Segmentação de Retas no Espaço Tridimensional

Marcellus Rosa Macêdo
Jorge Stolfi
Departamento de Ciência da Computação
Universidade Estadual de Campinas*
{stolfi,mrmacedo}@dcc.unicamp.br

14 de Junho de 1994

Sumário

Neste trabalho descrevemos algoritmos para deteção e localização de arestas de objetos poliédricos no espaço tridimensional apartir de um par de imagens estereoscópicas. Nossa abordagem é original, pois realizamos o agrupamento de pontos de arestas em segmentos de reta no espaço tridimensional, *depois* de casar pontos correspondentes nas duas imagens e calcular a profundidade dos mesmos.

Abstract

In this paper we describe algorithms to detect and locate polyedral objects edges in three-space from a pair of stereo-images. Our approach is unusual because we perform the clustering of edge points into line segments in the three-space, after matching the correspondings features in the two images and computing their depth.

*Endereço Atual: DCC–IMECC, UNICAMP, Caixa Postal 6065, 13081-970 Campinas SP – Brasil.

1 Introdução

A motivação para esta pesquisa é um problema específico de visão por computador: determinar a dimensão e posição no espaço de um objeto de forma conhecida, a partir de um par de imagens estereoscópicas do mesmo.

A abordagem usual para este problema é identificar primeiro os segmentos de reta presentes em cada imagem, e depois casar estes segmentos para determinar sua posição no espaço. Por motivos diversos, resolvemos explorar um caminho alternativo: primeiro casar pontos de descontinuidade nas duas imagens, obtendo assim um conjunto de pontos do $\mathbf{R}^3$; e em seguida agrupar estes pontos em segmentos de reta.

A resolução do problema completo requer várias outras etapas, incluindo a calibração das câmaras, e a identificação dos segmentos de reta no espaço com as arestas do objeto procurado. Não discutiremos essas etapas neste artigo, pois elas praticamente independem da abordagem empregada nas etapas de deteção e segmentação de arestas.

Na seção 2 descrevemos a deteção de pontos de descontinuidade nas duas imagens, e o casamento dos mesmos. Na seção 3, descrevemos um algoritmo para agrupar esses pontos em segmentos de reta. Na seção 4, apresentamos os resultados desses algoritmos aplicados a alguns pares de imagens simuladas.

2 Deteção e casamento de descontinuidades

2.1 Varredura unidimensional

A primeira etapa do nosso programa é a deteção de descontinuidades de intensidade em cada uma das duas imagens, separadamente.

A idéia do algoritmo é dividir cada linha de varredura da imagem em *carreiras*, que são seqüências de píxels 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.

Este método é obviamente pouco adequado para detetar uma aresta cuja projeção na imagem é aproximadamente horizontal. Em muitas aplicações, este defeito seria inaceitável, e portanto teríamos que usar algoritmos de deteção mais complexos, como os de Sobel [1].

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 esses pontos em geral pertencem também a arestas que não são horizontais.

Esta afirmação se aplica em parte também a arestas que formam um ângulo pequeno com a horizontal: devido e erros de medida, quantização, e calibração das câmaras, 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 deteção de descontinuidades fosse sensível a arestas quase-horizontais, teríamos que descartar essas arestas antes de tentar a reconstrução estereoscópica.

É importante observar que uma aresta do objeto pode cobrir vários pixels consecutivos numa mesma linha da imagem; quer por erro de focalização da câmara, 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. Para que o casamento funcione, é importante agrupar esses pixels, e devolver o ponto médio do grupo como sendo a posição nominal da aresta.

2.2 Determinação de carreiras

Definimos uma carreira como sendo sequência maximal $p_r, p_{r+1}, \ldots, p_s$ de pelo menos K pixels consecutivos que satisfaz

$$|p_i - p_j| \leq 2\delta\,|i - j|$$

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

Os parâmetros K e δ dependem da aplicação. O valor de δ é uma estimativa da diferença máxima entre dois píxels 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 píxel, devido por exemplo à focalização incorreta da câmara, seja incorretamente interpretada como duas arestas separadas por uma face minúscula.

