André Chalella das Neves

Reconstrução 3D a partir de Sequência de Imagens

Monografia de Conclusão de Curso apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro

São Paulo
2010

André Chalella das Neves

Reconstrução 3D a partir de Sequência de Imagens

Monografia de Conclusão de Curso apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro

Área de concentração:
Engenharia Mecatrônica

Orientador:
Prof. Dr. Marcos de Sales Guerra Tsuzuki

São Paulo
2010

Ficha Catalográfica

Neves, André Chalella das
Reconstrução 3D a partir de Sequência de Imagens. São Paulo, 2010. 46 p.

Monografia de Conclusão de Curso — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Visão Computacional. 2. Geometria Epipolar. 3. Nuvem de Pontos. I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos. II.

Ao meu pai, à minha mãe e ao meu irmão,

aos meus amigos,

à Rateria

e aos colegas da Mecatrônica.

Agradecimentos

Agradeço ao Prof. Dr. Marcos de Sales Guerra Tsuzuki, por ter me orientado, estimulado e confiado neste projeto, e ao Dr. Rogério Yugo Takimoto, pela ajuda dada em diversos momentos.

Resumo

Este projeto tem como objetivo construir um sistema capaz de gerar um modelo tridimensional de uma cena a partir de imagens dessa mesma cena e do conhecimento da localização tridimensional de 5 de seus pontos. O modelo tridimensional a ser obtido é uma nuvem de pontos desestruturada, isto é, pontos tridimensionais não ligados por estruturas como arestas, superfícies, etc. O sistema construído aceita um par de imagens e primeiro determina a sua matriz fundamental a partir de 8 ou mais correspondências de pontos característicos de um par de imagens, para depois utilizar o conhecimento dos 5 pontos para efetuar a reconstrução real. Vemos que o problema de determinação da matriz fundamental é malposto, e por isso exige a normalização dos seus parâmetros. Ao fim, verificamos que o sistema funciona e é robusto, apresentando pouca sensibilidade a imprecisões na determinação de pontos característicos.

Abstract

This project aims to build a system capable of creating a tridimensional model from images taken from a scene, along with the knowledge of the tridimensional position of 5 of its points. The tridimensional model to be created is an unstructured point cloud, which consists of tridimensional points not connected by elements such as edges, surfaces, etc. Our system accepts a pair of images and first determines its fundamental matrix by selecting at least 8 correspondences of feature points from two images, then uses the 5 known points to make the real reconstruction. We show that the problem of determining the fundamental matrix is ill-conditioned, thus requiring normalization of its parameteres. Finally, we see that our system works and is robust, presenting little sensibility to spatial inaccuracies in the selection of feature points.

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Introdução

A Visão Computacional pode ser definida como a ciência e o conjunto de tecnologias que propiciam a uma máquina a capacidade de ver [1, 2]. Seus avanços permitem o desenvolvimento de sistemas capazes de realizar tarefas eminentemente humanas, como guiar um veículo por um caminho, reconhecer expressões faciais e conceber modelos de objetos ou cenas que se enxerga. Como se nota, a automatização dessas tarefas traz grandes benefícios para o homem em diversas frentes; certas atividades podem ser tediosas, ou ineficientes, ou deixar a desejar em termos de exatidão quando executadas por humanos sem assistência computacional.

Com o crescente poder de processamento de dados que a tecnologia provê desde o advento do computador, torna-se possível — e cada vez mais conveniente — delegar a sistemas computacionais essas atividades. Tal possibilidade fomenta o estudo na área, criando tecnologias que ampliam as possibilidades e os desafios computacionais, perfazendo-se um círculo virtuoso. Hoje, existe grande maturidade no que tange a representação de sólidos por computador e a influência de erros na representação numérica [3, 4].

Como aponta Karlostroem [1], no campo de reconstrução de cenas e estimação de posição, a álgebra linear desempenha um papel importante. Com suas ferramentas, desenvolvem-se as entidades básicas e os algoritmos de reconstrução da estrutura espacial [5–7]. Utilizando essas ferramentas, iremos apresentar neste trabalho um sistema capaz de extrair informações relevantes sobre uma cena a partir de imagens dela e de algumas informações sobre seus pontos.

1.1 Definições

Para podermos enunciar o objetivo do projeto e dar início o seu desenvolvimento teórico, precisamos antes definir alguns conceitos importantes. Alguns desses conceitos necessitam do apoio de algum formalismo, que não queremos introduzir antes do capítulo seguinte. Portanto, iremos apresentá-los sem o devido rigor

Figura 1.1: Sem informações sobre a cena, é difícil determinar se este carro tem o tamanho de um carro real ou se é apenas uma miniatura. Essa incerteza é uma *ambiguidade de similaridade*.

matemático, deixando para redefini-los no momento apropriado.

1.1.1 Cena

Uma cena é um conjunto de objetos em um ambiente. Dos componentes de uma cena, deseja-se extrair informações métricas tridimensionais, como distância entre dois pontos, posição exata de cada ponto ou ângulo entre linhas.

Uma cena pode ser tão complexa quanto um escritório ou uma paisagem ou tão simples quanto um chip eletrônico, dependendo apenas da necessidade do usuário do sistema. Este pode estar interessado na representação tridimensional de uma turbina hidrelétrica para simulação usando elementos finitos, caso em que tiraria uma sequência de fotografias da turbina, já que dificilmente conseguiria enquadrar todas as características de seu interesse em uma só foto. Ou, na análise de vídeos filmados por câmeras de segurança, alguém poderia querer saber a distância entre dois elementos, como um veículo e um pedestre. Como dizíamos, o sistema proposto não discerne objetos de ambientes para efetuar a reconstrução da cena, bastando que os pontos de interesse permaneçam estáticos para todas as imagens obtidas.

1.1.2 Reconstrução de Cenas

Efetuar a reconstrução de uma cena significa determinar coordenadas tridimensionais para pontos dela a partir de imagens nas quais figurem esses pontos. O resultado dessa reconstrução é chamado de *nuvem de pontos desestruturada*[8],

"desestruturada" por não conter informações sobre arestas, superfícies, etc.

Quando temos pouca informação, não é possível construir um modelo muito útil da cena. Começamos com um exemplo básico: olhando a Fig. 1.1, é possível dizer se o carrinho mostrado é do tamanho de um carro de verdade ou se é apenas uma miniatura detalhada? Isso ilustra como a utilidade do processo de reconstrução é limitada pelo conhecimento que se tem sobre as imagens e as câmeras. Se soubermos a distância real entre as duas rodas dianteiras do carro, podemos então, como humanos, deduzir com alguma precisão qual o seu tamanho real. Isso porque sabemos que a imagem trata-se de um carro e que, portanto, suas rodas devem estar contidas em planos paralelos, entre outras hipóteses que nos permitem construir um modelo mental útil do objeto, embora a escala desse modelo seja inicialmente indeterminada. Quando sabemos a distância entre as rodas, no entanto, podemos eliminar essa ambiguidade de escala.

O processo de reconstrução por computador funciona de maneira análoga, no que diz respeito à eliminação de ambiguidades com informações externas. Inicialmente, sem informação alguma quanto às câmeras ou quanto à cena, o modelo que podemos construir é completamente inútil para qualquer informação métrica: nem sequer os ângulos entre linhas estão corretos. No entanto, com algumas informações, podemos melhorar esse modelo, até que obtemos uma representação em escala real da cena inicial.

Como neste trabalho desejamos ser independentes das câmeras utilizadas para retratar as imagens, iremos trabalhar apenas com informações sobre a cena, que serão o conhecimento prévio da localização tridimensional de alguns pontos, de acordo com um sistema de coordenadas qualquer. Isso nos liberta de ter que saber, para cada fotografia utilizada, onde exatamente estava posicionada a câmera, quanto *zoom* foi utilizado, etc. Portanto, daqui em diante, não nos referiremos mais a informações sobre as câmeras. Além disso, como, no nosso sistema, tais informações serão fornecidas na forma de coordenadas de pontos no espaço, "ter informações sobre a cena" será sinônimo de "conhecer a localização verdadeira de alguns pontos no espaço".

Hierarquia de Reconstruções e Reconstrução Real Podemos denominar e classificar as possíveis reconstruções de uma cena em uma hierarquia de reconstruções: [9]

1. Reconstrução projetiva;

2. Reconstrução afim;

3. Reconstrução similar;

4. Reconstrução real.

A reconstrução projetiva, a mais geral de todas, é aquela que podemos efetuar sem informação alguma sobre a cena. A reconstrução real é aquela que desejamos: as distâncias entre todos os pontos devem ser exatamente aquelas da cena real, o que determina que ela seja de fato a representação perfeita da cena original.

A reconstrução similar dista da reconstrução real por uma transformação de escala e por uma movimentação (translação e rotação) que a leva ao centro de coordenadas desejado. Já a reconstrução afim dista da real por uma transformação mais complexa, que envolve três rotações, uma transformação de escala anisotrópica (isto é, fatores de escala diferentes para x, y e z) e uma translação. Os nomes das reconstruções refletem o tipo da transformação que as leva à reconstrução real.

