{D_2^2 + D_3^2}$$

Esta grandeza é uma medida da "colinearidade" relativa dos pontos. O valor de Q varia entre 1 (quando a nuvem de pontos é aproximadamente esférica) e infinito (quando os pontos estão perfeitamente alinhados sobre um segmento de comprimento muito maior que ϵ).

3.4 Algoritmo detalhado

3.4.1 Geração inicial dos candidatos

Seja P o conjunto de pontos obtidos na etapa de casamento de degraus (seção 2.5). Nesta etapa, calculamos para cada ponto dado p de P seu ponto mais próximo p'. Os n pares $\{p, p'\}$, assim obtidos, são a coleção inicial de candidatos, desde que a distância $|p - p'|$ seja menor que um parâmetro δ_{ger} fornecido ao programa. Usamos esta condição para impedir que pontos muito distantes dos demais, presentes devidos a ruído e a falsos casamentos, venham formar falsos candidatos a segmentos.

3.4.2 Amadurecimento

Os candidatos (pares de pontos) resultantes da etapa anterior são submetidos ao seguinte processo de *amadurecimento* até obtermos um candidato com um número mínimo M de pontos. (O valor de M depende da natureza dos objetos; para nossos teste, usamos M entre 4 e 9).

Dentre todos os candidatos, escolhemos um que tenha ainda apenas dois pontos; ou, se todos tiverem mais que dois pontos, escolhemos aquele cuja nuvem tem a forma mais alongada (maior Q). Seja R o candidato escolhido. Dentre todos os pontos de P que não estão em R e ainda não foram incluídos em nenhum segmento completo, escolhemos o ponto u_{min} que está mais próximo ao baricentro b de R, na métrica associada a R. Mais precisamente, só acrescentamos u_{min} a R se

$$|u_{min}|_R \ \leq \ 2.3 + \frac{\delta_{amd}}{D_1} \tag{3.2}$$

Nesta fórmula δ_{amd} é a distância máxima aceitável entre dois pontos consecutivos ao longo de um segmento. Ou seja, δ_{amd} é o comprimento máximo, em coordenadas isotrópicas, de uma "quebra" na seqüência de pontos que pode ser ignorada.

A justificativa para a constante 2.3 é que se R é uma seqüência de k pontos igualmente espaçados ao longo de uma reta, a distância (na métrica de R) entre o próximo ponto x_{k+1} da seqüência e o baricentro pode ser calculado pela fórmula:

$$|x_{k+1}|_R = \frac{\sqrt{5k^2 + 8k + 36\epsilon^2 + 3}}{\sqrt{k^2 + k + 12\epsilon^2}}$$

cujo resultado está no intervalo $[2.2360, 2.2439]$. Portanto, se quisermos que uma seqüência de pontos uniformemente espaçados seja reconhecida como um segmento, devemos sempre aceitar pontos com $|u_{min}|_R \leq 2.2439$.

A condição (3.2) tem o objetivo de diminuir a formação de candidatos espúrios na fase de amadurecimento. Sem essa condição, o algoritmo continuaria acrescentando pontos ao melhor candidato, até completar M pontos, mesmo que esses pontos estejam distantes ou pouco alinhados com o candidato. Veja a figura 3.2. A condição (3.2) não evita completamente este problema, mas reduz bastante sua incidência.

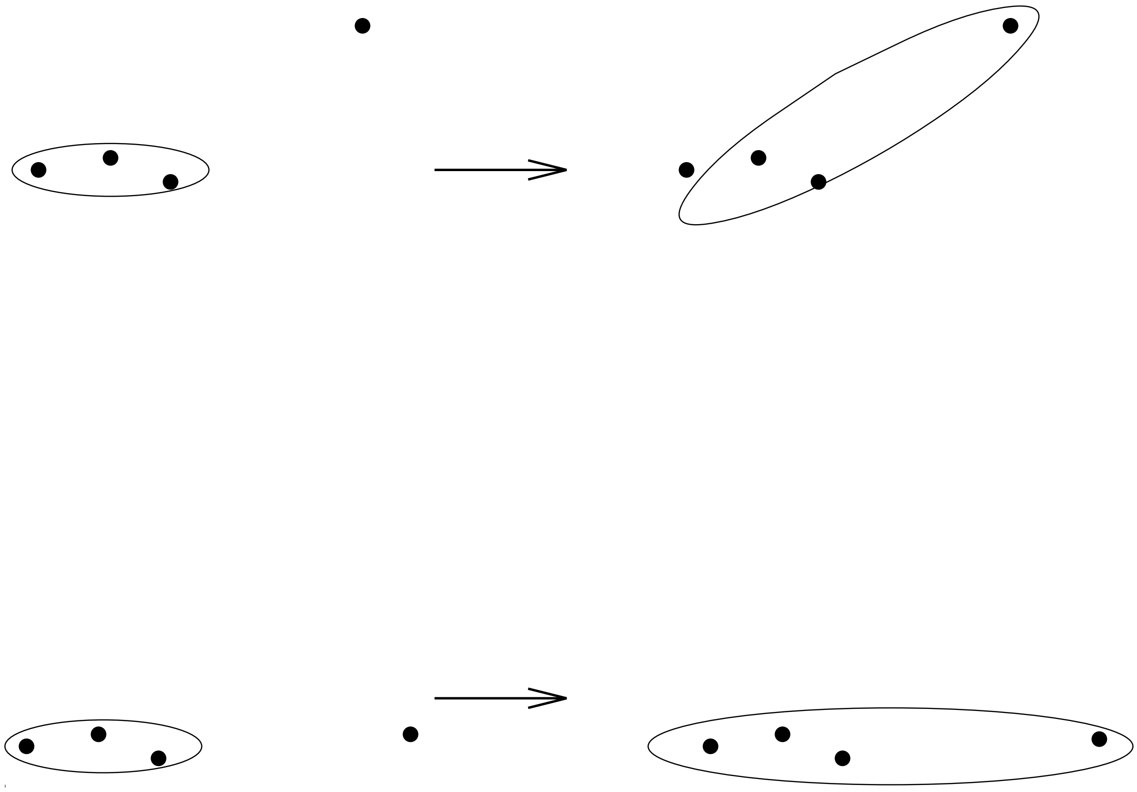

Figura 3.2: Justificativa para condição (3.2).

Observe que, depois de no máximo $n = |P|$ interações deste processo, todos os candidatos terão pelo menos três pontos. Observe também que a inclusão de u_{min} no candidato R pode aumentar ou diminuir sua qualidade, de modo que o mesmo candidato pode ser escolhido a cada interação, ou vários candidatos podem se revezar.

O objetivo da fase de amadurecimento é obter um conjunto representativo de candidatos, pequenos e compactos. Idealmente, deveríamos obter pelo menos um candidato maduro em cada aresta do objeto.

O número mínimo M de pontos para o candidato não deve ser demasiadamente alto. Apesar da condição (3.2), se a cena analisada possui arestas com menos de M pontos, a fase de amadurecimento pode juntar pontos de várias arestas não colineares num único candidato para poder completar o número mínimo M de pontos. O resultado é um candidato espúrio que não coincide com nenhuma dessas arestas.

O número mínimo M também não deve ser demasiado baixo, pois é importante que o candidato ao sair da fase de amadurecimento, possa definir através de seus eixos principais a direção aproximada do segmento de reta do qual ele faz parte. Quanto maior o número de pontos nos candidatos que realmente pertençam à segmentos de retas, maior será a precisão no cálculo da direção deste candidato. Como veremos na seção 3.4.3, é importante que a direção do candidato esteja bem definida quando ele passar para o processo de crescimento.

3.4.3 Crescimento de um candidato

No decorrer do amadurecimento, eventualmente um dos candidatos R atinge o tamanho mínimo M, e nesse momento ele é considerado "maduro" e submetido ao processo de crescimento.

O objetivo desta fase é acrescentar ao candidato todos os pontos de P que realmente parecem pertencer à mesma aresta. Mais precisamente, dentre todos os pontos de $P \setminus R$ escolhemos um ponto u_{min} que satisfaz as seguintes condições: (1) está aproximadamente alinhado com a direção principal do candidato; (2) não está muito distante do baricentro do candidato na métrica $|x|_R$; e (3) está mais próximo do baricentro, nessa mesma métrica, dentre todos os pontos que satisfazem (1) e (2). Neste caso acrescentamos u_{min} a R.

Repetimos este processo, para o mesmo candidato R, até que não reste nenhum ponto que possa ser nele incluído.

Retiramos então R do conjunto de candidatos, e eliminamos os pontos que pertencem a R de todos outros candidatos imaturos. Todo candidato que ficar vazio, ou reduzido a um único ponto, é também eliminado.

A justificativa para este último passo é que um candidato imaturo S que contém pontos do segmento recém-completado R ou é praticamente um subconjunto de R, ou contém muitos pontos de outros segmentos.

No primeiro caso, o candidato S merece ser eliminado, por ser redundante. Esta precaução é necessária, e geralmente suficiente, para evitar redundâncias na saída (dois ou mais segmentos completos que praticamente coincidem).

No segundo caso (que geralmente ocorre na junção entre duas ou mais arestas), a retirada dos pontos provavelmente melhora a qualidade de S, e portanto aumenta as chances desses outros segmentos serem reconhecidos. Note que, neste caso, os pontos retirados do candidato imaturo podem ser re-incluídos mais tarde quando este candidato passar para a fase de crescimento. Portanto, pontos comuns a várias arestas do objeto normalmente são incluídos em todos os segmentos completos correspondentes a essas arestas.

Condição de alinhamento

Para satisfazer a condição (1) acima — que o ponto u_{min} esteja aproximadamente alinhado com a direção principal de R — exigimos que suas coordenadas (y_1, y_2, y_3), relativas a R, satisfaçam a desigualdade

$$y_1^2 - y_2^2 - y_3^2 + 1 \geq 0 \tag{3.3}$$

Esta condição define o interior de um hiperbolóide de uma folha, que, no sistema de coordenadas do candidato, tem seção circular, eixo Y_1, e ângulo de abertura de 90°. No sistema de coordenadas global, este hiperbolóide é alongado na direção do eixo principal V_1 do candidato, e seu ângulo de abertura é tanto menor quanto mais colineares forem os pontos. Veja a figura 3.3.

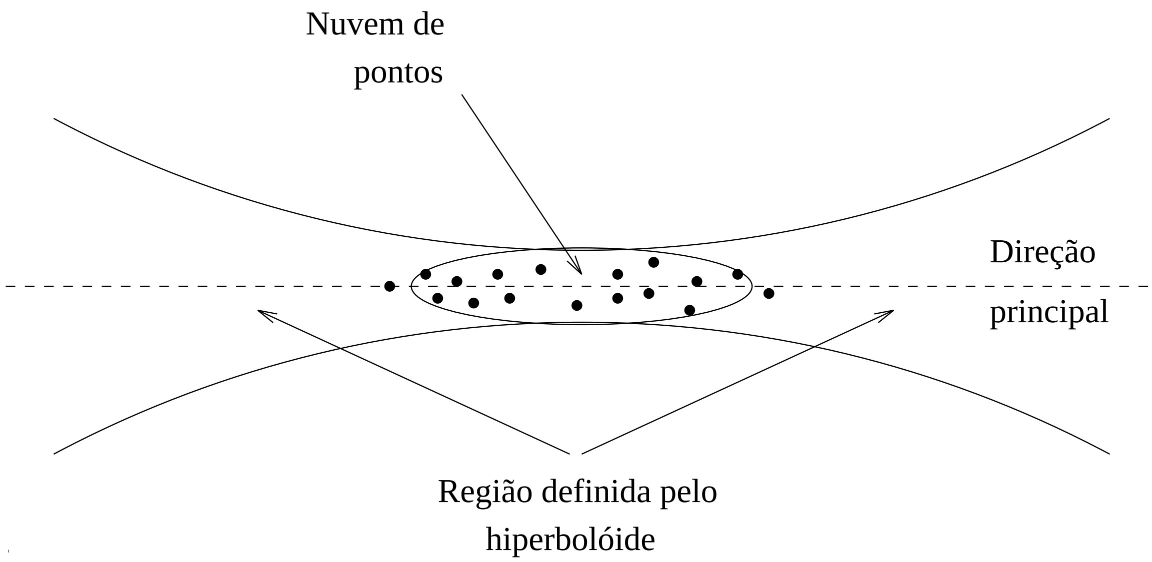

Figura 3.3: Seção transversal do hiperbolóide definido pela condição (1)

A razão para limitar a inclusão de pontos por um hiperbolóide, em vez de por um cilindro com eixo V_1, é que a direção deste vetor é apenas uma estimativa da direção real da aresta. Portanto, quanto mais longe um ponto estiver do baricentro, maior deve ser nossa tolerância quanto a seu desvio do eixo V_1. Por outro lado, quanto mais alongado for o candidato, menor deve ser essa tolerância. A condição (3.3) tem precisamente estas propriedades.