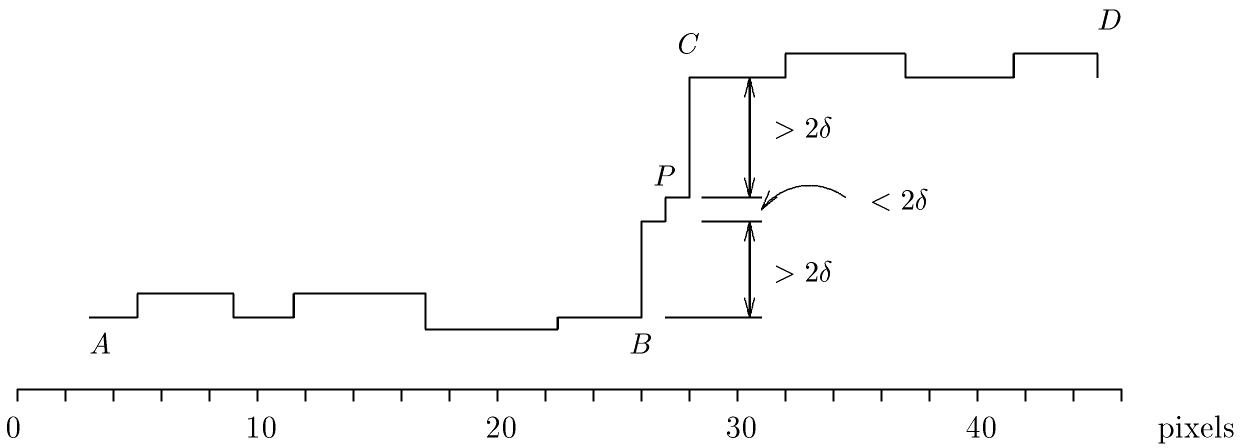

Figura 1: Duas carreiras ($A..B$ e $C..D$) separadas por um degrau ($B..C$). Note que o patamar P não é interpretado como uma carreira, por não ter largura suficiente.

Definimos a *amplitude* de uma aresta da imagem como sendo a diferença absoluta entre as intensidades das faces adjacentes à mesma. Note que a detetabilidade de uma aresta é uma função complicada da amplitude Δ e do ângulo θ da aresta com a horizontal. Pode-se verificar que uma aresta com $\theta \geq 45°$ é sempre reconhecida se $\Delta > 4\delta$, e nunca é reconhecida se $\Delta \leq 2\delta$. Para $\theta < 45°$, a aresta é sempre reconhecida se $\Delta > \frac{4\delta}{\tan\theta}$, e nunca é reconhecida se $\Delta \leq \frac{2\delta}{\tan\theta}$. (Se Δ está entre esses dois limites, a probabilidade da aresta ser detetada varia de maneira complicada com Δ e θ; mas de qualquer maneira aumenta com os dois.)

Observe que qualquer borramento da imagem (devido a focalização imperfeita, poeira, etc.) sempre diminui a probabilidade de deteção de arestas com Δ e θ baixo.

O algoritmo para deteção 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 do píxel anterior; caso contrário, a carreira do pixel anterior é encerrada, e p_i é considerado o início de uma nova carreira. Uma vez examinados todos os píxels, descartamos as carreiras com menos de K píxels.

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 píxel p_s $(r < s)$, então adotamos $x' = \frac{1}{2}(r + s)$ como sendo a abscissa nominal do degrau, para a ordenada comum y' do centro dos píxels desta linha.

2.3 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 píxels 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 uma *qualidade*

$$q = \Delta(w_{\max} - w)$$

onde $\Delta = |p_r - p_s|$ é a diferença absoluta entre o último píxel p_r da carreira precedente e o primeiro píxel p_s da carreira seguinte; e w é a largura $s - r$ do degrau. O parâmetro $w_{\max}$ é a largura máxima permitida para um degrau.

Dentre todos os degraus encontradas numa linha de varredura, escolhemos os N melhores, onde N é um parâmetro do algoritmo. (De modo geral, N 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 a cena consiste de um único cubo contra um fundo uniforme, podemos tomar $N = 5$, pois uma linha traçada sobre qualquer projeção de um cubo cruza no máximo 4 arestas visíveis do mesmo.)

2.4 Casamento dos degraus

Para cada linha de varredura, a etapa anterior devolve duas listas de degraus, uma em cada imagem. Na etapa de casamento, procuramos formar pares de degraus, um de cada lista, que provavelmente correspondem a duas vistas do mesmo ponto de uma aresta do objeto.

Para avaliar a probabilidade de dois degraus pertencerem à mesma aresta, usamos a medida de *discrepância*

$$f = \alpha \min\{|e_1 - d_1|, |e_2 - d_2|\} + \beta \max\{|e_1 - d_1|, |e_2 - d_2|\} \tag{1}$$

onde (e_1, e_2) são as intensidades dos píxels imediatamente antes e depois do degrau na imagem da esquerda, e (d_1, d_2) são os mesmos valores para a imagem da direita. Nos nossos testes, usamos $\alpha = 0.9$ e $\beta = 0.1$.