1.1.3 Pontos Característicos e Pontos Correspondentes

Pontos característicos são classicamente definidos como pontos de uma imagem que são "discretos, confiáveis e relevantes"[10]. No nosso sistema, pontos característicos serão primariamente os pontos do interesse de quem utilizará a reconstrução, além de pontos auxiliares que porventura sejam necessários para assistir a reconstrução. São esses pontos que, se selecionados corretamente, irão figurar na reconstrução efetuada pelo sistema.

Para servir como substrato ao sistema, um ponto característico deve ser identificado em duas imagens e ser alimentado ao sistema na forma de um par de coordenadas 2D. Chamamos isto de *mapeamento de pontos correspondentes*, e o par de coordenadas é chamado de *correspondência*.

Além disso, como já mencionamos, o sistema precisa da localização tridimensional de alguns pontos. Para esses pontos, essa informação é fornecida junto com a respectiva correspondência, na forma de uma coordenada 3D (de acordo com um sistema de coordenadas qualquer).

A seleção de pontos característicos em uma imagem pode ser feita por algoritmos computacionais. A Transformada de Hough permite determinar segmentos existentes em uma imagem[11], com a intersecção entre os segmentos definindo pontos característicos. No entanto, esse procedimento não foi satisfatório no nosso

caso, pois os pontos característicos selecionados eram normalmente de pouco interesse para a reconstrução. Por isso, decidimos selecionar os pontos manualmente, na interface gráfica de computador que construímos.

1.2 Objetivo

O objetivo deste projeto é descrever e construir um sistema computacional capaz de efetuar a reconstrução real de uma cena, a partir de alguns pontos correspondentes entre duas imagens dessa cena e da localização tridimensional de alguns desses pontos correspondentes. Mostraremos que é imediato utilizar esse sistema com as informações de uma sequência de imagens maior do que apenas duas, desde que cada uma das imagens tenha suficientes correspondências com outra qualquer.

A entrada do sistema será um conjunto de correspondências, na forma descrita na Seção 1.1.3. Na entrada, deve haver:

- pelo menos 8 correspondências cujos pontos verdadeiros não sejam coplanares 4 a 4;
- a localização tridimensional de pelo menos 5 pontos não coplanares 4 a 4.

A saída do sistema será uma lista de coordenadas tridimensionais, referentes aos pontos característicos fornecidos na forma de correspondências.

1.3 Estrutura do Trabalho

O Capítulo 2 é dedicado à determinação da matriz fundamental. Por ser o capítulo introdutório da metodologia, nele também são apresentados os conceitos de geometria projetiva, geometria epipolar e os métodos e questões computacionais que serão abordados.

No Capítulo 3, discutimos a reconstrução real a partir da matriz fundamental. Nesse capítulo apresentamos os conceitos de matriz de câmera, ambiguidade projetiva, objeto de calibração e, ao fim, encerramos a parte de metodologia mostrando como proceder com o método para múltiplas imagens (mais de 2).

A seguir, apresentamos o protótipo construído no Capítulo 4, que se divide em seções sobre a interface gráfica (GUI) e o núcleo do programa, este responsável

pela parte computacional. Enfim, no Capítulo 5 mostramos os resultados obtidos e, no Capítulo 6, tecemos as conclusões.

2 Determinação da Matriz Fundamental

A primeira parte da tarefa consiste em analisar as imagens que temos e extrair relações entre elas. Esse conjunto de relações é denominado *geometria epipolar* entre as imagens, sendo caracterizado pela *matriz fundamental*, como veremos a seguir.

Este capítulo fornece as bases para o estudo da geometria epipolar e meios de determinar a matriz fundamental entre duas imagens.

2.1 Elementos de Geometria Projetiva

A geometria epipolar é o estudo das relações entre diversas imagens de uma mesma cena. Se dispusermos de informação suficiente (isto é, um número mínimo de imagens e correspondências entre elas), podemos tirar conclusões importantes a partir do estudo da geometria epipolar.

A geometria projetiva oferece ferramentas mais adequadas do que a geometria euclidiana para procurar a geometria epipolar. Nesta seção, iremos discutir alguns elementos básicos da Geometria Projetiva, para dar embasamento para as técnicas que utilizaremos mais à frente.

2.1.1 Coordenadas Homogêneas

Em geometria projetiva, costumamos representar pontos e linhas por coordenadas homogêneas. Para pontos, essa representação difere da representação em coordenadas euclidianas — por exemplo, um ponto (x, y) em $\mathbb{R}^2$ pode ser representado por $(x, y, 1)$ em coordenadas homogêneas.

Fator de Escala e Classe de Equivalência O terceiro elemento no exemplo dado se refere a um *fator de escala*, presente em todos os elementos representados

por coordenadas homogêneas, como pontos e linhas (vetores) e transformações lineares (matrizes). Em coordenadas homogêneas, uma entidade com fator de escala não-nulo tem infinitas representações: qualquer multiplicação dessa entidade por escalar não-nulo levará a outra representação da mesma entidade. O conjunto de todas as representações de uma mesma entidade é chamado de *classe de equivalência*.

2.1.2 Pontos em Coordenadas Homogêneas

Quando representados em coordenadas homogêneas, pontos ganham uma dimensão extra. Por exemplo, um ponto (x, y, z) em $\mathbb{R}^3$ pode ser representado, em coordenadas homogêneas, por $(x, y, z, 1)$.

Se o fator de escala for não-nulo, temos que o elemento (a, b, c) em coordenadas homogêneas representa o ponto $(a/c, b/c)$ em $\mathbb{R}^2$. O mapeamento para pontos em $\mathbb{R}^n$ é análogo.

2.1.3 Linhas em Coordenadas Homogêneas

Sabemos que a equação de uma linha em $\mathbb{R}^2$ é a seguinte:

$$ax + by + c = 0 \tag{2.1}$$

Podemos decompor essa equação em um produto escalar:

$$(a, b, c) \cdot (x, y, 1) = 0 \tag{2.2}$$

E, assim, podemos definir a linha (a, b, c) como aquela cujos pontos satisfazem a equação acima.

2.1.4 Pontos Pertencentes a Linhas

A partir da seção acima, é claro que um ponto $\mathbf{p} = (x, y, 1)$ pertence à linha $\mathbf{l} = (a, b, c)$ se, e somente se:

$$\mathbf{p}^{\mathrm{T}} \cdot \mathbf{l} = 0 \tag{2.3}$$

2.1.5 Transformações Lineares

Uma propriedade interessante da representação em coordenadas homogêneas é que, sob ela, diversas operações em pontos e linhas são transformações lineares.

As vantagens disso são muitas e se encontram no fato de que essas operações podem ser representadas por matrizes, e aplicar essas operações a entidades nada mais é do que multiplicar a entidade pela operação desejada.

Se f for uma transformação linear, sabemos da álgebra linear que $kf(x) = f(kx)$. Como, em coordenadas homogêneas, kx é equivalente a x ("Fator de Escala", Seção 2.1.1), temos que $f(kx) = f(x)$, portanto f também é invariante a escala.

Veremos a seguir alguns exemplos de transformações que usaremos no desenvolvimento do nosso projeto.

Translação de Pontos Transladar um ponto de d_x na direção x e d_y na direção y:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x + d_x \\ y + d_y \\ 1 \end{pmatrix} \tag{2.4}$$

Uma matriz que satisfaz essa operação é dada a seguir:

$$\begin{pmatrix} x + d_x \\ y + d_y \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.5}$$

Ampliação de Pontos Ampliar um ponto de k_x na direção x e k_y na direção y:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} k_x x \\ k_y y \\ 1 \end{pmatrix} \tag{2.6}$$

Uma matriz que satisfaz essa operação é dada a seguir:

$$\begin{pmatrix} k_x x \\ k_y y \\ 1 \end{pmatrix} = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.7}$$

Composição de Operações Uma matriz de transformação linear pode realizar uma série de operações de uma vez só. Vejamos um exemplo que será útil a seguir. Imagine que desejamos transladar um ponto de d_x em x e d_y em y, e depois ampliá-lo de k_x em x e k_y em y. A operação seria a seguinte, na ordem

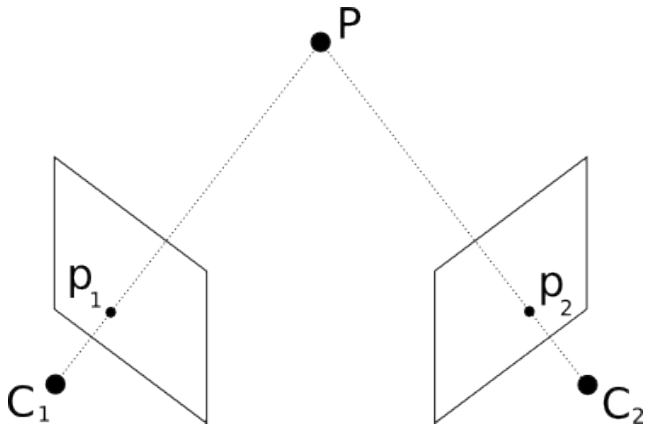

Figura 2.1: Duas imagens de uma mesma cena. $\mathbf{C_1}$ e $\mathbf{C_2}$ são centros de câmera; $\mathbf{p_1}$ e $\mathbf{p_2}$ são as projeções de $\mathbf{P}$ nas imagens.

dada:

$$\begin{pmatrix} k_x(x+d_x) \\ k_y(y+d_y) \\ 1 \end{pmatrix} = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \right) \tag{2.8}$$

Repare na ordem imposta pelos parênteses. Ela não é importante, no entanto, porque a multiplicação matricial é associativa. Assim, podemos realizar primeiro a multiplicação entre as duas matrizes, e ficamos com:

$$\begin{pmatrix} k_x(x+d_x) \\ k_y(y+d_y) \\ 1 \end{pmatrix} = \begin{bmatrix} k_x & 0 & k_x d_x \\ 0 & k_y & k_y d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.9}$$

A matriz acima representa uma composição entre translação e subsequente ampliação, de parâmetros arbitrários (d_x, d_y, k_x, k_y).