No lugar da condição (3.3) poderíamos verificar a medida da qualidade do conjunto $R' = R \cup \{u_{min}\}$. Se a qualidade de R' for maior que a de R, significa que a nuvem de pontos R' é mais alongada que R, e portanto u_{min} está basicamente alinhado com a direção principal de R'. Porém, verificar se u_{min} pertence ao hiperbolóide é mais barato do que calcular a qualidade de R'.

Como já observamos, um candidato que sai da fase de amadurecimento precisa estar razoavelmente alinhado com a aresta da qual ele faz parte, caso contrário a condição do hiperbolóide poderá impedir que ele cresça até englobar a aresta toda. Veja a figura 3.4.

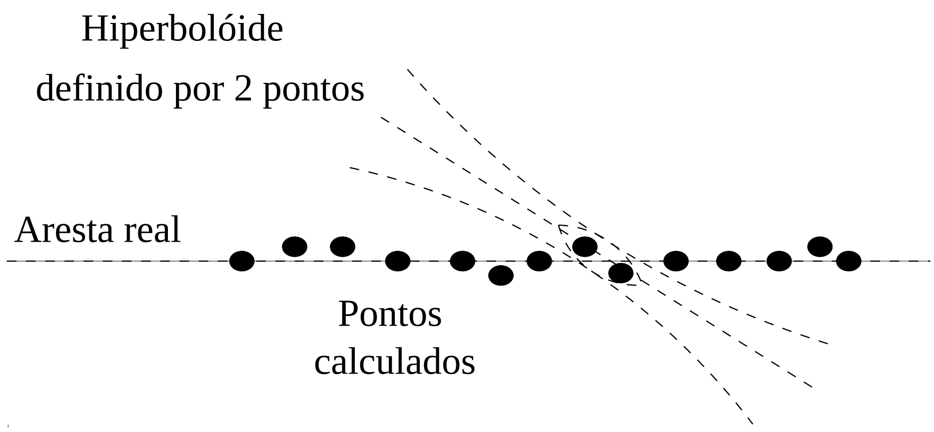

Figura 3.4: Justificativa para condição $|R| \geq M$.

Condição de distância máxima

A condição (2) é implementada pelo mesmo teste usado na fase de amadurecimento

$$|u_{min}|_R \;\leq\; 2.3 + \frac{\delta_{crs}}{D_1}$$

O propósito desta condição — que a distância do ponto u_{min} ao baricentro seja menor ou igual a $2.3 + \delta_{crs}/D_1$ — é impedir que um ponto isolado que está muito além do extremo do candidato, mas que por coincidência está perfeitamente alinhado com ele, seja indevidamente incluído no segmento. Esta condição também impede que dois segmentos colineares sejam considerados como um único segmento, se a separação entre eles for maior que δ_{crs}.

Critério de desempate

Finalmente, a condição (3) — que o ponto u_{min} seja o mais próximo possível ao baricentro, na métrica do candidato — favorece pontos que estão alinhados com o eixo principal, sem excluir pontos que estão menos alinhados porém próximos ao baricentro.

Tamanho mínimo

Caso nenhum ponto possa ser incluído a um candidato, nesta etapa, ele não é considerado um segmento completo e é eliminado, pois tem baixa probabilidade de representar uma aresta real do objeto. Ou seja, um candidato completo só é aceito se tiver pelo menos $M + 1$ pontos.

Representação dos segmentos completos

Cada segmento completo R produzido pelo algoritmo acima é passado à etapa seguinte na forma de um segmento idealizado de reta.

Os extremos p, q deste segmento são obtidos projetando-se todos os pontos de R no seu eixo principal V_1, e tomando-se os pontos projetados que estão mais distantes do baricentro nas duas direções.

É importante lembrar que tanto os pontos dados quanto os segmentos encontrados estão descritos no sistema de coordenadas isotrópicas (seção 2.6).

3.4.4 Trabalhos relacionados

Existem alguns trabalhos na literatura que fazem o reconhecimento de segmentos de retas de uma maneira similar ao nosso método apresentado. Por exemplo, no trabalho de McLaughlin e Alder [16, 15] é apresentado um método para o reconhecimento de segmentos de retas no plano a partir de conjuntos de pixels em uma imagem. Este método consiste basicamente em identificar pequenos subconjuntos de pontos da imagem, que podem ser cobertos com elipses de tamanho fixo. As arestas da cena são determinadas procurando-se conjuntos de elipses aproximadamente alinhadas.

Além da dimensão do espaço, uma principal diferença entre nosso método e esse trabalho é que nossos elipsóides mudam de forma e tamanho à medida que pontos são acrescentados, até obtermos um único elipsóide para cada aresta da cena.

O reconhecimento de segmentos de reta a partir de pontos no espaço também é um passo intermediário na tese de doutorado de G. Klinker [12]. Neste trabalho, que trata da segmentação de imagens monoculares coloridas, os pontos dados são as cores de todos os pixels de uma imagem (em coordenadas RGB), e o objetivo é identificar subconjuntos de pixels cujas cores se distribuem ao longo de um segmento de reta. (Estes conjuntos correspondem a reflexos polidos, ou *highlights*, dos objetos da cena). Infelizmente, a tese não descreve o algoritmo usado para identificar tais conjuntos.

3.5 Análise de complexidade

Se n é o número de pontos dados, a geração de candidatos iniciais custa $\Theta(n^2)$ e gera $\Theta(n)$ candidatos no pior caso. Não é difícil ver que, com o algoritmo aqui descrito, o custo de se conseguir o primeiro candidato maduro é $\Theta(n^2)$ no pior caso; e o custo de crescer esse candidato até um segmento completo com k pontos é $\Theta(nk)$. O número de segmentos devolvidos é menor ou igual a n/M.

Portanto, o custo total do algoritmo de segmentação é $\Theta(n^3)$ no pior caso; pois o algoritmo pode encontrar $\Theta(n)$ segmentos completos, cada qual com $\Theta(n)$ pontos. (Note que segmentos completos não são necessariamente disjuntos.)

Felizmente, estes limites só podem ser atingidos para conjuntos de pontos bastante peculiares, onde boa parte dos pontos estão concentrados em nuvens de raio aproximadamente igual a ϵ. Se esta situação ocorrer com pontos gerados pela fase de detecção e casamento, as imagens originais provavelmente são excessivamente complexas ou de baixa qualidade, e provavelmente inadequadas para quaisquer algoritmos de visão estereoscópica.

Em situações mais realistas, podemos esperar que cada ponto dado pertencerá a um ou dois segmentos completos, em média. Nesse caso, o custo esperado do algoritmo de segmentação será $\Theta(n^2)$. Nota-se que, se a fase de detecção e casamento de descontinuidades retorna um número fixo de casamentos por linha de varredura, o número de dados n será proporcional à altura da imagem; e, portanto, o custo esperado do algoritmo de segmentação será proporcional ao número de pixels das imagens de entrada.

Como esta implementação é apenas um protótipo para testar a validade dos critérios de segmentação, não nos preocupamos em tornar os algoritmos mais eficientes. Parece razoável supor que, com estruturas de dados adequadas, a etapa de segmentação poderia ser efetuada bem mais rapidamente – em tempo $O(n \log n)$, ou mesmo $O(n)$.

Capítulo 4

Localização do modelo

Na etapa de localização do modelo, os dados são um conjunto de segmentos de retas no espaço (ainda em coordenadas isotrópicas). O objetivo é localizar neste conjunto um objeto cujas forma e dimensões são conhecidas.

O método que usamos é específico para objetos poliédricos (limitados por superfícies planas) e convexos. Da coleção de segmentos obtida na etapa anterior, extraímos inicialmente uma coleção de *vértices observados* U – essencialmente as extremidades desses segmentos, modificados como descrito mais adiante. Temos também uma descrição matemática do objeto procurado, incluindo em particular o conjunto de seus vértices V. A localização do objeto é feita procurando-se, dentre os pontos U, um subconjunto U' que coincide aproximadamente com um subconjunto V' de V, adequadamente rotacionado e transladado.

Definimos uma medida de discrepância para avaliar a similaridade (ou ajuste) entre os subconjuntos U' e V'. Dentre todos os pares U',V' considerados, escolhemos o que possui menor discrepância. Este par determina a posição e orientação que o objeto se encontra na cena analisada.

O modelo do objeto poliédrico utilizado é uma estrutura de dados simples, que descreve a topologia dos vértices, arestas e faces. A estrutura também inclui as coordenadas homogêneas dos vértices, e os coeficientes homogêneos dos planos das faces, expressas em relação a um sistema de coordenadas própria do objeto. Com essas informações é possível determinar quais são os vértices e arestas visíveis para um observador em qualquer posição dada, usando técnicas padrões de computação gráfica [5, 17].

4.1 Geração dos vértices

A etapa de segmentação devolve uma coleção de segmentos de retas, sendo que cada segmento é identificado pelos seus pontos extremos. Devido a erros nas etapas anteriores, via de regra, arestas que terminam num mesmo vértice do objeto dão origem a segmentos com extremos próximos mas distintos. Portanto, devemos tentar identificar os segmentos que provavelmente correspondem a arestas adjacentes do objeto observado e tentar unir os seus extremos para obter vértices do objeto.

Em primeiro lugar, para cada par de segmentos que parecem ser adjacentes, calculamos a posição aproximada do vértice comum hipotético, extrapolando os dois segmentos se necessário. Feito isso, identificamos subconjuntos de extremidades (originais ou calculadas) que estão próximas entre si. Cada um desses subconjuntos é interpretado como um possível vértice do objeto.

4.1.1 Extrapolação dos extremos

O objetivo desta etapa é identificar entre os segmentos dados os que aparentemente correspondem a arestas adjacentes do objeto, e extrapolá-los de modo que seus extremos fiquem mais próximos do vértice comum a essas arestas.

Para esse fim examinamos todos os pares de segmentos, calculando para cada par a distância mínima entre as respectivas retas-suporte. Se esta distância for suficientemente pequena então calculamos os pontos (p, p'), um em cada reta suporte, que definem esta distância mínima. Se esses pontos estiverem suficientemente próximos dos segmentos, então os pontos (p, p') serão considerados na etapa seguinte. Chamamos esses pontos de *extremos extrapolados*.

Mais especificamente, consideramos a distância entre as retas suficientemente pequena se for menor que um múltiplo fixo $\rho\epsilon$ do erro ϵ de medida, definido na seção 3.3, sendo ρ um parâmetro do programa. Consideramos, também, que um extremo extrapolado está próximo ao segmento se ele estiver no interior do mesmo ou até $L/2$ de distância do extremo original mais próximo, onde L é o comprimento do segmento.

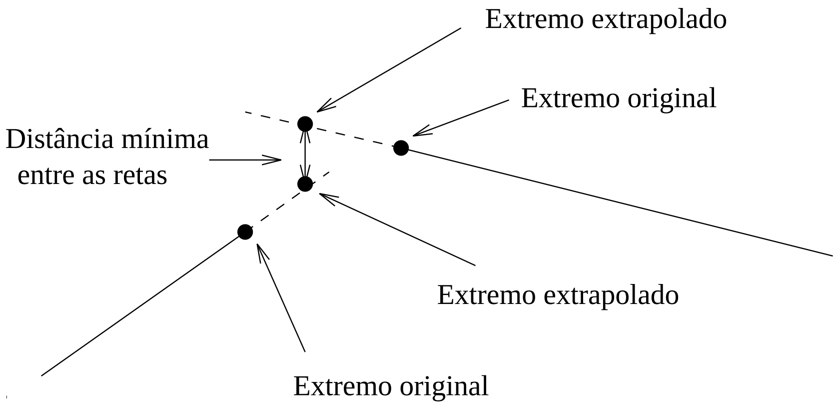

Figura 4.1: Vértices extremos e extrapolados de dois segmentos que poderão formar um único vértice dado pelo baricentro dos dois vértices extrapolados.

O resultado deste processo é um conjunto E de extremos originais e extrapolados.

4.1.2 União de extremos em vértices

Nesta etapa, tentamos unir os pontos de E que provavelmente formam um único vértice do objeto.