A idéia por trás da fórmula (1) é que se houver semelhança entre os pixels de um mesmo lado dos dois degraus, eles são considerados semelhantes — mesmo que os pixels do outro lado sejam muito diferentes. A justificativa para este critério é que uma aresta de silueta, quando vista de dois ângulos diferentes, pode estar projetada contra duas partes do fundo com cores diferentes.

2.5 Cálculo da posição espacial

Sejam x'_l e x'_r as posições horizontais de dois degraus, um em cada imagem, que foram casados na fase anterior. Seja y' a ordenada comum aos dois. Supondo-se que os dois degraus correspondem a um mesmo ponto numa aresta do objeto, as coordenadas desse ponto no espaço podem ser então determinadas pelas fórmulas padrão de estereoscopia [2]:

$$x = b\frac{(x'_l + x'_r)/2}{x'_l - x'_r} \quad y = b\frac{y'}{x'_l - x'_r} \quad z = b\frac{f}{x'_l - x'_r} \tag{2}$$

Veja a figura 2.

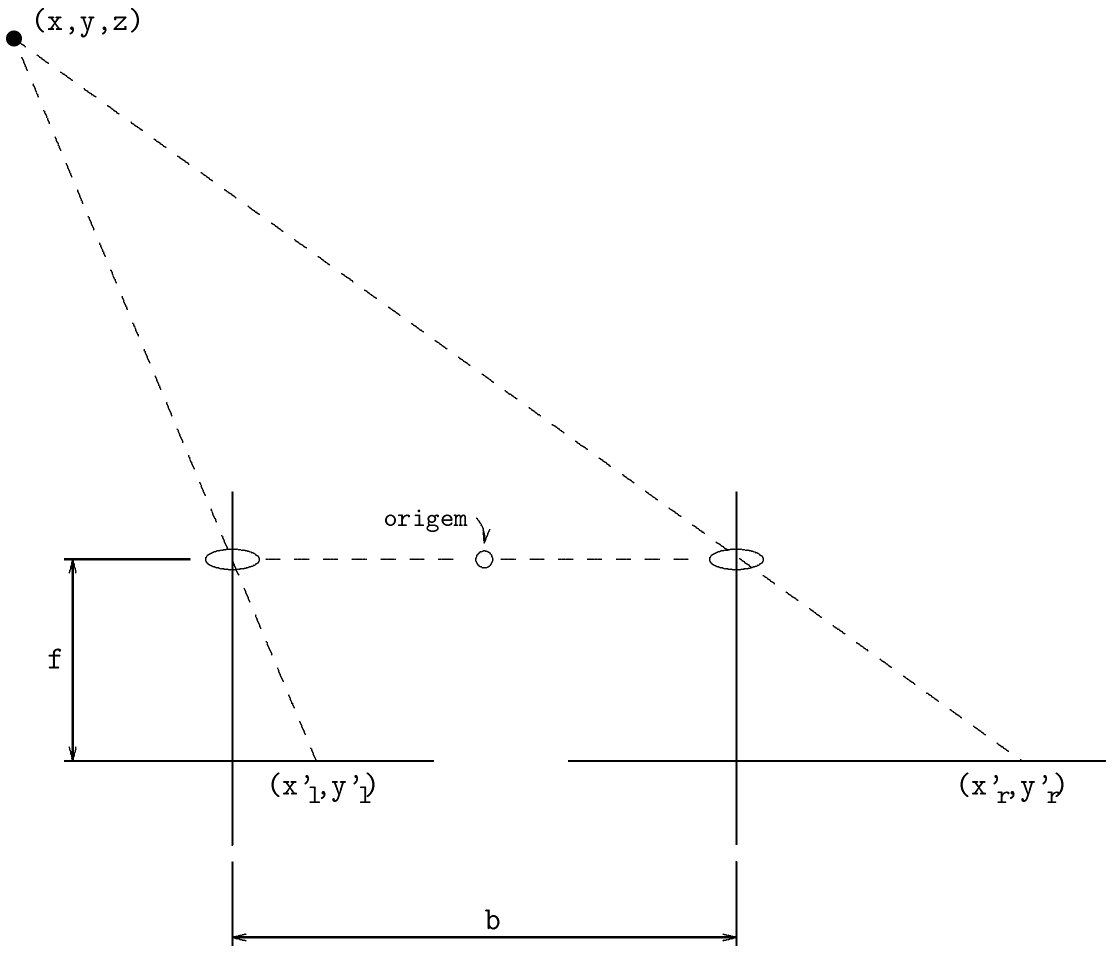

Figura 2: Geometria básica da visão estereoscópica

Entretanto, 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, a etapa de segmentação trabalha com as *coordenadas isotrópicas*

$$x'' = \frac{(x'_l + x'_r)}{2} \quad y'' = \frac{y'}{\sqrt{2}} \quad z'' = \frac{x'_l - x'_r}{2} \tag{3}$$

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 $\sqrt{2}$.

