Recuperação automática da estrutura tri-dimensional a partir de múltiplas imagens utilizando geometria projetiva

Tiago Tagliari Martinez

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO DE MESTRE
EM
CIÊNCIAS

Área de Concentração: Ciência da Computação
Orientador: Prof. Dr. Carlos Hitoshi Morimoto

São Paulo, junho de 2007

Recuperação automática da estrutura tri-dimensional a partir de múltiplas imagens utilizando geometria projetiva

Este exemplar corresponde à redação final da dissertação devidamente corrigida e apresentada por Tiago Tagliari Martinez e aprovada pela Comissão Julgadora.

São Paulo, 22 de junho de 2007

Banca Examinadora:

- Prof. Dr. Carlos Hitoshi Morimoto (orientador) [IME-USP]
- Profa. Dra. Nina Sumiko Tomita Hirata [IME-USP]
- Prof. Dr. Mario Fernando Montenegro Campos [UFMG]

à minha esposa e à minha família.

Agradecimentos

Ao meu orientador, Professor Doutor Carlos Hitoshi Morimoto, pela sugestão do tema e pela inestimável ajuda durante a elaboração desta dissertação.

À minha família, por sempre estar à disposição quando precisei de apoio e por acreditar no sucesso da realização deste trabalho.

Aos meus amigos, que compreenderam minha dedicação e abdicaram de minha atenção em prol do desenvolvimento deste projeto.

À minha esposa, que me deu inestimável suporte e apoio ao longo de toda minha pesquisa.

A todos os pesquisadores que escreveram os ótimos artigos que foram lidos durante a elaboração da dissertação, e cuja pesquisa contribui para o avanço tecnológico.

Resumo

Ferramentas para a criação de modelos tri-dimensionais de objetos são importantes para as áreas de Computação Gráfica e Visão Computacional. Boa parte das ferramentas atuais fazem uso de equipamentos especiais, caros e difíceis de usar. O uso de imagens obtidas a partir de câmeras digitais é uma solução atrativa devido ao seu baixo custo e alta flexibilidade.

Uma técnica relativamente recente de visão computacional, para a reconstrução de objetos a partir de duas ou mais imagens se baseia na geometria projetiva. Este método exige o uso de ao menos duas imagens de pontos de vista diferentes. Para o caso de apenas duas imagens, conhecida como visão estéreo, a geometria epipolar descreve a relação entre as câmeras, que é representada pela matriz fundamental.

Desenvolvemos um sistema para a reconstrução de objetos a partir de um conjunto de imagens. A saída do sistema é uma nuvem de pontos 3D que correspondem a pontos automaticamente selecionados e rastreados pelo método de Kanade, Lucas e Tomasi.

Criamos uma biblioteca de objetos sintéticos e seqüências de imagens reais, que utilizamos para testar o sistema. Os resultados experimentais obtidos mostram que o sistema depende bastante da qualidade e quantidade dos pontos rastreados, mas em condições adequadas é possível gerar uma boa reconstrução.

Como principais contribuições deste trabalho, podemos citar a elaboração de uma síntese dos conceitos de geometria projetiva necessários para a reconstrução de objetos a partir de uma seqüência de imagens e a implementação de código aberto de um sistema para reconstrução de objetos. Este sistema será utilizado pelo Laboratório de Tecnologias de Interação (LaTIn) do IME-USP, para o estudo de técnicas de interação em realidade virtual e estendida.

Abstract

We have developed a system for the reconstruction of objects from a set of images. The ouput of this system is a cloud of 3D points that corresponds to the points automatically selected and tracked using the Kanade, Lucas and Tomasi method.

Tools for the creation of three-dimensional models of objects are important for the fields of Computer Graphics and Computer Vision. Most of the current tools use special, expensive, and hard to use equipments. The use of images captured by digital cameras is an atractive solution, because of its low cost and high flexibility.

A computer vision technique for object reconstruction from two or more views is based on projective geometry. This method requires at least two images of different points of view. The case when only two images are used is known as stereo vision, which uses epipolar geometry to describe the relation between the cameras, represented by the fundamental matrix.

We have also developed a library of synthetic objects and sequences of real images that we used to test the system. Experimental results show that the system highly depends on the quality and quantity of the tracked points, but in adequate conditions it is possible to generate a good reconstruction.

The main contributions of this work, are the elaboration of a synthesis of the concepts of projective geometry necessary for the 3D reconstruction of scenes from a sequence of images, and an open source implementation of a system to reconstruct 3D scenes. This system will be used in the Laboratory of Technologies for Interaction (LaTIn) of IME-USP, for the study of interactive virtual and extended reality environments.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Um dos principais problemas existentes atualmente na área de Computação Gráfica é o trabalhoso processo de criação de modelos tri-dimensionais de objetos.

Com a evolução da capacidade de processamento dos computadores e o barateamento de *hardware* especializado, os sistemas gráficos atuais podem apresentar para o usuário, em tempo real, uma quantidade cada vez maior de informação gráfica, e com um detalhamento e realismo consideravelmente maior. Um efeito prático disso é que a quantidade de informação necessária para descrever uma cena aumentou consideravelmente nos últimos anos.

Atualmente, a criação de um modelo requer a participação intensa de um usuário humano, que em geral demora um tempo considerável. Muitas ferramentas tem sido desenvolvidas para facilitar este processo, porém ele se mantém, em grande parte, uma arte que exige grande trabalho e dedicação.

Por exemplo, vejamos o longa metragem de animação *Shrek* produzido pela *DreamWorks SDK*, que levou mais de quatro anos e meio para ser produzido com o trabalho de mais de 275 pessoas [IMD], a maior parte delas animadores e desenhistas. O processo de renderização utilizava, em cada dia, ate 2,5GiB de informação, excedendo duzentos milhões de polígonos

renderizados por dia, em média, com uma precisão que ultrapassava dez polígonos para cada *pixel* da imagem gerada [Wex]. Mais de mil ambientes e objetos foram criados pelos animadores para a produção do filme [Cin].

1.1 Motivação

Muita pesquisa tem sido feita a fim de desenvolver técnicas para a construção automática de cenas ou reconstrução de objetos já existentes. Chamamos de construção de cenas o ato de criar um modelo de objeto ou cena e reconstrução quando um modelo físico já existente é transportado à forma digital de alguma maneira. A construção de cenas é normalmente associada a área de computação gráfica, enquanto a reconstrução se associa a área de visão computacional. Porém, essa separação é cada vez mais tênue, dada a grande necessidade da computação gráfica de gerar modelos automaticamente, e do uso de modelos complexos em visão computacional para facilitar o processo de análise.

Na prática é válido dizer que a composição de uma cena complexa pode ser feita a partir de um conjunto de construções e reconstruções. Na área de realidade virtual, por exemplo, é comum definir um cenário através da *construção* de uma cena (criação de um modelo), e inserir vários objetos virtuais *reconstruídos* a partir de objetos individuais (cadeiras, mesas, etc) já existentes.

A aplicação e o escopo das técnicas desenvolvidas variam de acordo com o uso pretendido. Abaixo uma lista resumida das técnicas de reconstrução 3D mais comuns:

Triangulação Laser um feixe laser projeta uma linha que se movimenta pela superfície do objeto sendo modelado. Uma câmera, colocada em um ângulo específico em relação à origem do feixe laser, registra as variações na curvatura da linha gerada, estimando a topologia do objeto [Evi96].

Profundidade Laser um feixe laser é direcionado diretamente ao objeto, e através de variações de freqüência e comprimento de onda, o tempo de retorno do feixe à câmera é calculado, e a distância do ponto em relação à câmera é encontrada.

Sonda Mecânica uma espécie de "caneta" presa a um braço mecânico com sensores em cada junta é movida pela superfície do objeto. Pelo movimento das juntas, é possível encontrar a posição tri-dimensional, em relação à base do braço, da ponta da "caneta" em diversos momentos. Esse conjunto de pontos é utilizado para definir a estrutura do objeto. A operação de um equipamento deste porte exige grande tempo e paciência. Em contrapartida, os resultados são em geral extremamente precisos.

Reconstrução a partir de sombras um conjunto de técnicas que utilizam as propriedades de reflexão da superfície do objeto sendo analisado para estimar sua estrutura tri-dimensional. Essa família de técnicas é aplicada, com grande sucesso, para estimar a topologia de um terreno a partir de fotos de satélites [Hor86]

Reconstrução Estéreo a partir de duas câmeras cuja posição relativa é conhecida, podemos utilizar um ramo da geometria conhecido como *geometria epipolar* para estruturar a relação entre as imagens capturadas pelas câmeras. Isto nos permite estimar a posição de um ponto no espaço que aparece em cada uma das imagens e, a partir dessa informação, podemos encontrar a distância deste ponto em relação às câmeras.

Reconstrução a partir de Movimento a estimação da estrutura a partir do movimento é equivalente ao caso de reconstrução estéreo, porém utilizando apenas uma câmera. O desafio agora é encontrar a movimentação tri-dimensional da câmera entre duas posições distintas.

Dentre os métodos descritos acima, se destacam os métodos baseados em imagens pois se utilizam de equipamentos mais simples, que poderiam ser utilizadas em aplicações mais

variadas, como por exemplo, nas áreas de realidade virtual e realidade estendida, onde a reconstrução de objetos também é fundamental para facilitar a visualização, simulação e manipulação de objetos virtuais. Para popularizar essas tecnologias é necessário portanto criar ferramentas de reconstrução automáticas que sejam fáceis de usar, flexíveis e de baixo custo, onde a precisão possa não ser tão relevante.

1.2 Objetivos

Nosso objetivo é a exploração de técnicas de reconstrução de objetos a partir de imagens, para o desenvolvimento de um sistema interativo que permita a reconstrução de objetos rígidos pequenos e simples. Tal sistema deve funcionar em computadores pessoais e aceitar como entrada imagens em formatos de uso comum.

Esse projeto está sendo desenvolvido dentro do Laboratório de Técnicas de Interação (LaTIn) do Instituto de Matemática e Estatística da Universidade de São Paulo, e facilitará o estudo de técnicas de interação em ambientes virtuais e estendidos. Alguns estudos voltados à criação de interfaces para esses ambientes já vem sendo desenvolvidos, como por exemplo, interfaces baseadas no olhar e em gestos.

O código desenvolvido é aberto para o uso e modificação a qualquer instituição interessada, e inclui um conjunto de rotinas básicas desenvolvidas na linguagem de programação orientada à objetos C++, que podem ser facilmente re-utilizadas por outros projetos em desenvolvimento no LaTIn.

Dentro desse contexto, podemos definir algumas restrições para o sistema de reconstrução:

- Uso de equipamentos não especiais, facilmente disponíveis comercialmente.
- Simples de operar, sem necessitar de treinamento especial.

- Fácil de transportar, que possa ser carregado no bolso ou numa mochila.
- Baixo custo, idealmente necessitando apenas de equipamentos que um usuário já possuiria por outras razões.

Como boa parte desses requisitos já são satisfeitos atualmente por câmeras e filmadoras digitais, decidimos por utilizar um método que utilize tais equipamentos e faça a reconstrução de objetos a partir de um conjunto de imagens. Uma técnica mais recente que já demonstrou bons resultados é a técnica de *reconstrução projetiva* [FL01, HA04].

1.2.1 Contribuições

Nossas principais contribuições são as seguintes:

- Criação de um texto sintetizando a técnica de reconstrução de objetos a partir de imagens usando o formalismo de geometria projetiva, onde são apresentados os diversos tipos de reconstrução possíveis a partir do caso projetivo e sua formulação, com enfoque maior na visão estéreo.
- Implementação de um conjunto de algoritmos para reconstrução de objetos a partir de imagens, utilizando o formalismo de geometria projetiva. Todo o código fonte desenvolvido é aberto e disponível para reuso sem qualquer restrição, um dos objetivos é que esta implementação sirva de base para implementações mais completas no futuro.
- Geração de uma biblioteca de conjuntos e pares de imagens reais e sintéticas para a avaliação destes algoritmos. Tal biblioteca poderá ser futuramente utilizada por outros projetos do LaTIn e por outras instituições.
- Condução de experimentos para comparar o desempenho das técnicas de reconstrução desenvolvidas, e discussão dos resultados.

1.3 Metodologia

Este é um trabalho exploratório em desenvolvimento no LaTIn. O laboratório já possui experiência no desenvolvimento de vários sistemas para processamento de imagens, como para o rastreamento de objetos e construção de imagens panorâmicas em tempo real [Cer04, Mim04, San04, Cab05, Bar05]. Esse porém é o primeiro trabalho utilizando múltiplas câmeras e por isso será dado um enfoque especial ao caso estéreo, por ser mais simples, embora o formalismo de geometria projetiva permita o tratamento de múltiplas câmeras. A partir do desenvolvimento de um sistema simples de reconstrução, desejamos adquirir a experiência prática necessária para a utilização dessas técnicas em dispositivos de interação, desenvolver as rotinas básicas para sua implementação, e possibilitar o estudo de outras técnicas correlatas no futuro.

Um sistema de reconstrução baseado em imagens tipicamente possui os módulos mostrados no diagrama de blocos da figura 1.1.

Câmera : uma imagem é capturada por uma câmera e transferida para algum formato digital que possa ser processada por computador.

Pré-Processamento : para facilitar a extração de características da imagem, é comum realizar algum pré-processamento, por exemplo suavizações e realces.

Detecção de Características : a partir da imagem pré-processada são extraídas características locais relevantes à aplicação. Este passo restringe a quantidade de informação a ser processada nas demais imagens do conjunto, tornando o processamento computacionalmente mais eficiente.

Rastreamento de Características : tendo uma imagem cujas características foram selecionadas e uma segunda imagem que passou pelo pré-processamento, procuramos nesta

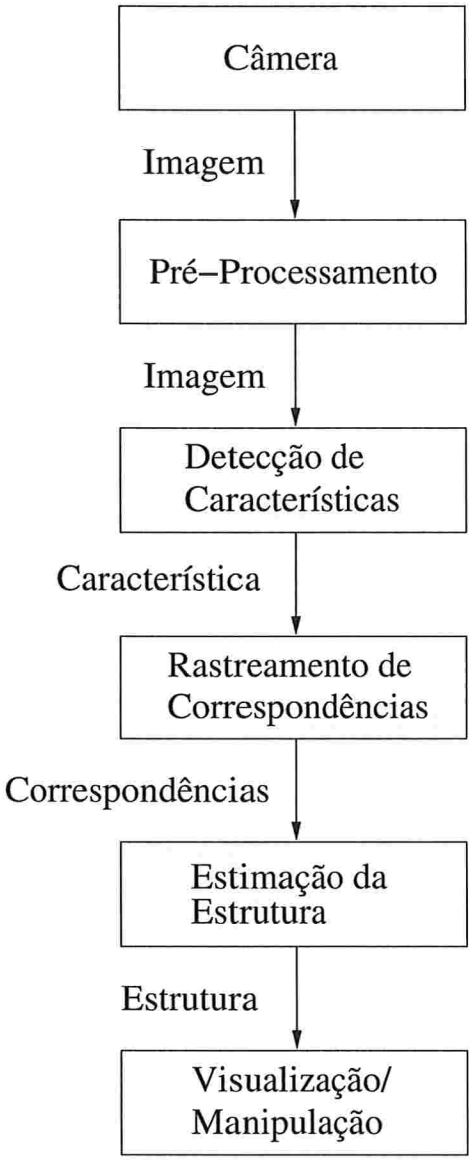

Figura 1.1: Módulos de um sistema de reconstrução.

segunda imagem pelas características que foram selecionadas na primeira. Nosso objetivo é encontrar, para posições selecionados na cena, sua movimentação entre os quadros, ou seja, a correspondência entre características de imagens distintas.

Estimação da Estrutura : o conjunto de correspondências é utilizado para estimar a movimentação entre as câmeras. A partir dessa informação (ou conjuntamente), é possível calcular as posições tri-dimensionais dos pontos correspondentes entre as imagens, ou seja, a estrutura 3D da cena, usando triangulação.

Para entender melhor o processo de estimação da estrutura 3D de objetos, vamos apresentar com um pouco mais de detalhes o modelo de câmera e geometria utilizados nesse trabalho.

Através de um conhecimento maior sobre como imagens são projetadas no plano da câmera, podemos iniciar uma discussão sobre quais informações podem ser extraídas no caminho inverso: das imagens aos objetos no espaço 3D.

1.3.1 Formação de Imagens

Para compreendermos quais informações podemos extrair sobre os objetos apresentados em um conjunto de imagens, precisamos compreender o processo de formação de imagens bidimensionais a partir de objetos tri-dimensionais.

O modelo de formação de imagens a partir de objetos 3D que utilizamos é similar ao funcionamento de uma câmera: os raios de luz que partem dos objetos passam atravéz de um pequeno buraco e são projetados no negativo (Figura 1.2). Este modelo será mais detalhado na sessão (2.1).

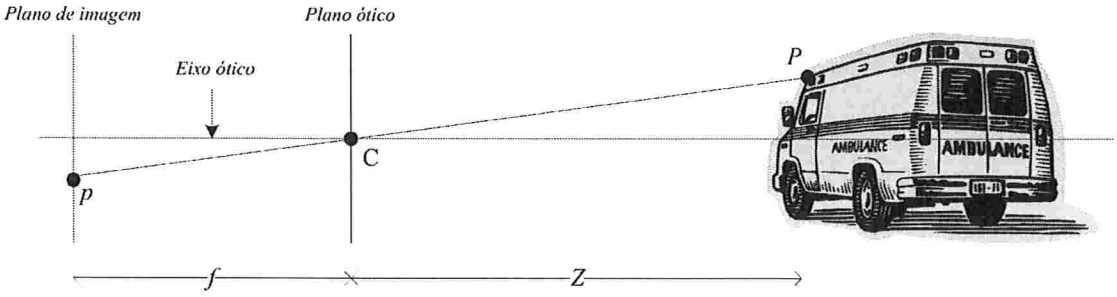

Figura 1.2: O modelo de câmera *pinhole*.

1.3.2 Estimação da Estrutura

Conhecendo o processo de formação de imagens, como é possível extrair informações 3D das superfícies que as geraram? Nós somos capazes de inferir muita informação 3D a partir de uma única foto, como quais objetos estão mais próximos, quais os mais distantes e o tamanho relativo entre objetos, mesmo sem conhecer os objetos contidos na imagem. Esse fenômeno foi já bastante estudado no ramo da psicologia conhecida com *gestalt* [Wik], e indica que nossa percepção considera várias propriedades visuais para construir uma interpretação.

Em nosso caso, dispondo apenas de conhecimento geométrico, é possível estimar a posição 3D de pontos a partir de 2 ou mais imagens distintas. A geometria epipolar é usada para reconstruir objetos a partir das imagens de câmeras distintas, conhecida como visão estéreo, e mais recentemente, um formalismo ainda mais elegante, conhecido como geometria projetiva, tem sido utilizado para a reconstrução de objetos utilizando múltiplas imagens.

Visão Estéreo

As perguntas básicas que a visão estéreo nos responde é: se temos a informação sobre a posição relativa de duas câmeras, o que podemos dizer sobre a relação entre as imagens capturadas por elas? E quais informações podemos extrair desta relação? A resposta para essas perguntas

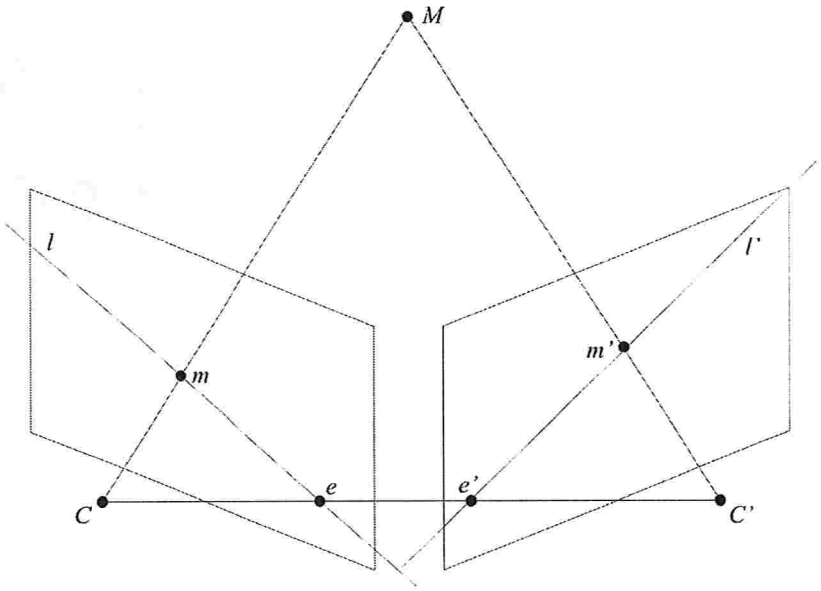

Figura 1.3: Geometria epipolar.

pode ser descrita através da *geometria epipolar*.

A geometria epipolar se baseia na observação que, para um par de câmeras, seus centros de projeção C e C', e um ponto M no espaço, definem um plano. Esse plano contém as projeções m e m' de M em ambas as câmeras. Tal plano é chamado de *plano epipolar* e o conjunto de todos os planos epipolares é chamado de *feixe epipolar* (figura 1.3). Pode-se perceber que, conforme M se move no espaço, m e m' se movem nos planos de imagem de cada câmera, porém e e e' permanecem fixos. Tais pontos são chamados de *epipoles* e são a interseção da reta que une os dois centros de projeção com os planos de imagem. As retas que unem e à m e e' à m' são formados pela intersecção dos planos de imagem e o plano formado por CMC' e são chamadas de linhas epipolares.

Dada essa estrutura, é possível, de uma maneira que será melhor detalhada no capítulo 2, definirmos uma transformação $\mathbf{F}$ que descreve a relação entre pontos na primeira e na segunda

imagem. Tal transformação é chamada de *matriz fundamental* e nos dá a seguinte relação: $l' = \mathbf{F}m$. Assim, somos capazes de restringir a posição de m' na segunda imagem, dada a posição de m na primeira imagem.

Sabendo a posição relativa das câmeras, determinamos $\mathbf{F}$, e com isso podemos varrer todas as linhas epipolares correspondentes de ambas as imagens (o número de linhas epipolares distintas é um conjunto finito, dado que as imagens são compostas de *pixels* discretos). Como sabemos que pontos correspondentes devem estar em linhas epipolares correspondentes, podemos utilizar um teste simples de correlação por intensidade ou cor para determinarmos o quanto a projeção de um ponto se moveu dentre as duas imagens. O efeito paralaxe indica que pontos cujas projeções se movem mais, estão mais próximos da câmera do que pontos cujas projeções se movem menos. Chamamos estas movimentações de *disparidades*.

Se, além da posição relativa entre as câmeras, temos os parâmetros intrínsicos delas, podemos associar dimensões às disparidades e efetuar uma reconstrução métrica. Além de sabermos a profundidade de cada ponto em relação à câmera, temos este valor em uma determinada unidade.