Para cada ponto p de E, construímos o subconjunto N_p de todos os pontos de E que estão no máximo a uma distância $\tau\epsilon$ de p, onde τ $(\tau \geq \rho)$ é um parâmetro do programa. Feito isso, construímos o conjunto U de vértices observados da seguinte maneira: agrupamos os subconjuntos N_p que possuem pontos em comum em *supergrupos* disjuntos dois a dois. Para cada supergrupo assim obtido, que contenha pelo menos um extremo *extrapolado*, acrescentamos um vértice ao conjunto U, cuja posição é o baricentro dos extremos *extrapolados* pertencentes ao supergrupo.

Acrescentamos também a U, todo extremo *original* p de E cujo supergrupo não contenha nenhum extremo extrapolado. Consideramos essa hipótese pelo fato da etapa de segmentação poder formar um segmento correto que não tem outros segmentos adjacentes. Isto ocorre, por exemplo, quando há arestas parcialmente ocultas, horizontais, ou de pequena amplitude.

4.2 Processo de casamento

Na etapa de casamento estamos interessados em identificar, dentre os vértices de U, o subconjunto U' que mais se aproxima de um subconjunto V' do conjunto de vértices do modelo V, adequadamente posicionados como descrito abaixo. Nosso algoritmo determina vários posicionamentos para o conjunto V e, para cada um, determina um valor de discrepância que qualifica a similaridade entre os subconjuntos V' e U' e, portanto o ajuste do modelo com a cena.

4.2.1 Posicionamento do modelo

Posicionar o modelo na cena consiste em especificar transformações de escalonamento, de rotação e translação do sistema de coordenadas do modelo para o sistema de coordenadas da cena. Para o cálculo destas matrizes de transformação necessitamos de apenas três vértices (v_1, v_2, v_3) do modelo e de três vértices (u_1, u_2, u_3) extraídos das imagens.

O efeito destas transformações pode ser descrito através de uma única matriz 4×4 [5]:

$$M = T_1 S R T_2 \tag{4.1}$$

onde a matriz T_1 translada o baricentro dos três vértices (v_1, v_2, v_3) do modelo para origem. A matriz S é um escalonamento (ampliação ou redução) uniforme do modelo. A matriz R é uma rotação que mantém a origem fixa. Finalmente T_2 é uma translação que leva a origem para o baricentro dos três vértices (u_1, u_2, u_3) dos dados. As matrizes T_1, S, R, eT_2 são calculadas como descrito a seguir.

Na realidade, a matriz M, calculada segundo a fórmula 4.1 deveria ser utilizada apenas como uma aproximação inicial. A matriz final deveria ser obtida por algum processo de otimização numérica. Não tivemos tempo de implementar esta otimização.

A matriz T_1 que desloca o baricentro $\bar{v}$ de (v_1, v_2, v_3) para a origem, e a matriz T_2 que desloca a origem para o baricentro $\bar{u}$ de (u_1, u_2, u_3), são dadas pelas fórmulas abaixo:

$$T_1 = \begin{pmatrix} 1.0 & -\bar{v}_x & -\bar{v}_y & -\bar{v}_z \\ 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix}$$

$$T_2 = \begin{pmatrix} 1.0 & \bar{u}_x & \bar{u}_y & \bar{u}_z \\ 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix}$$

A matriz de escalonamento S tem a forma

$$S = \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & \alpha & 0.0 & 0.0 \\ 0.0 & 0.0 & \alpha & 0.0 \\ 0.0 & 0.0 & 0.0 & \alpha \end{pmatrix}$$

onde α é o fator de escalonamento. Para determinar o fator α calculamos o raio médio quadrático d_u dos vértices u_1, u_2, u_3 em relação a seu baricentro. Calculamos também o valor análogo d_v para os vértices v_1, v_2, v_3, do modelo; sendo o fator de escalonamento dado pelo quociente entre os dois resultados. Mais precisamente:

$$\begin{aligned}
d_u &= \sqrt{\frac{1}{3}\sum_{i=1}^{3}|u_i - \bar{u}|^2} \\
d_v &= \sqrt{\frac{1}{3}\sum_{i=1}^{3}|v_i - \bar{v}|^2} \\
\alpha &= \frac{d_u}{d_v}
\end{aligned}$$

Para construir a matriz de rotação, precisamos definir três vetores ortogonais entre si (v_1^*, v_2^*, v_3^*) a partir dos vértices do modelo, e três vetores correspondentes (u_1^*, u_2^*, u_3^*) a partir dos vértices observados. A matriz R é então a única matriz que leva (v_1^*, v_2^*, v_3^*) para (u_1^*, u_2^*, u_3^*). Ou seja,

$$R = \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & v_{1x}^* & v_{1y}^* & v_{1z}^* \\ 0.0 & v_{2x}^* & v_{2y}^* & v_{2z}^* \\ 0.0 & v_{3x}^* & v_{3y}^* & v_{3z}^* \end{pmatrix}^{-1} \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & u_{1x}^* & u_{1y}^* & u_{1z}^* \\ 0.0 & u_{2x}^* & u_{2y}^* & u_{2z}^* \\ 0.0 & u_{3x}^* & u_{3y}^* & u_{3z}^* \end{pmatrix}$$

Os vetores (v_1^*, v_2^*, v_3^*) são calculados pelas fórmulas:

$$\begin{aligned}
v_1^* &= \mathrm{dir}((v_2 - v_1) \times (v_3 - v_1)) \\
v_2^* &= \mathrm{dir}(v_1 - \bar{v}) \\
v_3^* &= v_1^* \times v_2^*
\end{aligned}$$

onde $\mathrm{dir}(v) = v/|v|$. Veja a figura 4.2. Os vetores (u_1^*, u_2^*, u_3^*) são calculados de modo análogo a partir de (u_1, u_2, u_3).

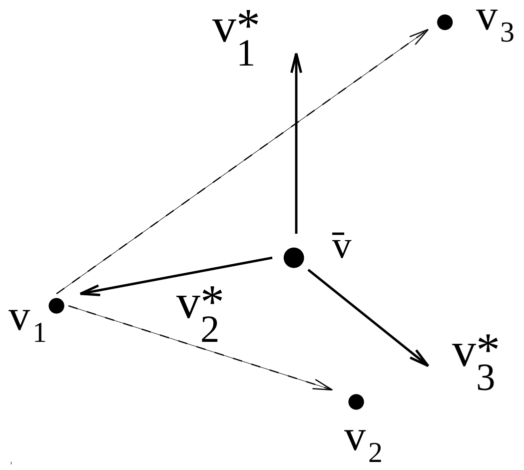

Figura 4.2: Os três vetores ortogonais entre si calculados a partir de três pontos.

Poderíamos considerar transformações mais gerais, incluindo por exemplo escalonamentos com fatores diferentes em cada eixo. Uma aplicação seria por exemplo o reconhecimento de caixas (paralelepípedos) de dimensões arbitrárias [23].O algoritmo seria semelhante porém um pouco mais complexo, usando quatro vértices, ao invés de três.

4.2.2 Enumeração das posições

Uma vez que apenas alguns dos vértices do modelo estarão visíveis na imagem, precisamos considerar todas as triplas (v_1, v_2, v_3) de vértices do modelo, e tentar casá-las com todas as triplas (u_1, u_2, u_3) de vértices da imagem. Em cada instância calculamos a matriz de transformação e posicionamos o modelo.

4.2.3 Avaliação de um posicionamento

A avaliação de um posicionamento (isto é, de uma matriz M calculada como descrito acima) leva em conta os seguintes critérios: (1) o fator de escalonamento α; (2) o número de vértices visíveis de V que, depois de mapeados, coincidem aproximadamente com vértices de U; e (3) a distância entre esses dois conjuntos de vértices.

Restrições sobre o fator de escalonamento

Podemos ter em uma cena vários objetos de tamanhos diferentes. Como estamos interessados em achar um determinado objeto de dimensões previamente definidas, calculamos a fator de escalonamento entre os dados e o modelo. Se este fator estiver além de um intervalo de valores, tendo a unidade como o centro, descartamos o posicionamento em questão.

Desta maneira não só eliminamos posicionamentos de outros objetos de outras dimensões como também posicionamentos falsos resultantes de ruídos das etapas anteriores.

Número mínimo de coincidências

Para cada posicionamento M, determinamos o conjunto V' de vértices do modelo que mapeados por M, deveriam ser visíveis por ambas as câmeras.

Para cada vértice v de V', determinamos o vértice u_v de U que está mais próximo do vértice vM do modelo posicionado. Se a distância $|vM - u_v|$ for menor que δ_{mod} (um parâmetro do algoritmo), consideramos que u_v é a manifestação nos dados do vértice v do modelo, e dizemos que v foi *encontrado* em U.

O posicionamento M só é considerado aceitável se pelo menos três vértices do modelo forem encontrados em U.

Medida de discrepância

Para todo posicionamento M que satisfaz as condições (1) e (2) acima, calculamos uma medida de discrepância $d(M)$ pela fórmula

$$d(M) = \sqrt{\frac{1}{|V'|} \sum_{v \in V'} |vM - u_v|^2} \tag{4.2}$$

Isto é, $d(M)$ é a distância media quadrática entre os vértices visíveis do modelo posicionado e os pontos de U mais próximos a eles. (Essas distâncias são calculadas no sistema de coordenadas isotrópicas).

4.3 Análise de complexidade

Na nossa implementação, a geração do conjunto de vértices observados U tem custo assintótico $\Theta(m^2)$, onde m é o número de segmentos devolvidos pela etapa de segmentação. Teoricamente, o número $k = |U|$ de vértices gerados pode ser $\Theta(m^2)$ no pior caso, pois cada par de segmentos pode dar origem a um vértice. Entretanto, em cenas típicas o número de vértices raramente ultrapassa $2m$.

Na nossa implementação, o processo de casamento tem custo muito alto, $\Theta(k^4 r^4)$ onde r é o número de vértices do modelo.

Convém salientar novamente, que na nossa implementação, não nos preocupamos com a eficiência dos algoritmos utilizados. Trata-se de um protótipo e o objetivo principal é a experimentação e comprovação (ou não) do princípio apresentado. Sendo o princípio aprovado, os algoritmos envolvidos devem ser pesquisados e otimizados afim de atingir a eficiência desejada geralmente determinada pela aplicação em si.

Capítulo 5

Testes e resultados

Apresentamos neste capítulo alguns testes realizados com imagens sintéticas e reais. Apresentamos os testes envolvendo todos as etapas descritas nos capítulos anteriores.

5.1 Testes com imagens sintéticas

A maioria dos pares de imagens estereoscópicas que usamos para testar o algoritmo foram gerados sinteticamente, com um programa simples de traçado de raios [13]. Com este programa podemos simular a aquisição de duas imagens de uma mesma cena a partir de pontos de observação diferentes. Desta forma podemos obter duas imagens perfeitamente deslocadas entre si de modo que as linhas de varredura das imagens coincidam com as linhas epipolares, simulando um perfeito deslocamento entre duas câmeras. Evitamos assim a necessidade da calibração das câmeras e retificação nas imagens.

As cenas utilizadas consistiam de um pequeno número de cubos ou paralelepípedos, contra um fundo uniforme com uma fonte de iluminação pontual e luz ambiente difusa.

Nas imagens geradas, as faces e arestas dos objetos eram uniformes e perfeitas. Para tornar os testes um pouco mais realistas, procuramos submeter as imagens sintéticas (512×512 pixels) a um pré-processamento que simula alguns dos defeitos esperados em imagens reais. Em primeiro lugar, "borramos" ligeiramente cada imagem, substituindo cada pixel por uma média ponderada do mesmo com seus 8 vizinhos. Em seguida, introduzimos "ruído" artificial, somando a cada pixel, originalmente entre 0 e 255, um inteiro aleatório pseudo-gaussiano entre -6 e $+6$ e com desvio padrão 1.7. Finalmente, procuramos simular o efeito de uma filtragem (imperfeita) do ruído, aplicando a cada linha de varredura um filtro de mediana unidimensional [18]. Este filtro substitui cada pixel pela mediana dele e de seus $2K-2$ vizinhos mais próximos na mesma linha, onde K é a largura mínima de uma carreira. (Veja o capítulo 2).

Note que tais imagens não possuem texturas, fundos complexos, ou objetos estranhos.

Entretanto, elas possuem sombras, oclusões, arestas borradas, e ruído; portanto, elas não são, necessariamente, triviais de se analisar.

5.1.1 Parâmetros gerais

Apresentamos aqui os parâmetros que permaneceram constantes em todos os testes com imagens sintéticas.