Depois da fase de segmentação, os extremos das arestas encontradas são convertidos do sistema de coordenadas isotrópicas (3) para coordenadas do mundo real (2), pelas seguintes fórmulas:

$$x = \frac{bx''}{2z''} \quad y = \frac{\sqrt{2}by''}{2z''} \quad z = \frac{fb}{2z''} \tag{4}$$

3 Segmentação dos pontos no espaço

O objetivo desta etapa é agrupar um conjunto de pontos no espaço numa coleção de *segmentos* — subconjuntos de pontos, cada um dos quais aproxima a uma linha reta. Quando os pontos são o resultado da etapa anterior, a coleção de segmentos resultante deve incluir as arestas dos objetos que são visíveis por ambas as câmaras.

3.1 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 dados 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.

Em seguida, na etapa de *amadurecimento*, acrescentamos repetidamente ao candidato mais promissor o ponto que está mais alinhado com seu eixo principal, até obter um candidato com um certo número mínimo de pontos. Este candidato é então submetido a um processo de *crescimento*, que acrescenta ao mesmo todos os pontos que estão aproximadamente alinhados com ele, em ordem de distância crescente. O conjunto resultante é um *segmento completo*.

As etapas de amadurecimento e crescimento são repetidas até que todo ponto dado pertença a algum candidato.

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

3.2 Direções principais e momentos de inércia do candidato

A posição, direção e extensão aproximadas de um candidato podem ser descritas pelo *baricentro*, *eixos principais* e *momentos de inércia* de sua nuvem de pontos, definidos a seguir [1]. (Estes nomes são emprestados da física, mas os mesmos conceitos são também usados em estatística, controle, modelagem, etc.)

O *baricentro* de um candidato é simplesmente a média aritmética das posições dos pontos; ou seja, seu centro de massa, 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

$$M = X^T X$$

onde X é a matriz de coordenadas (isotrópicas) dos pontos transladados,

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

Pode-se mostrar que esta matriz 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 realmente 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 os 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 podem ser estimadas pelos números

$$D_i = \sqrt{\frac{\lambda_i}{n-1} + \epsilon^2} \tag{5}$$

Nesta fórmula, o parâmetro ϵ mede a incerteza em cada coordenada; isto é, a magnitude dos erros de medida, quantização e aproximação cometidos na etapa deteção de descontinuidades.

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 ser portanto bem pequenas — da ordem de ϵ.

A justificativa para definir a dispersão D_i pela fórmula (5), 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 [4, pages 335–380].

3.2.1 Sistema de coordenadas do candidato

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

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 isotrópicas globais $x = (x_1, x_2, x_3)$ pela fórmula

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

3.2.2 Métrica definida por um candidato

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

$$d = \sqrt{y_1^2 + y_2^2 + y_3^2}$$

Esta norma é usada para comparar distâncias entre os pontos dados e o baricentro do candidato. 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 rD_i.

3.2.3 Qualidade

Definimos a *qualidade* de um candidato como sendo a razão entre a dispersão máxima (longitudinal) 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).

3.2.4 Amadurecimento

Na etapa de amadurecimento, repetimos o seguinte processo até obtermos um candidato com um número mínimo M de pontos. (O valor de M depende da aplicação; tipicamente, M está entre 4 e 8).

Dentre todos os candidatos, escolhemos um candidato que tenha 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 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 de R, na métrica associada a R; e acrescentamos u_{min} a R.

É importante notar que, quando um ponto é incluído num candidato, ele não é excluído do conjunto de pontos. Portanto, nesta fase os candidatos geralmente tem vários pontos em comum, e uma mesma aresta do objeto pode estar representada por vários candidatos.

Observe que, depois de no máximo n iterações 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 iteração, ou vários candidatos podem se revezar.

3.2.5 Crescimento de um candidato

De qualquer maneira, eventualmente um dos candidatos R atinge o tamanho mínimo M, e nesse momento ele é considerado "maduro" e submetido ao processo de crescimento.

Nesta fase, dentre todos os pontos que não estão em R escolhemos o ponto u_{min} que (1) está aproximadamente alinhado com a direção principal do candidato, e (2) está no máximo a uma distância d_{max} do candidato, na métrica associada ao mesmo, e (3) está mais próximo do candidato, nessa mesma métrica, dentre todos os pontos que satisfazem (1) e (2). 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 de R de quaisquer 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 que contém pontos do segmento recém-completado ou é praticamente um subconjunto deste mesmo segmento, ou contém muitos pontos de outro(s) segmento(s) (Tais candidatos geralmente aparecem na junção entre duas ou mais arestas do objeto.)