2.2 Linha Epipolar, Epipolo e Matriz Fundamental

Nesta seção conheceremos algumas entidades da geometria epipolar que serão essenciais para o desenvolvimento do projeto. Para esta seção, usaremos o modelo de duas câmeras gerando imagens de uma mesma cena, ilustrado na Fig. 2.1. A cena é representada pelo ponto $\mathbf{P}$.

Para entendermos o que são os conceitos acima, façamos um pequeno exercício mental. Imaginemos que nós conheçamos apenas os centros de câmera $\mathbf{C_1}$ e $\mathbf{C_2}$ e a projeção do ponto $\mathbf{P}$ na imagem 1, isto é, $\mathbf{p_1}$ — não conhecemos $\mathbf{P}$! É possível tirar alguma conclusão acerca da posição de $\mathbf{P}$ e de sua projeção na imagem 2,

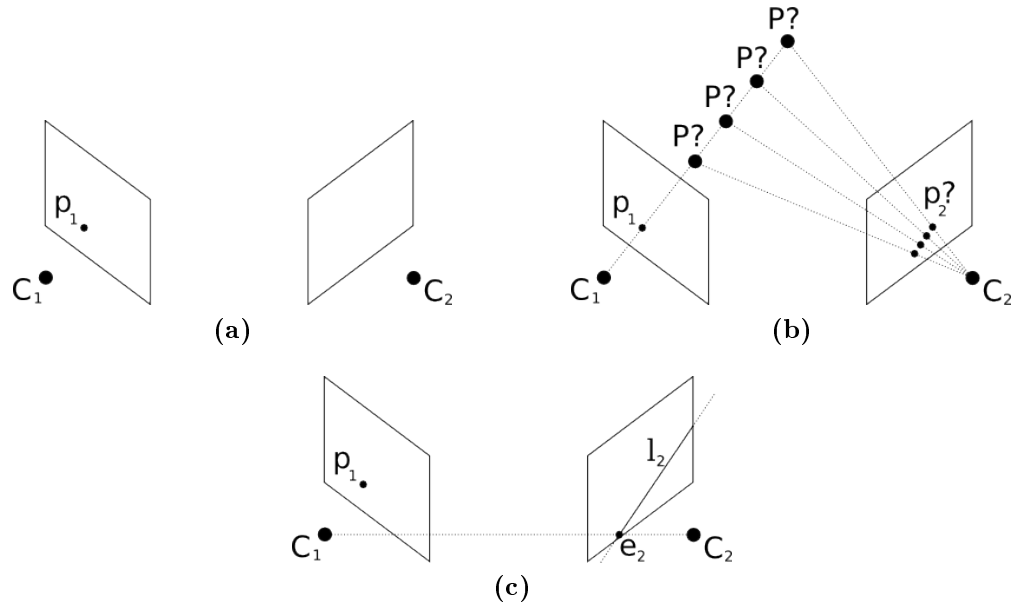

Figura 2.2: Dedução da linha epipolar. (a) dados iniciais: centros de câmera e imagem $\mathbf{p_1}$. (b) extrapolação de $\mathbf{C_1}$-$\mathbf{p_1}$ e projeção dos pontos hipotéticos em 2. (c) linha epipolar l_2, que restringe p_2, e epipolo e_2.

$\mathbf{p_2}$?

Bem, podemos certamente intuir que o ponto $\mathbf{P}$ deve situar-se em algum ponto na reta que passa por $\mathbf{C_1}$ e $\mathbf{p_1}$, naturalmente, por ser $\mathbf{p_1}$ uma projeção. Assim, imaginemos alguns lugares possíveis onde $\mathbf{P}$ pode se situar, como mostrado na Fig. 2.2b. Projeções possíveis desses pontos hipotéticos na imagem 2 nos levam a crer que o procedimento resulta alguma restrição quanto à localização de $\mathbf{p_2}$.

De fato, se projetarmos todos os infinitos pontos hipotéticos **P?** na imagem 2, obtemos o lugar da projeção $\mathbf{p_2}$! Esse lugar geométrico é uma linha e sua propriedade restritiva da localização de $\mathbf{p_2}$ é tão importante que foi batizado de *linha epipolar*.

Podemos também apresentar o *epipolo*, que, embora não tenha utilidade direta aqui, é uma entidade importante da geometria epipolar. O epipolo é o ponto onde todas as linhas epipolares possíveis (geradas a partir de diferentes pontos $\mathbf{p_1}$) se interceptam. O epipolo da imagem 2 se encontra na intersecção da reta que passa por $\mathbf{C_1}$ e $\mathbf{C_2}$ e do plano de imagem 2. Analogamente, o epipolo da imagem 1 se encontra na intersecção dessa reta e do plano de imagem 1.

Eis que surge a *matriz fundamental*. Ela representa a transformação linear que leva um ponto $\mathbf{p_1}$ (na imagem 1) à linha epipolar $\mathbf{l_2}$ (na imagem 2), que restringe a localização de $\mathbf{p_2}$. Assim, sendo F a matriz fundamental, temos:

$$\mathbf{l_2} = \mathrm{F} \cdot \mathbf{p_1} \tag{2.10}$$

Como sabemos que $\mathbf{p_2}$ pertence a $\mathbf{l_2}$, podemos enunciar a importantíssima *condição de correspondência da Matriz Fundamental*:

$$\mathbf{p_2^T} \cdot \mathrm{F} \cdot \mathbf{p_1} = 0 \tag{2.11}$$

2.3 Determinação da Matriz Fundamental

Analisando a matriz fundamental, podemos derivar relações que irão nos ajudar a determiná-la. Primeiramente, esta equação mostra o formato da matriz fundamental:

$$\begin{pmatrix} l_{2,1} \\ l_{2,2} \\ l_{2,3} \end{pmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} \tag{2.12}$$

2.3.1 Restrição de Singularidade

Como a matriz leva vetores de dimensão 3 a vetores de dimensão 3, ela tem dimensão 3x3. No entanto, como a imagem de F tem dimensão 2 (linhas no espaço bidimensional), a matriz F tem posto 2.

Essa propriedade é conhecida como restrição de singularidade e pode ser expressa através da equação:

$$\det \mathrm{F} = 0 \tag{2.13}$$

2.3.2 Restrição de Norma

Como já dissemos, por ser uma transformação linear de coordenadas homogêneas, a matriz F pode ser multiplicada à revelia por valores escalares não-nulos sem que deixe de representar a mesma transformação. Essa propriedade é conhecida como restrição de norma.

2.3.3 Condição de Correspondência

Se conhecermos um par de pontos correspondentes $\mathbf{p_1} = (x_1, y_1, 1)$ e $\mathbf{p_2} = (x_2, y_2, 1)$ entre duas imagens, a condição de correspondência nos dá:

$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = 0 \tag{2.14}$$

Efetuando a multiplicação, temos:

$$x_2x_1f_{11}+x_2y_1f_{12}+x_2f_{13}+y_2x_1f_{21}+y_2y_1f_{22}+y_2f_{23}+x_1f_{31}+y_1f_{32}+f_{33}=0 \quad (2.15)$$

Arranjando as incógnitas (f_{ij}) em um vetor-coluna $\mathbf{f}$, teremos:

$$[\ x_2x_1 \quad x_2y_1 \quad x_2 \quad y_2x_1 \quad y_2y_1 \quad y_2 \quad x_1 \quad y_1 \quad 1\ ]\mathbf{f}=0 \quad (2.16)$$

Assim, temos que o conhecimento de uma correspondência resultou em uma equação cujas incógnitas são os elementos da matriz fundamental.

2.3.4 Algoritmo dos 7 Pontos Correspondentes

Se conhecermos 7 correspondências entre imagens, podemos montar 7 equações com as condições de correspondência. Deve-se tomar cuidado para não utilizar quatro ou mais pontos coplanares, pois isso geraria equações linearmente dependentes entre si.

Assim, com os 7 pares de pontos correspondentes $\mathbf{a} \leftrightarrow \mathbf{b}, \mathbf{c} \leftrightarrow \mathbf{d}, \ldots, \mathbf{m} \leftrightarrow \mathbf{n}$ podemos montar o sistema:

$$\mathrm{A}\cdot\mathbf{f}=\begin{bmatrix} b_1a_1 & b_1a_2 & b_1 & b_2a_1 & b_2a_2 & b_2 & a_1 & a_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n_1m_1 & n_1m_2 & n_1 & n_2m_1 & n_2m_2 & n_2 & m_1 & m_2 & 1 \end{bmatrix}\cdot\mathbf{f}=0 \quad (2.17)$$

A matriz A será 7x9, e o sistema será, naturalmente, indeterminado. Assim, podemos resolvê-lo de modo a encontrar dois vetores $\mathbf{f_1}$ e $\mathbf{f_2}$ dos quais qualquer combinação linear $\mathbf{f}=k_1\mathbf{f_1}+k_2\mathbf{f_2}$ resulte $\mathrm{A}\cdot\mathbf{f}=0$.