Geometria de Múltiplas Câmeras

O caso de visão estéreo assume que sabemos a posição relativa entre as câmeras, a *estrutura* do sistema. Apesar desta suposição ser válida em alguns casos, ela é difícil de se manter quando temos o caso de uma única câmera se movendo para capturar imagens em pontos de vista diferentes, que é o caso que pretendemos estudar.

Portanto devemos estimar o quanto a câmera se movimentou entre as imagens, para podermos determinar a estrutura do sistema, e podermos efetuar a reconstrução. Felizmente existem técnicas que nos permitem estimar a informação do movimento entre quadros. A generalização destes conceitos, conhecida por *geometria projetiva* será apresentada em mais

detalhes no capítulo 2, em especial na sessão (2.2.2), comentamos sobre maneiras robustas de estimar a matriz fundamental de um par de câmeras qualquer, dado um conjunto de correspondências entre pontos das imagens.

1.4 Organização do Texto

No capítulo 2 vamos apresentar uma síntese dos conceitos de geometria projetiva e outros tópicos importantes ao desenvolvimento do sistema. Iniciaremos com uma apresentação breve de técnicas normalmente utilizadas para a representação de modelos 3D, depois em maior detalhe a geometria projetiva, com ênfase nas técnicas utilizadas neste trabalho. Finalizando, uma descrição da técnica de rastreamento utilizada pelo sistema.

No capítulo 3 descrevemos a implementação do sistema e o funcionamento de seus módulos. Os formatos de entrada e saída suportados são descritos, e algumas partes que não puderam ser implementadas por restrições de tempo são comentadas.

No capítulo 4 apresentaremos resultados do uso do sistema em um conjunto de imagens sintéticas. Os resultados incluem o caso de visão estéreo simples, e o caso de múltiplas câmeras. E no capítulo 5 concluímos o texto com um resumo e com propostas para trabalhos posteriores.

Capítulo 2

Fundamentos Teóricos

O objetivo desse capítulo é fornecer ao leitor os fundamentos teóricos necessários para o entendimento dos métodos implementados para a reconstrução de objetos a partir de imagens. Em particular serão introduzidos conceitos de geometria projetiva, que nos permitirão descrever, sob um mesmo formalismo, o problema de reconstrução a partir de visão estéreo e de múltiplas imagens (2.1).

Também apresentamos os métodos de rastreamento de imagens utilizados para gerar os conjuntos de correspondências necessários para que a geometria de múltiplas imagens seja utilizada (2.4).

Por fim, apresentamos uma introdução a diversos métodos de representação e armazenamento digital de modelos 3D que podem ser utilizados para armazenar o resultado final do processo de reconstrução (2.5).

2.1 Geometria

Nesta sessão, sintetizamos o formalismo da geometria de múltiplas câmeras para o problema de reconstrução de objetos a partir de pontos correspondentes entre as imagens. A técnica de reconstrução de objetos estudada é baseada na pesquisa de quanta informação conseguimos extrair de um conjunto de N visualizações (fotografias, projeções) de um mesmo objeto a partir de pontos de vista diferentes.

Portanto, para compreender seu funcionamento precisamos primeiro compreender como um objeto no espaço tri-dimensional é projetado em uma imagem, como em uma fotografia.

O assunto de geometria projetiva já foi tratado por diversos autores, como em [HGC92, ZDFL95, LF96, Tri98, MT98, Ioc98, Bir98, ZCHS03]. Grande parte do conteúdo deste capítulo é um resumo do conteúdo dos livros de Faugeras e Luong [FL01], e Hartley e Zisserman [HA04]. O leitor interessado deve procurar estas fontes para um tratamento mais detalhado.

2.1.1 Coordenadas Homogêneas e Transformações no Espaço

Utilizando a notação vetorial, um ponto 3D qualquer pode ser apresentado por $P = (x, y, z)^T$ e podemos efetuar as operações básicas de translação, escala e rotação:

$$P' = P + D$$
$$P' = S \cdot P$$
$$P' = R \cdot P,$$

onde D é o vetor de translação, S e R são as matrizes de escala e rotação, respectivamente, e P' é o ponto P após sofrer a respectiva transformação.

Estas três transformações são tão comuns na computação gráfica, que uma maneira de

tratá-las de forma combinada é interessante. Para isso utilizamos as coordenadas homogêneas [Wat99]. Efetivamente, aumentamos a dimensionalidade do espaço para tornar as transformações lineares. Na prática, isso proporciona um sistema linear unificado para a especificação das transformações.

Utilizando coordenadas homogêneas, um ponto:

$$P = (x, y, z)^T$$

do sistema de coordenadas cartesianas é representado como

$$P = (X, Y, Z, w)^T$$

para qualquer fator de escala $w \neq 0$. A representação tri-dimensional das coordenadas cartesianas é dada por:

$$\begin{aligned} x &= \frac{X}{w} \\ y &= \frac{Y}{w} \\ z &= \frac{Z}{w}. \end{aligned}$$

Em outras palavras, pontos 3D com coordenadas homogêneas $(X, Y, Z, w)^T$ possuem coordenadas cartesianas $(X/w, Y/w, Z/w)^T$. Homologamente, denotamos um ponto 2D em coordenadas homogêneas em uma imagem plana como $(X, Y, w)^T$, sendo $(X/w, Y/w)^T$ suas coordenadas cartesianas correspondentes. De maneira mais formal, o espaço definido por $(n+1)$-tuplas, com a regra que $(n+1)$-tuplas proporcionais – isto é, na forma $(\lambda X, \lambda Y, \lambda)$ – representam o mesmo ponto, é chamado de *espaço projetivo* de dimensão n, e denotado por $\mathbb{P}^n$.

Com o uso de coordenadas homogêneas podemos combinar uma rotação e translação em uma única operação:

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ w' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{2.1}$$

onde $r_{11} \cdots r_{33}$ definem uma rotação e t_x, t_y e t_z definem uma translação. Nesse caso temos seis graus de liberdade (três para rotação, mais três componentes da translação). Chamamos a esse tipo de transformação de *transformação euclidiana*. Em uma transformação euclidiana, todas as dimensões e ângulos dos objetos são preservadas.

Além da transformação euclidiana, outras três importantes categorias de transformações no espaço são: similaridade (ou métrica), afim e projetiva. Uma introdução às propriedades de cada uma destas categorias é importante para a compreensão dos tipos de reconstrução que podemos realizar, e serão apresentadas a seguir.

Se, além de uma transformação euclidiana, desejarmos permitir uma mudança de escala, podemos modificar a matriz de transformação para:

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ w' \end{bmatrix} = \begin{bmatrix} \sigma r_{11} & \sigma r_{12} & \sigma r_{13} & t_x \\ \sigma r_{21} & \sigma r_{22} & \sigma r_{23} & t_y \\ \sigma r_{31} & \sigma r_{32} & \sigma r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{2.2}$$

onde σ denota nosso fator de escala. Nesse caso temos uma *transformação métrica*, ou de similaridade. Os valores absolutos das dimensões não são mais preservados, devido ao fator de escala, mas sua relação (isto é, se um lado de um triângulo tem o dobro do tamanho de outro) e os ângulos ainda são preservados. Temos agora sete graus de liberdade (os mesmo

seis da transformação euclidiana, mais escala).

Se desejarmos ainda mais liberdade na transformação, podemos permitir escalas diferentes em cada eixo, além do efeito de cisalhamento (movimento de pontos na direção de um plano, em dimensão proporcional a sua distância ao dito plano). Nesse caso temos uma *transformação afim*, cuja forma é dada por:

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ w' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{2.3}$$

E temos um total de doze graus de liberdade. Neste tipo de transformação preservamos o paralelismo entre planos e retas, e as distâncias relativas de pontos em uma mesma reta.

Ainda podemos remover algumas restrições para chegarmos a forma mais geral de transformação no espaço, a *transformação projetiva*, dada pela matriz:

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ w' \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{2.4}$$

O elemento p_{44} é apenas um fator de escala, e podemos fixá-lo em $p_{44} = 1$. Dessa forma temos quinze graus de liberdade, e nossa transformação preserva apenas uma propriedade, a razão cruzada, que é uma razão das distâncias entre quatro pontos.

A figura (2.1) apresenta um exemplo de cada tipo de transformação apresentada.

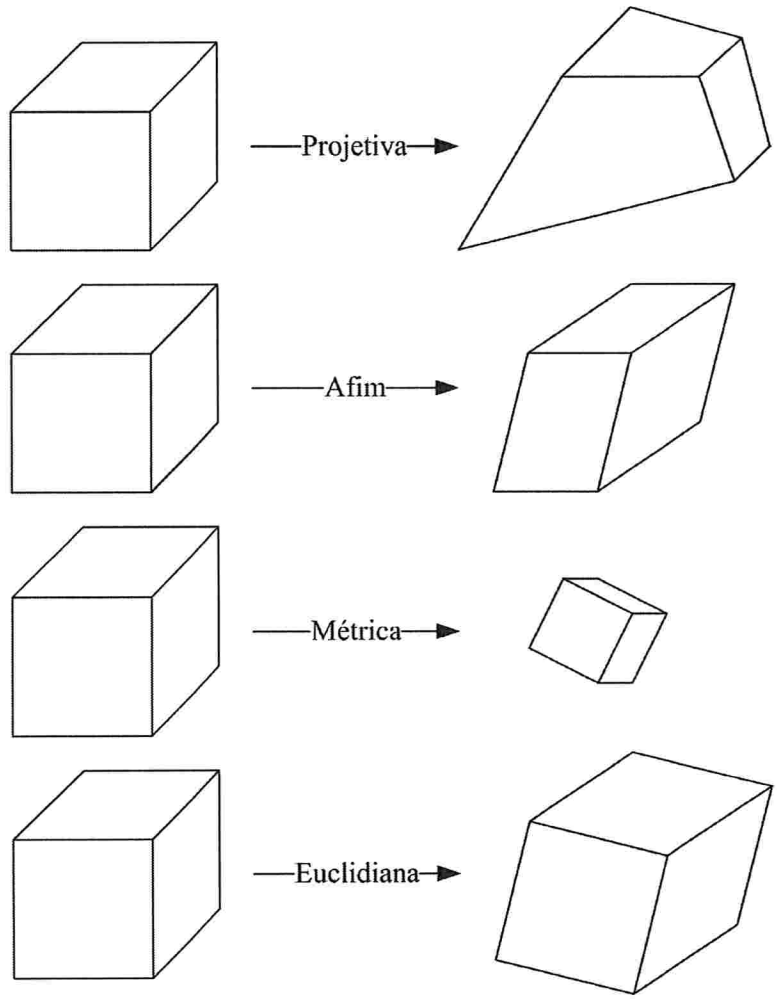

Figura 2.1: Os tipos comuns de transformações geométricas no espaço, da menos restritiva (transformação projetiva) à mais restritiva (transformação euclidiana). Quanto mais restritiva, mais propriedades do objeto são preservadas.

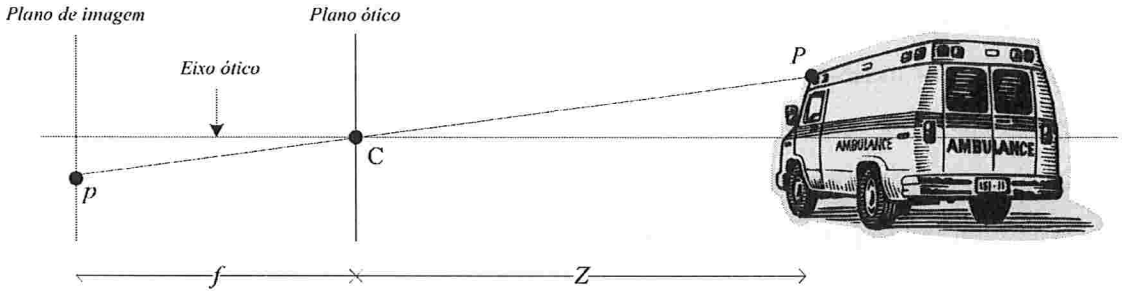

Figura 2.2: O modelo de câmera *pinhole*.

2.1.2 Câmeras e Transformações Projetivas

As propriedades geométricas envolvidas na representação de um objeto 3D em um plano 2D já eram conhecidas desde a época dos pintores renascentistas.

O modelo de câmera mais simples utilizado para tal transformação é conhecido como *modelo pinhole* (figura 2.2), que corresponde ao comportamento de feixes de luz passando por um pequeno buraco em uma superfície plana opaca.

Geometricamente podemos definir tal modelo a partir de:

- um plano Γ, chamado de *plano da retina* ou *plano da imagem*,
- um ponto C, não pertencente a Γ: o *centro ótico*.

A projeção m de um ponto M no espaço é a intersecção do *raio ótico* (C, M) com o plano da imagem. O *eixo ótico* é a linha que passa por C e é perpendicular ao plano da imagem. Ela cruza este plano no *ponto principal* c.

Considerando um sistema ortonormal de coordenadas sobre o plano da imagem podemos definir um sistema de coordenadas tri-dimensional chamado de *sistema de coordenadas da câmera*, centrado no centro ótico C com dois eixos paralelos ao plano da imagem e o terceiro paralelo ao eixo ótico.

A *distância focal* ou *comprimento focal* é a distância entre o ponto C e o plano Γ. Nesta descrição vamos utilizar a distância focal como a unidade no sistema de coordenadas. A mudança desta unidade corresponde a uma simples mudança de escala.

De maneira geral, a projeção de um ponto $P = (X, Y, Z, w)^T$ no sistema de coordenadas global em um ponto $p = (x', y', 1)$ no sistema de coordenadas da câmera é dada por:

$$\mathcal{P} \simeq \underbrace{\begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathcal{P}_0} \underbrace{\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0_3^T & 1 \end{bmatrix}}_{\mathcal{D}} = \mathbf{A}[\mathbf{R}\ \mathbf{t}] \tag{2.5}$$

onde:

- $\mathcal{D}$ descreve a posição e orientação da câmera no sistema de coordenadas global. É uma matriz 4×4 descrevendo a mudança do sistema de coordenadas como uma rotação $\mathbf{R}$ e uma translação $\mathbf{t}$, os chamados *parâmetros extrínsecos* da câmera. Essa matriz nos leva do sistema de coordenadas global para o sistema de coordenadas da câmera;

- $\mathcal{P}_0$ é a matriz de projeção, que descarta a coordenada homogênea, mantendo as outras coordenadas. Ela é particularmente simples devido a nossa escolha do sistema de coordenadas;

- $\mathbf{A}$ descreve as características da câmera, ou, mais precisamente, do sistema da imagem. Como uma matriz 3×3 ela representa uma mudança no sistema de coordenadas da imagem. Seus cinco parâmetros são chamados de *parâmetros intrínsecos* da câmera. α_u e α_v representam a distância focal expressa em unidades de *pixels* em cada direção. Eles representam a magnificação total do sistema de imagem resultante da ótica e da amostragem da imagem. A razão entre eles, chamada de *razão aparente* (*aspect ratio*)

é normalmente fixa, porém nem sempre igual a 1. (u_o, v_o) representam as coordenadas do ponto principal. O parâmetro γ é o cizalhamento (*skew*) e é zero exceto em alguns casos particulares (como *pixels* não ortogonais). Também é conhecida como *matriz de calibração*.

Uma câmera para a qual os parâmetros intrínsecos são conhecidos é chamada de *calibrada*, enquanto uma para a qual tais parâmetros são desconhecidos é chamada de *não-calibrada*.

Quando conhecemos $\mathbf{A}$ temos condições de extrair métricas da imagem, pois somos capazes de medir os ângulos entre raios óticos. Se também conhecemos $\mathcal{D}$ então conseguimos relacionar a posição da câmera com o sistema de coordenadas global ou com o sistema de coordenadas de outras câmeras. A maneira clássica de calcularmos os parâmetros intrínsecos de uma câmera é determinando sua matriz de projeção $\mathcal{P}$ utilizando pontos de controle conhecidos no espaço 3D.

Tendo os pontos 3D, realizamos a seguinte operação: sejam $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ vetores representando as três linhas de $\mathcal{P}$. Para cada correspondência $m \leftrightarrow M$ entre 2D e 3D, obtemos duas equações lineares sobre as entradas de $\mathcal{P}$:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{m} \simeq \mathcal{P}\mathbf{M} = \begin{bmatrix} \mathbf{U}^T\mathbf{M} \\ \mathbf{V}^T\mathbf{M} \\ \mathbf{W}^T\mathbf{M} \end{bmatrix} \text{; portanto } \begin{cases} u\mathbf{W}^T\mathbf{M} - \mathbf{U}^T\mathbf{M} = 0. \\ v\mathbf{W}^T\mathbf{M} - \mathbf{V}^T\mathbf{M} = 0. \end{cases} \tag{2.6}$$

Os pontos de referência M são medidos em algum sistema de coordenadas 3D, e suas projeções m detectadas. Em geral, um objeto especial é construído de maneira que ambas as operações podem ser feitas com boa precisão.

Como $\mathcal{P}$ possui 11 entradas independentes, a partir de seis correspondências 2D para 3D é possível determinar a matriz de projeção.

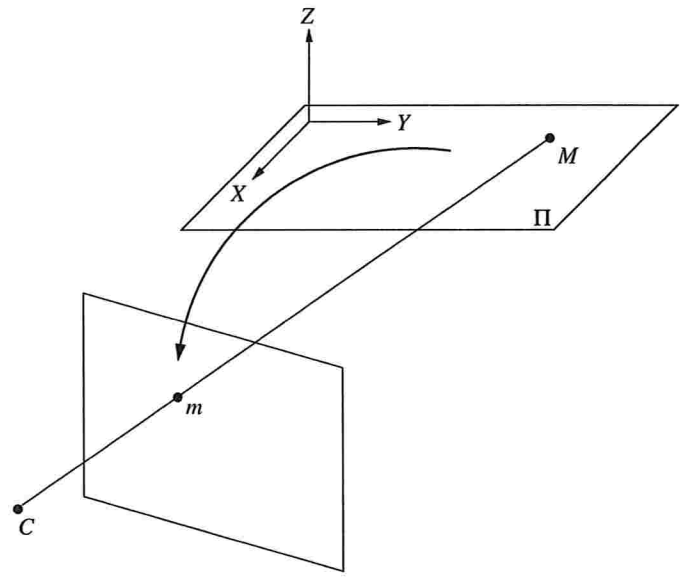

Figura 2.3: Homografia entre um plano no espaço e o plano de imagem.

2.1.3 Homografias entre Plano e Imagem

Uma transformação projetiva muito importante para o estudo da geometria de múltiplas imagens é a *homografia* entre um plano Π no espaço e o plano de imagem.

Se escolhemos o sistema de coordenadas de maneira que os dois primeiros eixos definam o plano Π (figura 2.3) , a projeção dos pontos de Π pode ser vista como uma transformação entre dois espaços $\mathbb{P}^2$, já que para tais pontos:

$$\underbrace{\begin{bmatrix} x \\ y \\ w \end{bmatrix}}_{\mathbf{m}} \simeq \mathcal{P} \begin{bmatrix} X \\ Y \\ 0 \\ T \end{bmatrix} = \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{14} \\ h_{21} & h_{22} & h_{24} \\ h_{31} & h_{32} & h_{34} \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} X \\ Y \\ T \end{bmatrix}}_{\mathbf{p}} . \tag{2.7}$$

Cada correspondência entre pontos (m, p) nos dá duas equações:

$$\frac{x}{w} = \frac{h_{11}X + h_{12}Y + h_{13}T}{h_{31}X + h_{32}Y + h_{33}T}, \quad \frac{y}{w} = \frac{h_{21}X + h_{22}Y + h_{23}T}{h_{31}X + h_{32}Y + h_{33}T}$$

das quais podemos extrair um sistema linear sobre os elementos de $\mathbf{H}$:

$$\begin{cases} h_{11}wX + h_{12}wY + h_{13}wT + h_{31}xX - h_{32}xY - h_{33}xT = 0 \\ h_{21}wX + h_{22}wY + h_{23}wT + h_{31}yX - h_{32}yY - h_{33}yT = 0 \end{cases}$$

$\mathbf{H}$ possui oito graus de liberdade devido ao fator de escala, portanto a partir de quatro correspondências (m, p), com a restrição que nenhum trio de pontos pode ser colinear (pois neste caso as equações não seriam linearmente independentes), $\mathbf{H}$ pode ser determinada resolvendo um sistema linear de equações.

Com $\mathbf{H}$ podemos determinar as posições de pontos em Π a partir de um único quadro. A transformação $\mathbf{H}$ é chamada de *homografia* ou *transformação projetiva* de $\mathbb{P}^2$. De maneira geral, uma homografia é uma transformação $\mathbf{H}$ de $\mathbb{P}^n$ que é linear em coordenadas projetivas e inversível. Tal transformação pode ser descrita por uma matriz não singular $\mathbf{H}$ $(n+1) \times (n+1)$, tal que a imagem de $\mathbf{x}$ é $\mathbf{x}'$:

$$\mathbf{x}' \simeq \mathbf{H}\mathbf{x}.$$

Para definir uma homografia em $\mathbb{P}^n$ precisamos de dois conjuntos de $n + 2$ pontos, sendo que nenhum conjunto de $n + 1$ deles pode ser linearmente dependente. Tais conjuntos são chamados de uma *base projetiva.*

2.1.4 Projeção Afim

A matriz projetiva $\mathcal{P}$ deve ter posto 3, caso contrário sua imagem seria uma linha projetiva, ao invés de um plano projetivo. Como ela possui 4 colunas, seu espaço nulo deve ter dimensão 1;

qualquer vetor $\mathbf{C}$ neste espaço nulo define um ponto C para o qual a projeção não é definida: este ponto é o *centro ótico*.

Vamos particionar a matriz $\mathcal{P}$ na concatenação de uma sub-matriz $\mathbf{P}_{3\times3}$ e um vetor $\mathbf{p}_{3\times1}$. A origem do sistema de coordenadas, o ponto $[0,0,0,1]$, é projetado em p.

O centro ótico também é decomposto, separando sua última coordenada das outras três:

$$\mathcal{P} = [\mathbf{Pp}], \quad \tilde{C} = \begin{bmatrix} \mathbf{C} \\ c \end{bmatrix}. \tag{2.8}$$

A equação que determina o centro ótico é $\mathcal{P}\tilde{C} = 0$. Utilizando a decomposição apresentada, $\mathcal{P}\tilde{C} = \mathbf{PC} + \mathbf{p}c$, temos $\mathbf{PC} = -\mathbf{p}c$. Assim, se $\det(\mathbf{P}) \neq 0$, então a solução é dada por:

$$\tilde{C} \simeq \begin{bmatrix} -\mathbf{P}^{-1}\mathbf{p} \\ 1 \end{bmatrix}. \tag{2.9}$$

Portanto o centro ótico é finito. Quando $\det(\mathbf{P}) = 0$, podemos verificar, utilizando o fato que $\mathcal{P}$ possui posto 3, que o centro ótico está no plano no infinito (i.e. $c = 0$).