Na etapa de detecção e casamento de descontinuidades utilizamos os seguintes parâmetros:

Nível de ruído $\delta = 2$

Largura mínima de carreiras $K = 4$

Largura máxima para um degrau $W = 15$

Número máximo de degraus por linha de varredura $D = 10$

Coeficientes de discrepância $\alpha = 0.9$, $\beta = 0.1$.

Na etapa de segmentação utilizamos os seguintes parâmetros de distância:

Geração inicial de candidatos: $|p - p'| \leq \delta_{ger} = 12\epsilon$

Amadurecimento: $\delta_{amd} = 4$

Crescimento: $\delta_{crs} = 16$

Na etapa seguinte de casamento com o modelo, utilizamos os seguintes parâmetros na fase de união dos extremos em vértices:

Distância entre retas suporte $\leq \rho\epsilon = 4\epsilon$

Distância entre extremidades do mesmo grupo $\leq \tau\epsilon = 7\epsilon$.

5.1.2 Teste 1

No primeiro teste usamos uma cena contendo dois paralelepípedos com arestas de $40, 40, 50$ e $10, 20, 20$ mm respectivamente, com os seguintes parâmetros:

Separação entre as câmeras $b = 22.4$ rm

Distância focal $f = 441.7$ *pixels*

A figura 5.1 mostra as duas imagens sintéticas após o pré-processamento citado acima.

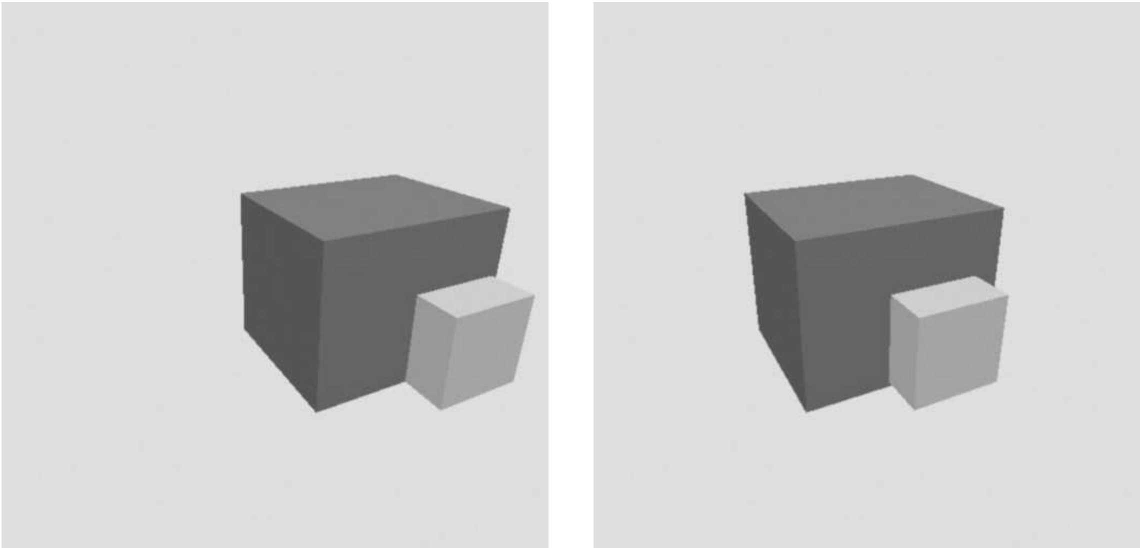

Figura 5.1: Um par estéreo sintético com dois paralelepípedos.

A figura 5.2 mostra as posições dos degraus determinados em cada imagem pelo algoritmo de detecção de descontinuidades (capítulo 2).

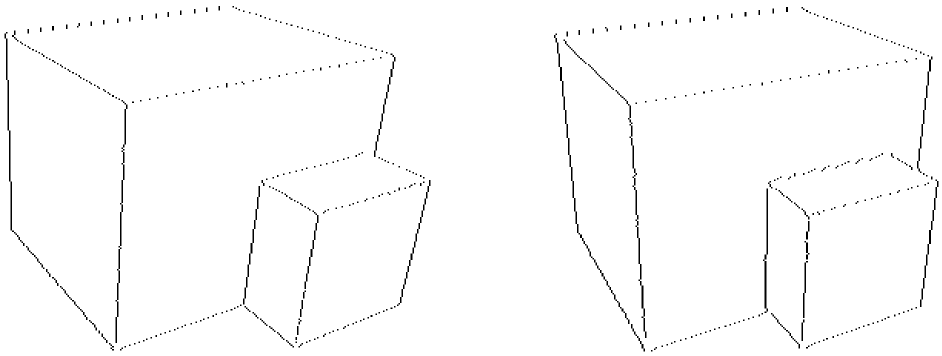

Figura 5.2: Degraus de intensidade detectados nas imagens da figura 5.1.

Podemos notar que a detecção dos degraus ficou praticamente perfeita nas duas imagens. Nota-se entretanto, na imagem da direita, a pequena falha na extremidade inferior de uma das arestas verticais do cubo maior (a mais próxima da câmera). Esta falha é devida à restrição de largura mínima de carreiras.

Na etapa de casamento de degraus e cálculo de profundidade, utilizamos os seguintes parâmetros:

$z_{min} = 0\ mm$

$z_{max} = 989.4\ mm\ (x'_l - x'_r \geq 10)$

$\kappa_{min} = 1.0$

$\kappa_{max} = 1.2$

Os emparelhamentos mais prováveis desses degraus, escolhidos pelo algoritmo de casamento dos degraus (capítulo 2), definiram o conjunto de 1041 pontos no $\mathbf{R}^3$ mostrado na figura 5.3.

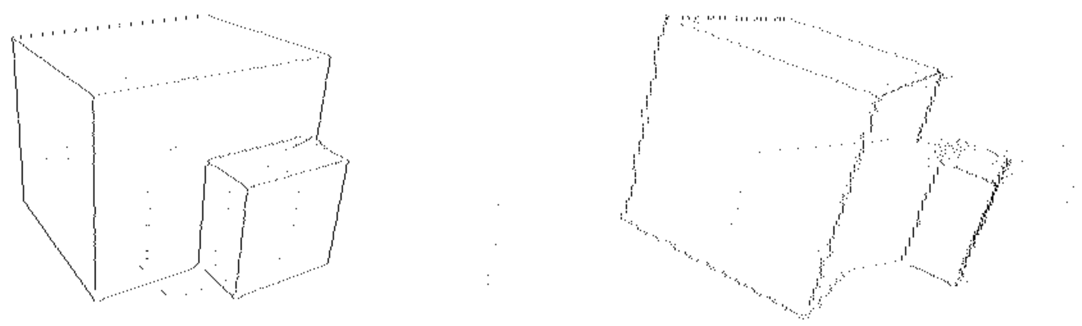

Figura 5.3: Pontos no $\mathbf{R}^3$ resultantes do casamento dos degraus da figura 5.2.

A imagem da esquerda mostra a projeção em perspectiva frontal do conjunto de pontos, calculada para um observador situado ao ponto médio entre as câmeras. A imagem da direita mostra outra projeção em perspectiva dos mesmos pontos, calculada a partir de outro ponto de vista. Podemos notar nesta figura que a maioria dos casamentos resultou em pontos de arestas reais do objeto, com profundidades substancialmente corretas.

Alguns dos casamentos falsos decorrem da tentativa de casar degraus do paralelepípedo maior com degraus do menor. Esses casamentos falsos serão eventualmente descartados nas etapas seguintes. Notamos também uma certa deformação na extremidade superior direita do paralelepípedo menor devido à oclusão de uma parte da aresta do paralelepípedo maior pelo menor.

Na figura 5.4 mostramos esse mesmos pontos no sistemas de coordenadas isotrópicas.

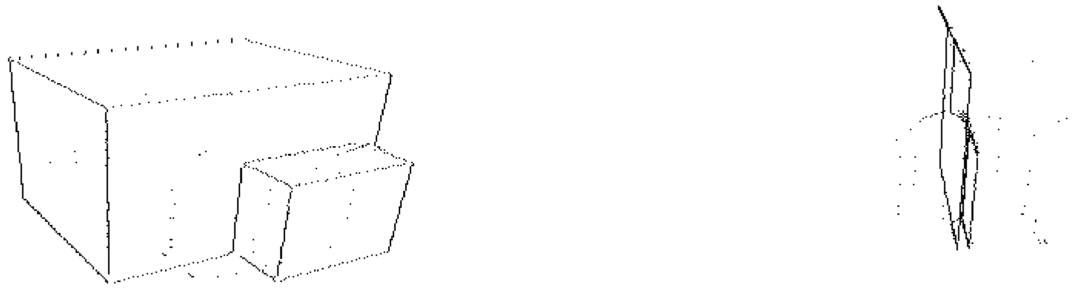

Figura 5.4: Pontos no $\mathbf{R}^3$, em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

As figuras 5.5 e 5.6 mostram o resultado da etapa de segmentação aplicada aos pontos acima (capítulo 3). As projeções são as mesmas da figura 5.3. Adotamos os seguintes valores para parâmetros:

Erro na posição dos pontos $\epsilon = 0.6$

Número mínimo de pontos para crescimento de candidatos $M = 4$

O algoritmo de segmentação encontrou 21 segmentos completos, dos quais 18 (com 1009 pontos no total) correspondem a arestas reais do objeto, e 3 (com 31 pontos no total) são arestas falsas devidas a casamentos incorretos, ruído, quantização, erros de arredondamento, etc.

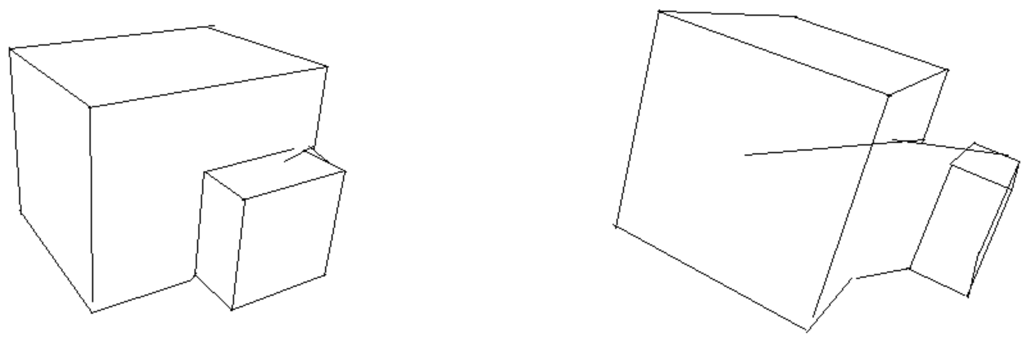

Figura 5.5: Segmentos completos determinados a partir dos pontos da figura 5.4.

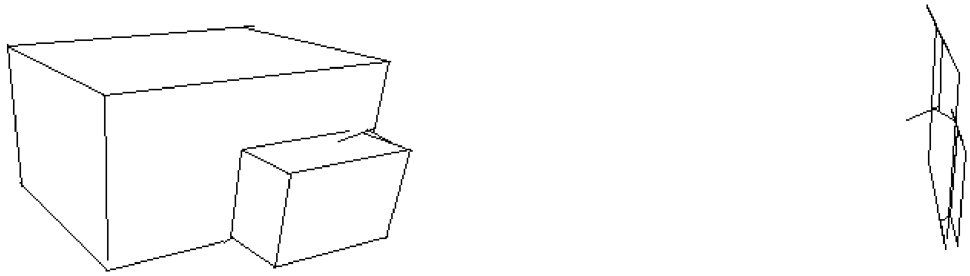

Figura 5.6: Segmentos da figura 5.5, em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Notamos que todas as arestas reais dos paralelepípedos estão presentes no final da etapa de segmentação.

A figura 5.7 mostra o resultado da união de extremidades dos segmentos em vértices possíveis dos paralelepípedos (capítulo 4). Nota-se o aparecimento de vértices extrapolados, como por exemplo P (resultado da extrapolação dos segmentos A e B). Nota-se também que uma extremidade (como Q, na figura) que não está suficientemente próxima do respectivo vértice extrapolado (R), sobrevive sozinha no conjunto final de vértices U.