No primeiro caso, o candidato 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, a retirada dos pontos provavelmente melhora a qualidade do candidato, e portanto aumenta as chances do(s) outro(s) segmento(s) ser(em) reconhecido(s). 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.

A condição (1) acima — que o ponto u_{min} está aproximadamente alinhado com a direção principal de R — significa que suas coordenadas (y_1, y_2, y_3), relativas a R, satisfazem a desigualdade

$$y_1^2 - y_2^2 - y_3^2 + 1 \geq 0 \tag{6}$$

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 do candidato, e seu ângulo de abertura é tanto menor quanto mais colineares forem os pontos. Veja a figura 3.

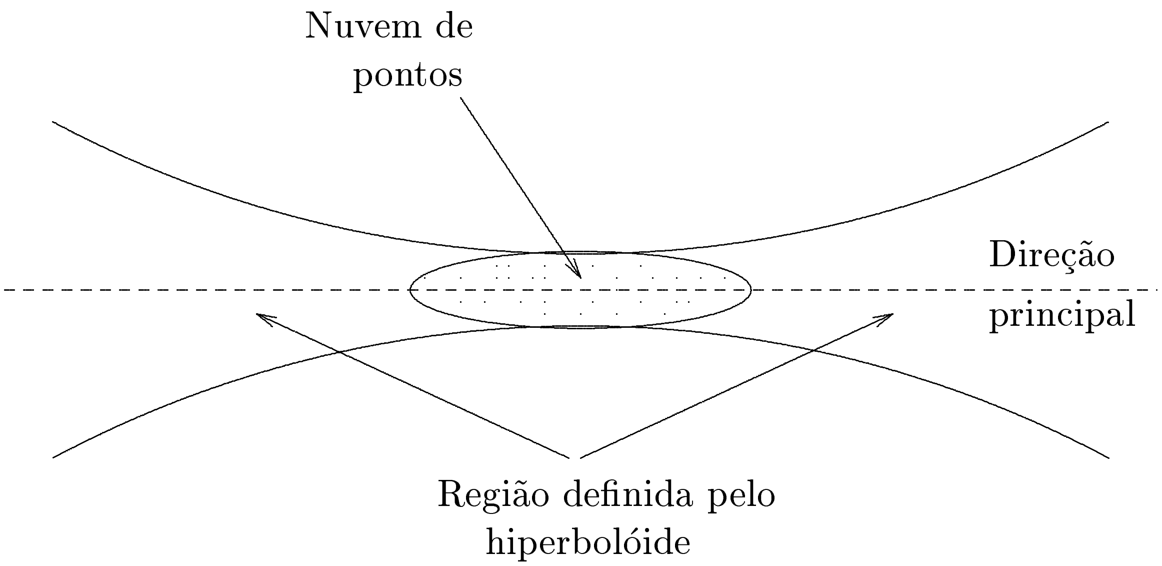

Figura 3: Hiperbolóide definido pela condição (1).

A razão para limitar a inclusão de vértices 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 (1) tem precisamente estas propriedades.

O propósito da condição (2) — que a distância do ponto u_{min} ao baricentro seja menor ou igual a d_{max} — 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 $d_{max}/2$ vezes seu comprimento.

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.

3.3 Análise de complexidade

Se n é o número de pontos dados, o custo de se conseguir o primeiro candidato maduro pode ser $\Theta(n^2)$ no pior caso; e o custo de crescer esse candidato até um segmento completo com k pontos é $\Theta(nk)$.

Portanto, o custo total do algoritmo de segmentação pode ser $\Theta(n^3)$ 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 $\approx \epsilon$. Se esta situação ocorrer com pontos gerados pela fase de deteção e casamento, as imagens originais provavelmente são excessivamente complexas ou de baixa qualidade, e provavelmente inadequadas para quaisquer algoritmos de visão esteroscópica.

Em situações mais realistas, podemos esperar que que cada ponto dado pertencerá a um ou dois segmentos completos, em média. Nesse caso, o custo esperado do algoritmo de segmentação será $\Theta(n^2)$. Observe que, se a fase de deteção 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 píxels das imagens de entrada.

4 Testes e resultados

4.1 Testes com imagens sintéticas

A maioria dos pares de imagens estereoscópicas que usamos para testar o algoritmo foram gerados sintéticamente, com um programa simples de traçado de raios [3]. As cenas utilizadas consistiam de um pequeno número de cubos ou paralelepípedos, contra um fundo uniforme.

Para tornar os testes 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 media 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 às mesmas um filtro de mediana unidimensional. 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 a seção 2.2).

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 necessáriamente triviais. Veja a figura 4.