Qualquer matriz projetiva que venha de um sistema físico (real) tem de satisfazer $\det(\mathbf{P}) \neq 0$, já que o centro ótico deve estar em um espaço afim (chamamos isso de *projeção perspectiva*), e, para simplificar, será o caso que vamos assumir daqui por diante.

2.2 Visão Estéreo

Vamos agora estudar o caso onde possuímos uma segunda imagem, obtida com a mesma câmera, mas em posições diferentes. Dois pontos, m na primeira imagem, e m' na segunda imagem, são chamados de *pontos correspondentes* ou de *correspondências* se ambos são projeções de um mesmo ponto 3D M. Nesta configuração podemos realizar as seguintes perguntas:

- Dado um ponto m na primeira imagem, onde está seu ponto correspondente m' na segunda imagem?
- Qual é a geometria 3D da cena?
- Qual é a posição relativa das duas câmeras?

Vimos que, de uma única imagem, mesmo se conhecermos os parâmetros da câmera, apenas podemos determinar o eixo ótico por m, não a posição 3D do ponto M. Com duas imagens e a correspondência (m, m'), podemos intersectar os raios óticos de m e m' e assim determinar M (Figura 1.3).

Assim, para encontrar a posição 3D de um ponto M, precisamos encontrar a posição de dois raios óticos em um mesmo sistema de coordenadas. Portanto precisamos conhecer a posição da segunda câmera em relação à primeira, que nós chamamos de *movimento* da câmera.

Algebricamente, tendo as matrizes de projeção $\mathcal{P}$ e $\mathcal{P}'$, encontramos M a partir de m e m' resolvendo o sistema de quatro equações:

$$\begin{cases} \mathcal{P}\mathbf{M} & \simeq & \mathbf{m} \\ \mathcal{P}'\mathbf{M} & \simeq & \mathbf{m}' \end{cases}. \tag{2.10}$$

Dessa maneira, para determinar a estrutura 3D de uma cena, precisamos antes determinar as matrizes de projeção $\mathcal{P}$ e $\mathcal{P}'$, que codificam a geometria das câmeras. Caso não tenhamos essa informação, devemos inicialmente estimá-la. Este caso, onde a posição das câmeras não é fixa e pré-determinada, chamamos de *estrutura a partir do movimento* (*structure from motion*).

Os dois problemas de estimação de movimento e determinação de estrutura são interrelacionados e vamos chamá-los de *problema de reconstrução*.

2.2.1 Homografia entre duas imagens de um plano

Vamos primeiro estudar o caso particular onde os pontos 3D estão em um mesmo plano Π. Este é um caso importante, pois planos aparecem com freqüência em muitas cenas.

Como já vimos, existe uma homografia entre o plano de imagem da primeira câmera e o plano Π, e também uma homografia entre o plano de imagem da segunda câmera e o plano Π, assim, por composição, existe uma homografia H entre os dois planos de imagem chamada *homografia planar* (figura 2.4), porque ela é induzida pelo plano Π e descrita por uma matriz $H_{3\times3}$. Se m e m' são projeções de um ponto M pertencente a Π, então:

$$\mathbf{m}' \simeq \mathbf{H}\mathbf{m}.$$

Invertendo as funções das duas imagens, transformamos $\mathbf{H}$ em sua inversa. Assim como no caso descrito na sessão (2.1.3), $\mathbf{H}$ pode ser determinada a partir de 4 correspondências. Uma vez que $\mathbf{H}$ é conhecida, para qualquer projeção m de um ponto de Π, é possível prever a posição de sua correspondência na outra imagem.

Um caso especial importante é quando Π é o plano no infinito Π_∞. Então $\mathbf{H}_\infty$ possui uma expressão particularmente simples em função das duas matrizes projetivas $\mathcal{P}$ e $\mathcal{P}'$, obtida utilizando a mesma decomposição apresentada em (2.1.4):

$$\mathbf{H}_\infty \simeq \mathbf{P}'\mathbf{P}^{-1}. \tag{2.11}$$

2.2.2 Geometria Epipolar

Dado um ponto qualquer no espaço e duas câmeras, existe grande liberdade na posição do ponto m' correspondente ao ponto m, porque esta posição depende da posição do ponto 3D M sobre o raio ótico. No entanto, geometricamente, esta posição não é arbitrária: M deve

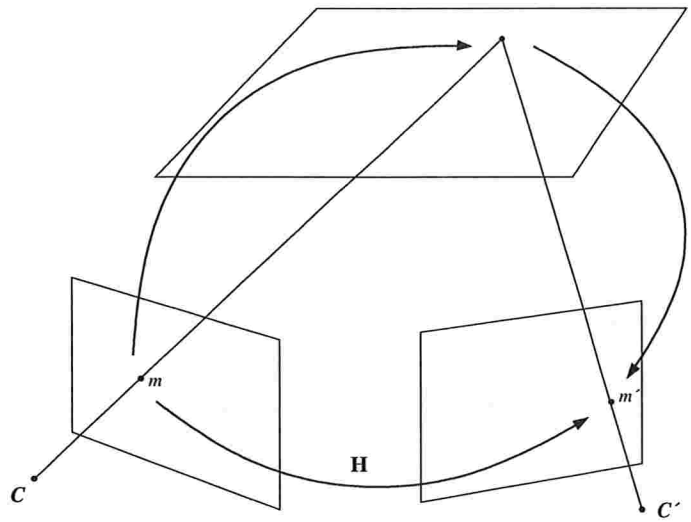

Figura 2.4: A homografia entre dois planos de imagens é obtida pela composição de duas homografias.

estar sobre o raio ótico de m, portanto m' está necessariamente localizado na projeção desse raio ótico na segunda câmera. Esta linha é chamada de *linha epipolar* do ponto m na segunda imagem. Se conseguirmos encontrar tal linha, então quando procurarmos pela correspondência de m, não precisaremos pesquisar por toda a imagem, mas apenas sobre a linha epipolar.

Uma maneira de observar isso é perceber que os raios óticos (M, m) e (M, m') e a linha que une os dois centros óticos (C, C') definem um plano, chamado *plano epipolar* (figura 2.5).

A Matriz Fundamental

A matriz que descreve a relação entre um ponto m e sua linha epipolar l'_m na segunda imagem é chamada de *matriz fundamental*: $\mathbf{l}'_m = \mathbf{Fm}$.

Se dois pontos m e m' são correspondências, então m' pertence à linha epipolar de m, e

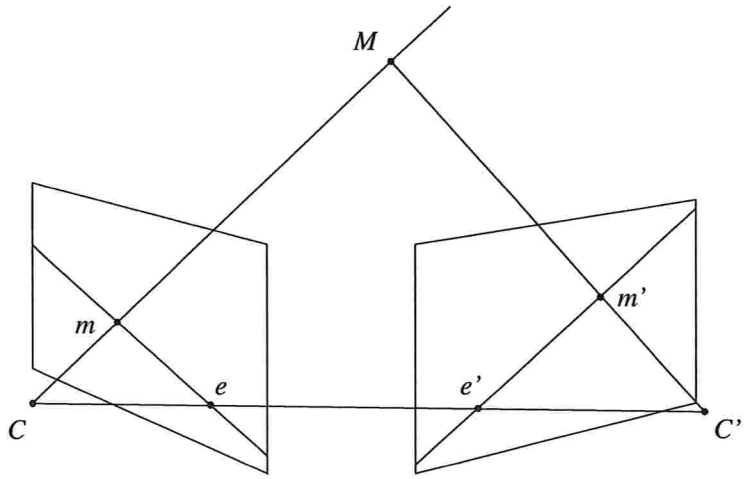

Figura 2.5: Geometria epipolar entre duas imagens.

tal relação satisfaz a *restrição epipolar*:

$$\mathbf{m}'^T\mathbf{F}\mathbf{m} = 0 \tag{2.12}$$

e a relação inversa entre as duas imagens é definida pela transposta de $\mathbf{F}$.

A matriz fundamental depende apenas da configuração das câmeras (parâmetros intrínsecos e extrínsecos) e não dos pontos 3D na cena. No caso mais genérico, onde não assumimos nenhuma relação espacial entre os pontos no espaço, a única informação disponível vem das *correspondências projetivas.*

Como todos os raios óticos contém o centro ótico C da primeira câmera, todas as linhas epipolares contém a projeção de C na segunda câmera, chamada de *epipolo* (os pontos e e e' na Figura 2.5). Isto implica que $\mathbf{e}'^T\mathbf{F}\mathbf{m} = 0$, para qualquer m, portanto $\mathbf{e}'^T\mathbf{F} = 0$, ou, de maneira equivalente, $\mathbf{F}^T e' = 0$. Invertendo as duas imagens, $\mathbf{F}\mathbf{e} = 0$. Podemos concluir que $\mathbf{F}$

é uma matriz de posto 2:

$$\det(\mathbf{F}) = 0.$$

Calculando a Matriz Fundamental

Cada correspondência (m, m') nos dá uma equação (2.12), portanto com um número suficiente de correspondências podemos determinar $\mathbf{F}$. Nenhuma outra informação sobre as câmeras e a cena é necessária.

A equação (2.12) é linear sobre as entradas de $\mathbf{F}$ e pode ser re-escrita como:

$$\mathbf{U}^T\mathbf{f} = 0,$$

onde $\mathbf{m} = [u, v, 1]^T$ e $\mathbf{m}' = [u', v', 1]^T$, portanto:

$$\mathbf{U} = [uu', vu', u', uv', vv', v', u, v, 1]^T, \tag{2.13}$$

$$\mathbf{f} = [F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}]^T. \tag{2.14}$$

Combinando as linhas de $\mathbf{U}$ para cada correspondência temos um sistema linear na forma $\tilde{\mathbf{U}}\mathbf{f} = 0$. Com sete pontos podemos calcular $\mathbf{F}$ utilizando a restrição $\det(\mathbf{F}) = 0$, no entanto como esta restrição é cúbica, podem existir três soluções. Com oito correspondências existe uma única solução que pode ser obtida linearmente.

Na prática nós temos mais que oito correspondências, mas elas não são exatas, portanto podemos procurar por uma solução por mínimos-quadrados:

$$\min_{\mathbf{f}} \|\tilde{\mathbf{U}}\mathbf{f}\| \text{ sujeito a } \|\mathbf{f}\| = 1 \tag{2.15}$$

A restrição $\|\mathbf{f}\| = 1$ é necessária pois $\mathbf{F}$ é definida a menos de um fator de escala.

Este algoritmo apresenta duas dificuldades. Primeiro, diferente do caso de sete pontos, a restrição de posto não é mais satisfeita. Segundo, a função de erro utilizada na equação (2.15) foi obtida de maneira algébrica, sem darmos importância a sua relevância geométrica. No entanto, se re-normalizarmos as coordenadas dos *pixels* para o intervalo $[-1, 1]$ para melhorar o condicionamento numérico da matriz $\tilde{\mathbf{U}}$, é possível obter resultados aceitáveis.

Homografias e a Matriz Fundamental

Vimos na sessão (2.2.1) que, dado um plano Π, a correspondência é determinada por uma homografia $\mathbf{H}$. Em outras palavras, $\mathbf{H}$ pode ser utilizada para encontrar pontos correspondentes na segunda imagem a partir de pontos na primeira imagem.

Também vimos que para pontos quaisquer no espaço 3D, a matriz fundamental pode ser utilizada para restringir as correspondências sobre uma direção, a da linha epipolar.

Dadas duas câmeras e a matriz fundamental, uma homografia planar é definida por seu plano associado. Como um plano depende de três parâmetros, e a homografia de oito parâmetros, nem todas as matrizes inversíveis 3×3 definem uma homografia, portanto $\mathbf{H}$ deve satisfazer seis restrições, dado $\mathbf{F}$.

Por outro lado, uma homografia restringe a matriz fundamental, pois ela pode ser utilizada para gerar um ponto na linha epipolar de qualquer ponto: se m é um ponto na primeira imagem, então seu raio ótico intersecta o plano Π em M_Π. $\mathbf{Hm}$ representa a projeção de M_Π na segunda imagem.

Como, por construção, M_Π pertence ao raio ótico de m, o ponto $\mathbf{Hm}$ pertence à linha epipolar de m. Assim, dado $\mathbf{H}$ é suficiente conhecer o epipolo e' para se determinar a matriz fundamental: a linha epipolar l'_m contém $\mathbf{Hm}$ e o epipolo e', portanto $\mathbf{l}'_m = e' \times Hm$. Como,

por definição de $\mathbf{F}$, $\mathbf{l}'_m = \mathbf{F}\mathbf{m}$, concluímos que:

$$\mathbf{F} \simeq [\mathbf{e}']_\times \mathbf{H}. \tag{2.16}$$

Tendo $\mathbf{F}$ também conseguimos encontrar uma transformação planar utilizando a *matriz especial* definida como:

$$\mathbf{S} = [\mathbf{e}']_\times \mathbf{F}. \tag{2.17}$$

Pode ser verificado que $\mathbf{S}$ satisfaz $\mathbf{F} \simeq [\mathbf{e}']_\times \mathbf{F}$, portanto é equivalente sabermos $\mathbf{F}$ ou $\mathbf{S}$. Pode ser demonstrado que o plano gerador de $\mathbf{S}$ passa através do centro ótico C' [FL01], portanto $\mathbf{S}$ é singular, e se projeta na linha $\mathbf{e}'$ na segunda imagem. Por estar atrelado ao sistema de duas câmeras, ele é conhecido como o *plano intrínseco* Π_S.

Também podemos chegar à matriz fundamental em função das matrizes de projeção das câmeras. Sejam as duas matrizes de projeção $\mathcal{P} = [\mathbf{P}\mathbf{p}]$ e $\mathcal{P}' = [\mathbf{P}'\mathbf{p}']$, assumindo apenas o caso perspectivo ($\det(\mathbf{P}) \neq 0$). O epipolo $\mathbf{e}'$ é a projeção $\mathcal{P}'\tilde{C}$ do centro ótico $\tilde{C}$, portanto:

$$\mathbf{e}' \simeq \mathcal{P}\tilde{C} \simeq [\mathbf{P}'\mathbf{p}'] \begin{bmatrix} -\mathbf{P}^{-1}\mathbf{p} \\ 1 \end{bmatrix} \simeq \mathbf{p}' - \mathbf{P}'\mathbf{P}^{-1}\mathbf{p}.$$

Agora que obtemos o epipolo, aplicamos a equação (2.16) para encontrarmos $\mathbf{F}$. Apenas necessitamos agora de uma homografia entre as duas imagens. Neste caso podemos utilizar a definida pelo plano no infinito, dada pela equação (2.11). Utilizando-a, obtemos a expressão da matriz fundamental em função de duas projeções:

$$\mathbf{F} = [\mathbf{p}' - \mathbf{P}'\mathbf{P}^{-1}\mathbf{p}]_\times \mathbf{P}'\mathbf{P}^{-1}$$

RANSAC e LMedS

Em geral possuímos um grande conjunto de correlações, com precisão variável. Para estimarmos a matriz fundamental precisamos de algum método para encontrar a estimativa que dê o melhor resultado para todos os pontos apresentados e também seja capaz de descartar pontos que apresentem resultados muito diferentes, chamaremos tais valores de "aberrantes".

Neste trabalho foram estudadas duas técnicas parecidas, que se baseiam na escolha de conjuntos aleatórios de correlações e em métricas para definir qual desses conjuntos nos dá o melhor resultado [HA04, FL01, CRB99, FB81]. Primeiro devemos definir melhor o que chamamos de "melhor resultado".

Dada uma matriz fundamental $\mathbf{F}$ e um conjunto de $\mathbf{m}_i$ pontos na primeira câmera e $\mathbf{m}'_i$ pontos na segunda câmera, sabemos que $\mathbf{m}'_i\mathbf{F}\mathbf{m}_i = 0$, isto é, o ponto $\mathbf{m}'_i$ deve estar sobre a linha epipolar de $\mathbf{m}_i$ na segunda imagem, e, inversamente, $\mathbf{m}_i$ deve estar na linha epipolar de $\mathbf{m}'_i$ na primeira imagem. Assim sendo, uma boa estimativa da qualidade de uma matriz fundamental estimada é a distância entre os pontos e as linhas epipolares a que eles devem pertencer.

Infelizmente, existe ainda uma outra dificuldade: pontos cuja correlação esteja errada (artefatos resultantes de "erros" do algoritmo de rastreamento) podem influenciar adversamente a matriz fundamental estimada. Portanto devemos encontrar uma forma de detectar e remover tais pontos.

Tanto o algoritmo *RANSAC* (*RANdom SAmple Consensus*) e o *LMedS* (*least-median-of-squares*) nos dão uma maneira de contornar esse problema e encontrar uma estimativa adequada da matriz fundamental.

Dada a proximidade dentre os dois métodos, podemos descrevê-los em um mesmo fluxo, sejam p correspondências $(\mathbf{m}_i, \mathbf{m}'_i)$, fazemos:

1. Utilizamos uma técnica de Monte Carlo para selecionar n conjuntos aleatórios com $q = 7$

correspondências cada (7 é o número mínimo de correspondências para determinarmos a matriz fundamental).

2. Para cada sub-conjunto (indexaremos por j), calculamos uma estimativa da matriz fundamental (dada uma característica do sistema definido por $\mathbf{m}_i'\mathbf{F}\mathbf{m}_i$, podemos ter até 3 matrizes diferentes para cada conjunto de 7 pontos).

3. Para cada matriz calculada no passo anterior, calculamos as distâncias residuais $r_i = \mathbf{m}_i'\mathbf{F}\mathbf{m}_i$.

   - No caso do *RANSAC*, crescemos esse conjunto inicial com todas as características tais que $r_i^2 < \sigma^2$, onde σ é um parâmetro definindo a distância máxima, em unidades da imagem (*pixels*) de uma correlação e sua linha epipolar.
   - No caso do *LMedS*, calculamos $M_j = mediana[d^2(\mathbf{m}_i', \mathbf{F_j m_i}) + d^2(\mathbf{m}_i, \mathbf{F^T m_i'})]$

4. Para *RANSAC* o melhor conjunto é aquele com o maior número de pontos que satisfazem a restrição acima, no caso de *LMedS* o melhor conjunto é aquele com menor M_j. A matriz fundamental estimada a partir de cada um desses conjuntos é o resultado do algoritmo.

Outra pergunta importante é como definimos o número n de amostras aleatórias e como selecionamos as correlações que fazem parte de cada amostra.

Um subconjunto é "bom" se ele consiste de q correlações precisas. Assumindo que no total de correspondências temos uma fração de até ε elementos aberrantes, a probabilidade que pelo menos um dentre os n conjuntos seja boa, é dada por:

$$P = 1 - [1 - (1 - \varepsilon)^q]^n.$$

Com o requisito que P seja próximo a 1, podemos determinar n para valores de ε e q:

$$n = \frac{\log(1-P)}{\log[1-(1-\varepsilon)^q]}.$$

Por exemplo, para $\varepsilon = 40\%$ e $P = 0.99$ temos $n = 163$.

Para um melhor resultado, os 7 pontos de cada conjunto selecionado devem estar bastante espalhados sobre as imagens. Para auxiliar isso, uma técnica para segmentar as correlações por regiões (*bucketing*) pode ser utilizada.

2.2.3 Reconstrução

O problema de reconstrução estéreo envolve determinarmos as matrizes de projeção $\mathcal{P}$ e $\mathcal{P}'$, assim como os pontos 3D M_i, dado um conjunto de N correspondências (m_i, m_i'). No entanto a solução não é única, pois depende da escolha de um sistema de coordenadas, expresso pela matriz $\mathcal{H}_{4\times4}$. Se $(\mathcal{P}, \mathcal{P}', \mathbf{M}_1, \ldots, \mathbf{M}_N)$ é uma solução para o problema, então $(\mathcal{P}\mathcal{H}^{-1}, \mathcal{P}'\mathcal{H}^{-1}, \mathcal{H}\mathbf{M}_1, \ldots, \mathcal{H}\mathbf{M}_N)$ também é uma solução, pois:

$$\left\{ \begin{array}{lcll} \mathbf{m} & \simeq & \mathcal{P}\mathbf{M} & = & (\mathcal{P}\mathcal{H}^{-1})(\mathcal{H}\mathbf{M}) \\ \mathbf{m}' & \simeq & \mathcal{P}'\mathbf{M} & = & (\mathcal{P}'\mathcal{H}^{-1})(\mathcal{H}\mathbf{M}) \end{array} \right. . \tag{2.18}$$

Em outras palavras, todos os pares de matrizes projetivas $(\mathcal{P}\mathcal{H}, \mathcal{P}'\mathcal{H})$, onde $\mathcal{H}$ é uma transformação projetiva arbitrária, são potencialmente equivalentes. No, entanto, se conseguirmos alguma restrição sobre a correspondência, talvez possamos limitar a ambigüidade sobre $\mathcal{H}$ reforçando que tal restrição deve ser satisfeita pelo par $(\mathcal{P}\mathcal{H}, \mathcal{P}'\mathcal{H})$.

Veremos adiante que, para conseguirmos uma *reconstrução de similaridade* (isto é, a menos de uma transformação euclidiana e escala), precisamos de algum conhecimento anterior sobre o mundo, que nos permita determinar o plano no infinito e os parâmetros intrínsecos da

câmera, ou alguma informação anterior sobre a câmera, que nos permite tentar alguma forma de *auto-calibração.*

Reconstrução Projetiva

No caso em que temos apenas as correspondências (m, m') entre pontos, toda a informação de correspondência projetiva pode ser resumida por uma matriz fundamental $\mathbf{F}$. Um par de matrizes projetivas $(\mathcal{P}, \mathcal{P}')$ é uma solução válida se, e somente se, sua matriz fundamental é compatível com a correspondência entre pontos, ou seja $\mathbf{F}$.

É possível demonstrar que um par $(\mathcal{P}\mathcal{H}, \mathcal{P}'\mathcal{H})$ possui uma matriz fundamental $\mathbf{F}$ se, e somente se, ele é da forma:

$$\begin{cases} \mathcal{P} & \simeq & [\mathbf{I}_3\mathbf{0}_3]\mathcal{H} \\ \mathcal{P}' & \simeq & [\mathbf{H}\mu\mathbf{e}']\mathcal{H} \end{cases} \text{com } \mathcal{H} = \begin{bmatrix} \mathcal{P} \\ \Pi^T \end{bmatrix} \tag{2.19}$$

onde

- $\mathcal{P}$ é uma matriz de projeção arbitrária (11 parâmetros), Π é a equação projetiva de um plano arbitrário (3 parâmetros), μ é uma constante arbitrária (1 parâmetro), que é a escala comum de $\mathcal{P}$ e Π na matriz $\mathcal{H}$. Juntos, esses 15 parâmetros formam a ambigüidade projetiva na reconstrução: a escolha arbitrária de uma base 3D, ou, de maneira equivalente, da matriz $\mathcal{H}$.