Assim, reduzimos o problema de nove incógnitas a apenas duas: k_1 e k_2. No entanto, façamos:

$$\lambda=\frac{k_2}{k_1} \quad (2.18)$$

Então vem:

$$\mathbf{f}=k_1(\mathbf{f_1}+\lambda\mathbf{f_2}) \quad (2.19)$$

Assim, k_1 é meramente um fator de escala, que a restrição de norma nos diz que é irrelevante. Portanto, a busca reduz-se a encontrar o fator de ponderação λ entre as duas soluções.

Resta-nos a restrição de singularidade. Podemos montar a matriz F da se-

guinte maneira:

$$\mathrm{F} = \begin{bmatrix} f_{11}^1 + \lambda f_{11}^2 & f_{12}^1 + \lambda f_{12}^2 & f_{13}^1 + \lambda f_{13}^2 \\ f_{21}^1 + \lambda f_{21}^2 & f_{22}^1 + \lambda f_{22}^2 & f_{23}^1 + \lambda f_{23}^2 \\ f_{31}^1 + \lambda f_{31}^2 & f_{32}^1 + \lambda f_{32}^2 & f_{33}^1 + \lambda f_{33}^2 \end{bmatrix} \tag{2.20}$$

e executar:

$$\det \mathrm{F} = 0 \tag{2.21}$$

Como se trata de uma matriz 3x3, a equação acima nos levaria a um polinômio de terceiro grau em λ. Quaisquer soluções provenientes de raízes desse polinômio devem ser checadas, verificando se resultam em matrizes de posto 2. Isso porque a restrição de singularidade expressa em termos do determinante também envolve soluções de posto 1, e estas devem ser descartadas.

2.3.5 Algoritmo dos 8 Pontos Correspondentes

O Algoritmo dos 7 Pontos Correspondentes é válido para a determinação de matrizes fundamentais. No entanto, a aplicação da restrição de singularidade, com a resolução de um polinômio de terceiro grau e subsequente verificação das soluções é uma desvantagem desse algoritmo. Assim, Longuett-Higgins desenvolveu o Algoritmo dos 8 Pontos Correspondentes [12], que não necessita desses passos extras.

A chave para a melhora do algoritmo é evitar aplicar a restrição de singularidade expressa pelo determinante nulo de F. Isso nos leva, naturalmente, a ficarmos desfalcados de uma equação para determinar completamente a matriz fundamental. Compensamos esse desfalque com o uso de uma ou mais correspondências extras.

Deve-se notar que o uso de pares de pontos correspondentes a mais pode levar o sistema à sobredeterminação. Nesse caso, nós deixamos de procurar $\mathbf{f}$ tal que $\mathrm{A}\cdot\mathbf{f} = 0$, e passamos a procurar $\mathbf{f}$ que minimize a norma de $\mathrm{A}\cdot\mathbf{f}$, segundo alguma definição. No entanto, como veremos a seguir, isso não irá alterar o procedimento computacional de solução do sistema.

Outra questão importante neste algoritmo é que, por não aplicarmos a restrição de singularidade, normalmente acabaremos com uma solução cujo posto é 3. Nesse caso, precisamos fazer uma aproximação da matriz de posto 3 encontrada por uma matriz de posto 2 que satisfaça algum critério de similaridade com a matriz original. Veremos também a seguir como proceder tal manobra.

Tabela 2.1: Representação numérica do SVD da Equação (2.23)

Valor singular	Vetor singular à esquerda	Vetor singular à direita
9,51	(-0,39; -0,92)	(-0,43; -0,57; -0,70)
0,77	(-0,92; -0,39)	(0,81; 0,11; -0,58)
0	–	(-0,41; 0,82; -0,41)

2.4 Decomposição em Valores Singulares (SVD)

Em álgebra linear, a decomposição em valores singulares (do inglês *Singular Value Decomposition*) estende para matrizes o conceito de fatoração. Pode-se demonstrar, por ele, que qualquer matriz composta por valores complexos pode ser reduzida à multiplicação de três matrizes, que detêm propriedades úteis.

No domínio das matrizes de números reais, uma aplicação do SVD resultaria no seguinte:

$$\mathrm{A_{mn}} = \mathrm{U_{mm}} \cdot \mathrm{D_{mn}} \cdot \mathrm{V_{nn}^{T}} \tag{2.22}$$

A matriz D é sempre uma diagonal de valores não negativos chamados *valores singulares*. A cada valor singular corresponde um *vetor singular à esquerda* e um *vetor singular à direita*, que são, respectivamente, os vetores formados pelas colunas correspondentes de U e de V. Para ilustrar, considere o exemplo a seguir.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} -0{,}39 & -0{,}92 \\ -0{,}92 & -0{,}39 \end{bmatrix} \begin{bmatrix} 9{,}51 & 0 & 0 \\ 0 & 0{,}77 & 0 \end{bmatrix} \begin{bmatrix} -0{,}43 & 0{,}81 & -0{,}41 \\ -0{,}57 & 0{,}11 & 0{,}82 \\ -0{,}70 & -0{,}58 & -0{,}41 \end{bmatrix}^{\mathrm{T}} \tag{2.23}$$

Como podemos ver na Tabela 2.1, embora não haja um valor singular 0 na matriz D, consideramos que o elemento que falta para que a matriz D seja diagonal seja esse valor 0. Assim, podemos associar a esse valor os vetores singulares que aparentemente não teriam associação (como a última coluna de V, por exemplo).

2.4.1 Soluções Não-Triviais de Equações Homogêneas

O SVD é uma grande ferramenta computacional para resolver equações homogêneas, quando não se deseja a solução trivial. Pode-se mostrar que, se uma matriz A gera um sistema linear indeterminado, seu SVD terá um número de valores singulares nulos igual ao grau de indeterminação do sistema (incluindo-se os valores singulares que não aparecem na matriz, como no exemplo dado), e cada

vetor singular à direita $\mathbf{x}$ associado a esses valores singulares satisfará a equação $\mathrm{A} \cdot \mathbf{x} = 0$.

Por outro lado, se uma matriz A gerar um sistema linear determinado ou sobredeterminado, pode-se mostrar que o vetor singular $\mathbf{x}$ à direita do menor valor singular será aquele que minimiza a norma de $\mathrm{A}\mathbf{x}$, segundo a norma de Frobenius (raiz da soma dos quadrados dos elementos).

2.4.2 Redução de Posto de Matrizes

O SVD oferece também uma maneira de se aproximar uma matriz de posto n por uma matriz de posto $m < n$. O procedimento consiste simplesmente em aplicar o SVD em uma matriz A, zerar os $n - m$ menores valores singulares e multiplicar novamente as matrizes da decomposição, obtendo-se uma matriz $\tilde{\mathrm{A}}$. Essa matriz é a aproximação de posto reduzido que minimiza a norma de $\mathrm{A} - \tilde{\mathrm{A}}$, novamente de acordo com a norma de Frobenius.

2.5 Questão Computacional: Mau Condicionamento

Um grande problema dos algoritmos propostos para se determinar a matriz fundamental é que eles gerarão sistemas lineares malcondicionados. A seguir, faremos uma breve introdução sobre sistemas lineares malcondicionados e, a seguir, analisaremos por que o nosso sistema será malcondicionado e como resolver esse problema.

2.5.1 Condicionamento de um Sistema Linear

Examinemos o seguinte sistema de equações lineares:[13]

$$x + y = 1 \tag{2.24}$$

$$x + 1{,}01y = 2 \tag{2.25}$$

Podemos resolver facilmente o problema e obter a solução (-99, 100). No entanto, suponhamos que, por uma questão de arredondamento, chegássemos ao seguinte sistema, ao invés de ao anterior:

$$x + y = 1 \tag{2.26}$$

$$x + 1{,}02y = 2 \tag{2.27}$$

A solução para este sistema é (-49, 50). Como se vê, um erro de aproximadamente 1% em um dos coeficientes resultou em um erro de 50% nos valores encontrados. Chamamos sistemas anomalamente sensíveis como esse de *sistemas malcondicionados*. Tais sistemas apresentam um problema considerável para a resolução numérica, pois, como visto, arredondamentos corriqueiros podem levar a erros grotescos.

Uma das raízes do mau condicionamento reside na pequena variação dos valores dos coeficientes relativamente a seus valores absolutos. Por isso mesmo, muitas vezes podem ser encontradas matrizes equivalentes à original que são bem-condicionadas. Para o exemplo dado, podemos combinar linearmente as equações[1] de modo a encontrar:

$$3x + 4y = 103 \tag{2.28}$$

$$4x - 3y = -696 \tag{2.29}$$

Nesse sistema, um erro de mesma ordem no mesmo coeficiente gera um erro máximo de 0,48% na solução.