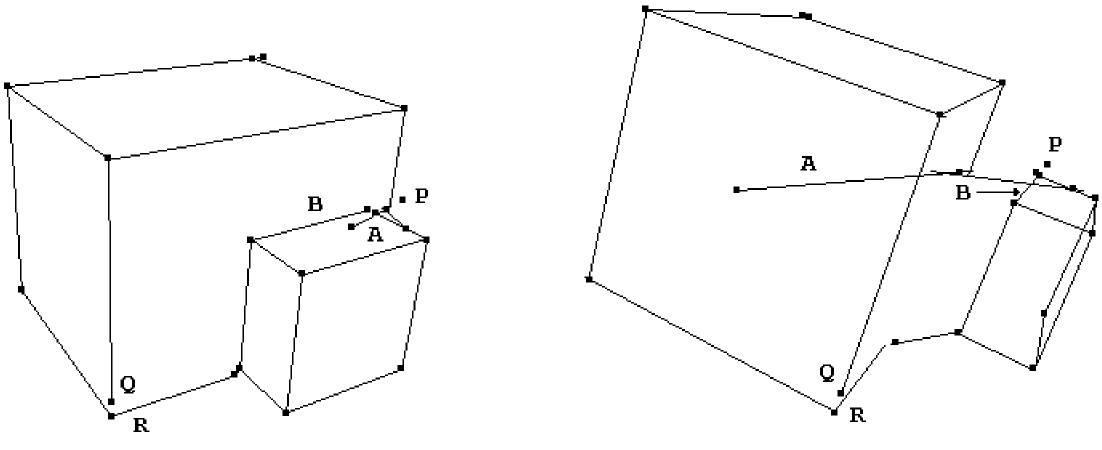

Figura 5.7: Vértices calculados a partir dos segmentos da figura 5.6.

Na figura 5.8 mostramos os resultados do casamento dos vértices acima com os modelos dos paralelepípedos.

Nesta etapa a distância máxima permitida entre os vértices do modelo mapeado e os vértices dos dados $\delta_{mod} = \epsilon = 0.6$.

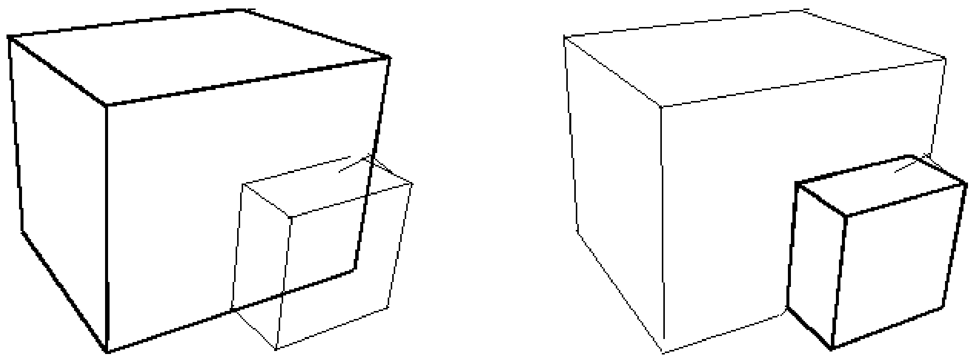

Figura 5.8: Os melhores casamentos dos vértices com os modelos.

A figura mostra os segmentos resultantes da fase de segmentação (linhas finas) e as arestas do paralelepípedo posicionado (linhas grossas). Podemos notar que foi possível ajustar os modelos precisamente sobre os segmentos. Estes são os posicionamentos que apresentam as menores discrepâncias (fórmula 4.2, seção 4.2.3)

Os segmentos falsos foram ignorados, pois todos os posicionamentos que usaram seus vértices tiveram discrepâncias altas.

5.1.3 Teste 2

Nas figuras 5.9–5.17 apresentamos os resultados realizados com uma cena sintética constando de 3 cubos com arestas de 16, 20 e 30 mm cada, com os seguintes parâmetros:

Separação entre as câmeras $b = 60.0\ mm$

Distância focal $f = 441.7\ pixels$

Observa-se que esta cena contém várias características problemáticas, em particular sombras e casamentos ambíguos (devidos ao fato dos três cubos terem a mesma cor e iluminação).

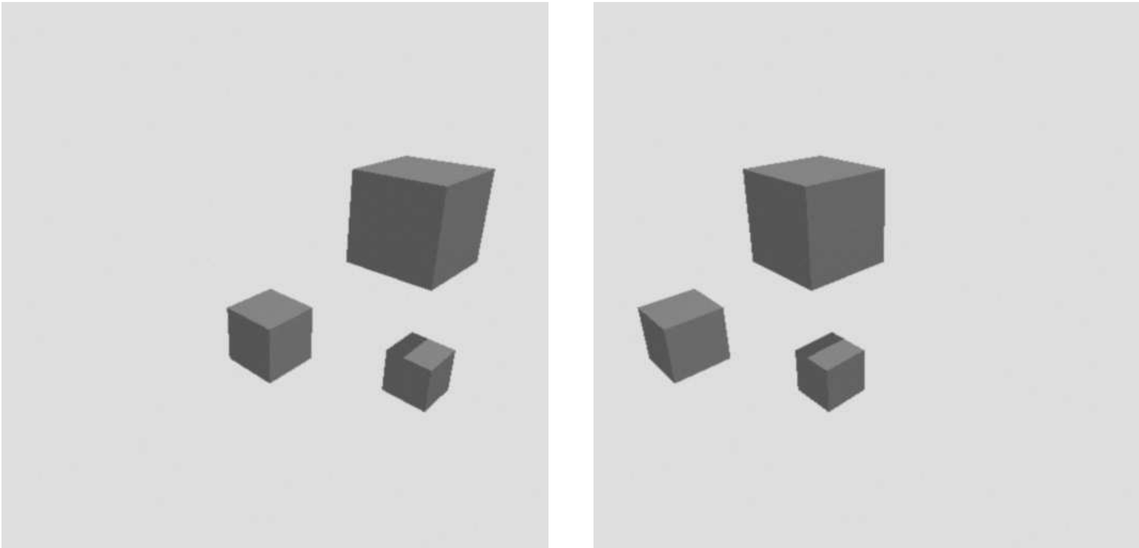

Figura 5.9: Um par estéreo sintético com 3 cubos.

A figura 5.10 mostra as posições dos degraus determinados em cada imagem.

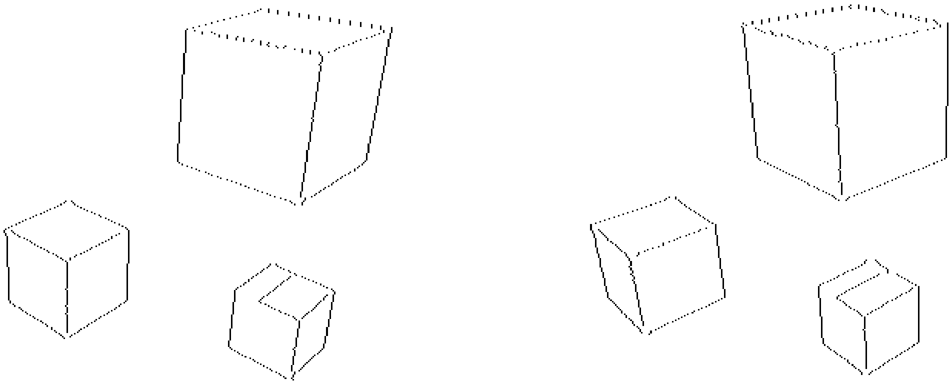

Figura 5.10: Degraus de intensidade detectados nas imagens da figura 5.9.

A detecção dos degraus ficou praticamente perfeita nas duas imagens, sendo que a borda da sombra do cubo maior sobre o cubo menor também foi detectada como um degrau. Essa sombra impossibilitou a detecção de parte de uma aresta do cubo menor.

Nas figuras 5.11 e 5.12 mostramos os 998 pontos obtidos pela etapa de casamento. Utilizamos os seguintes parâmetros:

$z_{min} = 0\ mm$

$z_{max} = 2650.2\ mm\ (x'_l - x'_r \geq 10)$

$\kappa_{min} = 1.0$

$\kappa_{max} = 1.2$

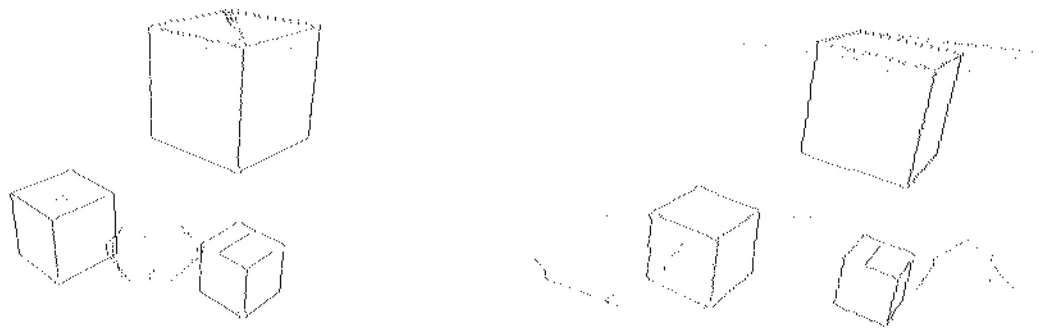

Figura 5.11: Pontos no $\mathbf{R}^3$ resultantes do casamento dos degraus da figura 5.10.

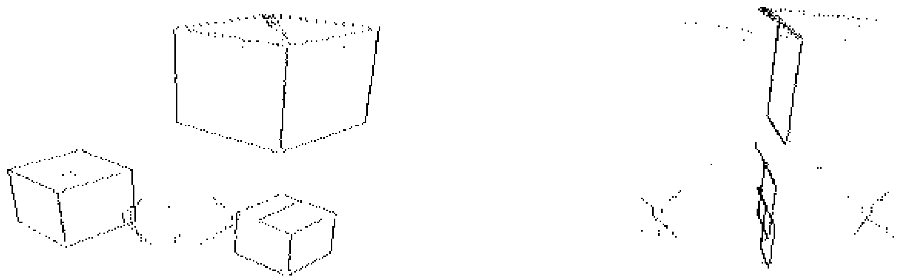

Figura 5.12: Pontos da figura 5.11 em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Nas figuras 5.13 e 5.14 mostramos o resultado da etapa de segmentação. Adotamos os seguintes valores para parâmetros:

Erro na posição dos pixels $\epsilon = 1.2$

Número mínimo de pontos para crescimento de candidatos $M = 9$

Nesta cena, o algoritmo de segmentação encontrou 31 segmentos completos, dos quais 27 (com 1005 pontos no total) correspondem a arestas reais do objeto, 3 (com 52 pontos no total) são falsos, resultantes de falsos casamentos, e um segmento (com 18 pontos) que corresponde à borda da sombra projetada sobre o cubo menor.

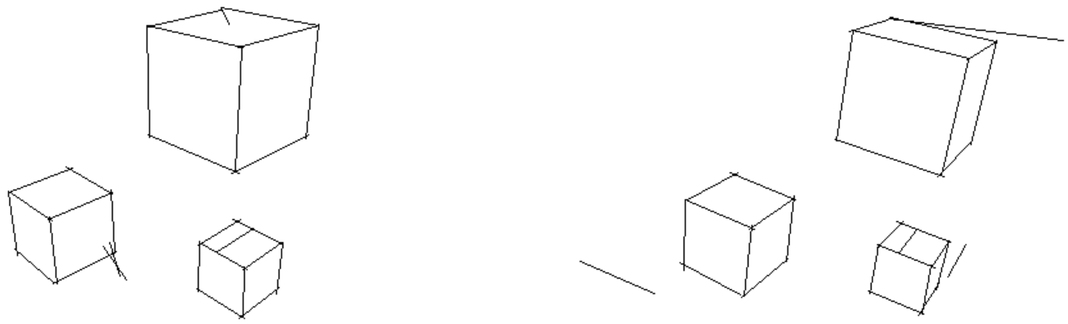

Figura 5.13: Segmentos completos determinados a partir dos pontos da figura 5.12.

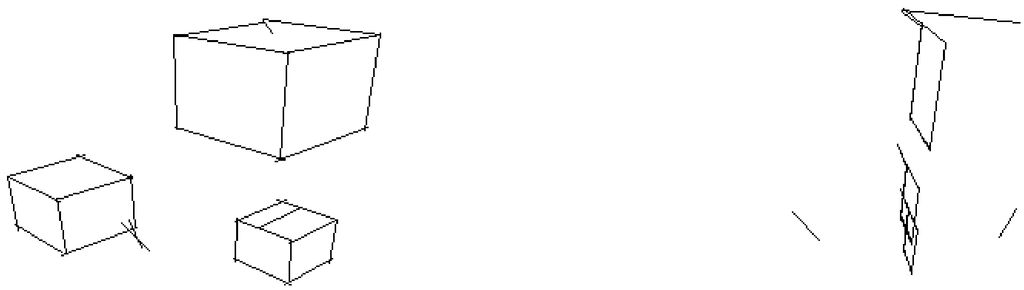

Figura 5.14: Segmentos da figura 5.13, em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Podemos notar que a falha devido a sombra do cubo não prejudicou a computação dos segmentos.