- Os elementos em $\mathcal{P}'$ são: o epipolo $\mathbf{e}'$ de $\mathbf{F}$ na segunda imagem e a homografia $\mathbf{H}$, compatível com $\mathbf{F}$ e gerada pelo plano Π. Estas entidades são unicamente determinadas por $\mathbf{F}$ e Π.

Assim, particionamos os parâmetros de um par de matrizes projetivas em dois tipos: a correspondência projetiva do par de câmeras, inclusa na matriz fundamental (7 parâmetros), e

uma transformação projetiva (15 parâmetros) que representa a ambigüidade na reconstrução.

Assim, para obtermos uma reconstrução projetiva:

- Obtemos pares de correspondências (m_i, m_i').
- Encontramos a matriz fundamental usando a equação (2.12).
- Calculamos $\mathbf{e}'$ por $\mathbf{F}^T\mathbf{e}' = 0$.
- Calculamos a matriz especial $\mathbf{S} = [\mathbf{e}']_\times \mathbf{F}$.
- Calculamos um par de matrizes projetivas, chamadas de *representação projetiva canônica*, obtida a partir da equação (2.19), utilizando $\mathbf{S}$ como uma instância particular de $\mathbf{H}$:

$$\begin{cases} \mathcal{P} & \simeq & [\mathbf{I}_3\mathbf{0}_3], \\ \mathcal{P}' & \simeq & [\mathbf{S}\mu\mathbf{e}']. \end{cases} \tag{2.20}$$

- Encontramos M_i utilizando a equação (2.10).

O algoritmo anterior geralmente nos dá uma reconstrução com uma distorção projetiva significativa, porque a transformação projetiva não preserva ordenação nas profundidades, sequer distâncias relativas. Isto significa, por exemplo, que não podemos utilizar tal reconstrução para extrair medidas precisas.

Reconstrução Afim

Acabamos de ver que, a partir apenas de correspondências entre duas imagens somos capazes de reconstruir pontos e matrizes projetivas a menos de uma transformação projetiva. Se, adicionalmente, possuirmos alguma informação afim, podemos reduzir a ambigüidade da reconstrução de uma transformação projetiva em $\mathbb{P}^3$ para uma transformação afim em $\mathbb{P}^3$. Isto

significa que podemos chegar aos pontos "reais" por uma transformação que é mais restrita e induz menos deformações, preservando mais propriedades.

No entanto a informação afim tem de vir de um conhecimento adicional da cena ou das câmeras. Apenas correspondências não nos proporcionam informação afim.

Uma transformação afim é um caso particular de transformação perspectiva que preserva o plano no infinito Π_∞. É fácil ver que uma transformação $\mathcal{A}$ conserva Π_∞ se, e somente se, a última linha da matriz $\mathcal{A}$ é da forma $[0, 0, 0, \mu]$, com $\mu \neq 0$. Como esta matriz é definida a menos de um fator de escala, podemos assumir $\mu = 1$, e então a transformação $\mathcal{A}$ é descrita por sua sub-matriz $\mathbf{A}_{3\times3}$ e pelas primeiras três coordenadas da última coluna, o vetor $\mathbf{b}$:

$$\mathcal{A} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0}_3^T & 1 \end{bmatrix},$$

que nos dá a descrição clássica de uma transformação afim em $\mathbb{R}^3$: $\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}$. Pontos no infinito representam direções, uma transformação afim, no entanto, preserva o paralelismo. Outras propriedades como a ordem nas profundidades e a proporção na distância de três pontos alinhados também são preservadas. Isto limita a distorção gerada na reconstrução.

O espaço afim é caracterizado pelo plano Π_∞ no infinito em $\mathbb{P}^3$, que possui 3 parâmetros. A correspondência afim entre duas imagens é codificada como a correspondência das projeções de pontos de Π_∞. A correspondência de pontos de Π_∞, como a correspondência de pontos em qualquer plano é descrita por uma homografia planar chamada *homografia no infinito* $\mathbf{H}_\infty$, cuja expressão em função das matrizes de projeção foi dada na equação (2.11). Uma vez que $\mathbf{F}$ é conhecida, os três parâmetros adicionais necessários para descrever $\mathbf{H}_\infty$ estão no vetor $\mathbf{r}_\infty$:

$$\mathbf{H}_\infty \simeq \mathbf{S} + \mathbf{e}'\mathbf{r}_\infty^T.$$

Nesta equação o vetor $\mathbf{r}_\infty$ representa a projeção da primeira imagem da intersecção do plano intrínseco Π_S com o plano no infinito, a *linha de fuga* de Π_S, contendo os pontos de fuga de todos os conjuntos de linhas paralelas de Π_S.

A partir de $\mathbf{F}$, precisamos de três correspondências de pontos no infinito para determinar $\mathbf{H}_\infty$. Uma maneira de obtê-las é considerarmos três pontos de fuga correspondentes. Como linhas paralelas em $\mathbb{P}^3$ se intersectam no plano no infinito, um ponto de escape é a projeção de um ponto de Π_∞. Outras maneiras de conseguirmos restrições no plano no infinito incluem a correspondência de pontos no horizonte, que estão a grande distância das câmeras, e utilizarmos o conhecimento da proporção de distâncias entre três pontos alinhados.

Quando a correspondência afim é conhecida, podemos restringir mais os pares $(\mathcal{P}, \mathcal{P}')$ de matrizes projetivas possíveis. Além de sua matriz fundamental $\mathbf{F}$, necessitamos de $\mathbf{H}_\infty$ sua homografia no infinito. Na verdade, se possuirmos $\mathbf{H}_\infty$, podemos utilizar a equação (2.16) e uma epipole para definir $\mathbf{F}$.

Podemos verificar que os pares de matrizes projetivas que possuem homografia no infinito $\mathbf{H}_\infty$ e epipole $\mathbf{e}'$ são da forma:

$$\begin{cases} \mathcal{P} & \simeq & [\mathbf{I}_3 \mathbf{0}_3]\mathcal{A} \\ \mathcal{P}' & \simeq & [\mathbf{H}_\infty \mu \mathbf{e}']\mathcal{A} \end{cases} \text{com } \mathcal{A} = \begin{bmatrix} \mathcal{P} \\ 0_3^T \; 1 \end{bmatrix} \tag{2.21}$$

Esta decomposição particiona os parâmetros de um par de câmeras em:

- 12 parâmetros que correspondem à ambigüidade na reconstrução afim: a escolha arbitrária da base afim (11 por fixarmos $\mathcal{P}$, o outro é dado por μ).
- 10 parâmetros que descrevem a correspondência afim: 8 da homografia no infinito e 2 da epipole.

Em suma, quando identificamos o plano no infinito, um par de imagens e sua epipole, e

a homografia no infinito determinam a reconstrução a menos de uma transformação afim em $\mathbb{P}^3$. Podemos realizar a reconstrução utilizando um par particular de matrizes de projeção, a *representação canônica afim*, cujos elementos não dependem da escolha da base afim em $\mathbb{P}^3$. Essa representação pode ser obtida da equação (2.20) por uma multiplicação pela matriz $\mathbf{Q}_A^{-1}$:

$$\begin{cases} \mathcal{P} &= [\mathbf{I}_3\mathbf{0}_3] &= [\mathbf{I}_3\mathbf{0}_3]\mathbf{Q}_A^{-1} \\ \mathcal{P}' &= [\mathbf{H}_\infty\mu\mathbf{e}'] &= [\mathbf{S}\mu\mathbf{e}']\mathbf{Q}_A^{-1} \end{cases} \text{com } \mathbf{Q}_A^{-1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{r}_\infty^T & \mu \end{bmatrix} \tag{2.22}$$

Como transformações afins incluem cisalhamento e escalas diferentes para cada eixo, as distâncias relativas entre pontos não são preservadas. No entanto, as distâncias relativas de pontos colineares são preservadas e portanto já podemos realizar algumas medidas quantitativas, como encontrar o ponto médio de um segmento.

Reconstrução Euclidiana

Agora vamos adiante para estudar o caso onde além da correspondência afim, possuímos informação euclidiana. Esta informação torna possível reduzir a ambigüidade para uma *transformação de similaridade* (translação, rotação e escala) e obter reconstruções nas quais podemos medir ângulos e distâncias relativas.

Assim como transformações afins são casos particulares de transformações projetivas, transformações de similaridade são casos particulares de transformações afins, onde a primeira sub-matriz satisfaz $\mathbf{A}\mathbf{A}^T = s\mathbf{I}_3$.

Portanto existe uma hierarquia de transformações: similaridade é um sub-conjunto de afim, que é um sub-conjunto de projetiva. Cada vez que restringimos a transformação, mais conseguimos restringir a reconstrução, porém para isso necessitamos de mais informação.

A informação euclidiana em uma imagem é codificada como a matriz de parâmetros intrínsecos da câmera, que chamaremos de $\mathbf{A}$. Em geral, esta matriz representa cinco parâmetros

que são conhecidos quando a câmera é calibrada. Eles podem ser determinados a partir de uma combinação de conhecimento sobre a câmera e conhecimento sobre a cena.

Quando $\mathbf{A}$ é conhecida, podemos restringir ainda mais os pares $(\mathcal{P}, \mathcal{P}')$ de reconstruções possíveis. Primeiro devemos notar que, se decompusermos cada matriz projetiva em seus parâmetros intrínsecos e extrínsecos, temos a decomposição clássica para um par de matrizes projetivas:

$$\begin{cases} \mathcal{P} & \simeq & \mathbf{A}[\mathbf{R}_1\ \mathbf{t}_1] & = & [\mathbf{A}\mathbf{0}_3]\mathcal{S} \\ \mathcal{P}' & \simeq & \mathbf{A}'[\mathbf{R}_2\ \mathbf{t}_2] & = & \mathbf{A}'[\mathbf{R}\ \mu\mathbf{t}]\mathcal{S} \end{cases} \text{com } \mathcal{S} = \begin{bmatrix} \mathbf{R}_1 & \mathbf{t}_1 \\ \mathbf{0}_3^T & 1/\mu \end{bmatrix},$$

onde $\mathbf{R} = \mathbf{R}_2\mathbf{R}_1^T$ e $\mathbf{t} = \mathbf{t}_2 - \mathbf{R}_2\mathbf{R}_1^T\mathbf{t}_1$ representam o deslocamento relativo entre os sistemas de coordenadas das duas câmeras. Agora, dos 22 parâmetros totais da reconstrução:

- 7 correspondem à transformação de similaridade representando a escolha arbitrária da base euclidiana (6 fixando o sistema de coordenadas da primeira câmera através de $\mathbf{R}_1$ e $\mathbf{t}_1$, e 1 sendo μ que representa a escala).
- 15 descrevem os parâmetros intrínsecos (5 para cada câmera) e a transformação euclidiana relativa $\mathbf{R}$ e $\mathbf{t}$ (posição e orientação) das câmeras.

A direção da translação é determinada, porém sua norma não, devido à ambigüidade profundidade-velocidade: não é possível distinguir entre um ponto próximo se movendo devagar ou um ponto distante se movendo proporcionalmente mais rápido.

A partir da matriz de projeção, obtemos:

$$\mathbf{e}' = \mathbf{A}\mathbf{t},\ \mathbf{H}_\infty = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}.$$

Deste resultado concluímos que podemos caracterizar a correspondência euclidiana por qualquer um dos seguintes conjuntos de 15 parâmetros:

- A correspondência afim mais os parâmetros intrínsecos de uma câmera: $\mathbf{H}_\infty$, $\mathbf{e}'$, $\mathbf{A}$.
- Os parâmetros intrínsecos de ambas as câmeras e o deslocamento entre elas: $\mathbf{A}$, $\mathbf{A}'$, $\mathbf{R}$, $\mathbf{t}$.

Podemos agora obter uma *representação euclidiana canônica* como uma especialização do caso afim e projetivo. Neste caso usaremos como sistema de coordenadas, o sistema de coordenadas da primeira câmera:

$$\begin{cases} \mathcal{P} \simeq [\mathbf{A}0_3] = [\mathbf{I}_3 0_3]\mathbf{Q}_A^{-1}\mathbf{Q}_E^{-1} \\ \mathcal{P}' \simeq \mathbf{A}'[\mathbf{R}\ \mu\mathbf{t}] = [\mathbf{S}\mu\mathbf{e}']\mathbf{Q}_A^{-1}\mathbf{Q}_E^{-1} \end{cases} \tag{2.23}$$

com:

$$\mathbf{Q}_A^{-1} = \begin{bmatrix} \mathbf{I}_3 & 0_3 \\ \mathbf{r}_\infty^T & \mu \end{bmatrix} \text{ e } \mathbf{Q}_E^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{A}_3 \\ 0^T & 1 \end{bmatrix}. \tag{2.24}$$

A partir de uma reconstrução projetiva, que requer apenas correspondências entre pontos, podemos avançar para uma reconstrução afim quando $\mathbf{r}_\infty$ é conhecido (3 graus de liberdade), aplicando $\mathbf{Q}_A$ para os pontos M_i, e então para uma reconstrução euclidiana quando $\mathbf{A}$ é conhecida (5 graus de liberdade), aplicando $\mathbf{Q}_E$. Cada avanço reduz a ambigüidade da reconstrução, primeiro de uma homografia (15 graus de liberdade), para afim (12 graus de liberdade) e finalmente para similaridade (7 graus de liberdade).

Matriz Essencial

No caso de ambas as câmeras possuírem a mesma matriz de calibração $\mathbf{A}$, podemos trabalhar com a *matriz essencial* ao invés da matriz fundamental.

Sejam duas câmeras calibradas (isto é, os parâmetros intrínsecos conhecidos) $\mathcal{P} = [I_3 0_3]$ e

$\mathcal{P}' = [\mathbf{R}\ \mathbf{t}]$, neste caso a *matriz essencial* entre elas é definida como:

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}. \tag{2.25}$$

A matriz essencial também possui uma relação direta com a matriz fundamental, por:

$$\mathbf{E} \simeq \mathbf{A}^T \mathbf{F} \mathbf{A}. \tag{2.26}$$

Se utilizarmos os métodos de estimação da matriz fundamental apresentados acima com pontos calibrados (isto é, transformados por $\mathbf{A}$) teremos como resultado a matriz essencial, e não a matriz fundamental [FL01, HA04].

2.2.4 Triangulação dos Pontos no Espaço

Com a posição relativa das câmeras no espaço e as correspondências de pontos nas imagens, podemos fazer una triangulação para encontrarmos a posição do ponto no espaço.

No entanto, a partir dessa idéia simples, existem complicações: podem ocorrer imprecisões e acúmulos de erros durante a estimação da matriz essencial e das câmeras, o que faz com que os raios óticos que passam pelas projeções em cada imagem não se cruzem no espaço. Uma solução simples para isso é encontrar o ponto médio da distância dentre os raios.

Assim, para estimarmos a posição 3D de um ponto, sejam $\mathbf{P} = [\mathbf{R}\ \mathbf{t}]$ e $\mathbf{P}' = [\mathbf{R}'\ \mathbf{t}']$ as matrizes calibradas associadas a duas câmeras. Sejam $\mathbf{m} = (u, v)$ e $\mathbf{m}' = (u', v')$ um par de correspondências entre as imagens. Decompondo $\mathbf{P}$ e $\mathbf{P}'$ em:

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} \mathbf{P}' = \begin{bmatrix} \mathbf{p}'_1 \\ \mathbf{p}'_2 \\ \mathbf{p}'_3 \end{bmatrix}.$$

Chegamos ao sistema [HA04]:

$$\begin{bmatrix} u\mathbf{p}_3 - \mathbf{p}_1 \\ v\mathbf{p}_3 - \mathbf{p}_2 \\ u'\mathbf{p}'_3 - \mathbf{p}'_1 \\ v'\mathbf{p}'_3 - \mathbf{p}'_2 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} = 0. \tag{2.27}$$

Com o sistema montado, a decomposição por valores singulares (SVD) é utilizada para encontrarmos (X, Y, Z, w). Este sistema de coordenadas é suficiente para o caso de visão estéreo, porém no caso de múltiplas imagens os erros na estimação das câmeras podem levar a que um mesmo ponto no espaço, encontrado a partir de um par de câmeras, seja estimado em uma posição 3D diferente em outro par de câmeras da mesma seqüência.

Para solucionar este problema, montamos um sistema linear que calcula as triangulações levando em conta todos os quadros onde uma correspondência do ponto foi encontrada, que é uma extensão simples da equação (2.27):

$$\begin{bmatrix} u_0\mathbf{p}[0]_3 - \mathbf{p}[0]_1 \\ v_0\mathbf{p}[0]_3 - \mathbf{p}[0]_2 \\ u_1\mathbf{p}[1]_3 - \mathbf{p}[1]_1 \\ v_1\mathbf{p}[1]_3 - \mathbf{p}[1]_2 \\ \vdots \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} = 0. \tag{2.28}$$

Onde $\mathbf{m}_i = (u_i, v_i)$ é a correspondência da característica $\mathbf{m}_i$ na imagem i e $\mathbf{p}[i]_j$ é a linha j da matriz $\mathbf{P}[i]$, que é a matriz da câmera associada a imagem i.

Com isso temos uma nuvem de pontos 3D extraída a partir de correlações em múltiplas imagens.

2.3 Ajuste de Grupo

Após o processo de reconstrução descrito acima temos:

1. Um conjunto de m câmeras, definidas por suas matrizes $\mathtt{P}^i$;
2. Um conjunto de n pontos 3D $\mathbf{X}_j$;
3. Um conjunto de pontos $\mathbf{x}_j^i$, onde cada ponto $\mathbf{x}_j^i$ é a coordenada do ponto $\mathbf{X}_j$ quando observado pela câmera $\mathtt{P}^i$.

Idealmente todos os valores satisfazem a relação $\mathtt{P}^i\mathbf{X}_j = \mathbf{x}_j^i$, para todo $i = 1,..,m$ e $j = 1,..,n$. No entanto, devido ao acúmulo de erros no rastreamento de características, estimação dos parâmetros das câmeras e reconstrução dos pontos, a equação pode não ser satisfeita de maneira exata.

Assumindo que o ruído segue uma distribuição Gaussiana, queremos a solução mais provável: estimar as matrizes de projeção $\hat{\mathtt{P}}^i$ e os pontos 3D $\hat{\mathbf{X}}_j$ que definem os pontos reprojetados $\hat{\mathbf{x}}_j^i = \hat{\mathtt{P}}^i\hat{\mathbf{X}}_j$ minimizando a distância entre estes pontos e os pontos $\mathbf{x}_j^i$ amostrados nas imagens:

$$\min_{\hat{\mathtt{P}}^i,\hat{\mathbf{X}}_j} \sum_{ij} d(\hat{\mathtt{P}}^i\hat{\mathbf{X}}_j, \mathbf{x}_j^i)^2,$$

onde $d(\mathbf{x},\mathbf{y})$ é a distância geométrica entre dois pontos homogêneos $\mathbf{x}$ e $\mathbf{y}$ na imagem. Esta estimação minimizando o erro de reprojeção é conhecida como *ajuste de grupo* [HA04, TMHF00] pois envolve o ajuste de um grupo de feixes de raios entre o centro de cada câmera e os pontos 3D.

O ajuste de grupo é utilizado como um passo final do algoritmo de reconstrução. Ele é tolerante a dados ausentes e nos dá verdadeiramente a estimativa mais provável para a solução do sistema. No entanto ele apresenta limitações: (i) ele exige uma boa estimativa inicial e (ii)

ele pode se transformar em um problema extremamente grande de minimização, devido ao grande número de parâmetros envolvidos.

Minimização Iterativa

Como cada câmera possui 11 graus de liberdade e cada ponto no espaço 3 graus de liberdade, uma reconstrução envolvendo n pontos e m pontos de vista requer uma minimização sobre $3n + 11m$ parâmetros. Conforme m e n aumentam, a minimização se torna extremamente demorada. Existem algumas soluções para esse problema:

Reduzir m e/ou n Não incluir todos os pontos ou câmeras na minimização ou particionar a entrada em conjuntos, ajustá-los separadamente e depois uni-los.

Intercalação Alternadamente melhoramos o erro de reprojeção modificando as câmeras com os pontos 3D fixos e melhoramos o erro de reprojeção modificando os pontos com as câmeras fixas. Neste caso a maior matriz que precisamos inverter é a matriz 11×11 utilizada para estimar uma câmera. No entanto, apesar de cada iteração ser mais rápida, o algoritmo pode demorar mais para convergir [TMHF00].

Métodos Esparsos Um passo central do algoritmo de minimização, a solução das equações, é computacionalmente custoso e repetido diversas vezes durante o processamento. No entanto, no caso do ajuste de grupo, a minimização opera sobre uma matriz esparsa com uma forma definida. Aproveitando isso é possível conseguir grandes reduções no tempo de execução do algoritmo de minimização [TMHF00].

No sistema desenvolvido foi utilizada uma implementação do método de gradientes conjugados de Powell baseada em [PFTV92]. Esse método foi escolhido por convergir em poucos passos e possuir uma implementação confiável facilmente disponível. Na nossa implementação

nenhuma das técnicas descritas acima para reduzir o tempo de processamento foi utilizada, porém melhorias neste sentido estão planejadas para o futuro.

Solução Inicial

Um ponto importante de qualquer algoritmo de minimização é a aproximação inicial fornecida a ele. No caso do nosso sistema, essa estimativa é o resultado da reconstrução realizada conforme os passos descritos na sessões anteriores. Relembrando:

1. Geramos um conjunto de correlações entre as imagens da seqüência de entrada. Esse conjunto pode ser obtido de maneira automática, através de um rastreador de características baseado na técnica de Lucas-Kanade-Tomasi, manualmente com a ajuda de um operador humano, ou automaticamente no caso de testes com valores sintéticos;

2. A partir das correspondências de pontos e dos parâmetros de calibração da câmera (obtidos em um sistema separado) estimamos a *matriz essencial* entre cada par de câmeras. Isso nos dá a posição de cada câmera C_i em relação à câmera C_{i-1}. Como o sistema é sobre-definido, utilizamos uma implementação do algoritmo RANSAC (já apresentado em 2.2.2). Neste passo as características que não fazem parte do grupo de consenso são descartadas;

3. As posições de câmeras encontradas no passo anterior são movidas para um *frame* de coordenadas comum (que considera a primeira câmera na origem);

4. Dado o conjunto de câmeras e correspondências, utilizamos um sistema de mínimos quadrados para encontrarmos a solução mais provável para os pontos de cruzamento dos raios saindo de cada câmera. Isso nos dá um conjunto de pontos 3D.

Os valores encontrados em cada um desses três passos são utilizados como a estimativa inicial para o ajuste de grupo. O uso do método de Powell para minimização implica em uma

suposição que o mínimo global está próximo da estimativa inicial, o que nem sempre é verdade neste caso, e é uma limitação da versão atual do sistema.