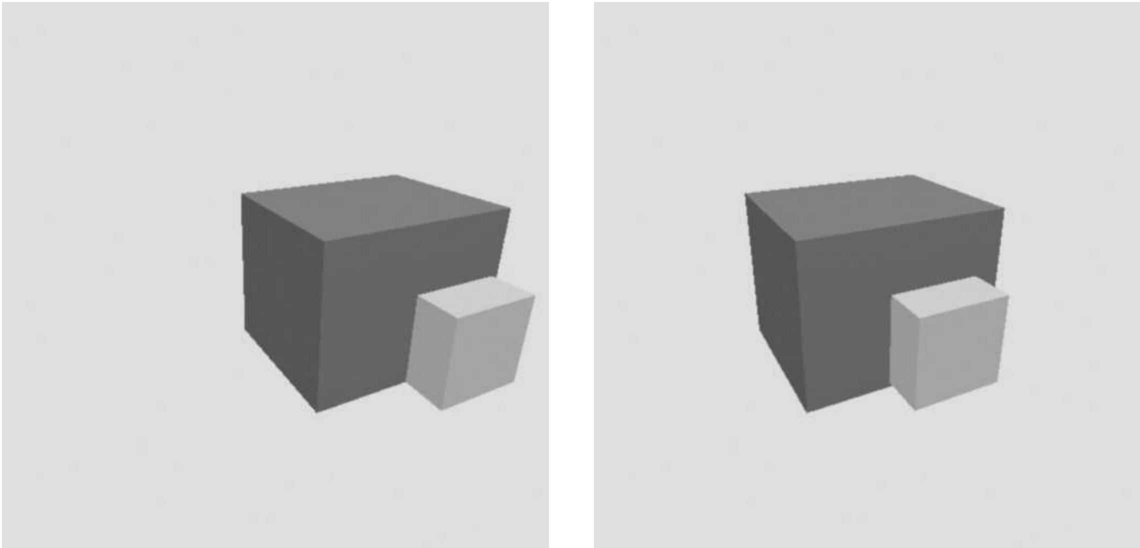

Figura 4: Um par estereo sintético.

A figura 5 mostra as posições dos degraus determinados em cada imagem pelo algoritmo de deteção de descontinuidades.

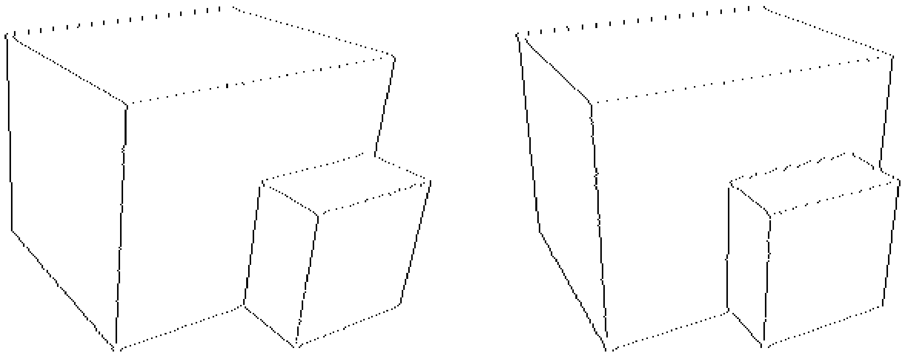

Figura 5: Degraus de intensidade detetados nas imagens da figura 4.

Os emparelhamentos mais prováveis desses degraus, escolhidos pelo algoritmo de casamento, definiram o conjunto de 1041 pontos tridimensionais mostrado na figura 6.

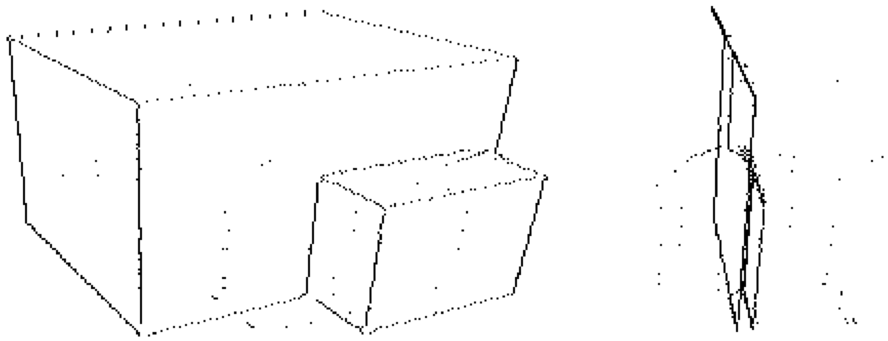

Figura 6: Pontos de arestas no espaço, determinados pelo casamento dos degraus da figura 5. Sistema de coordenadas isotrópicas, eixos x''–y'' (esq.) e z''–y'' (dir.).