A etapa de geração dos vértices produziu o conjunto de pontos ilustrado na figura 5.15.

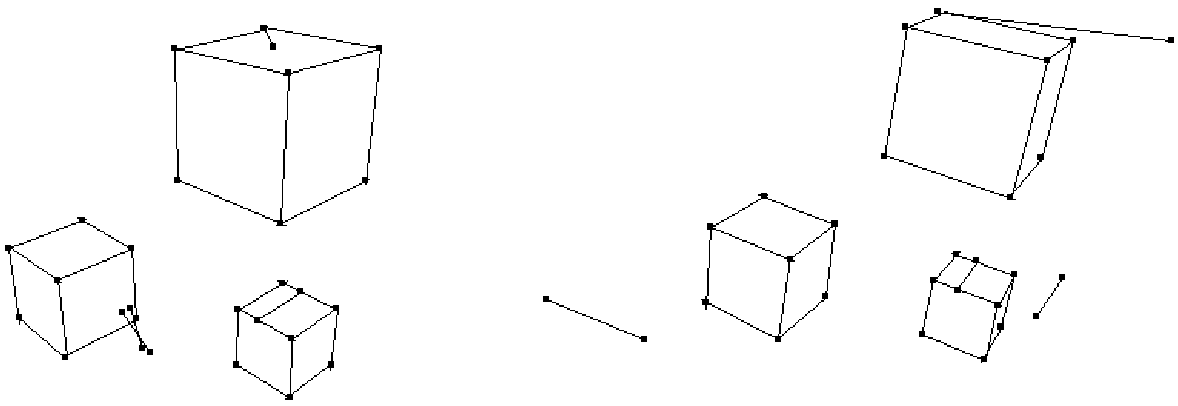

Figura 5.15: Vértices extraídos dos segmentos da figura 5.14.

Podemos notar que todos os vértices visíveis dos cubos foram encontrados. Os extremos do segmento correspondente à sombra foram mantidos e formam cada um, isoladamente, um vértice.

Nas figuras seguintes (5.16–5.17) apresentamos os resultados do casamento dos vértices acima com os modelos de cada cubo. Nesta etapa, a distância máxima permitida entre os vértices do modelo mapeado e os vértices dos dados é $\delta_{mod} = \epsilon = 1.2$.

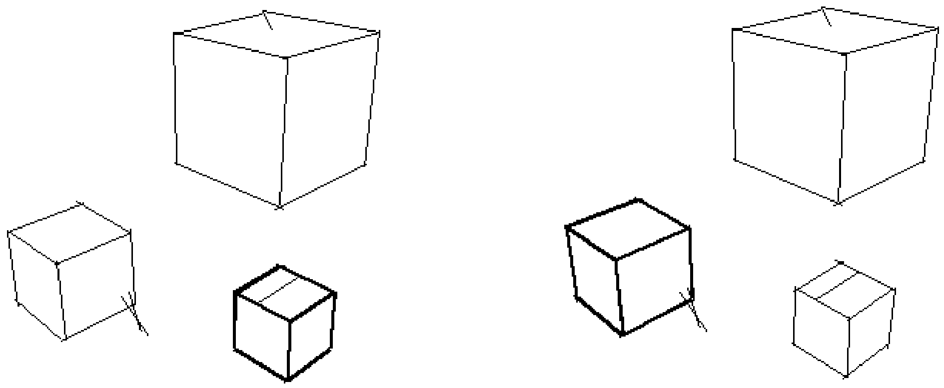

Figura 5.16: Os melhores casamentos dos vértices da figura 5.15 com os modelos 1 e 2.

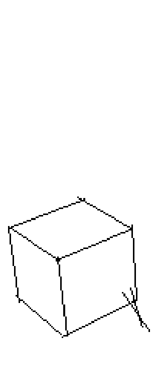

Figura 5.17: Melhor casamento dos vértices com o modelo 3.

Podemos notar que foi possível ajustar, precisamente os modelos sobre os cubos. Para cada casamento utilizamos um modelo diferente, correspondente ao cubo casado. Neste teste verificou-se a necessidade da condição no cálculo do fator de escala da matriz de escalonamento, que permite um fator de escala próximo da unidade. Temos vários objetos com mesmo formato e com dimensões diferentes, e se esta condição não for utilizada poderemos não identificar na cena o cubo de dimensões desejadas.

5.1.4 Teste 3

Nas figuras 5.18–5.27 apresentamos os resultados realizados com uma cena sintética consistindo de 2 cubos de mesma cor, com arestas de 20 e 30 mm cada, com os seguintes parâmetros:

Separação entre as câmeras $b = 60.0$ mm

Distância focal $f = 441.7$ *pixels*

Observa-se na imagem da direita que um pedaço do cubo menor está escondido pelo maior. Observa-se também a presença de casamentos ambíguos devido ao fato dos dois cubos terem a mesma cor e iluminação e estarem posicionados lado a lado e, portanto, ocupando as mesmas linhas de varredura.

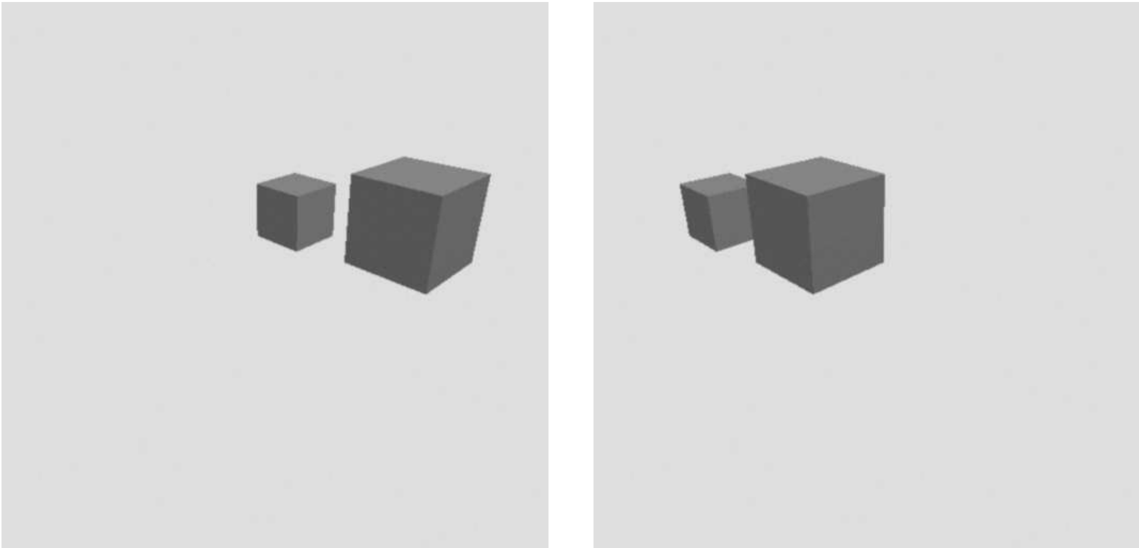

Figura 5.18: Um par estéreo sintético com 2 cubos.

A figura 5.19 mostra as posições dos degraus determinados em cada imagem (os rótulos são usados nas discussões a seguir).

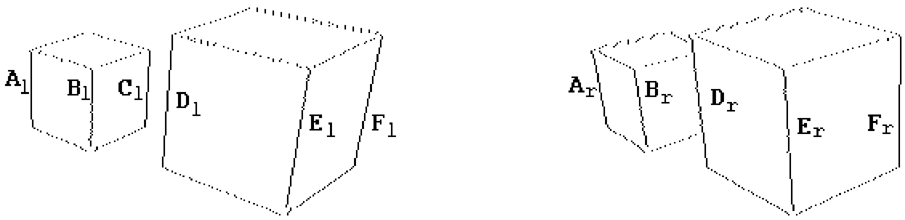

Figura 5.19: Degraus de intensidade detectados nas imagens da figura 5.18.

As figuras 5.20 e 5.21 mostram os 686 pontos resultantes da etapa de casamento de degraus. Utilizamos os seguintes parâmetros, nesta etapa:

$z_{min} = 0 \ mm$

$z_{max} = 2650.2 \ mm \ (x'_l - x'_r \geq 10)$

$\kappa_{min} = 1.0 \ , \ \kappa_{max} = 2.0$

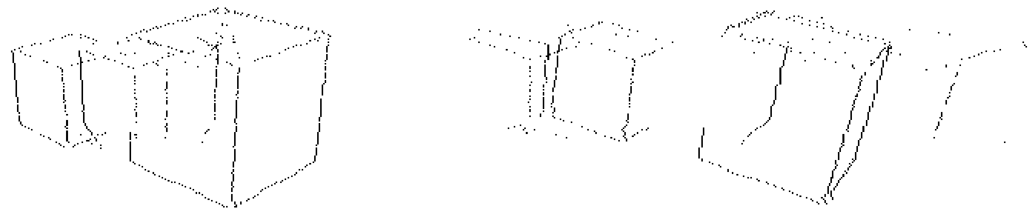

Figura 5.20: Pontos no $\mathbf{R}^3$ resultantes do casamento dos degraus da figura 5.19.

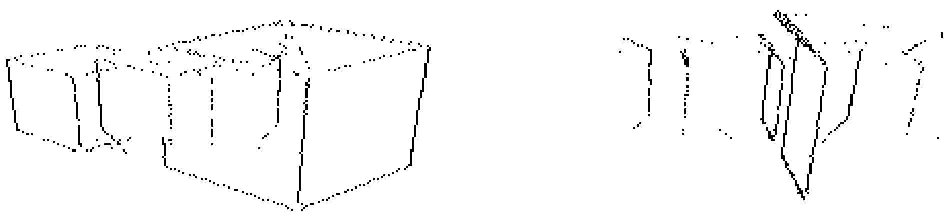

Figura 5.21: Pontos da figura 5.20, em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Este exemplo ilustra a função do parâmetro κ_{min}. Podemos notar que muito dos degraus pertencentes a aresta D do cubo maior não foram casados corretamente. Isto ocorre porque parte da aresta D está projetada contra fundos de cor diferente nas duas imagens e, com isso, o casamento correto (D_l, D_r) tem discrepância (dada pela fórmula 2.5 da seção 2.5) ligeiramente maior que os casamentos incorretos (B_l, E_r), (C_l, F_r), (D_l, A_r), (E_r, B_l); como o número mínimo de casamentos considerados neste caso é igual a 6 (maior número de degraus detectados entre as linhas de varredura da esquerda e da direita), o casamento correto é sempre deixado de fora.

A solução para este problema é aumentar o número mínimo de casamentos devolvidos por linha de varredura. Por exemplo, adotando $\kappa_{min} = 2.0$ obtemos os casamentos mostrados nas figuras 5.22 e 5.23.

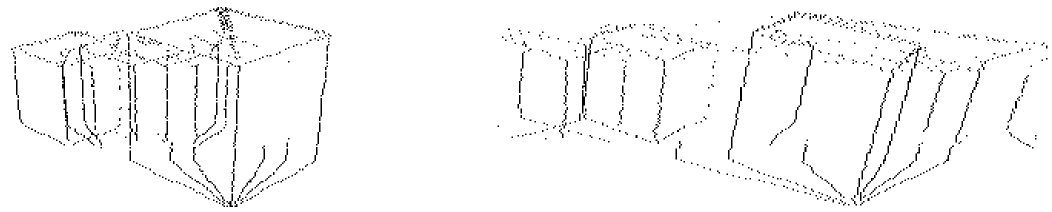

Figura 5.22: Pontos no espaço determinados pelo casamento dos degraus para $\kappa_{min} = 2.0$. (Dois pontos de observação)

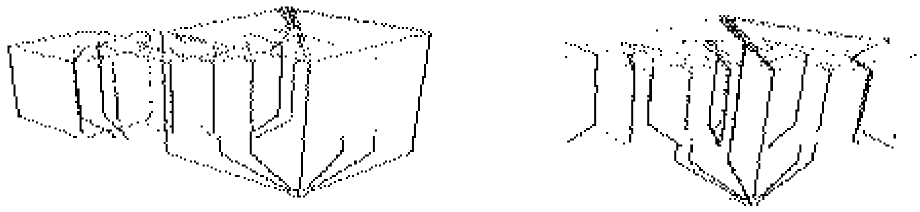

Figura 5.23: Pontos no espaço, determinados pelo casamento dos degraus para $\kappa_{min} = 2.0$. Sistema de coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Podemos notar agora que o número de casamentos falsos aumentou, mas em compensação os casamentos corretos estão quase todos presentes.