2.4 Rastreamento

Já temos agora uma base para encontrarmos, dado um par de imagens e correspondências, a posição relativa das câmeras no espaço, e a profundidade dos pontos na cena.

A fim de estimar a matriz fundamental, devemos encontrar um conjunto de correspondências de características entre duas imagens.

O rastreamento de características é um problema semelhante a encontrar o movimento de cada ponto da imagem (*optical flow*, fluxo ótico).

2.4.1 Fluxo Ótico

O fluxo ótico (*optical flow*) é uma aproximação do campo de movimentação (*motion field*) de uma seqüência de imagens. O campo de movimentação é um conceito teórico que nos dá, para cada ponto da imagem, a movimentação do elemento "real" associado a este ponto [BFBB92, Hor86].

Por exemplo, uma seqüência de vídeo com uma esfera girando sobre seu próprio eixo implica em um campo de movimentação associado a todos os pontos da superfície da esfera, exceto os dois pontos nas extremidades do eixo de rotação. A magnitude dos vetores associados nos indica a velocidade de movimentação de cada posição da esfera.

Para a construção do fluxo ótico, no entanto, só dispomos da seqüência de vídeo gerada, assim sendo só podemos aproximar o campo de movimentação a partir da variação de intensidade dos pontos pertencentes à imagem.

Por exemplo, na seqüência de vídeo com a esfera girando como mencionada acima, se

a superfície da esfera for perfeitamente lamberciana, sem nenhuma irregularidade, ou se a superfície for perfeitamente reflexiva, apesar do campo de movimentação existir e ser bem definido, não será possível estimar o fluxo ótico a partir das imagens.

Diversas técnicas para calcular o fluxo ótico de uma imagem foram desenvolvidas. Vamos apresentar algumas delas [Hor86, BFBB92, LK81].

Técnicas Diferenciais

As técnicas diferenciais calculam o fluxo ótico a partir de derivadas espaço-temporais da intensidade de versões filtradas das imagens originais (usando filtros *passa-baixa* ou *passa-faixa*).

Estes filtros são necessários, pois para que existam as derivadas das imagens, precisamos de uma função contínua, o que não é verdade dada uma imagem qualquer. Portanto devemos suavizá-las para remover descontinuidades.

As primeiras versões utilizavam derivadas de primeira ordem com suporte apenas à translação:

$$I(x,t) = I(x - vt, 0). \tag{2.29}$$

Onde $v = (u, v)^T$. A partir da expansão de *Taylor* de (2.29) ou, de maneira mais geral, assumindo a conservação de intensidade ($dI(x,t)/dt = 0$), a *equação de restrição do gradiente* pode ser derivada:

$$\nabla I(x,t) \cdot v + I_t(x,t) = 0. \tag{2.30}$$

Onde $I_t(x,t)$ denota a derivada parcial de $I(x,t)$ no tempo, $\nabla I(x,t) = (I_x(x,t), I_y(x,t))^T$ e $[\cdot]$ denota produto escalar.

A equação (2.30) nos dá a componente normal do movimento de contornos espaciais com

intensidade constante, $v = s \cdot n$. As componentes de magnitude s e direção n são dadas por:

$$s(x,t) = \frac{-I_t(x,t)}{\| \nabla I(x,t) \|}, \quad n(x,t) = \frac{\nabla I(x,t)}{\| \nabla I(x,t) \|}. \tag{2.31}$$

Por (2.30) temos duas variáveis, para apenas uma equação, assim sendo mais restrições são necessárias.

Diferenciais de Segunda Ordem

As técnicas neste grupo utilizam derivadas de segunda ordem (a *Hessiana* de I) para restringir a movimentação 2D:

$$\begin{bmatrix} I_{xx}(x,t) & I_{yx}(x,t) \\ I_{xy}(x,t) & I_{yy}(x,t) \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} + \begin{pmatrix} I_{tx}(x,t) \\ I_{ty}(x,t) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.32}$$

A equação (2.32) pode ser derivada diretamente de (2.29) ou da conservação de $\nabla I(x,t)$, $d\nabla I(x,t)/dt = 0$. A conservação de $\nabla I(x,t)$ implica que deformações de primeira ordem de intensidade (rotação ou dilatação, por exemplo) não podem estar presentes, uma restrição mais forte nos fluxos óticos possíveis.

No entanto, se o *problema da abertura* (ver abaixo) estiver presente em uma região, devido a sensibilidade numérica da diferenciação de segunda ordem, derivadas de segunda ordem normalmente não podem ser medidas com precisão suficiente para determinar a tangente de v.

Problemas e Limitações

Ambas as técnicas de encontrar o fluxo ótico possuem restrições nas velocidades que podem ser estimadas e nos tipos de entrada onde funcionam bem.

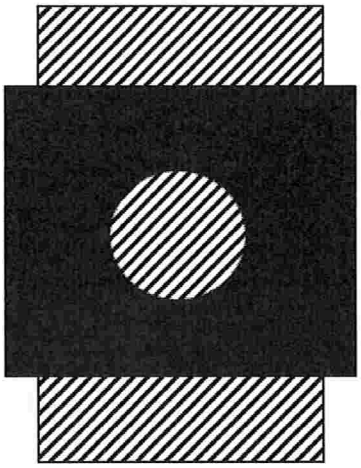

Figura 2.6: O problema da abertura.

O **problema da abertura** é a observação que a movimentação de um contorno homogêneo, vista localmente, é ambígua [Hor86]. Isto ocorre porque o sistema observa a movimentação por uma pequena "abertura". Neste ponto de vista, por exemplo, a movimentação de retas diagonais de baixo para cima possuem a mesma estrutura espaço-temporal que uma movimentação das mesmas da direita para a esquerda (figura 2.6).

Outro problema intrínseco às técnicas que utilizam derivadas é que elas se aplicam a **funções contínuas**. No entanto imagens são, em geral, funções discretas. Para mitigar este problema as imagens passam por um filtro de suavização inicial, diminuindo suas descontinuidades. No entanto, mesmo depois desta filtragem, certos pontos de descontinuidades, como cantos e bordas, permanecem.

Isto cria um paradoxo: os pontos que intuitivamente parecem os melhores para estimar o campo de movimentação (cantos, bordas) são justamente as regiões onde a técnica não

funciona.

Outro problema que advém do uso de derivadas espaço-temporais é que as velocidades medidas são pequenas, na ordem de 1 *pixel* por quadro. Para reduzir este problema, algumas implementações utilizam uma técnica conhecida como pirâmides: a estimação de movimento é primeiro realizada em versões reduzidas das imagens, e esses valores estimados são utilizados como base para estimar a movimentação nas imagens originais.

2.4.2 Rastreamento de Características Selecionadas

Grande parte dos problemas inerentes às técnicas de estimação do fluxo ótico provém delas visarem a estimação de um campo denso (com estimativas para cada *pixel* na seqüência de imagens).

Uma outra maneira de encarar o problema, e que tem encontrado maior uso prático, é de se ater apenas a pequenas regiões da imagem que são consideradas "mais fáceis" de rastrear dentre os quadros de uma seqüência.

A definição do que seriam regiões "mais fáceis" de rastrear é dependente do conteúdo das imagens e do tipo de problema sendo analizado. Aplicações distintas geralmente utilizam definições diferentes.

Existem algumas vantagens nesta aproximação ao problema:

Eficiência As técnicas densas envolvem o uso de diferenciais e integrais em todos os *pixels* das imagens envolvidas, além de derivadas no tempo, tal processamento pode ser extremamente lento.

Qualidade Não existe nenhuma métrica para medir a qualidade do fluxo ótico gerado [BFBB92], portanto é difícil estimar a qualidade do resultado gerado. Atendo-nos a regiões que foram previamente consideradas, por alguma métrica, mais confiáveis para estimação de

movimento aumenta nossa segurança no resultado.

Robustez Tendo um conjunto de características e sua movimentação dentre os quadros de uma seqüência, é mais fácil eliminar alguns destes pontos do que tentar eliminar regiões inteiras do fluxo ótico.

Neste trabalho utilizamos uma implementação das técnicas desenvolvidas por Shi e Tomasi [ST94], baseadas no trabalho de Lucas e Kanade [LK81], que se enquadra nesta classificação. Vamos apresentar agora o seu funcionamento.

Introdução

Assim como na estimação do campo de movimentação, assumimos constância de intensidade (2.29, 2.30).

Vamos chamar a quantidade de movimentação $\delta = (\varepsilon, \eta)$ de *deslocamento* de um ponto $\boldsymbol{x} = (x, y)$. O deslocamento depende da posição na imagem $\boldsymbol{x}$, e variações em δ são perceptíveis mesmo em regiões pequenas. Assim sendo, não faz muito sentido falarmos do deslocamento de uma característica como um valor constante para toda a região, mas um campo de movimentação afim é uma representação melhor:

$$\delta = D\boldsymbol{x} + \boldsymbol{d},$$

onde:

$$D = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix}$$

é a matriz de deformação e $\boldsymbol{d}$ é a translação do centro da região da característica sendo rastreada.

Assim, um ponto $\boldsymbol{x}$ na primeira imagem I se movimenta para um ponto $A\boldsymbol{x}+d$ na segunda imagem J, onde $A = \boldsymbol{I} + D$ e $\boldsymbol{I}_{2\times 2}$ é a matriz identidade.

$$J(A\boldsymbol{x} + d) = I(\boldsymbol{x}). \tag{2.33}$$

Dadas duas imagens I e J, e uma região na imagem I, rastrear significa determinar os seis parâmetros que aparecem na matriz de deformação D e no vetor de deslocamento d.

Regiões pequenas são, em geral, preferíveis para o rastreamento por mais dificilmente se perderem por descontinuidades na profundidade. Por essa razão um modelo unicamente de translação é preferível durante o rastreamento, onde assumimos que a matriz de deformação D é zero:

$$\delta = \boldsymbol{d}.$$

Experimentos realizados pelos autores concluíram que a melhor combinação dentre estes dois modelos de movimentação é utilizar um modelo puramente de translação para rastreamento, e o modelo de movimentação afim para comparar características entre o primeiro quadro e o quadro atual para monitorar a qualidade do rastreamento.

Estimando a Movimentação

Devido a ruídos na imagem e como o modelo de movimentação afim não é perfeito, a equação (2.33) em geral não é satisfeita exatamente. Podemos resolver esse problema encontrando A e $\boldsymbol{d}$ que minimizem uma medida de *dissimilaridade*:

$$\epsilon = \int\int_W [J(A\boldsymbol{x} + \boldsymbol{d}) - I(\boldsymbol{x})]^2 \, w(\boldsymbol{x}) \, d\boldsymbol{x} \tag{2.34}$$

onde W é uma região na imagem origem e $w(\boldsymbol{x})$ é uma função com pesos. No caso mais simples $w(\boldsymbol{x}) = 1$. De maneira alternativa, w poderia ser uma gaussiana para dar mais ênfase à área central da região. No caso de termos apenas translação, a matriz A é restrita à matriz identidade.

Para minimizar (2.34), podemos diferenciar sobre as entradas desconhecidas de D e $\boldsymbol{d}$ e igualar a zero. Então, linearizamos o sistema resultante utilizando uma expansão de Taylor:

$$J(A\boldsymbol{x} + \boldsymbol{d}) = J(\boldsymbol{x}) + \boldsymbol{g}^T(\boldsymbol{u}). \tag{2.35}$$

Isto nos leva ao sistema linear 6×6:

$$T\boldsymbol{z} = \boldsymbol{a} \tag{2.36}$$

onde $z^T = [d_{xx}, d_{yx}.d_{xy}, d_{yy}, d_x, d_y]$ incluí as entradas desconhecidas de D e $\boldsymbol{d}$. O vetor de erro:

$$\boldsymbol{a} = \int\int_W [I(\boldsymbol{x}) - J(\boldsymbol{x})] \begin{bmatrix} xg_x \\ xg_y \\ yg_x \\ yg_y \\ g_x \\ g_y \end{bmatrix} w\, d\boldsymbol{x},$$

que depende da diferença entre as duas imagens e da matriz $T_{6\times6}$, que pode ser encontrada a partir de uma imagem e pode ser escrita como:

$$T = \int\int_W \begin{bmatrix} U & V \\ V^T & Z \end{bmatrix} w\, d\boldsymbol{x} \tag{2.37}$$

onde:

$$U = \begin{bmatrix} x^2 g_x^2 & x^2 g_x g_y & xy g_x^2 & xy g_x g_y \\ x^2 g_x g_y & x^2 g_y^2 & xy g_x g_y & xy g_y^2 \\ xy g_x^2 & xy g_x g_y & y^2 g_x^2 & y^2 g_x g_y \\ xy g_x g_y & xy g_y^2 & y^2 g_x g_y & y^2 g_y^2 \end{bmatrix}$$

$$V^T = \begin{bmatrix} x g_x^2 & x g_x g_y & y g_x^2 & y g_x g_y \\ x g_x g_y & x g_y^2 & y g_x g_y & y g_y^2 \end{bmatrix}$$

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}.$$

Mesmo quando a transformação afim é um modelo adequado, a equação (2.36) é apenas parcialmente satisfeita, devido a linearização da equação (2.35).

Durante o rastreamento, a magnitude da transformação afim D da característica será provavelmente pequena, dado que o movimento entre quadros deve ser pequeno para que o rastreamento funcione. É portanto seguro forçar a matriz D a zero.

Assim, quando o objetivo é determinar $\boldsymbol{d}$, o sistema:

$$Z\boldsymbol{d} = \boldsymbol{e} \tag{2.38}$$

deve ser resolvido, onde $\boldsymbol{e}$ contêm as duas últimas entradas do vetor $\boldsymbol{a}$ da equação (2.36).

No entanto, quando estamos monitorando as características por diferenças em suas aparências entre o primeiro quadro e o quadro atual da seqüência, a movimentação é grande demais para ser descrita adequadamente apenas por translação.

Seleção de Características

Tendo um sistema definido para o rastreamento, devemos nos preocupar na seleção de quais regiões da imagem devem ser selecionadas como características e rastreadas.

Muitas técnicas procuram um conjunto que preencha algum requisito pré-definido, como cantos e bordas. No entanto, Shi e Tomasi [ST94] propõem uma técnica diferente, definindo como boas características aquelas que podem ser mais facilmente rastreadas (isto é, aquelas para as quais as técnicas previamente apresentadas dão os melhores resultados).

Características são bem rastreadas entre quadros se o sistema (2.38) apresenta boas medidas, e se ele pode ser resolvido de maneira confiável. Conseqüentemente, a matriz simétrica $Z_{2\times 2}$ deve estar acima do nível de ruído da imagem e ser bem condicionada.

A exigência de estar acima do nível de ruído implica que os dois autovalores de Z sejam grandes e que não sejam distantes por muitas ordens de magnitude. Dois autovalores pequenos indicam uma região com intensidade aproximadamente constante, enquanto um valor muito grande e outro muito pequeno indicam regiões com padrões uni-direcionais (que podem nos levar ao *problema da abertura*), enquanto dois valores grandes representam regiões com cantos e padrões que são mais facilmente rastreados.

Na prática é observado que quando o valor do menor autovalor é suficientemente grande para passar do nível de ruído, a matriz Z é normalmente bem condicionada. De fato as variações de intensidade dentro de uma região da imagem possuem um limite superior, definido pelo valor máximo que um *pixel* pode assumir, portanto o maior autovalor não pode ser arbitrariamente grande.

Concluindo, sendo λ_1 e λ_2 os dois autovalores de Z, uma região é selecionada se:

$$\min(\lambda_1, \lambda_2) > \lambda \tag{2.39}$$

onde λ é um valor de corte pré-definido.

2.5 Representação de Modelos Tri-Dimensionais

De maneira geral, um *modelo tri-dimensional* é uma maneira de descrever o "formato" de um objeto no mundo. Neste caso não incluímos informação como cor ou textura, apenas a informação *estrutural* do objeto é levada em conta.

Tendo este conceito definido, precisamos nos preocupar em como representá-lo de maneira digital, própria para armazenamento em um computador.

Diversas maneiras para a representação de objetos já foram desenvolvidas [Req80], cada uma apresentando suas vantagens e desvantagens em domínios específicos. As principais técnicas se dividem em métodos de decomposição, onde objetos são decompostos de alguma forma em primitivas e/ou polígonos, decomposição espacial, métodos construtivos, métodos baseados em imagem e métodos baseados em interpolação e aproximação.

2.5.1 Decomposição em Primitivas

A decomposição em primitivas utiliza algumas formas básicas como quadrados, esferas e pirâmides, que são pré-definidas, e os objetos são modelados a partir da composição destas primitivas. Esta técnica possui a vantagem que, para os objetos pouco complexos, a modelagem é extremamente simples e eficiente.

Porém, para objetos mais complexos, a composição de primitivas se torna cada vez mais difícil de se aplicar. Normalmente, esta técnica é utilizada como uma maneira auxiliar de representar partes comuns de modelos [Man88].

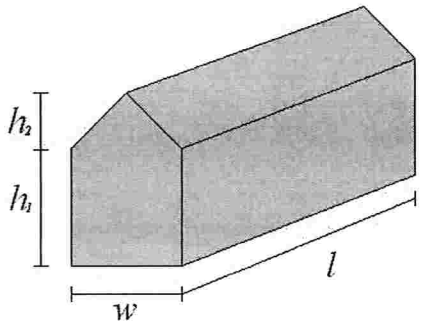

Figura 2.7: Exemplo de uma primitiva com parâmetros associados. Uma representação por decomposição em primitivas armazenaria qual primitiva, além de parâmetros básicos e informações de transformação [Req80].

2.5.2 Decomposição em Polígonos

Podemos definir um objeto no espaço tri-dimensional como uma coleção de pontos pertencentes à superfície deste objeto e um conjunto de sub-planos conectando estes pontos. Por exemplo, uma caixa pode ser representada no espaço tri-dimensional como um conjunto de seis sub-planos, definidos por um conjunto de oito pontos.

Esta representação, no entanto, possui limitações com relação as superfícies curvas, necessitando de uma grande quantidade de polígonos para defini-las. Por causa disto, existem técnicas que associam o uso de polígonos planos com informações sobre curvatura [Pie91], ou procuram simplificar modelos que necessitam de muitos polígonos [Gar99]. Também se torna mais difícil realizarmos certas operações básicas, como determinar o interior e o exterior de objetos.

2.5.3 Decomposição Espacial

Neste caso, o espaço é particionado em pequenas regiões, chamadas células, que definem o material (cor, textura, etc.) em cada região.

Quanto mais detalhado o modelo, maior é o número de células que o compõem, que pode se tornar muito grande. Para minimizar a quantidade de informação que deve ser processada e armazenada, uma estrutura hierárquica, como a *octree* (similar a uma árvore binária mas no espaço tri-dimensional), é comumente empregada [SW88].

O uso da decomposição espacial pode se tornar mais eficiente se outra forma de representação, como por exemplo a decomposição em primitivas, visto anteriormente, for utilizada ao invés do armazenamento da informação sobre o material de cada célula. Esta organização híbrida permite que células sejam rapidamente desconsideradas para efeitos de visualização e teste de colisão, porém permitindo acesso a um maior detalhamento das células pertinentes [Cla76, Boi81].

2.5.4 Métodos Construtivos

Os métodos construtivos modelam um objeto por uma seqüência de operações básicas de combinação de outros objetos. A representação mais comum é chamada *CSG* (*Constructive Solid Geometry*, geometria sólida construtiva) e utiliza operações booleanas (versões modificadas das operações de união, intersecção e diferença da teoria de conjuntos) para combinar os objetos básicos (como as primitivas descritas em 2.5). A seqüência de operações é normalmente armazenada como uma árvore (figura 2.8) [Req80].

Em *CSG*, podemos representar um objeto de mais de uma maneira. Por exemplo, no caso de uma representação bi-dimensional, uma forma em "L" pode ser composta como a união de dois retângulos, ou como a subtração de um retângulo menor sobre um retângulo maior. Esta

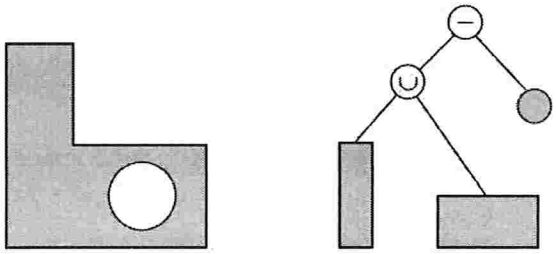

Figura 2.8: Criação de uma forma bi-dimensional por construção.

característica torna difícil a solução do problema de determinar se dois modelos são iguais, ou se um modelo é nulo.

2.5.5 Interpolação e Aproximação

Dado um conjunto de primitivas de menor dimensão (pontos e retas, por exemplo), podemos utilizá-los como base para determinar uma primitiva de maior dimensão (planos e curvas, por exemplo) [Gre].

Por exemplo, em um espaço n-dimensional, se possuímos um conjunto de m pontos, queremos encontrar uma curva $F(n)$ que aproxime ou interpole esses pontos.

Chamamos de interpolação se $n = m$, e o sistema pode ser determinado. Se temos mais pontos do que dimensões ($m > n$) não podemos resolver o sistema, e temos que realizar uma aproximação.

2.5.6 Modelos Baseados em Imagens

Uma técnica bem mais recente permite que o modelo seja construído a partir de imagens capturadas do objeto, em conjunto com outras informações geométricas. Existem diversas formas de representar um objeto a partir de imagens e algumas serão apresentadas com mais

detalhes a seguir.

Todas se baseiam no conceito de gerar um banco de dados de imagens do objeto sendo modelado, com o objetivo de gerar novas visualizações do mesmo a partir da busca e interpolação das amostras existentes [AB91, OB99].

Muitas das técnicas utilizadas em modelagem baseada em imagens utilizam o conceito da *função plenóptica* [AB91], dada por $P(\theta, \phi, \lambda, V_x, V_y, V_z, t)$, que determina a freqüência de onda captada por um raio partindo de um ponto com coordenadas (V_x, V_y, V_z) no espaço tri-dimensional, com um ângulo (θ, ϕ, λ), em um momento t no tempo.

Para determinar o que um observador, em determinada posição do espaço, está observando, podemos encontrar o valor da função plenóptica para todos os raios que fazem parte do seu campo de visão. Desta forma, o funcionamento básico de todas as técnicas que utilizam modelos baseados em imagens, é estimar novos valores da função plenóptica, dado um banco de dados com valores capturados para determinados pontos de vista.

Vamos apresentar em mais detalhes três técnicas que utilizam modelos baseados em imagens:

- Panoramas Cilíndricos: que é a técnica mais simples e também a mais utilizada, disponível em pelo menos um sistema comercial conhecido ([Che95]);
- Campos de Iluminação: uma técnica generalista e bastante flexível, que pode ser utilizada como base para métodos mais específicos;
- Imagens com Profundidade: um modelo que pode ser facilmente gerado como resultado final de um processo de reconstrução como o apresentado neste trabalho.