O algoritmo de segmentação encontrou 22 segmentos completos, dos quais 19 (com 1107 pontos no total) correspondem a arestas reais do objeto, e 3 (com 32 pontos no total) são arestas falsas devidas a casamentos incorretos, ruído, quantização, erros de arredondamento, etc. Veja figura 7.

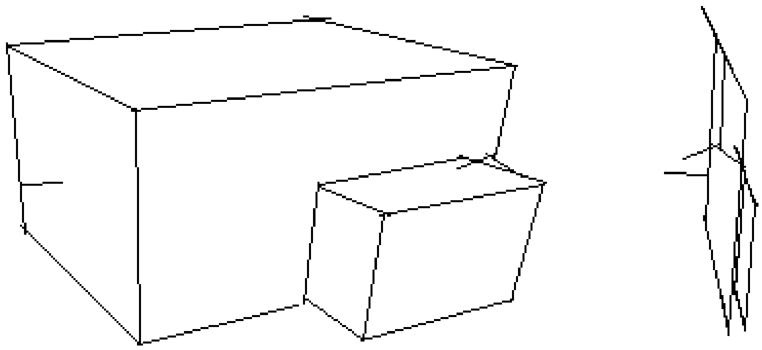

Figura 7: Segmentos completos determinados a partir dos pontos da figura 6. Sistema de coordenadas isotrópicas, eixos x''–y'' (esq.) e z''–y'' (dir.).

Nas figuras 8–11 apresentamos os resultados realizados com uma cena sintética constando de 4 cubos. Note que esta cena contém várias características problemáticas, em particular sombras, oclusões, e casamentos ambíguos (devidos ao fato dos quatro cubos terem a mesma cor e iluminação).

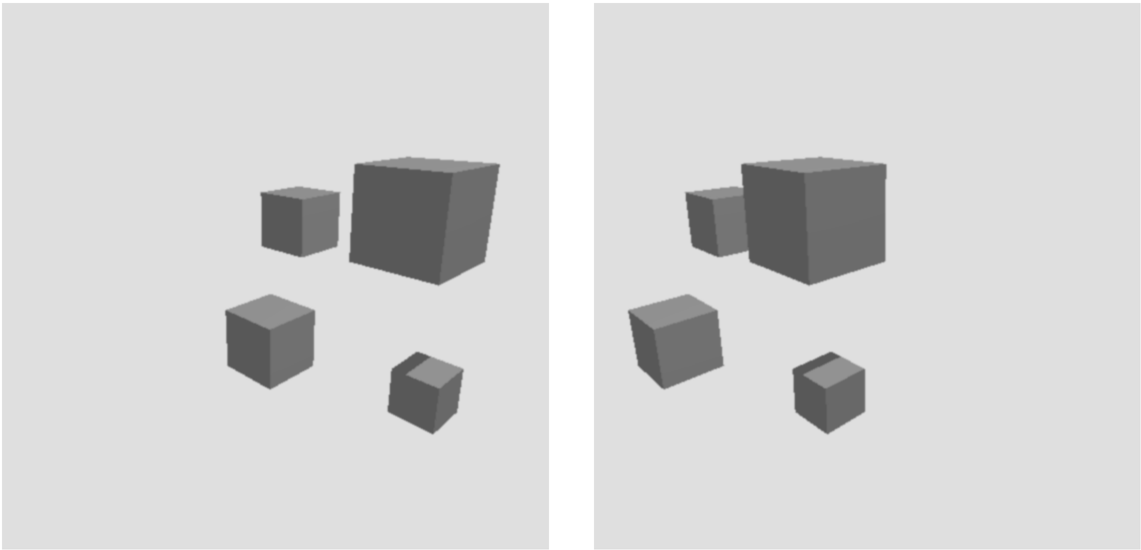

Figura 8: Um par estereo sintético.

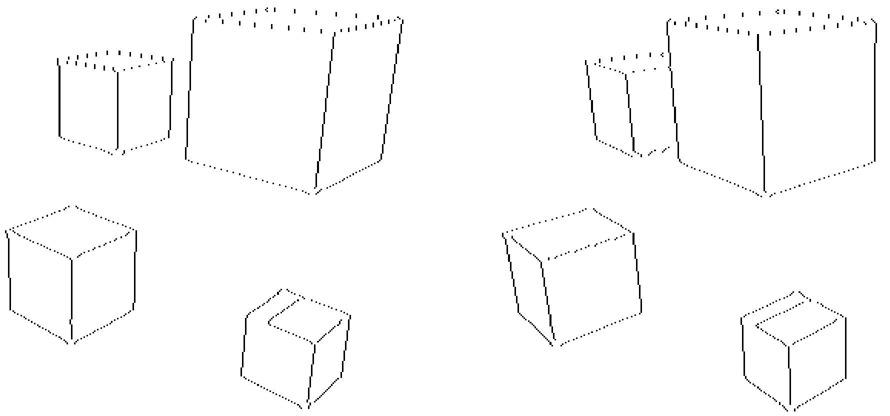

Figura 9: Degraus de intensidade detetados nas imagens da figura 8.

