



Frank Alexis Canahuire Cabello

CONTRIBUIÇÕES À DETECÇÃO DE FALHAS MECÂNICAS SOBRE  
PLACAS DE CIRCUITO IMPRESSO USANDO PROCESSAMENTO DE  
IMAGENS E O ALGORITMO SIFT.

Campinas  
2013



Universidade Estadual de Campinas  
Faculdade de Engenharia Elétrica e de Computação

**CONTRIBUIÇÕES À DETECÇÃO DE FALHAS MECÂNICAS SOBRE  
PLACAS DE CIRCUITO IMPRESSO USANDO PROCESSAMENTO DE  
IMAGENS E O ALGORITMO SIFT.**

**Frank Alexis Canahuire Cabello**  
**Orientador: Prof. Dr. Yuzo Iano**

**Dissertação de Mestrado** apresentada à Faculdade de Engenharia Elétrica e de Computação, da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Telecomunicações e Telemática.

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Frank Alexis Canahuire Cabello e orientado pelo Prof. Dr. Yuzo Iano

Assinatura do Orientador

---

Campinas

2013

FICHA CATALOGRÁFICA ELABORADA PELA  
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

C16c Canahuire Cabello, Frank Alexis  
Contribuições à detecção de falhas mecânicas sobre placas de circuito impresso usando processamento de imagens e o algoritmo SIFT / Frank Alexis Canahuire Cabello. --Campinas, SP: [s.n.], 2013.

Orientador: Yuzo Iano.  
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Algoritmos de computador. 2. Processamento de imagens. 3. Localização de falhas (Engenharia). I. Iano, Yuzo, 1950-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: Contributions to the detection of mechanical failures on printed circuit boards using image processing and the algorithm SIFT

Palavras-chave em Inglês: Computer algorithms, Image processing, Troubleshooting (Engineering)

Área de concentração: Telecomunicações e Telemática

Titulação: Mestre em Engenharia Elétrica

Banca examinadora: Yuzo Iano, Evaldo Gonçalves Pelaes, Rangel Arthur

Data da defesa: 25-01-2013

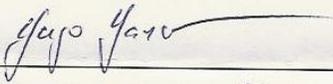
Programa de Pós Graduação: Engenharia Elétrica

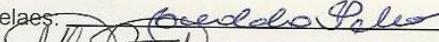
## COMISSÃO JULGADORA - TESE DE MESTRADO

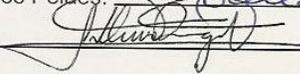
**Candidato:** Frank Alexis Canahuire Cabello

**Data da Defesa:** 25 de janeiro de 2013

**Título da Tese:** "Contribuições à Detecção de Falhas Mecânicas sobre Placas de Circuito Impresso Usando Processamento de Imagens e o Algoritmo SIFT"

Prof. Dr. Yuzo Iano (Presidente):  \_\_\_\_\_

Prof. Dr. Evaldo Gonçalves Pêlaes:  \_\_\_\_\_

Prof. Dr. Rangel Arthur:  \_\_\_\_\_

# Resumo

A área que trata de detectar falhas mecânicas em circuito impresso é, atualmente, de grande importância. Isso se deve ao fato de que cada dia as empresas procuram de novas máquinas mais eficientes e rápidas para detectar falhas em placas. Existem diferentes tipos de falhas mecânicas em placas de circuitos impressos, como por exemplo, falta de componentes, falta de soldadura, má colocação de dispositivos, etc. Este trabalho dará uma contribuição para solucionar o problema de má colocação de dispositivos nas placas de circuito impresso. Para isso serão usadas imagens de placas de circuito impresso e será usado um algoritmo de detecção de características chamado SIFT o qual será modificado para esse propósito. A modificação do método SIFT será na etapa de *matching* (encontrar correspondências semelhantes entre duas imagens) e será desenvolvido um algoritmo usando métodos geométricos para eliminar as correspondências que não pertencem à mesma característica. Será comparado o SIFT modificado com os métodos SURF, SIFT e RANSAC para determinar seu desempenho em variações de escala, rotação, ruído gaussiano e brilho. Será usado o método SIFT desenvolvido neste trabalho para solucionar o problema de má colocação de dispositivos em placas de circuito impresso.