Panoramas Cilíndricos

Panoramas cilíndricos (*Cylindrical Panoramas*) ou mapas de ambiente cilíndricos são utilizados para conseguir liberdade de orientação horizontal quando exploramos um ambiente a partir de um único ponto. A escolha de panoramas cilíndricos é justificada pela facilidade de obter as imagens e pelo fato que um mapa cilíndrico apenas se curva em uma direção, simplificando em muito a transformação necessária para construir novas visualizações.

O sistema funciona colhendo, a partir de uma câmera fixa em um ponto e tirando fotos dos objetos ao seu redor, informações suficientes para estimar o valor da função plenóptica para raios partindo da mesma posição da câmera e olhando ao seu redor.

Podemos construir panoramas cilíndricos utilizando câmeras panorâmicas especializadas [Adv02, Eve02], colando imagens fotográficas adquiridas com uma câmera comum ou utilizando imagens geradas por computador. O software *QuickTime VR*, desenvolvido pela *Apple Computers*, suporta renderização a partir de panoramas cilíndricos e foi o primeiro produto comercial a utilizar técnicas de renderização baseadas em imagens [Che95].

Depois do *QuickTime VR*, diversos sistemas para renderização de mapas de ambiente cilíndricos foram implementados. Estes sistemas suportam movimentação livre na direção horizontal e aproximação. A movimentação na direção vertical é limitada pelo campo de visão do panorama cilíndrico (normalmente próximo a 50 graus).

Durante uma movimentação, transformamos a parte visível do panorama para produzir imagens com a perspectiva adequada. A figura (2.9) apresenta um panorama cilíndrico com o campo de visão associado e ilustra a geometria básica do algoritmo de transformação [CSM95]. Como o panorama não se curva na direção vertical, a função de transformação se reduz a uma operação de escala aplicada nas colunas intersectadas pelo campo de visão. O fator de escala varia de coluna a coluna. Para um determinado ponto de vista, podemos pré-calcular os fatores, armazenando-os em um vetor de uma dimensão e reutilizando-os para acelerar a

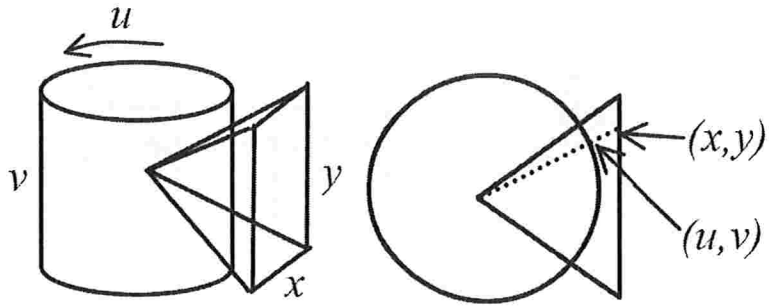

Figura 2.9: Esquerda: um panorama cilíndrico e seu campo de visão. Direita: a geometria do algoritmo de transformação. (Imagem obtida de [Oli02])

operação [CSM95].

Podemos implementar a transformação utilizando uma formulação inversa similar a utilizada no mapeamento de texturas. Assim, dadas as coordenadas (x, y) de um ponto no plano de imagem, obtemos diretamente os parâmetros (u, v) no mapa de ambiente cilíndrico (figura 2.9). Isto garante uma re-amostragem correta para todos os *pixels* em novas visões.

Implementamos a movimentação vertical em dois passos: primeiro, a porção visível do panorama é mapeada a um plano paralelo ao eixo do cilindro utilizando a técnica de transformação descrita acima; depois, a imagem resultante é mapeada ao plano de imagem desejado utilizando uma transformação projetiva planar. Durante a movimentação vertical, as imagens sofrem duas transformações, o que tende a degradar a qualidade da imagem final.

A grande limitação deste sistema é restringir a câmera a uma única posição (o ponto a partir do qual as imagens foram adquiridas).

Campo de Iluminação e *Grafo de Iluminação*

A função "campo de iluminação" (*light field*) descreve, para qualquer ponto determinado, a radiância captada em uma direção particular no espaço [LH96]. Renderização *light field*

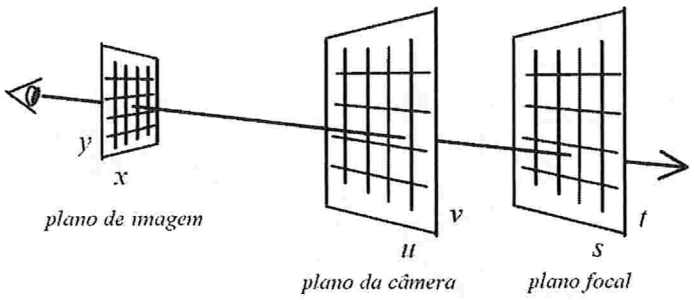

Figura 2.10: *Light field* e *Lumigraph*. Representação geométrica da criação de novas visões (Imagem obtida de [Oli02]).

e *lumigraph* criam novas visualizações de uma cena ou objeto re-amostrando um banco de dados de imagens representando uma amostra da função plenóptica representada por raios parametrizados por suas intersecções com dois planos paralelos, uma estrutura conhecida como *light slab*.

Para criar o banco de dados, um reticulado ortogonal é utilizado para definir as posições da câmera para captura. Este reticulado é associado a um plano *uv*, conhecido como *plano da câmera*. Em cada posição do reticulado uma imagem é capturada utilizando um plano *st* como plano de imagem. Como as posições de ambos os planos são fixas, o campo de visão se torna progressivamente inclinado conforme a câmera se move do centro do plano da câmera em direção as bordas.

Conceitualmente, a criação de novas visões é ilustrada na figura 2.10. Para cada *pixel* da visão nova, se computa a intersecção do raio de visualização com os planos de câmera e focal. As coordenadas (u, v) da intersecção são utilizadas para selecionar as imagens que serão utilizadas para re-amostragem; as coordenadas (s, t) da intersecção são utilizadas para selecionar o *pixel* exato das imagens selecionadas.

A estratégia de re-amostragem pode variar desde o vizinho mais próximo até a interpolação

quadrilinear [LH96].

Capturando imagens de um objeto de diversos pontos de vista, as técnicas de *light field* e *lumigraph* podem criar efeitos dependentes do ponto de vista, como destaques. Porém, o grande número de imagens necessárias para evitar que a imagem fique excessivamente borrada (causado pelo passo de interpolação durante re-amostragem) aumentam significantemente as necessidades de armazenamento quando comparada com outras técnicas. No sistema *lumigraph*, uma geometria aproximada pode ser utilizada durante a renderização para efetuar correção de profundidade [GGSC96], tornando assim a imagem final menos borrada.

Imagens com Profundidade

Uma imagem com profundidade é um par $\{i_d, K\}$, onde i_d é uma imagem digital e K é um modelo da câmera associado a i_d. Cada elemento do espaço de cores de i_d é aumentado para incluir um valor escalar por *pixel* representando a distância, no espaço Euclidiano, entre o ponto amostrado e a entidade de referência. Se K é uma câmera de projeção perspectiva, a imagem é chamada *imagem de projeção perspectiva com profundidade* e a entidade de referência é o centro de projeção de K. Se K é uma câmera de projeção paralela, a imagem é chamada *imagem de projeção paralela com profundidade* e a entidade de referência é o plano de imagem de K (figura 2.11).

2.6 Conclusão

Neste capítulo descrevemos o funcionamento de câmeras projetivas, e quais informações podemos extrair de uma ou duas imagens de uma mesma câmera. Para isto precisamos estimar a movimentação dentre as duas câmeras, que nos é descrita pela geometria epipolar ou, de maneira mais geral, pela geometria projetiva. Descrevemos também um método de buscar correlações

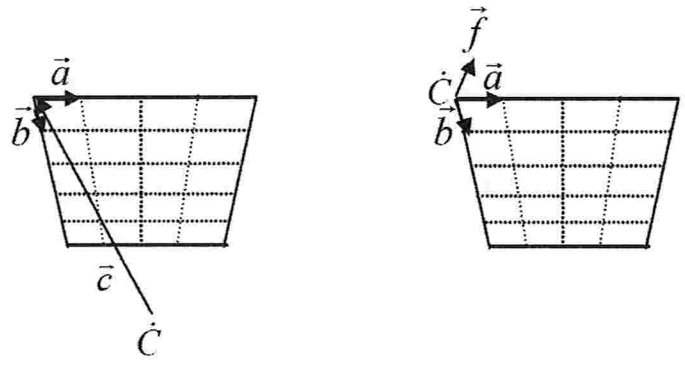

Figura 2.11: Esquerda: imagem de projeção perspectiva com profundidade. Direita: imagem de projeção paralela com profundidade (Imagem obtida de [Oli02]).

entre imagens, necessárias para o processo de estimarmos a estrutura do sistema, e para encontrarmos pontos 3D do objeto. Por fim, apresentamos maneiras de representar modelos 3D em um computador. Este conhecimento é importante para o objetivo final deste trabalho, um sistema para reconstrução interativa de objetos.

Capítulo 3

Descrição do Sistema

Da integração de um algoritmo robusto para detecção e rastreamento de características e das técnicas para estimação do movimento e estrutura de cenas, criamos um sistema para reconstruir a posição 3D de pontos rastreados em um conjunto de imagens capturadas por uma mesma câmera.

No momento, as imagens de entrada devem ser todas de um mesmo tamanho, nos formatos JPG, PNG ou BMP, e nenhuma interação humana é necessária para gerar o conjunto de coordenadas 3D dos pontos reconstruídos. Um passo futuro é, a partir dessa nuvem de pontos gerada, apresentarmos ao usuário uma interface interativa onde ele possa auxiliar o sistema a reconstruir o objeto ao qual tais pontos pertencem, e exportá-lo para algum dos formatos apresentados na sessão (2.5).

O sistema foi desenvolvido na linguagem orientada a objetos C++, com as seguintes bibliotecas auxiliares:

QT um conjunto de classes e ferramentas para a criação de interfaces gráficas portáveis. Existem versões para diversas distribuições Linux, para Windows e para outras interfaces gráficas. Existem duas versões da biblioteca, diferentes apenas na forma de licencia-

mento, a versão aberta, utilizada aqui, é de uso gratuito para projetos abertos [Tro].

OpenCV originalmente desenvolvida pela **Intel**, agrega a implementação de diversos algoritmos para visão computacional e processamento de matrizes, como RANSAC e diversos filtros [Int].

Boost um conjunto de bibliotecas de uso geral que visam estender a funcionalidade das bibliotecas ANSI C++ básicas de uma maneira portável. É utilizada para algumas funcionalidades auxiliares, como *smart pointers* e conversões entre tipos [Boo].

O sistema recebe como entrada um arquivo texto, em um formato específico, que descreve os quadros que compõem a seqüência. Futuramente pretendemos estender este arquivo (e o sistema) para que a entrada possa incluir vídeos.

Uma ferramenta, chamada apenas de **Acquisition**, foi desenvolvida para auxiliar na criação desses arquivos descritores do projeto. Este aplicativo apresenta uma lista com todas as imagens em um diretório e permite que o usuário adicione arquivos à lista de imagens do projeto, podendo visualizar miniaturas (*thumbnails*) dos quadros.

O arquivo de projeto possui um formato simples de ser editado manualmente, similar a um arquivo tipo *script*. Cada linha do arquivo pode conter:

- Um comentário: nesse caso o primeiro caractere não branco deve ser `#`.
- Um nome de sessão: formatado como `[nome da sessão]`, é utilizado para demarcar partes do arquivo.
- Um comando: uma *string* que determina qual comando, seguida de um espaço e uma lista opcional de parâmetros, separados por ";". Comandos são utilizados para adicionar quadros à seqüência.

Abaixo apresenta-se um exemplo de um pequeno arquivo de projeto com apenas três imagens:

```
#reconstruction project v1.0
# O comentário na primeira linha é obrigatório,
# porém podemos inserir outros comentários livremente
# basta que '#' seja o primeiro caractere na linha
[framelist]
image img01.png
image img02.png
image img03.png
# [end] marca o final do arquivo, não é estritamente necessário.
[end]
```

O único comando atualmente aceito é `image` que recebe um único parâmetro, o nome do arquivo de imagem a ser lido. São aceitos arquivos no formato JPEG, PNG e BMP.

A versão atual do sistema busca pela sessão chamada `[framelist]` e lê todos os comandos dentro desta sessão.

Uma vez carregado um projeto, é apresentada uma interface com três regiões principais: um menu, visualização do quadro selecionado, e uma lista navegável dos quadros da seqüência, permitindo que o usuário visualize cada quadro e navegue entre eles (figura 3.1).

Pelo menu é possível selecionar cada passo do processo (rastreamento, estimação das câmeras e estimação dos pontos 3D) em separado, ou todos em seqüência a partir de um único comando.

O sistema de reconstrução foi desenvolvido progressivamente, em uma primeira instância apenas consideramos o caso básico de visão estéreo (isto é, duas câmeras). Em uma segunda

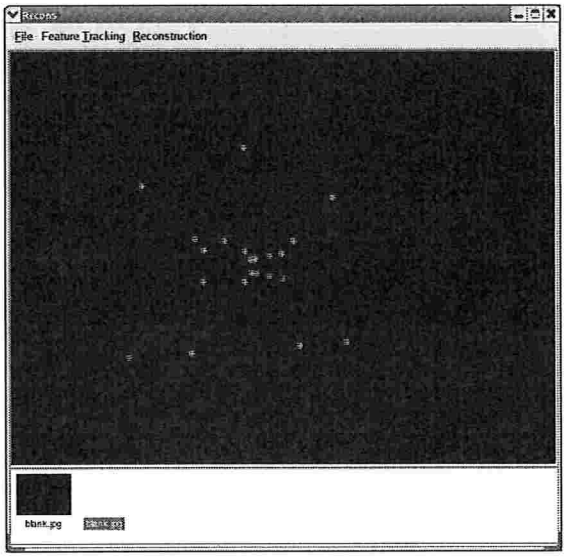

Figura 3.1: Uma tela do sistema em execução.

parte foi estudado o que pode ser feito a partir das mesmas primitivas (matriz fundamental, essencial, rastreamento) para estender o sistema para múltiplas câmeras.

3.1 Seleção e Rastreamento de Características

O módulo de seleção e rastreamento de características permite três modos de operação:

- *Automático*: todo o processo de seleção e rastreamento é automático, utilizando as técnicas apresentadas em [ST94, LK81];
- *Seleção Manual*: o usuário seleciona manualmente as características em um quadro da seqüência e o rastreamento é feito automaticamente;
- *Rastreamento Manual*: o usuário seleciona manualmente as correspondências entre dois quadros da seqüência;

Atualmente a escolha de qual modo de operação será utilizado é feita antes da execução e não pode ser modificada durante a operação. Se o modo automático esta ativo o usuário não pode ajudar o sistema selecionando características em um determinado quadro. Uma versão futura deve permitir que todos os três modos sejam utilizados em uma mesma execução do sistema.

3.1.1 Automático

Para a seleção e rastreamento automáticos utilizamos uma implementação aberta e gratuita do algoritmo desenvolvido por Lucas e Kanade (descrito na sessão 2.4.2), disponível em [Bir]. O seguinte processo é executado:

1. Seleciona até N características do primeiro quadro, onde N é determinado pelo usuário antes da execução do sistema;
2. Para cada quadro a seguir:
   (a) Tenta rastrear todas as N características;
   (b) Se não foi possível rastrear as N características, executa o algoritmo de seleção novamente.

Por exemplo, a imagem (3.2) apresenta o primeiro quadro de uma seqüência, com uma marcação para os pontos selecionados. A imagem (3.3) mostra uma imagem intermediária da seqüência, com os pontos rastreados a partir da imagem anterior marcados.

3.1.2 Seleção Manual

Neste modo de operação o usuário seleciona um número arbitrário de características e o sistema tenta rastrear estas características automaticamente nos próximos quadros. Se uma das

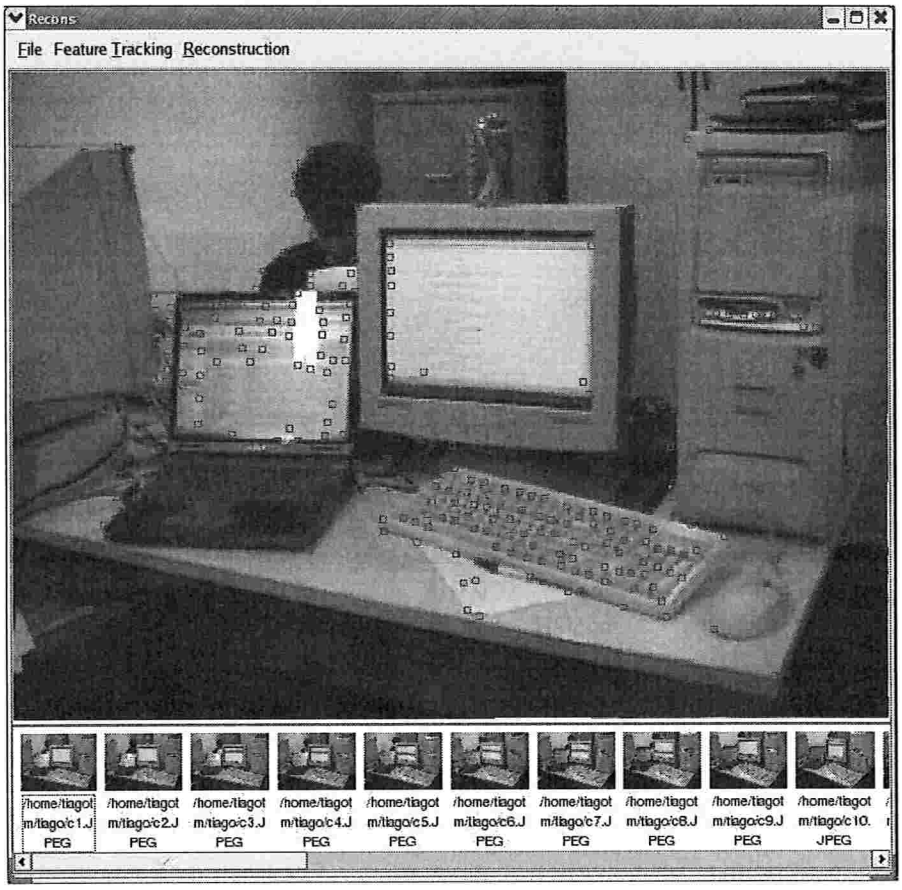

Figura 3.2: O primeiro quadro de uma seqüência. Os retângulos indicam regiões selecionadas para rastreamento.

Figura 3.3: Um quadro intermediário de uma seqüência. Os retângulos claros indicam regiões que foram rastreadas desde o último quadro.

características é perdida (isto é, não é rastreada entre dois quadros) o sistema não tenta executar a seleção automática. Desta forma, depois de um certo número de quadros o número de características sendo rastreadas tende a diminuir, até que eventualmente não exista mais nenhuma.

3.1.3 Rastreamento Manual

Neste modo de operação o usuário realiza o rastreamento de maneira manual. Para cada característica ele deve selecionar sua posição em dois quadros consecutivos.

3.1.4 Estrutura de Dados

Cada imagem possui uma lista de características associadas. O tamanho dessa lista é o mesmo para todas as imagens da seqüência. Cada característica encontrada é armazenada em uma estrutura com as seguintes informações: posição na imagem e estado (encontrada, rastreada, descartada, processada).

- **Encontrada**: a característica foi encontrada nesse quadro, isto é, não foi rastreada de um quadro anterior.
- **Rastreada**: a característica foi encontrada no quadro anterior.
- **Descartada**: essa posição da lista de características deve ser ignorada (acontece quando uma posição não é utilizada ou quando um passo do processo – RANSAC por exemplo – desativa a característica).
- **Processada**: esta característica já foi utilizada na estimação de pontos 3D.

Para facilitar o acesso às correspondências o sistema garante que, seja $C_{i,j}$ a característica j em um quadro i, então para uma imagem i onde $estado[C_{i,j}] = rastreada$, a característica $C_{i-1,j}$ contém a característica correspondente a $C_{i,j}$ no quadro $i-1$.

Assim sendo, se uma imagem i tem uma característica j com estado *rastreada*, podemos seguir "para trás" na seqüência de imagens até encontrarmos uma imagem k onde $C_{k,j}$ esta marcada como *encontrada* para descobrirmos o suporte temporal da característica j.

Dados Sintéticos

Para que o sistema possa ser testado com um objeto cuja estrutura 3D é conhecida (isto é, possuímos *ground truth*), o usuário pode passar como parâmetro o nome de um arquivo com uma lista de coordenadas 3D de pontos que definem os vértices deste objeto. O rastreamento é então substituído pelo seguinte processo:

- Uma câmera sintética é posicionada d unidades distante da origem pelo eixo Z (isto é, com coordenadas $(0, 0, d)$ "olhando" para $(0, 0, 0)$). O parâmetro d pode ser definido na linha de comando, porém os parâmetros intrínsecos da câmera (distância focal, ponto principal) são fixos.

- Para cada quadro i da seqüência, os pontos lidos do arquivo são transformados seguindo a fórmula: $P' = R_x(b + (i \cdot a_x)) \cdot R_y(b + (i \cdot a_y)) \cdot R_z(b + (i \cdot a_z)) \cdot P$, onde R_x, R_y e R_z são as matrizes de rotação ao redor dos eixos x, y e z respectivamente; b é o valor de "frame base" da seqüência, cujo valor padrão é zero, porém pode ser alterado por um parâmetro de linha de comando; e (a_x, a_y, a_z) são os ângulos de rotação entre quadros, em graus, cujos valores padrão são $(5, 5, 0)$ e podem também ser alterados pela linha de comando.

- Os pontos transformados são projetados no plano de imagem utilizando a câmera pre-

viamente construída. Os pontos $\mathbf{m}_{i,j}$ resultantes dessa projeção são armazenados no sistema como correlações no quadro i do ponto j, conforme detalhado acima.

O efeito disso é simular um rastreamento "perfeito" de características, sem a necessidade de nenhuma outra alteração no restante do sistema.

3.2 Câmeras

Com o conjunto de correspondências, o sistema pode estimar a posição relativa das câmeras utilizadas para capturar as imagens.

O processamento é feito em pares, a câmera estimada é depois levada ao frame de coordenadas global, que considera a câmera inicial na origem, apontando em direção ao eixo Z negativo.

Originalmente, para cada par de câmeras, era estimada apenas a matriz fundamental, utilizando o algoritmo RANSAC (2.2.2). Um dos intuitos iniciais do projeto era estudar a robustez do método e qualidade da estimação usando conjuntos de câmeras não calibradas.