Nesta cena, o algoritmo de segmentação encontrou 61 segmentos completos, dos quais 35 (com 1096 pontos no total) correspondem a arestas reais do objeto, e 26 (com 277 pontos no total) são falsos.

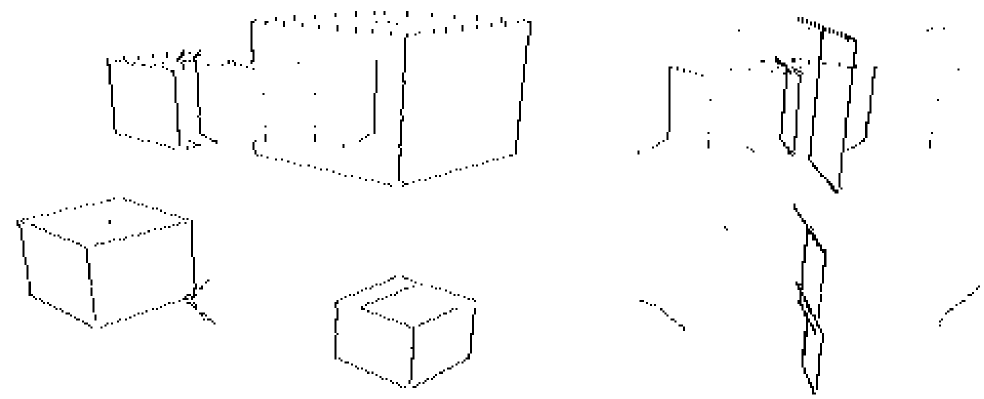

Figura 10: Pontos de arestas no espaço, determinados pelo casamento dos degraus da figura 9. Sistema de coordenadas isotrópicas, eixos x''–y'' (esq.) e z''–y'' (dir.).

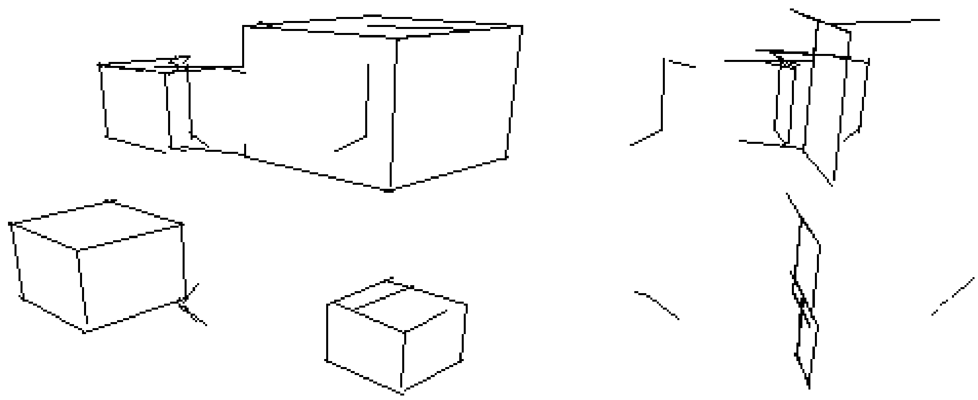

Figura 11: Segmentos completos determinados a partir dos pontos da figura 10. Sistema de coordenadas isotrópicas, eixos x''–y'' (esq.) e z''–y'' (dir.).

5 Conclusões e trabalho futuro

Os testes apresentados na seção anterior sugerem que nossa metodologia (deteção 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 de execução desses algoritmos não é desprezível, mas também não é exorbitante: com a escolha adequada de parâmetros, o tempo é proporcional no máximo ao número de píxels das duas imagens.

Ainda há muito a fazer nesta linha de pesquisa. Por exemplo, deve ser possível melhorar enormemente a eficiência do algoritmo de segmentação — de $O(n^2)$ para $O(n)$ — se usarmos estruturas de dados adequadas para armazenar os candidatos e localizar os pontos mais próximos aos mesmos.

Naturalmente, o próximo passo é testar estes algoritmos com imagens reais, em vez de sintéticas.

Referências

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

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

[3] 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.

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