2.5.2 Normalização

Como vimos anteriormente, sistemas lineares que apresentam pouca variação de coeficientes face a seus valores absolutos são candidatos ao mau condicionamento. Essa situação surge com frequência no campo de visão computacional. Basta imaginar que, hoje em dia, câmeras fotográficas comerciais têm resoluções na casa dos milhares de pontos em cada eixo e, frequentemente, a ordem de grandeza das diferenças de coordenadas entre dois pontos correspondentes se encontra bem abaixo de 10^3. Assim, não raro resultam sistemas lineares com coeficientes muito semelhantes e, nesse caso, ligeiras aproximações numéricas podem comprometer seriamente os resultados.

Uma maneira de se resolver esse problema, no entanto, é aplicar a normalização dos pontos correspondentes, sugerida por Hartley [14] sobre o algoritmo de Longuett-Higgins [12].

Trata-se de se transladar e escalonar as imagens de maneira a se minimizar o efeito deletério das coordenadas distantes e semelhantes. A normalização consiste em mudar as origens das coordenadas de cada uma das imagens, de modo que, em cada uma, a origem se posicione no centro geométrico dos pontos característicos

[1] $-97 \times (2.24) + 100 \times (2.25)$; $704 \times (2.24) - 700 \times (2.25)$

individuais. Então, aplica-se um fator de escala nos pontos, de maneira a tornar a distância média deles em relação à origem igual a $\sqrt{2}$.

2.5.3 Matrizes de Normalização

Para cada imagem, portanto, construiremos uma matriz de normalização e a aplicaremos em seus pontos característicos, de maneira que:

$$\hat{\mathbf{p}} = \mathrm{T} \cdot \mathbf{p} \tag{2.30}$$

A matriz de normalização dos pontos da imagem 1 será T_1 e, da imagem 2, T_2.

Como dissemos, a matriz de normalização translada os pontos de modo a neutralizar a distância entre o centroide dos pontos e a origem do sistema cartesiano. Depois disso, ela amplia os pontos de modo que a distância média dos pontos à origem seja $\sqrt{2}$.

Resgatando a transformação linear que combina translação e ampliação da Equação (2.9), temos:

$$\begin{pmatrix} k_x(x+d_x) \\ k_y(y+d_y) \\ 1 \end{pmatrix} = \begin{bmatrix} k_x & 0 & k_x d_x \\ 0 & k_y & k_y d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.31}$$

No nosso caso, temos que d_x e d_y são negativos e iguais em módulo às coordenadas em x e y do centroide do conjunto de pontos característicos. Já k_x e k_y são iguais, e de valor $\sqrt{2}$ dividido pela distância média dos pontos ao centroide. Assim, temos:

$$d_x = -\frac{1}{n}\sum_i^n x_i \tag{2.32}$$

$$d_y = -\frac{1}{n}\sum_i^n y_i \tag{2.33}$$

$$k_x = \frac{\sqrt{2}}{\frac{1}{n}\sum_i^n \sqrt{(x_i+d_x)^2 + (y_i+d_y)^2}} \tag{2.34}$$

$$k_y = k_x \tag{2.35}$$

E, assim, montamos as matrizes de normalização.

2.6 Algoritmo dos 8 Pontos Correspondentes Normalizado

Quando aplicamos a normalização antes de montar o sistema linear para determinar a matriz fundamental, no final do processo temos uma matriz fundamental que não serve para os pontos não-normalizados, mas apenas para os pontos normalizados.

Assim, deixa-se de satisfazer a condição de correspondência original para se satisfazer:

$$\hat{\mathbf{p}}_\mathbf{2}^\mathbf{T} \cdot \hat{\mathrm{F}} \cdot \hat{\mathbf{p}}_\mathbf{1} = 0 \tag{2.36}$$

O que procuramos, naturalmente, é uma matriz F que satisfaça a Equação (2.11):

$$\mathbf{p}_\mathbf{2}^\mathbf{T} \cdot \mathrm{F} \cdot \mathbf{p}_\mathbf{1} = 0 \tag{2.37}$$

Bem, sabemos que:

$$\hat{\mathbf{p}_\mathbf{1}} = \mathrm{T}_1 \cdot \mathbf{p}_\mathbf{1} \tag{2.38}$$

$$\hat{\mathbf{p}_\mathbf{2}} = \mathrm{T}_2 \cdot \mathbf{p}_\mathbf{2} \tag{2.39}$$

Então:

$$(\mathrm{T}_2 \cdot \mathbf{p}_\mathbf{2})^\mathrm{T} \cdot \hat{\mathrm{F}} \cdot \mathrm{T}_1 \cdot \mathbf{p}_\mathbf{1} = 0 \Leftrightarrow \mathbf{p}_\mathbf{2}^\mathbf{T} \cdot \mathrm{T}_2^\mathrm{T} \cdot \hat{\mathrm{F}} \cdot \mathrm{T}_1 \cdot \mathbf{p}_\mathbf{1} = 0 \Leftrightarrow \mathrm{F} = \mathrm{T}_2^\mathrm{T} \cdot \hat{\mathrm{F}} \cdot \mathrm{T}_1 \tag{2.40}$$

A isso chamamos de denormalização da matriz fundamental.

3 Reconstrução Real

Após encontrada a matriz fundamental entre duas imagens, deve-se utilizá-la, junto com as informações externas que se têm, para determinar a *reconstrução real* da cena. A reconstrução da cena é caracterizada pelas suas *matrizes de câmera*, com as quais pode-se determinar a localização tridimensional de pontos característicos (e correspondentes) das imagens.

Este capítulo mostra como efetuar a reconstrução real a partir da matriz fundamental e do conhecimento da localização tridimensional de 5 pontos correspondentes entre duas imagens.

3.1 Matriz de Câmera

Uma imagem de uma cena é um conjunto de pontos bidimensionais obtidos através da projeção de pontos tridimensionais em um plano. Em coordenadas homogêneas, o que temos é:

$$\mathbf{x} = \mathrm{P} \cdot \mathbf{X} \tag{3.1}$$

$\mathbf{X}$ é um vetor homogêneo que representa um ponto no espaço, e $\mathbf{x}$ é sua projeção no plano. Assim, P leva um vetor de quatro dimensões a um de três dimensões e, portanto, é uma matriz 3x4 e tem posto 3. P é a *matriz de câmera*. Sua operação é projetar pontos tridimensionais $\mathbf{X}$ no plano de imagem.

3.1.1 Parâmetros da Matriz de Câmera

Não será preciso entrar em detalhes sobre os elementos da matriz de câmera, mas é pertinente apontar que eles são os seguintes:

- distância focal f
- resolução em x e em y
- inclinação do plano de imagem s

- posição do ponto principal $(x_0, y_0)^{\mathrm{T}}$
- posição absoluta do centro de câmera $(X_0, Y_0, Z_0)^{\mathrm{T}}$
- rotação $\mathbf{R}$ da câmera em relação ao sistema de coordenadas absoluto

Os dois últimos itens dizem respeito à posição absoluta da câmera e, por isso, são chamados de *parâmetros externos* ou *extrínsecos*, tendo, naturalmente, 6 graus de liberdade. Os outros são *parâmetros intrínsecos* da câmera e totalizam 5 graus de liberdade (entre a distância focal, resolução e posição do ponto principal há apenas 4 graus de liberdade).

Duas fotografias diferentes, mesmo se tiradas com a mesma máquina fotográfica, podem ter (e normalmente terão) matrizes de câmera distintas. Em primeiro lugar, os parâmetros extrínsecos normalmente serão diferentes, pois a máquina provavelmente terá transladado ou rotacionado para pegar um ângulo diferente. Quanto aos parâmetros intrínsecos, a distância focal pode ter se alterado, ou mesmo a resolução.

Ou pode ser mesmo o caso de a segunda câmera ser uma máquina diferente da primeira. Como o objetivo deste trabalho é obter um sistema independente das câmeras utilizadas, trabalharemos sempre com duas matrizes de câmera genéricas, P_1 e P_2, para um par de imagens.

3.2 Triangulação de Matrizes de Câmera

O objetivo deste projeto pode ser traduzido em obter a coordenada absoluta de pontos a partir de suas imagens. Embora isso não seja possível diretamente a partir da matriz de câmera — já que ela não é inversível, nem faria sentido que fosse —, ela é extremamente útil na tarefa, pois nos dará uma linha no espaço tridimensional onde o ponto original deve estar.