**Palavras-chave:** características, circuito impresso, falhas mecânicas, SIFT.

# Abstract

The area which comes to detect mechanical failures in printed circuit board is currently of great importance. This is due to the fact that every day companies seek new machines more efficient and quick to detect flaws in printed circuit boards. There are different types of mechanical faults in printed circuit boards, such as lack of components, poor weld, bad placement devices and so on. This work will provide a contribution to solve the problem of poor placement of devices on printed circuit boards. For this, are used images of printed circuit boards and will use a method of feature detection called SIFT which will be modified for our purpose. The modification of the method SIFT is in the stage of matching (finding similar correspondences between two images) and will be developed an algorithm using geometric methods to eliminate the matches that do not belong to the same feature. Will be compared the modified SIFT with the methods SURF, SIFT and RANSAC to determine their performance in variations of scale, rotation, Gaussian noise and brightness. The SIFT method developed in this work will be used to solve the problem of poor placement of devices on printed circuit boards.

**Keywords:** feature, mechanical failures, printed circuit board, SIFT

# Agradecimentos

Aos meus pais, Esteban e Gloria, por ter me dado muito amor e compreensão.

Aos meus irmãos, Christian e Ruth, pela ajuda e compreensão durante meus estudos de Mestrado.

A minha amiga Angela por ser sempre compreensiva e ter me apoiado e escutado sempre que eu precisei.

Ao meu orientador, Prof. Yuzo Iano sou grato pela orientação, conselhos e compreensão.

A meus amigos pelo apoio durante esta jornada.

Aos integrantes dos grupos de pesquisa formados durante a minha vida acadêmica dentro e fora do Laboratório de Comunicações Visuais (LCV). Aos demais colegas de pós-graduação, pelas sugestões, ajudas, conselhos e amizades.

Meus agradecimentos às agências financiadoras do grupo de pesquisa do LCV como Capes (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - "*Coordination for the Improvement of Higher Level Personnel*"), CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico - "*National Counsel of Technological and Scientific Development*").

# Agradecimentos

Agradecimentos especiais ao programa CAPES RH-TVD da **Coordenação de Aperfeiçoamento de Pessoal de Nível Superior** tanto pelo apoio financeiro quanto pelo incentivo acadêmico para que este trabalho pudesse ser realizado.

# Sumário

<b>1. Introdução</b> .....	<b>1</b>
<b>2. Revisão teórica</b> .....	<b>4</b>
2.1 Introdução.....	4
2.2 Extração de características .....	4
2.3 <i>Matching</i> .....	5
2.3.1 <i>Template Matching</i> .....	5
2.3.2 <i>Feature Matching</i> .....	6
2.4 O algoritmo SIFT .....	7
2.4.1 Detecção de extremos no espaço de escalas.....	7
2.4.1.1 Espaço de escalas gaussiano .....	7
2.4.2 Localização de extremos .....	12
2.4.3 Localização precisa de pontos característicos .....	13
2.4.3.1 Remoção de pontos com baixo contraste .....	13
2.4.3.2 Remoção de pontos ao longo de uma aresta .....	17
2.4.4 Atribuição de orientações dos descritores .....	20
2.4.5 Construção do Descritor Local.....	22
2.4.6 <i>Matching</i> do SIFT .....	24
2.5 Algoritmo SIFT proposto .....	25
2.5.1 Parâmetros do SIFT.....	25
2.5.2 Algoritmo para eliminar falsas correspondências .....	25
<b>3. Testando a invariância e uso da Matriz de Homografia</b> .....	<b>28</b>
3.1 Introdução.....	28
3.2 Extração de características usando o algoritmo SIFT .....	28
3.3 Testando o algoritmo SIFT para comparar características entre duas imagens .....	34
3.4 Testando para variações em escala, orientação, ruído gaussiano e brilho .....	37
3.4.1 Variações em escala .....	37
3.4.2 Variações em rotação .....	38
3.4.3 Variações em ruído gaussiano.....	40
3.4.4 Variações em brilho.....	41
3.5 Matriz de Homografia .....	42
3.6 Eleição de pontos para calcular a Matriz de Homografia .....	43
<b>4. Uso do SIFT proposto sobre placas de circuito impressos</b> .....	<b>50</b>
4.1 Introdução.....	50
4.2 Aproximação da matriz de homografia .....	50
4.3 Testes usando o algoritmo SIFT com pós-processamento .....	52
4.3.1 Exemplo de ubicação errônea de componente .....	53
4.3.2 Exemplo de ubicação correta de componente .....	56