No entanto, tentando utilizar apenas a matriz fundamental sem nenhuma calibração, encontramos algumas dificuldades:

- Influência da reconstrução a menos de uma transformação projetiva. Inicialmente buscávamos compreender qual o impacto da deformação pela ambigüidade projetiva, e qual a dificuldade de levarmos os pontos reconstruídos a suas coordenadas "reais". A experiência demonstrou que os pontos são por demais deformados para que possamos utilizá-los.

- Processamento com múltiplas câmeras. Como as câmeras incluem os parâmetros intrínsecos, é difícil encontrar a transformação que leva uma câmera qualquer ao frame de outra. Essa transformação é necessária para que todas as câmeras utilizem o mesmo sistema

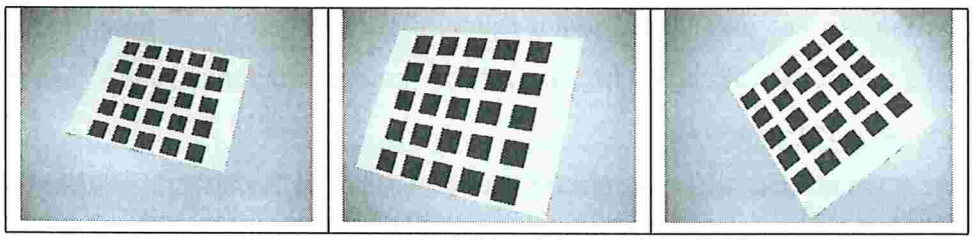

Figura 3.4: Exemplo de imagens usadas para calibrar a câmera.

de coordenadas. Pior, a segunda câmera estimada a partir da matriz fundamental está no infinito.

A partir dessas dificuldades, foi decidido que o sistema seria alterado para utilizar câmeras calibradas e reconstrução métrica. Por enquanto os parâmetros da câmera são fornecidos manualmente pelo operador, porém uma extensão planejada inclui adicionar um sub-sistema para realizar a auto-calibração das câmeras a partir de padrões e, posteriormente, a partir da própria seqüência de imagens.

Por exemplo, o método descrito por [Zha98], já utilizado em outros projetos do LaTIn (como [Per06]), permite que calibremos a câmera utilizada para a aquisição das imagens a partir de um conjunto de 6 imagens do padrão quadriculado, algumas dessas imagens são apresentadas na figura 3.4. Aplicando este método, seguido de uma rotina de minimização não linear sobre o erro de reprojeção obtivemos os seguintes parâmetros internos para uma câmera exemplo utilizada:

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} 756.09 & 9.68504 & 9.80734 \\ & 763.965 & -25.1164 \\ & & 1 \end{bmatrix}$$

Com os parâmetros de calibração da câmera, utilizamos o mesmo algoritmo RANSAC,

porém passamos como parâmetros para a minimização os pontos *calibrados*, isto é, transformados pela matriz de calibração. Isto nos retorna uma matriz fundamental "calibrada", que é chamada de *matriz essencial.*

Existe outra maneira de encontrarmos a matriz essencial, a partir da relação básica entre a matriz essencial e a matriz fundamental (equação 2.26).

Com a matriz essencial (2.2.3), é possível recuperarmos as câmeras euclidianas canônicas (2.23).

Na execução do RANSAC, nem todas as correlações são utilizadas para estimarmos a matriz fundamental. As correlações que não foram utilizadas, isto é, não fazem parte do consenso, são descartadas e não são levadas em conta na hora de estimarmos os pontos 3D.

Neste ponto já temos informação o suficiente para o caso de visão estéreo, que foi o primeiro objetivo do estudo. No entanto temos ainda o objetivo de estender os resultados para o caso de múltiplas câmeras. Agora descrevemos como essa informação foi obtida a partir da matriz essencial (que nos dá a geometria de duas câmeras) foi estendida para o caso de múltiplas câmeras.

Tendo cada par de câmeras definido pela matriz essencial estimada, movemos todas para ficarem no mesmo sistema de coordenadas, que considera a primeira câmera da seqüência na origem.

Como as câmeras estão calibradas, elas contém apenas seus parâmetros extrínsecos, elas são da forma $\mathcal{P} = [\mathbf{R}\ \mathbf{t}]$. Portanto, temos dois pares de câmeras canônicas: $(\mathcal{P}_0 = [\mathbf{I}_3\ \mathbf{0}_3], \mathcal{P}_1 = [\mathbf{R}\ \mathbf{t}])$ e $(\mathcal{P}'_1 = [\mathbf{I}_3\ \mathbf{0}_3], \mathcal{P}'_2 = [\mathbf{R}'\ \mathbf{t}'])$, queremos levar $\mathcal{P}'_2$ para o mesmo sistema de coordenadas de $\mathcal{P}_0$. Para isso, basta multiplicarmos $\mathcal{P}'_2$ pela direita por

$$\mathbf{T}_{4\times4} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_3^T & 1 \end{bmatrix}$$

onde $\mathbf{R}$ e $\mathbf{t}$ são os componentes da câmera $\mathcal{P}_1$, portanto:

$$\mathcal{P}_2 = \mathcal{P}_2' \, \mathbf{T}.$$

Executando essa transformação em cadeia, levamos todas as câmeras ao sistema de coordenadas do primeiro par de câmeras, com a primeira câmera na origem:

$$\mathcal{P}_i = \mathcal{P}_i' \, \mathbf{T}_{i-1},$$

onde $\mathbf{T}_i$ indica a matriz de transformação apresentada acima, formada a partir dos componentes de $\mathcal{P}_i$.

3.3 Triangulação dos Pontos no Espaço

Tendo as características rastreadas e as câmeras estimadas, podemos estimar a posição dos pontos no espaço 3D. Isto é realizado triangulando a partir da posição relativa das câmeras no espaço e a posição das características nas imagens processadas (2.2.4).

Uma solução de mínimos quadrados da equação (2.27) é encontrada utilizando a decomposição de valores singulares. Esta solução leva em conta todas as imagens onde cada característica foi rastreada e, por definição, minimiza a distância entre a projeção dos pontos sendo estimados pelas câmeras encontradas e as características rastreadas.

Para economizar processamento, e tentar aumentar a precisão da reconstrução, apenas características que foram rastreadas por pelo menos 3 quadros (isto é, possuem um *suporte temporal* de pelo menos 3 quadros) são consideradas. No final este algoritmo gera um conjunto de pontos 3D, cada um associado a uma característica rastreada por pelo menos 3 quadros.

3.4 Ajuste de Grupo

As câmeras e pontos estimados são utilizados como estimativa inicial no algoritmo de *ajuste de grupo* (sessão 2.3). O resultado do ajuste de grupo substitui os resultados obtidos anteriormente.

3.5 Visualização

Um dos objetivos do sistema é que ele continue a ser desenvolvido e que um módulo para a reconstrução de sólidos seja adicionada a ele, sobre o sistema para estimação da nuvem de pontos que já esta pronto.

Uma dificuldade crítica na reconstrução dos objetos 3D propriamente ditos, a partir da nuvem de pontos, é que existe muita ambiguidade com relação as quais seriam os planos e cantos do objeto. Isso torna muito difícil segmentar automaticamente os planos e polígonos que compões o objeto. Portanto, o sistema idealizado conta com um nível de interação humana auxiliar.

Em um estágio inicial o usuário veria os pontos 3D estimados sobrepostos às imagens que foram utilizadas para gerá-los. Nessa interface, o usuário deve selecionar pontos que pertencem a um mesmo plano. A partir do momento que mais que três pontos são selecionados, o sistema identifica, com uma cor diferente, todos os planos que ele considera coplanares a estes três selecionados, levando em conta uma distância máxima ponto-plano configurável. Para cada plano selecionado, o usuário escolhe uma opção confirmando sua escolha e estes pontos são removidos da lista de pontos disponíveis, e não mais aparecem na tela.

No próximo passo, o sistema intersecta os planos selecionados criando retas, e as intersecções destas retas geram vértices. Esse conjunto de retas e vértices serve como base para o usuário marcar, em uma ordem determinada, os polígonos que compõe o objeto. A saída final

do sistema seria uma descrição do modelo do objeto, utilizando alguma das formas apresentadas na sessão (2.5).

Capítulo 4

Resultados

Neste capítulo apresentamos resultados obtidos em vários experimentos utilizando imagens sintéticas. Nosso objetivo é estimar a qualidade da reconstrução gerada pelo sistema a partir de múltiplas imagens com ou sem a utilização do ajuste de grupo, que é um procedimento computacionalmente mais caro e demorado.

A qualidade da reconstrução foi julgada a partir de duas métricas: o erro de reprojeção dos pontos 3D encontrados nas câmeras estimadas, calculado como a distância absoluta, em *pixels*, dos pontos reprojetados aos pontos encontrados durante o rastreamento; e o erro de reconstrução 3D dos objetos, dado como a soma das distâncias das posições dos pontos reconstruídos aos pontos do objeto 3D original.

4.1 Metodologia dos Testes

Foram utilizados três objetos com características diferentes:

- *Paralelepípedo*: uma forma retangular descrita a partir de 12 pontos, os 8 cantos mais os os oito vértices de um paralelepípedo (figura 4.1).

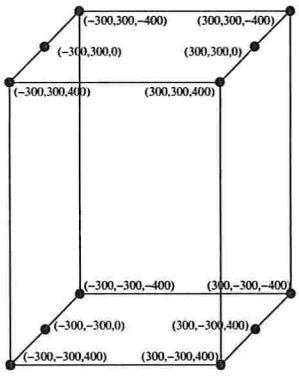

Figura 4.1: Paralelepípedo, uma forma retangular descrita a partir de 12 pontos.

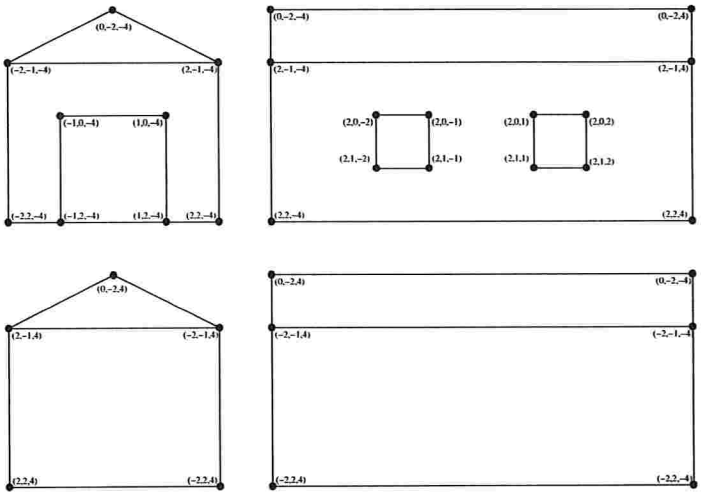

Figura 4.2: Casa, um paralelepípedo não simétrico.

- *Casa*: o desenho estilizado de uma casa com uma porta em uma das paredes frontais, e duas janelas em uma das paredes laterais (figura 4.2).
- *Esfera*: Cem pontos dispostos regularmente sobre a superfície de uma esfera.

Foram realizados 24 testes, utilizando 3 objetos com quatro casos de erro de amostragem nas correspondências (1, 3, 5 ou 7 *pixels* de trepidação na imagem – ver abaixo), e com ou sem o uso do ajuste de grupo.

Os valores de correspondência de pontos dos objetos sintéticos foram gerados projetando cada objeto sintético sobre 6 quadros, cada um com uma rotação de 15 graus ao redor do eixo Y ("para cima") em relação ao quadro anterior e agrupando as projeções de um mesmo ponto 3D original como correspondências, totalizando uma rotação de 90 graus.

Um erro de amostragem foi induzido para cada ponto na seguinte forma: $\mathbf{x} = (x \pm \mathtt{rnd}(T/2), y \pm \mathtt{rnd}(T/2))$, onde $\mathtt{rnd}(x)$ é uma função geradora de números pseudo-aleatórios que retorna um número entre 0 e $x - 1$, inclusive, e T é um parâmetro de entrada adicional do sistema indicando o limite máximo no erro, que chamamos de "trepidação" da amostra.

O erro de reprojeção é dado como a somatória das distâncias em *pixels* entre os pontos reconstruídos quando projetados pelas câmeras estimadas e os pontos originalmente rastreados. Assim, definimos o erro de reprojeção como:

$$\sum_{i,j} |o_{i,j} - r_{i,j}|,$$

onde o_i é o ponto do objeto associado à característica utilizada para a reconstrução de um ponto P_i e r_i é a projeção deste ponto. Isto é feito para cada câmera j da seqüência.

O erro de reconstrução é dado como a distância absoluta entre os pontos reconstruídos e os pontos originais dividida pela distância média dos pontos do modelo ao seu baricentro. Isto é realizado pois, além da reconstrução ser a menos da escala, cada objeto possui um tamanho

diferente e assim levamos todos a uma unidade similar. Portanto:

$$\sum_i \frac{O_i - P_i}{DB},$$

onde O_i é o ponto no objeto que foi utilizado pela reconstrução do ponto P_i e DB é a distância média dos pontos do objeto *reconstruído* ao seu baricentro.

Os testes foram reproduzidos utilizando ou não o *ajuste de grupo*, pois este sendo um algoritmo muito custoso em processamento nos interessa saber a sua real influência no resultado final.

Equipamento Utilizado

Os testes foram realizados em um PC com processador AMD *Athlon XP* operando a 1200MHz com 512MB de memória RAM. Foi utilizado o sistema operacional livre *Linux*, mais especificamente a distribuição *Fedora Core 2*.

O código foi compilado pelo GCC versão 3.0, com a opção *-O3* para forçar o uso de otimizações agressivas.

4.2 Resultados

Casa

As tabelas 4.1 e 4.2 apresentam os resultados da reconstrução do modelo "*casa*" com e sem o uso do ajuste de grupo.

Em ambas as tabelas fica claro que o erro de amostragem dos pontos influencia em muito tanto o erro de reprojeção como o erro de reconstrução. Um erro máximo de 2.754 (no caso de $T = 7$ sem ajuste de grupo) implica em um ponto estar distante da sua posição esperada

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	7.9	1.0	4.0	12.0	0.055	0.025	0.019	0.110	0.20
3	13.6	1.2	10.8	16.5	0.045	0.042	0.007	0.209	0.20
5	22.9	6.1	11.8	41.5	0.483	0.326	0.047	1.497	0.19
7	15.1	3.9	6.1	25.1	0.849	0.716	0.259	2.754	0.21

Tabela 4.1: Erros de reprojeção e reconstrução da casa, sem uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	2.3	0.3	1.6	3.0	0.047	0.023	0.020	0.104	785
3	1.4	0.4	0.3	2.1	0.046	0.015	0.014	0.073	1157
5	7.8	2.3	3.5	14.4	0.403	0.349	0.057	1.465	352
7	6.3	1.7	1.3	8.9	0.748	0.562	0.190	2.069	680

Tabela 4.2: Erros de reprojeção e reconstrução da casa, com o uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

mais que o dobro da distância média dos pontos ao baricentro.

Como esperado o uso do ajuste de grupo reduz em muito o erro de reprojeção, como conseqüência tendo uma influência positiva também no erro de reconstrução. No entanto o tempo de execução é diversas ordens de magnitude maior. A implementação de algumas das técnicas descritas acima pode reduzir esse tempo, e em trabalhos futuros pretendemos explorar e melhor analisar esta possibilidade.

Um fato curioso é que em ambas as tabelas o caso com $T = 7$ apresenta um erro de reprojeção menor que o caso $T = 5$, mesmo tendo um erro de reconstrução maior. Isso deixa claro uma situação encontrada muitas vezes no desenvolvimento do sistema: as técnicas utilizadas minimizam o erro de reprojeção (por ser a única informação disponível) e o menor erro de reprojeção não necessariamente corresponde ao menor erro de reconstrução. (apesar de "provavelmente" corresponder a um erro pequeno de reconstrução).

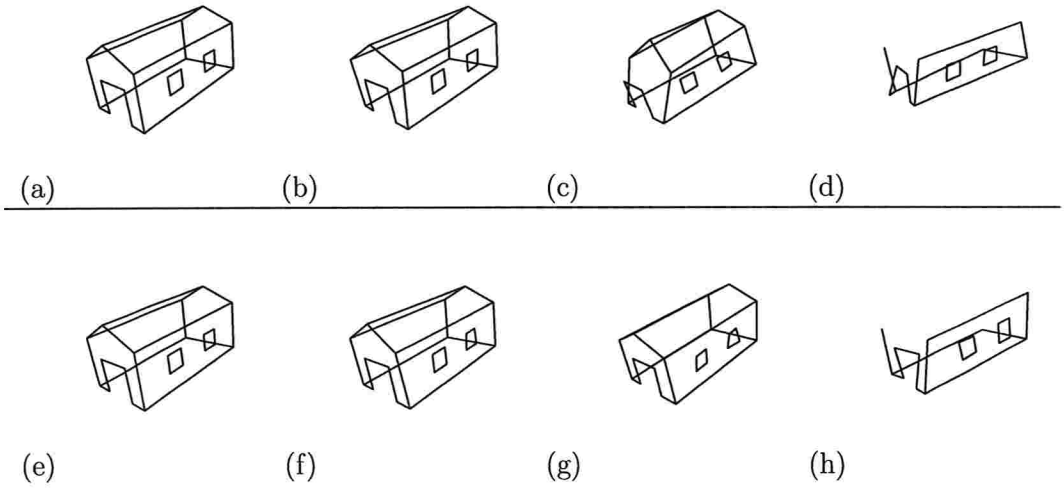

Figura 4.3: O resultado da reconstrução do modelo casa. Os pontos reconstruídos foram conectados por segmentos para facilitar a visualização. (a) - (d) são os modelos reconstruídos sem o uso do ajuste, com trepidação de 1, 3, 5 e 7 respectivamente. (e) - (h) é a mesma seqüência porém utilizando o ajuste de grupo. Em (d) e (h), onde o erro de trepidação é 7, nem todos os pontos foram reconstruídos, por isso o modelo esta incompleto.

A figura 4.3 apresenta alguns resultados da reconstrução da casa. No caso de $T = 7$ o RANSAC não inclui todos os pontos do modelo no grupo de consenso, assim nem todos os pontos foram reconstruídos.

Paralelepípedo 2

As tabelas 4.3 e 4.4 apresentam os resultados da reconstrução do modelo "*paralelepípedo 2*" com e sem o uso do ajuste de grupo.

Como no caso da casa, tanto o erro de reprojeção quanto o erro de reconstrução crescem conforme o nível de ruído na amostragem aumenta.

No caso deste modelo podemos ver que o ajuste de grupo ajuda mais que no caso da casa.

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	3.7	0.7	2.3	5.1	0.023	0.009	0.008	0.037	0.18
3	44.4	16.1	14.9	87.9	0.409	0.069	0.325	0.541	0.18
5	62.2	21.5	24.1	104.1	0.474	0.249	0.039	0.962	0.17
7	63.2	17.2	36.2	100.3	0.562	0.180	0.314	0.849	0.19

Tabela 4.3: Erros de reprojeção e reconstrução do paralelepípedo 2, sem uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	2.8	0.4	2.0	3.8	0.043	0.009	0.028	0.068	620
3	20.5	6.0	10.0	36.8	0.369	0.185	0.344	0.580	227
5	13.5	4.4	5.4	22.9	0.292	0.113	0.145	0.478	237
7	11.4	4.8	2.4	17.6	0.304	0.042	0.235	0.373	847

Tabela 4.4: Erros de reprojeção e reconstrução do paralelepípedo 2, com o uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

Um erro de reconstrução máximo de 0.962, no caso $T = 5$ sem ajuste de grupo, se reduz para um erro de 0.478 unidades com o ajuste de grupo. Uma redução ainda maior pode ser vista no erro de reprojeção. Devido a um menor número de pontos, o algoritmo de ajuste de grupo executou em um tempo menor do que no caso da casa, e também promoveu um refinamento maior nos resultados.

A figura 4.4 apresenta alguns resultados da reconstrução do paralelepípedo 2. O ponto de vista é o mesmo em todas as cenas, a aparência da câmera estar em uma posição diferente é devido aos erros de reconstrução.

Esfera

As tabelas 4.5 e 4.6 apresentam os resultados da reconstrução do modelo "*esfera*" com e sem o uso do ajuste de grupo.

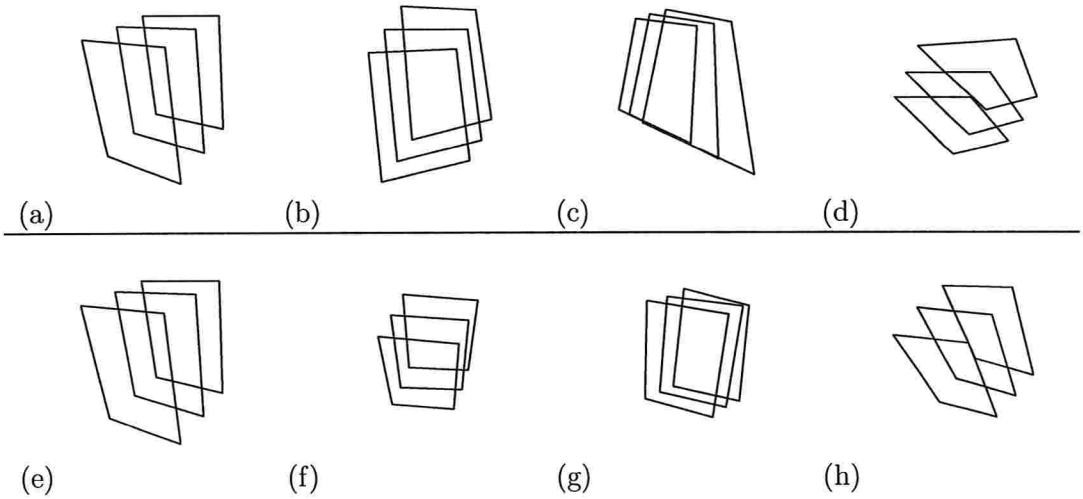

Figura 4.4: O resultado da reconstrução do modelo paralelepípedo 2. Os pontos reconstruídos foram conectados por segmentos para facilitar a visualização. (a) - (d) são os modelos reconstruídos sem o uso do ajuste, com trepidação de 1, 3, 5 e 7 respectivamente. (e) - (h) é a mesma seqüência porém utilizando o ajuste de grupo.

A esfera apresenta em todos os testes um erro tanto de reconstrução quanto de reprojeção menor que os outros modelos testados. Isso se deve a termos um maior número de pontos o que, considerando que o erro induzido é aleatório, aumenta a probabilidade do sistema encontrar o movimento correto entre as câmeras. O fato do modelo ser o mais simétrico dos três testados também pode ajudar a reconstrução.

Talvez pelos argumentos apresentados acima, o uso do ajuste de grupo não influencia tanto o erro de reconstrução quanto nos outros modelos, é provável que a estimativa inicial já esteja muito próxima do valor real. O erro de reprojeção, porém, melhora substancialmente, caindo em quase um terço no caso $T = 7$ entre a execução sem ajuste e a com ajuste.