Sabemos que um ponto $\mathbf{X}$, para ser projetado em $\mathbf{x}$, deve obedecer à Equação (3.1). Se tivermos $\mathbf{X} = \mathrm{P_p}\mathbf{x}$, com $\mathbf{P_p}$ sendo a pseudoinversa de P (tal que $\mathrm{PP_p} = \mathrm{I}$), teremos:

$$\mathrm{P} \cdot \mathbf{X} = \mathrm{P} \cdot \mathrm{P_p} \cdot \mathbf{x} = \mathrm{I} \cdot \mathbf{x} = \mathbf{x} \tag{3.2}$$

Repare que $\mathrm{P_p}$ *não é uma inversa*, pois P não é uma matriz quadrada. Também, I é uma matriz de dimensão 3, não de dimensão 4. Uma das expressões possíveis para $\mathrm{P_p}$ é:

$$\mathrm{P} \cdot \mathrm{P}^{\mathrm{T}} \cdot \left(\mathrm{P} \cdot \mathrm{P}^{\mathrm{T}}\right)^{-1} = \mathrm{I_{3x3}} \Rightarrow \mathrm{P_p} = \mathrm{P}^{\mathrm{T}} \cdot \left(\mathrm{P} \cdot \mathrm{P}^{\mathrm{T}}\right)^{-1} \tag{3.3}$$

Assim, temos que $\mathrm{P_p}\cdot\mathbf{x}$ é um dos pontos da linha que restringe $\mathbf{X}$. Outro ponto dessa linha certamente é o centro de câmera, $\mathbf{C}$. Se encontrarmos duas dessas linhas (uma em cada imagem), podemos encontrar o ponto $\mathbf{X}$ na intersecção entre elas. Isso é chamado de triangulação.

No entanto, o método utilizado não será exatamente esse, por ser esta uma abordagem idealizada. Na prática, as matrizes de câmera não serão determinadas com exatidão, o que faria com que as linhas obtidas não se interceptassem em nenhum ponto. Nosso procedimento será obter o ponto que melhor aproxime a intersecção entre linhas.

Sendo P qualquer uma das câmeras conhecidas $\mathrm{P_1}$ e $\mathrm{P_2}$, temos que $\mathrm{P}\cdot\mathbf{X}$ sempre pertencerá à classe de equivalência de $\mathbf{x}$, ou seja, $\mathrm{P}\cdot\mathbf{X} = \lambda\mathbf{x}$. Por isso, podemos fazer $\mathrm{P}\mathbf{X}\times\mathbf{x} = 0$.

Denotando as linhas de cada P da seguinte maneira:

$$\mathrm{P_i} = \begin{bmatrix} \mathbf{p_1^i} \\ \mathbf{p_2^i} \\ \mathbf{p_3^i} \end{bmatrix} \tag{3.4}$$

Temos do produto vetorial o seguinte conjunto de equações, para dado $\mathbf{x_i} = (x_i, y_i, 1)$:

$$x_i\left(\mathbf{p_3^i}\cdot\mathbf{X}\right) - \left(\mathbf{p_1^i}\cdot\mathbf{X}\right) = 0 \tag{3.5}$$

$$y_i\left(\mathbf{p_3^i}\cdot\mathbf{X}\right) - \left(\mathbf{p_2^i}\cdot\mathbf{X}\right) = 0 \tag{3.6}$$

$$x_i\left(\mathbf{p_2^i}\cdot\mathbf{X}\right) - y_i\left(\mathbf{p_1^i}\cdot\mathbf{X}\right) = 0 \tag{3.7}$$

Nota-se que a terceira equação é combinação linear das duas primeiras.

Fazendo a operação para ambas as imagens, temos:

$$\mathrm{A}\mathbf{x} = \begin{bmatrix} x_1\mathbf{p_3^1} - \mathbf{p_1^1} \\ y_1\mathbf{p_3^1} - \mathbf{p_2^1} \\ x_2\mathbf{p_3^2} - \mathbf{p_1^2} \\ y_2\mathbf{p_3^2} - \mathbf{p_2^2} \end{bmatrix}\mathbf{X} = 0 \tag{3.8}$$

Deve-se notar duas coisas sobre o sistema linear acima:

1. A matriz A tem dimensão 4x4, e não 4x1, como pode aparentar.

2. Como $\mathbf{X}$ deve ser determinado a menos de um fator de escala, o sistema linear 4x4 pode ser sobredeterminado. Idealmente, A teria posto 3, mas

isso não acontecerá com matrizes obtidas a partir de dados reais. Na SVD de A, a melhor solução se encontra no vetor singular à direita do menor valor singular, que, idealmente, seria zero.

3.3 Relação entre Matrizes de Câmera e Matriz Fundamental

Como já vimos, a matriz fundamental relaciona um par de imagens de uma mesma cena. Naturalmente, embora a matriz fundamental possa ser determinada a partir de correspondências entre pontos, ela não é função dos pares de pontos que escolhermos para determiná-la, isto é: se fossem outros objetos na frente das câmeras, ainda assim a matriz fundamental seria a mesma.

Em outras palavras, a matriz fundamental é função apenas das duas imagens em questão, que são determinadas pelas matrizes de câmera e pelos pontos tridimensionais projetados. Mas, como acabamos de dizer, os eventuais pontos projetados não têm influência na matriz fundamental. Segue que esta é função apenas do par de matrizes de câmera usado.

Assim, temos que um par de matrizes de câmera determina unicamente uma matriz fundamental. Como já aprendemos a determinar a matriz fundamental de um par de imagens, isso é uma boa notícia, pois iremos utilizá-la para determinar as matrizes de câmera.

3.4 Ambiguidade Projetiva de Câmeras a partir da Matriz Fundamental

Embora um par de matrizes de câmera determine uma matriz fundamental, essa relação não é bijetora: uma matriz fundamental não determina um par de matrizes de câmera. Na verdade, qualquer par de matrizes de câmera "separados por" uma transformação projetiva determina a mesma matriz fundamental.

Imagine que pontos $\mathbf{X_i}$ sejam projetados em imagens por matrizes de câmera P_1 e P_2. Agora, olhe para os pontos $\mathrm{H} \cdot \mathbf{X_i}$, sendo H uma transformação projetiva qualquer. Se olharmos as imagens deles segundo as câmeras $\mathrm{P}_1 \cdot \mathrm{H}^{-1}$ e $\mathrm{P}_2 \cdot \mathrm{H}^{-1}$, respectivamente, teremos as mesmas imagens dos pontos iniciais, pois:

$$\left(\mathrm{P}_1 \cdot \mathrm{H}^{-1}\right) \cdot (\mathrm{H} \cdot \mathbf{X_i}) = \mathrm{P}_1 \cdot \mathbf{X_i} \tag{3.9}$$

e idem para a matriz P_2. Assim, teremos as mesmas correspondências e, portanto, a mesma matriz fundamental.

Portanto, há mais de uma reconstrução (isto é, um par de matrizes de câmera e um conjunto de pontos tridimensionais) obtenível a partir de uma matriz fundamental, e essa ambiguidade só pode ser resolvida tendo-se informações sobre a cena ou sobre as câmeras. Tomaremos como lema que, dados uma matriz fundamental F e o epipolo $\mathbf{e_2}$ no contradomínio de F, um par de matrizes de câmera é:

$$P_1 = [I_{3x3}|\mathbf{0}] \tag{3.10}$$

$$P_2 = [[\mathbf{e_2}]_\times F|\mathbf{e_2}] \tag{3.11}$$

onde $[\mathbf{e_2}]_\times$ denota a matriz antissimétrica para produto vetorial a partir de $\mathbf{e_2}$. Esta matriz é construída da seguinte maneira, para um vetor qualquer $\mathbf{v} = (v_1, v_2, v_3)$ é:

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{3.12}$$

Tais P_1 e P_2 são chamadas de *câmeras canônicas* para dado F, e essas câmeras serão utilizadas para efetuar a triangulação.

3.5 Reconstrução Real a partir de Objeto de Calibração

Ao encontrar uma projetividade H que leve os pontos estimados a partir das câmeras canônicas a pontos em uma estrutura euclidiana na escala correta, estamos extinguindo a ambiguidade projetiva e criando uma *reconstrução real*, como já definido na Seção 1.1.2: um modelo do qual podem-se extrair medidas fielmente.

Uma das maneiras de se determinar tal H é conhecendo de antemão as posições absolutas de alguns pontos fotografados. Para termos tal conhecimento, podemos inserir na cena fotografada algum objeto de dimensões conhecidas e medir as posições de pontos característicos dele em relação a algum sistema de coordenadas conveniente. Esse sistema de coordenadas acabará sendo o sistema absoluto para toda a reconstrução.

Conhecendo a posição verdadeira no espaço (x, y, z) de algum ponto, inicial-

mente em suas coordenadas projetivas $\mathbf{X}$, e tendo as linhas de H como:

$$\mathrm{H} = \begin{bmatrix} \mathbf{h_1} \\ \mathbf{h_2} \\ \mathbf{h_3} \\ \mathbf{h_4} \end{bmatrix} \tag{3.13}$$

podemos escrever (em coordenadas não-homogêneas):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{\mathbf{h_4} \cdot \mathbf{X}} \begin{pmatrix} \mathbf{h_1} \cdot \mathbf{X} \\ \mathbf{h_2} \cdot \mathbf{X} \\ \mathbf{h_3} \cdot \mathbf{X} \end{pmatrix} \Rightarrow \begin{pmatrix} x\mathbf{h_4} \cdot \mathbf{X} - \mathbf{h_1} \cdot \mathbf{X} \\ y\mathbf{h_4} \cdot \mathbf{X} - \mathbf{h_2} \cdot \mathbf{X} \\ z\mathbf{h_4} \cdot \mathbf{X} - \mathbf{h_3} \cdot \mathbf{X} \end{pmatrix} = \mathbf{0} \tag{3.14}$$

e, assim, reorganizar em um sistema:

$$\begin{bmatrix} -\mathbf{X^T} & \mathbf{0^T} & \mathbf{0^T} & x\mathbf{X^T} \\ \mathbf{0^T} & -\mathbf{X^T} & \mathbf{0^T} & y\mathbf{X^T} \\ \mathbf{0^T} & \mathbf{0^T} & -\mathbf{X^T} & z\mathbf{X^T} \end{bmatrix} \begin{pmatrix} \mathbf{h_1^T} \\ \mathbf{h_2^T} \\ \mathbf{h_3^T} \\ \mathbf{h_4^T} \end{pmatrix} = \mathbf{0} \tag{3.15}$$

Repare que o sistema tem dimensão 3x16, e o vetor $\mathbf{h}$ tem 16 linhas, que são os elementos de H. Como H deve ser determinado a menos de um fator de escala, bastam 5 pontos conhecidos para determinar a reconstrução métrica.