Nas figuras a seguir apresentamos o resultado da segmentação , união de extremidades e casamento com modelo, para este conjunto de pontos.

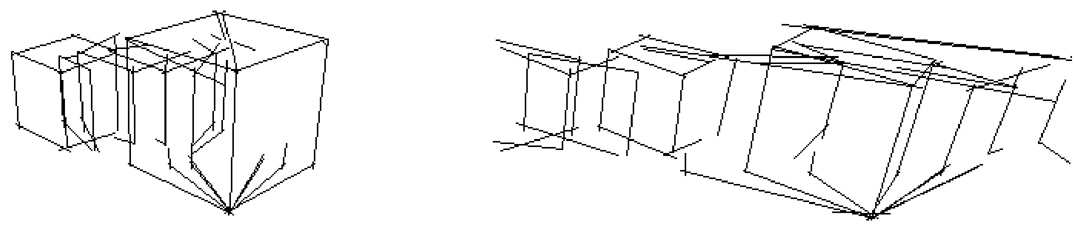

Figura 5.24: Segmentos completos determinados a partir dos pontos da figura 5.23.

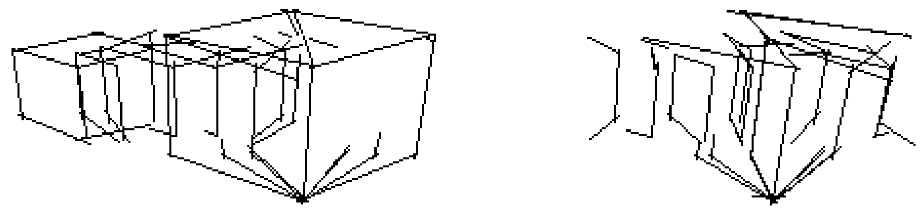

Figura 5.25: Segmentos completos da figura 5.24 em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

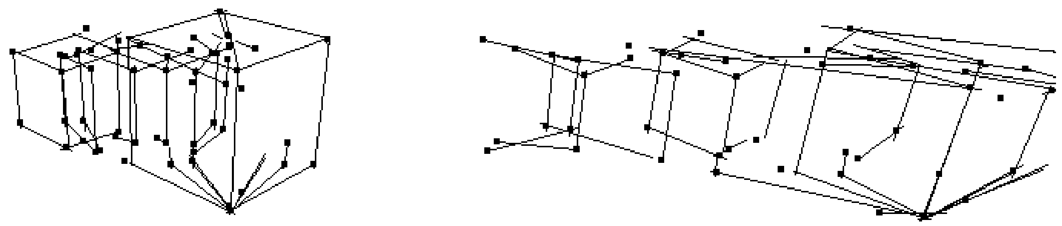

Figura 5.26: Vértices extraídos dos segmentos da figura 5.25.

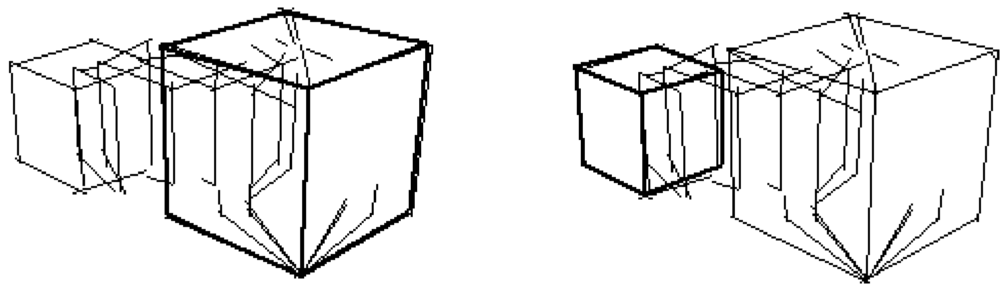

Figura 5.27: Melhores posicionamentos dos dois modelos.

Nestas etapas, utilizamos os seguintes parâmetros:

Erro na posição dos pontos $\epsilon = 1.2$

Número mínimo de pontos para crescimento de candidatos $M = 4$

Distância máxima entre os vértices do modelo mapeado e os vértices dos dados $\delta_{mod} = \epsilon = 1.2$

Podemos notar que foi possível encontrar o posicionamento correto dos modelos sobre os cubos, apesar do grande número de falsos segmentos presentes. Nesta última etapa eles foram todos eliminados.

5.2 Teste com imagens reais

Nesta seção apresentamos um teste realizado com imagens reais. Este teste tem por objetivo verificar o desempenho dos algoritmos aqui apresentados quando submetidos a condições reais de trabalho.

As imagens foram obtidas através de uma câmera CCD no Laboratório de Visão da Escola Federal de Engenharia de Itajubá – EFEI. Utilizamos somente uma câmera. Para obter as imagens deslocadas entre si, ao invés de deslocarmos a câmera, deslocamos os objetos mantendo a câmera fixa. Procuramos deslocar os objetos paralelamente ao eixo horizontal da câmera, de modo que as linhas epipolares das imagens obtidas coincidissem com as linhas de varredura.

Na realidade, mesmo tomando todos os cuidados, o deslocamento não foi perfeitamente paralelo ao eixo, e portanto, não conseguimos obter imagens com as características desejadas. Tivemos então que efetuar uma calibração manual aproximada das imagens baseada nas dimensões conhecidas dos objetos da cena.

A calibração consistiu em aplicar uma pequena redução vertical (aproximadamente 0.8%) seguida de uma pequena translação vertical (aproximadamente 4 pixels) na imagem da esquerda. Convém ressaltar que a calibração correta provavelmente exigiria uma transformação projetiva mais complexa.

Neste teste usamos uma cena contendo dois objetos esculpidos em madeira contra um fundo uniforme: um paralelepípedo com arestas de $40, 40$ e 25 mm, e um prisma triangular com base retangular com arestas de 40 e 25 mm e faces laterais formadas por triângulos eqüiláteros com arestas de 40 mm.

Os parâmetros óticos foram os seguintes:

Separação entre as câmeras $b = 100.0$ mm

Distância focal $f = 1335.0$ $pixels$

A figura 5.28 mostra as duas imagens reais após a retificação citada acima.

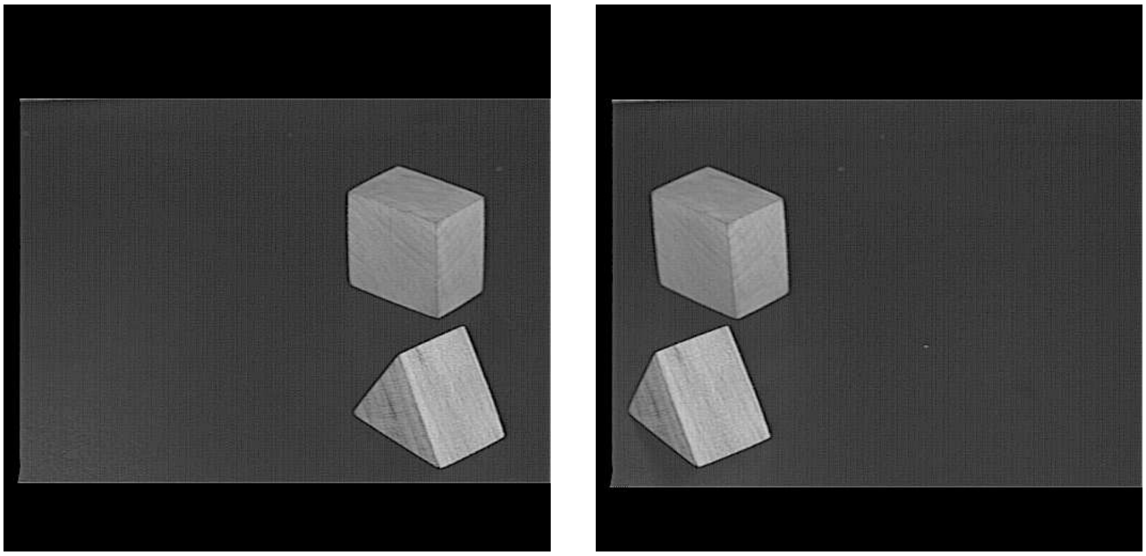

Figura 5.28: Um par estéreo real com dois objetos.

A figura 5.29 mostra as posições dos degraus determinados em cada imagem pelo algoritmo de detecção de descontinuidades. Devido a natureza das imagens, utilizamos os seguintes valores dos parâmetros na etapa de detecção de degraus:

Nível de ruído $\delta = 4$

Largura mínima de carreiras $K = 5$

Largura máxima para um degrau $W = 15$

Número máximo de degraus por linha de varredura $D = 10$

Coeficientes de discrepância $\alpha = 0.9$, $\beta = 0.1$.

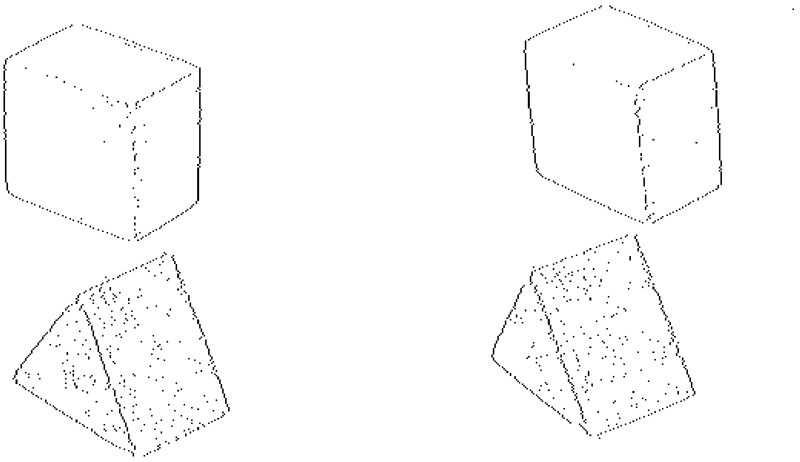

Figura 5.29: Degraus de intensidade detectados nas imagens da figura 5.28.

Podemos notar que a maioria dos degraus detectados pertence as arestas reais dos objetos da cena. Uma aresta do paralelepípedo não pôde ser detectada devido à pequena diferença entre os níveis das duas faces adjacentes. Nas faces do prisma o algoritmo detectou vários degraus que não correspondem a arestas do objeto, devidos à textura das faces.

Nas figuras 5.30 e 5.31 mostramos os 1383 pontos obtidos pela etapa de casamento de descontinuidades. Nesta etapa utilizamos os seguintes parâmetros:

$z_{min} = 0\ mm$

$z_{max} = 13350.0\ mm\ (x'_l - x'_r \geq 10.0)$

$\kappa_{min} = 1.0$

$\kappa_{max} = 1.2$

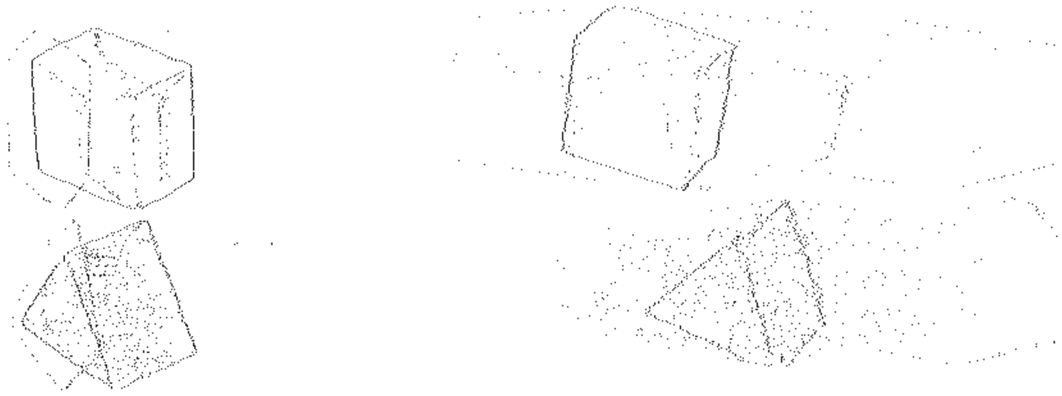

Figura 5.30: Pontos no $\mathbf{R}^3$ resultantes do casamento dos degraus da figura 5.29.

Figura 5.31: Pontos da figura 5.30, em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Podemos notar a presença de muitos casamentos falsos; porém as arestas reais dos objetos estão bem definidas.