Devido ao grande número de pontos o tempo de execução com ajuste é muito maior que nos outros modelos, chegando a 12802 segundos (mais de três horas e trinta minutos) no caso de

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	2.4	0.2	1.7	3.0	0.012	0.005	0.002	0.030	0.42
3	3.0	0.7	1.2	5.0	0.044	0.021	0.010	0.122	0.38
5	9.1	1.9	4.7	17.7	0.131	0.071	0.024	0.346	0.39
7	12.8	2.4	6.8	28.2	0.610	0.390	0.091	1.387	0.42

Tabela 4.5: Erros de reprojeção e reconstrução da esfera, sem uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

T	Média	D.P.	Mín.	Máx.	Média	D.P.	Mín.	Máx.	Tempo
1	0.8	0.2	0.3	1.2	0.011	0.005	0.001	0.024	4035
3	2.5	0.6	0.8	4.0	0.045	0.026	0.009	0.111	3500
5	4.5	1.1	1.6	8.8	0.090	0.046	0.015	0.219	5265
7	4.7	1.5	0.7	11.0	0.579	0.390	0.029	1.589	12802

Tabela 4.6: Erros de reprojeção e reconstrução da esfera, com o uso do ajuste de grupo. À esquerda os erros de reprojeção e, à direita, os erros de reconstrução. O tempo de execução do sistema, em segundos, também é dado. T indica a trepidação induzida no teste.

$T = 7$. Para modelos desse tamanho ou maiores, fica claro que é necessária a implementação de alguma das otimizações comentadas acima para o uso prático do sistema.

A figura 4.5 apresenta alguns resultados da reconstrução da esfera. Nesse modelo, cada lateral é definida por dois pontos, o que explica a duplicidade de segmentos nas versões com maior erro induzido. Nas versões com menor erro os dois segmentos estão tão próximos que são inconfundíveis.

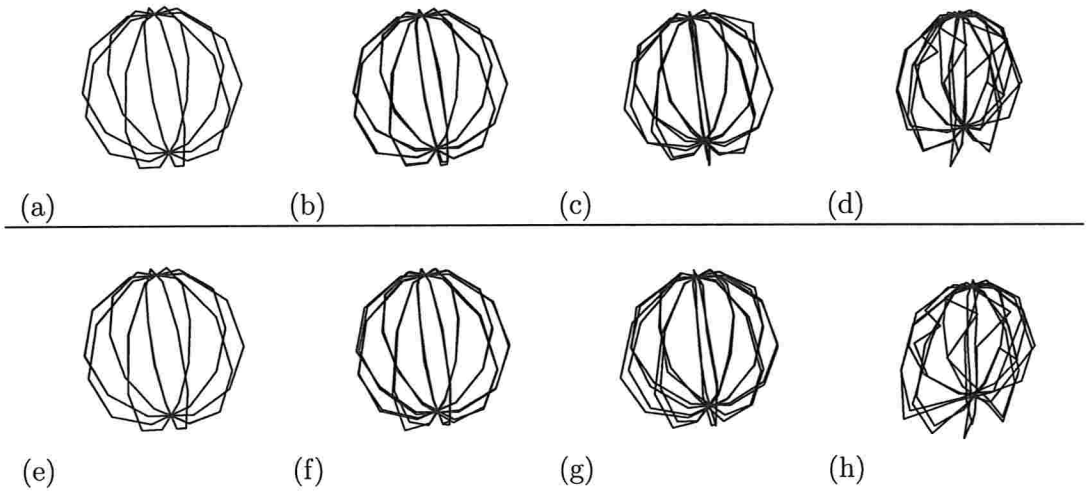

Figura 4.5: O resultado da reconstrução do modelo esfera. Os pontos reconstruídos foram conectados por segmentos para facilitar a visualização. (a) - (d) são os modelos reconstruídos sem o uso do ajuste, com trepidação de 1, 3, 5 e 7 respectivamente. (e) - (h) é a mesma seqüência porém utilizando o ajuste de grupo.

Capítulo 5

Conclusão

Neste documento tratamos um problema atual na computação gráfica, mais especificamente o de produzir modelos tri-dimensionais para uso na composição de cenas. Atualmente a quantidade de trabalho necessária para a criação de cenas para uso em simulações e entretenimento é muito grande, necessitando de muitas horas de esforço por um grande número de pessoas. Um sistema para automatizar e/ou facilitar esta tarefa seria útil e permitiria a utilização de modelos em várias aplicações.

Em particular, nós do Laboratório de Tecnologias de Interação (LaTIn) do Instituto de Matemática e Estatística da Universidade de São Paulo, temos interesse no desenvolvimento de um sistema como esse para o estudo de ferramentas de interação em ambientes de realidade virtual e estendida.

A princípio, tal sistema de reconstrução deve ser de fácil uso, não exigir equipamentos especiais, porém não precisa ter uma precisão muito grande. Devido também à popularização de câmeras digitais, acreditamos que um método baseado em imagens tornaria tal sistema acessível a uma grande variedade de aplicações.

Recentemente, a técnica mais pesquisada em visão computacional é a baseada em geometria

projetiva, capaz de reconstruir a estrutura 3D de objetos a partir de um conjunto de imagens. Neste trabalho apresentamos uma síntese dos conceitos de geometria projetiva necessários para resolver o problema de reconstrução a partir de um conjunto de correspondências entre imagens.

Um dos conceitos fundamentais é o modelo de câmera necessário para descrever as transformações entre os objetos e a imagem. Utilizamos o modelo de câmera *pinhole*, e descrevemos o que são os parâmetros intrínsecos e extrínsecos dessa câmera.

A geometria projetiva nos permite determinar quais informações podemos extrair de uma única imagem. Estas informações se provam insuficientes para reconstruir um objeto, portanto mais imagens são necessárias, que é explorada pela visão estéreo, descrita pela geometria epipolar, que descreve a estrutura de um sistema de duas câmeras, e pode ser definida matematicamente pela matriz fundamental. No entanto o sistema descrito somente pela matriz fundamental nos permite apenas uma reconstrução a menos de uma transformação projetiva, o que é insuficiente para as nossas necessidades.

Através da imposição de algumas restrições sobre as câmeras, é possível tornar a reconstrução projetiva em afim e até métrica, bastando para isso conhecermos os parâmetros intrínsecos da câmera. Nesse caso ao invés da matriz fundamental, podemos trabalhar com a matriz essencial.

Um outro objetivo importante deste trabalho foi o de automatizar todo o processo de reconstrução. Para isso estudamos várias técnicas de detecção e rastreamento automáticas de características. Neste trabalho utilizamos o método de Kanade, Lucas e Tomasi [ST94, LK81]. Este método fornece um conjunto de pontos que pode ser rastreado robustamente entre imagens obtidas com câmera que apresente um movimento significativo entre elas. Porém, como esses pontos são esparsos, a estrutura recuperada pelo nosso sistema também corresponde a uma nuvem esparsa de pontos 3D, embora as técnicas tradicionais para modelamento de

objetos utilizadas em computação gráfica em geral utilizem superfícies.

5.1 Trabalhos Futuros

Infelizmente os métodos de reconstrução baseados em geometria projetiva se mostraram numericamente bastante instáveis, o que resultou num atraso considerável em nosso projeto. Identificamos como problema principal o erro de rastreamento de característica, pois mesmo com dados sintéticos se o erro induzido no rastreamento aumenta o sistema se mostra incapaz de gerar um resultado preciso ou completo.

Assim que o sistema for capaz de gerar uma nuvem de pontos com maior precisão, pretendemos iniciar o trabalho em um sistema interativo para a reconstrução de sólidos, a partir dos pontos encontrados e das imagens utilizadas. Tal sistema apresentaria ao usuário o conjunto de imagens originais, com os pontos reconstruídos sobrepostos, e permitiria ao usuário selecionar arestas e planos para delimitar as faces do objeto.

Referências Bibliográficas

[AB91] E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. *M. Landy and J. A. Movshon, (eds) Computational Models of Visual Processing*, 1991. 61

[Adv02] Advantica. Roundshort 220vr, 2002. http://www.advantica.com/rs3.htm. 62

[Bar05] T.S. Barcelos. Interfaces prestativas baseadas em visão computacional e informação de contexto. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Junho 2005. 6

[Bar06] R.A. Barbosa. Detecção e estimação da orientação 3d de faces. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Abril 2006.

[BFBB92] J.L. Barron, D.J. Fleet, S.S. Beauchemin, and T.A. Burkitt. Performance of optical flow techniques. *CVPR*, 92:236–242, 1992. 47, 48, 51

[Bir] Stan Birchfield. Klt: An implementation of the kanade-lucas-tomasi feature tracker. http://www.ces.clemson.edu/ stb/klt/. 71

[Bir98] Stanley Birchfield. An introduction to projective geometry, 1998. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/BIRCHFIELD/projective.html. 14

[Boi81] S. Boinodiris. Hidden surface elimination for complex graphical scenes. *SIGGRAPH Comput. Graph.*, 14(4):153–167, 1981. 59

[Boo] Boost. Boost c++ libraries. http://www.boost.org/. 68

[BSSM05] R.A. Barbosa, F. Silva, T.T. Santos, and C.H. Morimoto. An extensible automatic video browsing tool. In *Workshop Teses e Dissertações - Sibgrapi 05*, Natal, RN, October 2005.

[Cab05] M.C. Cabral. Interação em ambientes de realidade virtual através de gestos utilizando visão computacional. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Abril 2005. 6

[Cer04] H.P.F. Ceribelli. Construção de imagens panorâmicas a partir de vídeo. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Abril 2004. 6

[Che95] Shenchang Eric Chen. QuickTime VR — an image-based approach to virtual environment navigation. *Computer Graphics*, 29(Annual Conference Series):29–38, 1995. 61, 62

[Cin] Cinema.com. Shrek production information. http://www.cinema.com/articles/463/shrek-production-information.phtml. 2

[Cla76] James H. Clark. Hierarchical geometric models for visible surface algorithms. *Commun. ACM*, 19(10):547–554, 1976. 59

[CMZ05] M.C. Cabral, C.H. Morimoto, and M.K. Zuffo. On the usability of gesture interfaces in virtual reality environments. In *CLIHC'05, Conferencia Latinoamericana de Interacion Humano-Computadora*, Cuernava, México, October 2005.

[CRB99] Roberto Cipolla, Duncan Robertson, and Edmond Boyer. Photobuilder – 3d models of architectural scenes from uncalibrated images. In *IEEE International Conference on Multimedia Computing and Systems,Firenze*, volume I, pages 25–31, June 1999. 32

[CSM95] Chen, Eric S., and Gavin Miller. Cylindrical to planar image mapping using scanline coherence. *United States Patent number 5,396,583*, March 1995. 62, 63

[DTM96] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Computer Graphics*, 30(Annual Conference Series):11–20, 1996.

[Eve02] Everent. Globuscope panoramic camera, 2002. http://www.everent.com/globus/. 62

[Evi96] Martin P. Evison. Computerised 3d facil reconstruction, 1996. http://www.shef.ac.uk/assem/1/evison.html. 2

[FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 32

[FL01] Olivier Faugeras and Quang-Tuan Luong. *The Geometry of Multiple Images.* The MIT Press, 2001. 5, 14, 31, 32, 42

[Gar99] Michael Garland. *Quadric-Based Polygonal Surface Simplification.* PhD thesis, Carnegie Mellon University, 1999. 58

[GGSC96] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. *Computer Graphics*, 30(Annual Conference Series):43–54, 1996. 65

[Gre] G. Greiner. Geometric modelling. http://atrey.karlin.mff.cuni.cz/projekty/vrr/doc/grafika/geometric%20modelling.pdf. 60

[HA04] R. I. Hartley and Zisserman A. *Multiple View Geometry in Computer Vision.* Cambridge University Press, second edition, 2004. 5, 14, 32, 42, 43, 44

[Har95] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision*, pages 1064–1070. Cambridge, 1995.

[HGC92] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 761–764. Urbana-Champaign, 1992. 14

[Hor86] Berthold K. Horn. *Robot Vision.* McGraw-Hill Higher Education, 1986. 3, 47, 48, 50

[IMD] IMDB.com. Shrek trivia. http://www.imdb.com/title/tt0126029/trivia. 1

[Int] Intel. Open computer vision library. http://sourceforge.net/projects/opencvlibrary/. 68

[Ioc98] Luca Iocchi. Multiresolution stereo vision system for mobile robots, 1998. http://www.dis.uniroma1.it/ iocchi/stereo/stereo.html. 14

[KCM95] O.J. Kwon, R. Chellappa, and C.H. Morimoto. Motion compensated subband coding of video acquired from a moving platform. In *Proc. of IEEE International Conf. on Acoustics, Speech, and Signal Processing*, pages 2185–2188, Detroit, MI, January 1995.

[LF96] Q.T. Luong and O. Faugeras. The fundamental matrix: Theory, algorithms and stability analysis. *International Journal of Computer Vision*, 17(1):43–76, 1996. 14

[LH96] Marc Levoy and Pat Hanrahan. Light field rendering. *Computer Graphics*, 30(Annual Conference Series):31–42, 1996. 63, 65

[LK81] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981. 48, 52, 70, 94

[MAF02] C.H. Morimoto, A.Amir, and M. Flickner. Free head motion eye gaze tracking without calibration. In *Proc. of CHI 2002 Extended Abstracts on Human Factors in Computing Systems*, pages 586–587, Minneapolis, Minnesota, 2002.

[MAFZ02] C.H. Morimoto, A. Amir, M. Flickner, and S. Zhai. Detecting eye position and gaze from a single camera and 2 light sources. In *Proc. of the ICPR 2002: 16th International Conference on Pattern Recognition*, volume VI, pages 314–317, Quebec, CA, August 2002.

[Man88] M. Mantyla. An introduction to solid modeling. *Principles of Computer Science*, 1988. 57

[MBC97a] C.H. Morimoto, S. Balakirsky, and R. Chellappa. Fast image stabilization and mosaicking for predator data. In *Proc. DARPA Image Understanding Workshop*, New Orleans, LA, May 1997.

[MBC97b] C.H. Morimoto, P. Burlina, and R. Chellappa. Hybrid video coding - using global motion transformations. In *Proc. International Conference on Image Processing*, Santa Barbara, CA, October 1997.

[MBCY96] C.H. Morimoto, P. Burlina, R. Chellappa, and Y.S. Yao. Performance analysis of model-based video coding. In *Proc. International Conference on Image Processing*, volume III, pages 279–282, Lausanne, Switzerland, September 1996.

[MC96a] C.H. Morimoto and R. Chellappa. Fast electronic digital image stabilization. In *Proc. International Conference on Pattern Recognition*, Vienna, Austria, August 1996.

[MC96b] C.H. Morimoto and R. Chellappa. Fast electronic digital image stabilization for off-road navigation. *Real-Time Imaging*, II(5):285–296, October 1996.

[MC97a] C.H. Morimoto and R. Chellappa. Evaluation of image stabilization algorithms. In *Proc. DARPA Image Understanding Workshop*, New Orleans, LA, May 1997.

[MC97b] C.H. Morimoto and R. Chellappa. Fast 3d stabilization and mosaicking. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 660–665, Puerto Rico,PR, June 1997.

[MDD+95] C.H. Morimoto, D. DeMenthon, L.S. Davis, R. Chellappa, and R. Nelson. Detection of independently moving objects in passive video. In I. Masaki, editor, *Proc. of Intelligent Vehicles Workshop*, pages 270–275, Detroit, MI, September 1995.

[MF00] C.H. Morimoto and M. Flickner. Real-time multiple face detection using active illumination. In *Proc. of the 3rd Int. Conf. on Automatic Face and Gesture Recognition*, Grenoble, France, March 2000.

[Mim04] M.R.M. Mimica. Desenvolvimento de um rastreador de olhar apropriado para interação humano computador. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Junho 2004. 6

[MKA+99] C.H. Morimoto, D. Koons, A. Amir, M. Flickner, and S. Zhai. Keeping an eye for hci. In *SIBGRAPI'99, XII Brazilian Symposium in Computer Graphics and Image Proc.*, pages 171–176, Campinas, Brazil, October 1999.

[MKAF99] C.H. Morimoto, D. Koons, A. Amir, and M. Flickner. Frame-rate pupil detector and gaze tracker. In *ICCV'99 FRAME-RATE workshop*, September 1999.

[MKAF00] C.H. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, 18(4):331–336, March 2000.

[ML97] McMillan and Leonard. *An Image-Based Approach to Three-Dimensional Computer Graphics.* PhD thesis, University of North Carolina, Apr 1997.

[MM03] M.R.M. Mimica and C.H. Morimoto. A computer vision framework for eye gaze tracking. In *SIBGRAPI'03, XVI Brazilian Symposium in Computer Graphics and Image Proc.*, pages 406–411, São Carlos, SP, October 2003.

[MM05] C.H. Morimoto and M.R.M. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98(1):4–24, 2005.

[Mor00a] C.H. Morimoto. Interfaces baseadas em técnicas de visão computacional. In *XI SEMAC*, São José do Rio Preto, São Paulo, Brasil, Novembro 2000.

[Mor00b] C.H. Morimoto. Virtual autonomous agents with vision. In *I WAICV - Workshop for Artificial Inteligence and Computer Vision, em conjunto com o SBIA 2000*, Atibaia, São Paulo, Brazil, 2000.

[MT98] Roger Mohr and Bill Triggs. Projective geometry for image analysis, 1998. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MOHR_TRIGGS/isprs96.html. 14

[MYD96] C.H. Morimoto, Y. Yacoob, and L.S. Davis. Recognition of head gestures using hidden markov models. In *Proc. International Conference on Pattern Recognition*, Vienna, Austria, August 1996.

[NLL+01] Nyland, Lars, Lastra, Anselmo, McAllister, David K., Popescu, Voicu, McCue, and Chris. Capturing,processing and rendering real-world scenes. In *Videometrics and Optical Methods for 3D Shape Measurement, Electronic Imaging 2001, Photonics West*, volume 4309, San Jose, CA, 2001. SPIE.

[NMP+99] Lars Nyland, David McAllister, Voicu Popescu, Chris McCue, Anselmo Lastra, Paul Rademacher, Manuel Oliveira, Gary Bishop, Gopi Meenakshisundaram, Matt Cutts, and Henry Fuchs. The impact of dense range data on computer graphics. In *MVIEW '99: Proceedings of the IEEE Workshop on Multi-View Modeling & Analysis of Visual Scenes*, page 3. IEEE Computer Society, 1999.

[OB99] Manuel M. Oliveira and Gary Bishop. Image-based objects. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 191–198. ACM Press, 1999. 61

[OBM00] Manuel M. Oliveira, Gary Bishop, and David McAllister. Relief texture mapping. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 359–368. ACM Press/Addison-Wesley Publishing Co., 2000.

[Oli02] Manuel M. Oliveira. Image-based modeling and rendering techniques: A survey. *RITA - Revista de Informática Teórica e Aplicada*, 2002. vi, 63, 64, 66

[PBM04] B. Pera, R.A. Barbosa, and C.H. Morimoto. Análise comparativa de diferentes espaços cromáticos para detecção de cor de pele. In *Anais do Simpósio de Realidade Virtual*, page 660, São Paulo, Brazil, Novembro 2004.

[Per06] B. Pera. Reconstrução de faces. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Abril 2006. 77

[PFTV92] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, second edition, 1992. 45

[Pie91] Les Piegl. On nurbs: a survey. *IEEE Comput. Graph. Appl.*, 11(1):55–71, 1991. 58

[Req80] Aristides G. Requicha. Representations for rigid solids: Theory, methods, and systems. *ACM Comput. Surv.*, 12(4):437–464, 1980. v, 57, 58, 59

[Res] Wolfram Research. B-spline. http://mathworld.wolfram.com/B-Spline.html.

[San04] T.T. Santos. Detecção de cortes em sequências de vídeo. Master's thesis, Departamento de Ciência da Computação do Instituto de Matemática e Estatística, Universidade de São Paulo, Agosto 2004. 6

[SGHS98] Jonathan W. Shade, Steven J. Gortler, Li-Wei He, and Richard Szeliski. Layered depth images. *Computer Graphics*, 32(Annual Conference Series):231–242, 1998.

[SM03] T.T. Santos and C.H. Morimoto. Estruturação e indexação de vídeo digital. In *Anais do Simpósio Brasileiro de Sistemas Multimídia e Web - WebMídia 2003*, Salvador, Brazil, Novembro 2003.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600, 1994. 52, 56, 70, 94

[SW88] Hanan Samet and Robert E. Webber. Hierarchical data structures and algorithms for computer graphics. part i. *IEEE Comput. Graph. Appl.*, 8(3):48–68, 1988. 59

[TMHF00] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000. 44, 45

[Tri87] H. P. Trivedi. Estimation of stereo and motion parameters using a variational principle. *Image Vision Comput.*, 5(2):181–183, 1987.

[Tri98] Bill Triggs. Projective stereo vision, 1998. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MOHR_TRIGGS/node46.html. 14

[Tro] Trolltech. Qt cross platform c++ library. http://www.trolltech.com/products/qt/index.html. 68

[Wat99] Alan Watt. *3D Computer Graphics*. Addison Wesley Professional, 1999. 15

[Wex] Daniel Wexler. Shrek rendering statistics. http://www.flarg.com/Graphics/ShrekRenderingStatistics.html. 2

[Wik] Wikipedia. Gestalt. http://pt.wikipedia.org/wiki/Gestalt. 9

[Wil83] Lance Williams. Pyramidal parametrics. In *SIGGRAPH '83: Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, pages 1–11. ACM Press, 1983.

[YR02] Y.K.Wong and R.Cipolla. Reconstruction of outdoor sculptures from silhouettes under approximate circular motion of an uncalibrated hand-held camera., 2002. http://citeseer.ist.psu.edu/658285.html.

[ZBJ05] Y. Zana, R.A. Barbosa, and R.M. Cesar Jr. Eye detection. In *SIBGRAPI'05, XVIII Brazilian Symposium in Computer Graphics and Image Proc.*, pages 171–176, Natal, Brazil, October 2005.

[ZCHS03] Li Zhang, Brian Curless, Aaron Hertzmann, and Steven M. Seitz. Shape and motion under varying illumination: Unifying structure from motion, photometric stereo, and multi-view stereo. In *The 9th IEEE International Conference on Computer Vision*, pages 618–625, Oct. 2003. 14

[ZDFL95] Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry., 1995. 14

[Zha98] Zheng Zhang. A new flexible technique for camera calibration. Technical report, Microsoft Research, Microsoft Corporation, 1998. 77

[ZMI99] S. Zhai, C.H. Morimoto, and S. Ihde. Manual and gaze input cascaded (magic) pointing. In *Proc. ACM SIGCHI - Human Factors in Computing Systems Conference*, pages 246–253, Pittsburgh, PA, May 1999.