3.6 Reconstrução a partir de Múltiplas Imagens

Como mostramos, o sistema descrito trabalha com imagens duas a duas, ou seja, uma sequência de imagens de uma cena não é utilizada toda de uma vez só. No entanto, muitas vezes desejaremos utilizar múltiplas imagens de uma cena, pois, por exemplo, pode ser impossível capturar, com apenas duas imagens, todos os detalhes relevantes de uma cena.

A solução para esse problema é simples. Dado que uma reconstrução a partir de um par de imagens já foi feita, a chave é alimentar os pontos conhecidos no próximo par de imagens utilizado. Por isso, é interessante que o próximo par de imagens tenha pelo menos 5 pontos característicos já reconstruídos, pois isso torna desnecessário o uso de objeto de calibração nesse par.

4 Protótipo

O protótipo construído foi dividido em duas partes: núcleo e interface gráfica (GUI). A GUI é responsável por coletar as informações necessárias para o sistema: coordenadas de pontos correspondentes e pontos tridimensionais conhecidos. O núcleo é responsável por fazer, a partir dos dados fornecidos pela GUI, os cálculos necessários para a reconstrução real: determinação da matriz fundamental e a reconstrução a partir dela.

Como se vê, faz-se necessário um meio de comunicação entre a GUI e o núcleo, para trocar dados como coordenadas de pontos característicos e a matriz fundamental calculada. Essa comunicação se dá por arquivos de texto formatados com um padrão preestabelecido.

4.1 Interface Gráfica (GUI)

A interface gráfica foi desenvolvida em Visual C#, utilizando o Microsoft Visual Studio 2010 (.NET Framework 4.0). A função básica que ela atende é possibilitar a seleção manual de pontos característicos e correspondências entre imagens, utilizando o mouse.

Inicialmente, é necessário carregar o par de imagens, o que se faz utilizando o

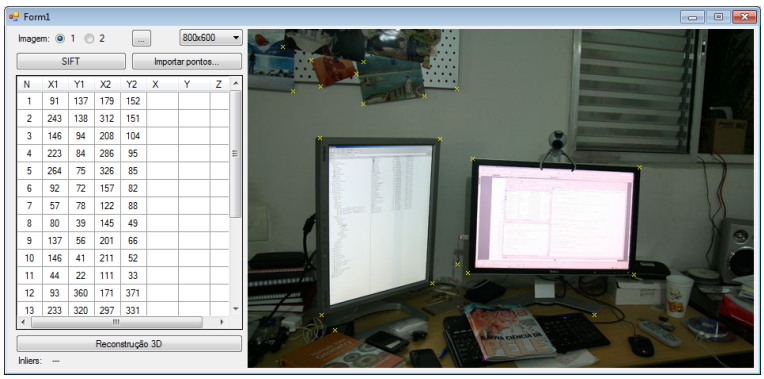

Figura 4.1: Seleção manual de pontos na GUI. Quando se clica em um ponto da imagem, a respectiva coordenada é inserida na tabela à esquerda, e a outra imagem do par é exibida para que o usuário indique a correspondência.

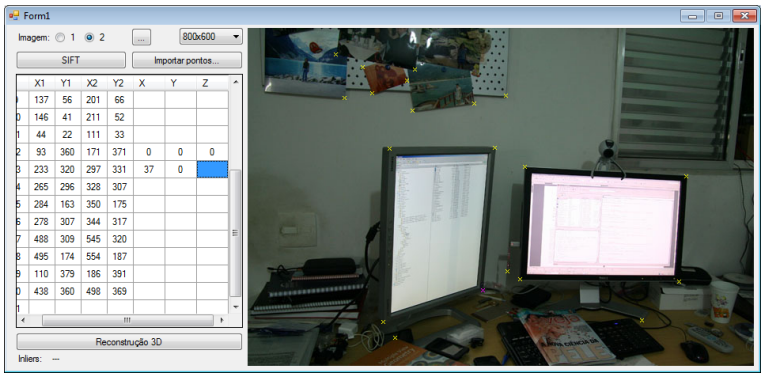

Figura 4.2: Inserção de coordenadas 3D na GUI.

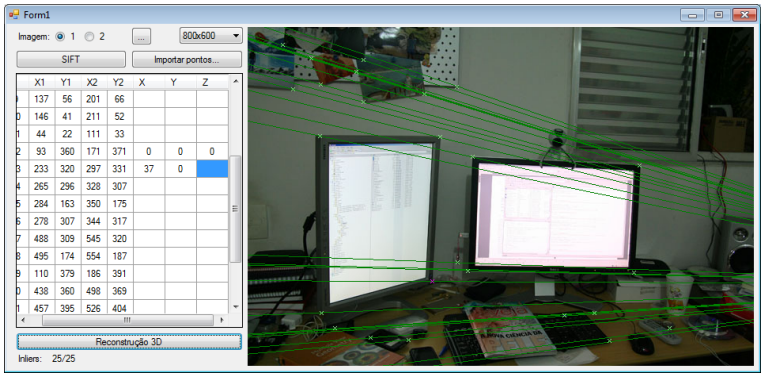

Figura 4.3: Linhas epipolares na GUI. Se houvesse algum ponto pelo qual não passasse linha epipolar, alguma correspondência estaria incorreta.

botão "...", que se encontra na região superior esquerda da Fig. 4.1. Então pode-se proceder à seleção de pontos característicos e suas correspondências. Para fazê-lo, o usuário seleciona o ponto em uma imagem e, em seguida, a interface exibirá a outra imagem do par, para que o ponto correspondente ao escolhido seja também selecionado. Se necessário, o usuário tem à disposição a funcionalidade *zoom* em regiões da imagem, utilizando o botão direito para zoom in e o do meio para zoom out.

Para informar as coordenadas 3D de pontos conhecidos, o usuário entra diretamente com os dados na tabela de pontos à esquerda, como ilustrado na Fig. 4.2.

Após todos os pontos terem sido selecionados e as coordenadas tridimensionais conhecidas terem sido informadas, clica-se no botão "Reconstrução 3D", que repassa os dados ao núcleo e lê de volta a saída gerada. A partir dessa leitura, a GUI mostra as linhas epipolares determinadas pela matriz fundamental (vide Fig. 4.3), o que é útil para se averiguar a exatidão da matriz determinada (se houver algum ponto pelo qual não passe nenhuma linha epipolar, as correspondências estão imperfeitas).

4.2 Núcleo

O núcleo do protótipo foi escrito em Visual C++, utilizando bibliotecas do livro Numerical Recipes [15] para SVD. Como já foi dito, o programa se comunica com a GUI por arquivos de texto, com formatação preestabelecida. Mais especificamente, o núcleo aceita um arquivo no qual os pontos correspondentes são linhas no seguinte formato:

```
x1 y1 x2 y2 X Y Z
```

onde os primeiros são as coordenadas do ponto característico em cada imagem, e X, Y e Z são dados apenas se a localização tridimensional for conhecida.

O núcleo emite duas saídas: uma para a GUI, com as linhas epipolares, e outra com apenas as coordenadas tridimensionais da reconstrução efetuada. A primeira saída tem as linhas no seguinte formato:

```
x1 y1 l1 l2 l3 x2 y2 l4 l5 l6
```

onde os l_i representam uma linha epipolar segundo a equação $l_1x + l_2y + l_3 = 0$ na primeira imagem (idem para os outros l_i, na segunda imagem). A segunda saída (da reconstrução em si) tem como linhas simplesmente:

```
X Y Z
```

representando pontos tridimensionais determinados pela reconstrução.

Ao ser chamado, o núcleo lê o arquivo de entrada e cria um vetor de objetos `Ponto`, com quantos forem as linhas do arquivo de entrada. Esse objeto contém dois objetos `Ponto2D`, que descrevem o par de pontos correspondentes, e dois objetos `Ponto3D`, que guardam o resultado da reconstrução projetiva e da reconstrução real. Nos pontos cuja coordenada tridimensional é conhecida, o objeto `Ponto3D` relativo à reconstrução real já é preenchido.

O primeiro passo do programa é chamar o algoritmo dos 8 pontos correspondentes, na função `oito_pontos()`. Essa função faz a normalização dos pontos correspondentes, monta o sistema linear a partir das condições de correspondência, resolve o sistema por SVD, diminui o posto da solução para 2 (novamente por SVD) e então denormaliza a solução com as transpostas das matrizes de normalização, que foram guardadas.