Nas figuras 5.32 e 5.33 mostramos o resultado da etapa de segmentação. Adotamos os seguintes valores para parâmetros:

Erro na posição dos pontos $\epsilon = 0.8$

Número mínimo de pontos para crescimento de candidatos $M = 8$

Distância máxima para candidatos iniciais: $|p - p'| \leq \delta_{ger} = 12\epsilon$

Distância máxima na fase de amadurecimento: $\delta_{amd} = 4$

Distância máxima na fase de crescimento: $\delta_{crs} = 16$

Com estes dados o algoritmo de segmentação encontrou 31 segmentos completos, dos quais 14 (com 674 pontos no total) correspondem a arestas reais do objeto, e 17 (com 284 pontos no total) são arestas falsas devidas a casamentos incorretos, ruído, quantização, erros de arredondamento, etc.

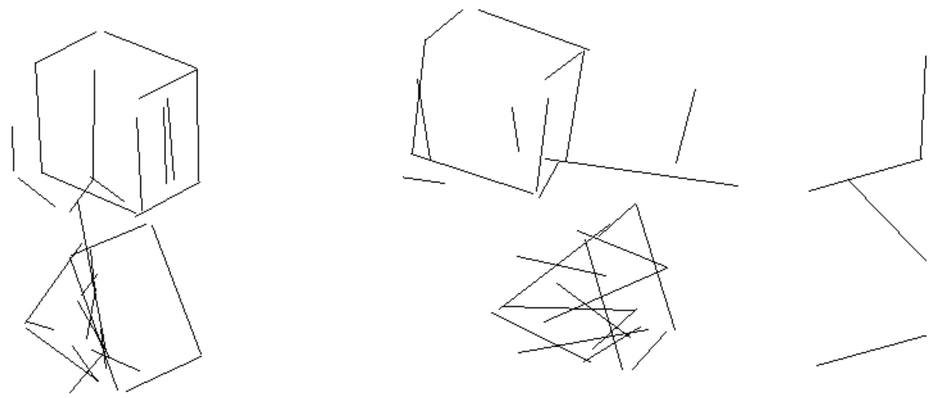

Figura 5.32: Segmentos completos determinados a partir dos pontos da figura 5.31.

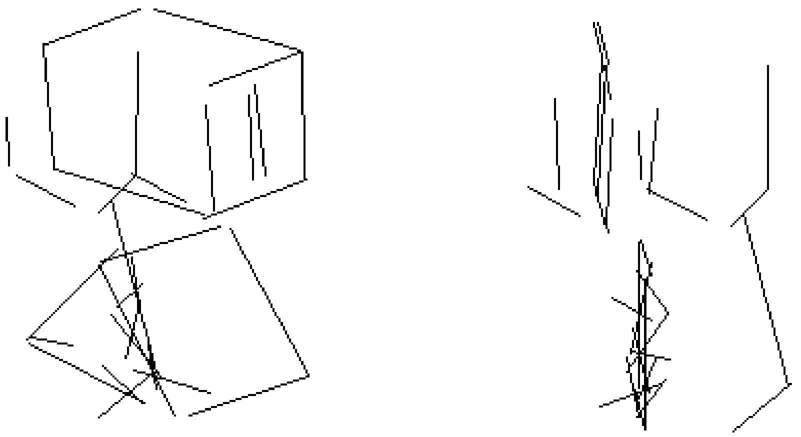

Figura 5.33: Segmentos da figura 5.32 em coordenadas isotrópicas: eixos x''–y'' (esq.) e z''–y'' (dir.).

Notamos que praticamente todas as arestas reais dos objetos estão presentes no conjunto de segmentos obtidos. Temos também vários segmentos falsos; mas, como veremos mais adiante, que esses segmentos não prejudicaram a etapa de localização do modelo.

A figura 5.34 mostra o conjunto de vértices extraídos destes segmentos. Utilizamos os seguintes parâmetros nesta etapa:

Distância máxima entre retas suporte $\rho\epsilon = 4\epsilon$

Distância máxima entre extremidades do mesmo grupo $\tau\epsilon = 7\epsilon$.

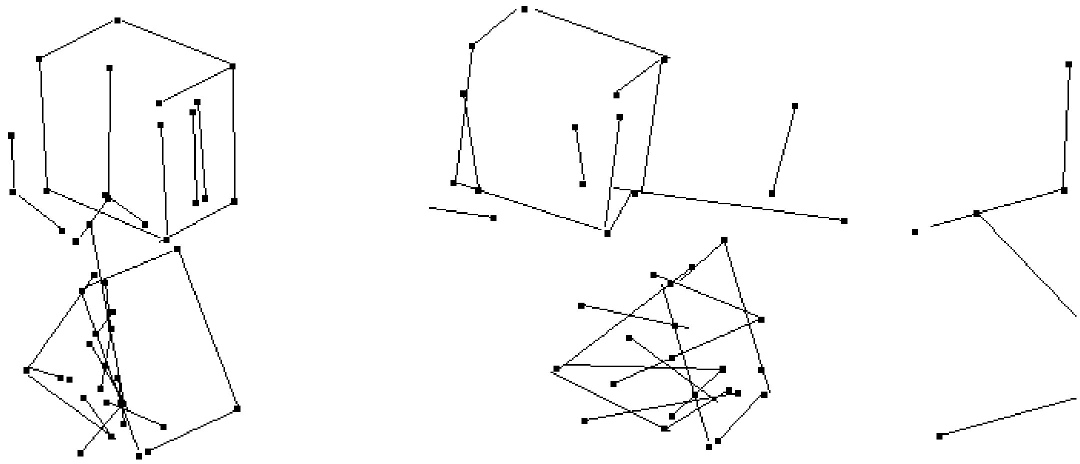

Figura 5.34: Vértices extraídos dos segmentos da figura 5.33.

Na figura 5.35 mostramos os resultados do casamento com os modelos do paralelepípedo e do prisma. Nesta etapa, adotamos para o parâmetro δ_{mod} (distância máxima entre vértices do modelo mapeado e vértices dos dados) o valor $10\epsilon = 8.0$

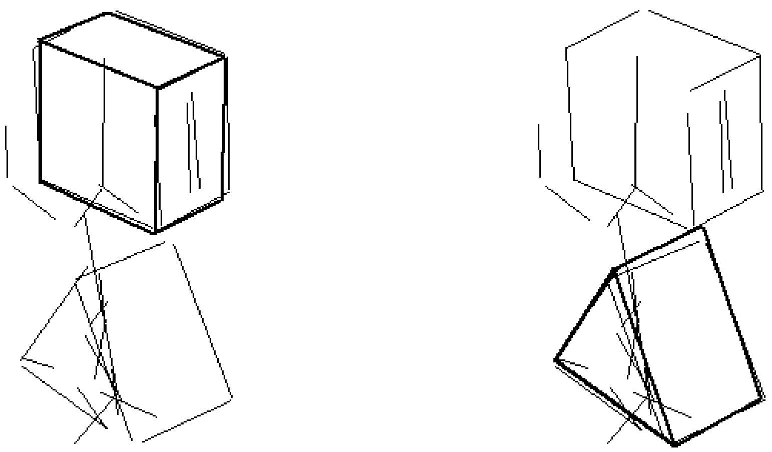

Figura 5.35: Casamento dos vértices com o modelo. Modelo do paralelepípedo (esq.) e do prisma triangular (dir.).

Podemos notar que mesmo no melhor posicionamento, as arestas do modelo não coincidem exatamente com os segmentos extraídos das imagens. Mesmo assim, o posicionamento com menor discrepância é uma aproximação muito boa da resposta ideal. As pequenas diferenças são quase que certamente devidas à calibração imperfeita das duas imagens.

Capítulo 6

Conclusões e trabalho futuro

Os testes apresentados na seção anterior sugerem que nossa metodologia (detecção unidimensional de descontinuidades, e segmentação de pontos no espaço) produz resultados comparáveis em qualidade com os da metodologia tradicional (em que o reconhecimento de segmentos é feito em cada imagem separadamente, antes da restituição).

Os algoritmos principais utilizados em cada etapa parecem ser originais, pelo menos dentro dos limites de nossa pesquisa bibliográfica. Na implementação atual, o tempo total de processamento é relativamente elevado, especialmente na etapa de localização do modelo. Entretanto, exceto por esta etapa, o custo assintótico é no pior caso proporcional ao número de pixels das imagens.

Ainda há muito a fazer nesta linha de pesquisa. Por exemplo, deve ser possível melhorar enormemente a eficiência dos algoritmos de segmentação ($\Theta(n^2)$) e localização ($\Theta(n^4)$) para $O(n \log n)$, ou mesmo $O(n)$ se usarmos estruturas de dados adequadas para armazenar os candidatos e localizar os pontos mais próximos aos mesmos.

Finalmente, seria necessário testar esta abordagem numa gama maior de imagens, possivelmente provindas de aplicações práticas.

Bibliografia

[1] Peter K. Allen. *Robotic Object Recognition Using Vision and Touch.* Kluwer, 1987.

[2] Nicholas Ayache. *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception.* Cambridge University Press, 1991.

[3] Dana H. Ballard and Christopher M. Brown. *Computer Vision.* Prentice-Hall, 1982.

[4] Stephen T. Barnard and Martin A. Fischler. Computational stereo. *ACM Computing Surveys*, 14(4):553–572, December 1982.

[5] James D. FOLEY and Andries Van DAM. *Fundamentals of Interactive Computer Graphics.* Addison Wesley, 1982.

[6] Ardeshir Goshtasby and William A. Gruver. Design of a single-lens stereo camera system. *Pattern Recognition*, 26(6):923–937, 1993.

[7] Min-Hong Han and Sangyong Rhee. Camara calibration for three-dimensional measurement. *Pattern Recognition*, 25(2):155–164, 1992.

[8] B. K. P. Horn. *Robot Vision.* The MIT Press and McGraw-Hill, 1986.

[9] Radu Houraud and Thomas Skordas. Structural matching for stereo vision. In *Proc. 9th International Conference on Pattern Recognition*, volume 1, pages 439–445, Rome, Italy, November 1988.

[10] John R. Jordan III and Alan C. Bovik. Using chromatic information in dense stereo correspondence. *Pattern Recognition*, 25(4):367–383, 1992.

[11] Behrooz Kamgar-Parsi and Roger D. Eastman. Calibration of a stereo system with small relative angles. *Computer Vison, Graphics and Image Processing*, 51:1–19, 1990.

[12] Gudrun J. Klinker. *A Physical Approach to Color Image Understanding.* PhD thesis, Carnegie Mellon University, May 1988.

[13] Roman Kuchkuda. An introduction to ray tracing. In *Proc. NATO Advanced Science Institute, Series F, vol. 40: Theorical Foundations of Computer Graphics and CAD*, pages 1039–1060, Lucca, Italy, July 1988. Springer-Verlag.

[14] Ruihuna Ma and Monique Thonnat. A robust and efficient stereo matching algorithm. Technical Report 1860, INRIA Institut National de Recherche en Informatique et en Automatique, 1993.

[15] Robert A. McLaughlin and Michael D. Alder. Syntactic pattern recognition of simple shapes. In *ANZIIS-93 Conference Proceedings*, Perth Australia, 1993.

[16] Robert A. McLaughlin and Michael D. Alder. Recognising cubes in images. *Pattern Recognition in Practice IV*, 1994.

[17] William M. Newman and Robert F. Sproull. *Principles of Interactive Computer Graphics.* Mc Graw-Hill, second edition, 1979.

[18] William K. Pratt. *Digital Image Processing.* Wiley, second edition, 1991.

[19] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, 1986.

[20] Jorge L.C. Sanz. *Advances in Machine Vision.* Springer-Verlag, 1989.

[21] Hironobu Takahashi and Fumiaki Tomita. Planarity constraint in stereo matching. In *Proc. 9th International Conference on Pattern Recognition*, volume 1, pages 446–449, Rome, Italy, November 1988.

[22] Harit P. Trivedi. A semi-analytic method of determining stereo camera geometry from matched points in a pair of images: Coincident meridional planes, exact or noisy data. *Computer Vison, Graphics and Image Processing*, 51:299–312, 1990.

[23] A. J. Vayda and A. C. Kak. A robot system for recognition of generic shaped objects. *CVGIP: Image Understanding*, 54(1):1–46, July 1991.

[24] Juyang Weng, Paul Cohen, and Marc Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, October 1992.

[25] Yifeng Wu and H. Maître. A new dynamic programming method for stereovision ignoring epipolar geometry. In *Proc. 9th International Conference on Pattern Recognition*, volume 1, pages 146–148, Rome, Italy, November 1988.