Nesse ponto, o programa pode emitir as linhas epipolares. Tendo a matriz fundamental, as linhas epipolares da imagem 2 são encontradas pela multiplicação dos pontos característicos da imagem 1 pela própria matriz, e, as linhas da imagem 1, pela multiplicação dos pontos da imagem 2 pela transposta.

Então o programa segue para a reconstrução real. Primeiro, ele monta as matrizes de câmera canônicas, com a função `P_canonicas()`. Em seguida, efetua a triangulação dos pontos, na função `triangulacao()`. Essa função preenche um dos `Ponto3D` de cada `Ponto` com o resultado da triangulação, que é a reconstrução projetiva. Enfim, o programa segue para a função `reconstrucao()`, que encontra a matriz de transformação H a partir dos pontos tridimensionais já conhecidos, para então determinar a reconstrução real a partir da multiplicação da reconstrução projetiva de cada ponto por H.

Finalmente, o programa emite as coordenadas da reconstrução real de cada ponto e encerra.

5 Resultados

Fizemos um teste de reconstrução de uma cena com vários objetos. Na Fig. 5.1, pode-se ver o par de imagens utilizado e os pontos característicos selecionados. No caso, os pontos conhecidos, que serviram de objeto de calibração, foram os quatro do laptop e o da tampa vermelha.

Na Fig. 5.2, vemos as linhas epipolares, encontradas a partir da matriz fundamental determinada pelo núcleo. Após fazer a reconstrução, com um programa simples de edição de imagens colorimos alguns pontos de destaque e plotamos no Scilab os pontos tridimensionais resultantes da reconstrução, obtendo os resultados apresentados na Fig. 5.3. É interessante notar que os gráficos apresentam eixos em escala real, podendo ser utilizados para tirar medidas verdadeiras da cena.

Embora o processo de seleção manual dos pontos tenha uma incerteza inerente, devida à quase sempre imperfeita identificação do ponto característico na imagem — às vezes é difícil tanto identificar visualmente o ponto característico quanto eleger o ponto exato para selecioná-lo com o mouse —, verificamos que o resultado não sofreu sérios prejuízos por causa dessa imprecisão. Outra demonstração de robustez foi a insensibilidade ao erro da estimativa de posição do ponto característico da tampa vermelha, que, por não ser solidária ao laptop, teve sua posição medida sem que se encostasse na mesma.

Figura 5.1: Par de imagens utilizado no primeiro teste e seus respectivos pontos característicos.

Figura 5.2: Linhas epipolares no par de imagens utilizado no primeiro teste. Nota-se que, em ambas as imagens, o epipolo encontra-se fora da região visível da imagem.

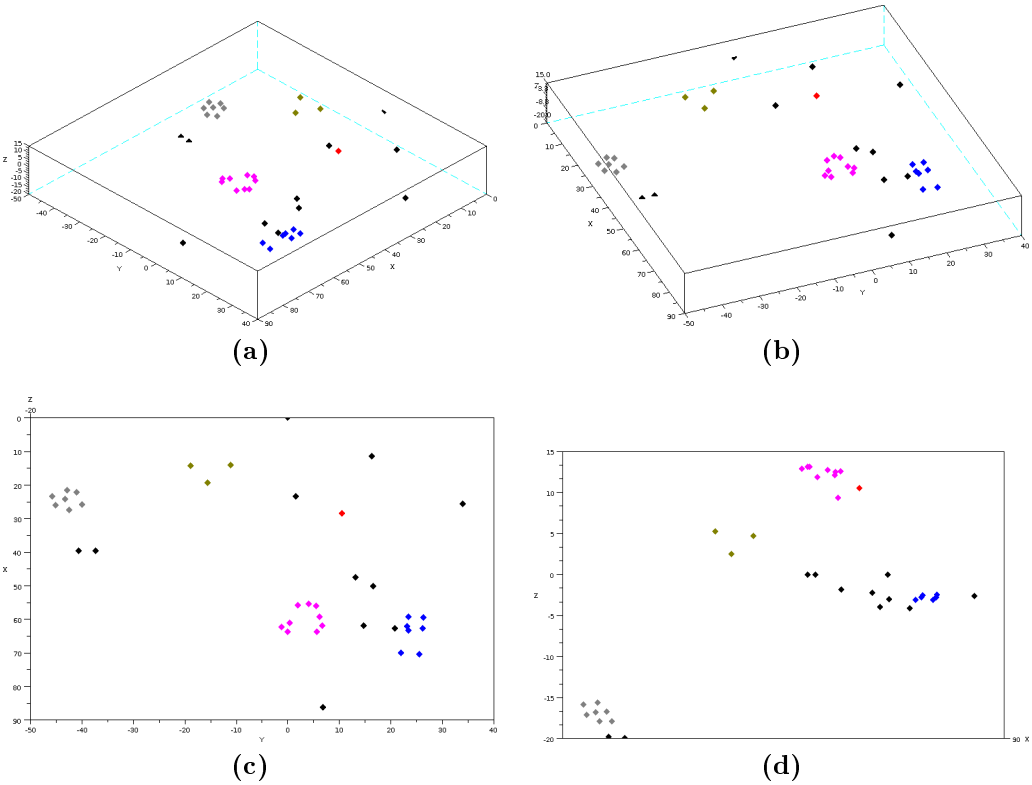

Figura 5.3: Reconstrução real da cena, plotada no Scilab – cinza: relógio; rosa: canetas; dourado: troféu; azul: celular; vermelho: tampa. (a) vista a partir da perspectiva da imagem 1. (b) idem, imagem 2. (c) vista de topo. (d) vista de frente.

6 Conclusão

A partir dos resultados obtidos, verificamos que o sistema construído satisfaz os objetivos especificados de maneira robusta. Este sistema demonstrou implementar com perfeição os algoritmos escolhidos para realizar a reconstrução real.

Pode-se acrescentar que, a partir do protótipo construído, seria simples e de grande benefício implementar algumas funcionalidades que o tornariam mais poderoso e ao mesmo tempo mais fácil de usar. Por exemplo, na interface poderia haver um mecanismo que possibilitasse o reaproveitamento automático de pontos correspondentes entre diferentes pares de fotos, o que tornaria mais fácil a reconstrução a partir de múltiplas imagens, captando vários ângulos.

Assim, encerramos comentando que obtivemos um protótipo sólido, cujo núcleo é confiável e permite que a utilidade do sistema seja aumentada amplamente apenas por conveniências implementadas na interface.

Referências

1 KARLSTROEM, A. *Estimação de Posição e Quantificação de Erro Utilizando Geometria Epipolar entre Imagens*. Dissertação (M. Eng. thesis) — Escola Politécnica da Universidade de São Paulo, São Paulo, 2007.

2 MARR, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York, NY, USA: Henry Holt and Co., Inc., 1982. ISBN 0716715678.

3 TSUZUKI, M. de S. G. *Solucionando sistemas polinomiais não lineares pelo metodo do poliedro projetado implementado em aritmética intervalar*. Tese (Livre-docência) — EPUSP, 2000.

4 SHIMADA, M.; TSUZUKI, M. de S. G. Implementação de operações booleanas usando aritmética intervalar em um modelador de sólidos B-REP. In: ABCM (Ed.). *Anais do XVI COBEM*. Uberlândia: ABCM, 2001.

5 MAYBANK, S. *Theory of Reconstruction from Image Motion*. Berlin, Germany: Springer-Verlag, 1993. ISBN 3-540-55537-4.

6 SEMPLE, J. G.; KNEEBONE, G. T. *Algebraic Projective Geometry*. Oxford, England: Oxford University Press, 1952. ISBN 0-19-8503636.

7 TSAI, R. An efficient and accurate camera calibration technique for 3d machine vision. *IEEE CVPR*, p. 364–374, 1986.

8 FABIO, R. From point cloud to surface: the modeling and visualization problem. In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Tarasp-Vulpera, Switzerland, 2003. XXXIV-5/W10.

9 HARTLEY, R.; ZISSERMAN, A. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

10 HARRIS, C.; STEPHENS, M. A combined corner and edge detector. In: *Proceedings of the 4th Alvey Vision Conference*. 1988. p. 147–151.

11 TAKIMOTO, R. Y.; NEVES, A. C.; MAFALDA, R.; TAVARES, R. S.; SATO, A. K.; STEVO, N.; TSUZUKI, M. S. G. Detecting function patterns with Interval Hough Transform. In: *Proceedings of the 9th IEEE/IAS International Conference on Industry Applications*. São Paulo: IEEE, 2010.

12 LONGUET-HIGGINS, H. C. A computer algorithm for reconstructing a scene from two projections. *Nature*, v. 293, p. 133–135, 1981.

13 MENDELSOHN, N. S. Some elementary properties of ill conditioned matrices and linear equations. *The American Mathematical Monthly*, Mathematical Association of America, v. 63, n. 5, p. 285–295, May 1956.

14 HARTLEY, R. I. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 19, p. 580–593, 1997.

15 PRESS, W. H.; TEUKOLSKY, S. A.; VETTERLING, W. T.; FLANNERY, B. P. *Numerical Recipes – The Art of Scientific Computing*. 3rd. ed. Cambridge, UK: Cambridge University Press, 2007.