<b>5. Conclusões .....</b>	<b>62</b>
<b>6. Trabalhos futuros .....</b>	<b>63</b>
<b>7. Referências bibliográficas.....</b>	<b>64</b>
<b>A Apêndice A: A Matriz Hessiana.....</b>	<b>68</b>
<b>B Apêndice B: Propriedades da Gaussiana .....</b>	<b>69</b>
<b>C Apêndice C: Espaço de Escala Gaussiano.....</b>	<b>71</b>

# Lista de Figuras

Figura 2.1. Exemplo do Template Matching no lado esquerdo está o objeto que será buscado na imagem .....	6
Figura 2.2. Duas imagens do mesmo objeto. ....	6
Figura 2.3. Matching entre as duas imagens para obter as características parecidas usando o algoritmo CSIFT.....	7
Figura 2.4. Rpresentação do procedimento de obtenção das Diferenças de Gaussianas(DoG) para diversas oitavas de uma imagem partindo da pirâmide gaussiana[3].....	9
Figura 2.5. Obtendo a pirâmide gaussiana e a Diferencia de Gaussiano para uma imagem de olho humano. ....	11
Figura 2.6. Localização dos máximos e mínimos do DoG .....	11
Figura 2.7. Detecção de extremos no espaço-escala. ....	12
Figura 2.8. Detecção de canto,borda ou região segundo os valores $\lambda_1$ e $\lambda_2$ .....	20
Figura 2.9. Histograma de orientações de um ponto chave. ....	21
Figura 2.10. Calculo da orientação do ponto chave. ....	22
Figura 2.11. Descritor do ponto chave para uma janela de 8 x 8. ....	23
Figura 2.12. Descritor do ponto chave para uma janela de 16 x 16.....	23
Figura 2.13. Correspondências entre duas imagens usando o algoritmo SIFT.....	27
Figura 2.14. Correspondências depois de usar o algoritmo para eliminar as correspondências erradas.....	27
Figura 3.1. Imagem de teste Lena. ....	29
Figura 3.2. Oitava 1. ....	30
Figura 3.3. Oitava 2. ....	30
Figura 3.4. Oitava 3. ....	30
Figura 3.5. Oitava 4. ....	30
Figura 3.6. Diferença de Gaussiano da imagem Lena. ....	31
Figura 3.7. Máximos e mínimos na primeira oitava. ....	32
Figura 3.8. Máximos e mínimos na segunda oitava.....	32
Figura 3.9. Máximos e mínimos na terceira oitava.....	33
Figura 3.10. Características da imagem Lena 240 x 320.....	33
Figura 3.11. Características da imagem Lena 512 x 512.....	34
Figura 3.12. Características da imagem Lena girada 10 graus sexagesimais .....	34
Figura 3.13. Correspondências das duas imagens anteriores.....	35
Figura 3.14. Correspondências das duas imaagens giradas 60 graus sexagesimais. ....	35
Figura 3.15. Eliminação das correspondências errôneas pela aplicação do algoritmo proposto..	36
Figura 3.16. Correspondências de duas imagens que contem o mesmo objeto. ....	36
Figura 3.17. Eliminação das correspondencias errôneas .....	37
Figura 3.18. Correspondências para imagens de diferente escala. ....	38
Figura 3.19. Resultados de testes para duas imagens, $E$ é a escala que serão variados os eixos $x$ e $y$ da imagem .....	38
Figura 3.20. Correspondências para imagens de diferente ângulo. ....	39
Figura 3.21. Resultados para duas imagens, sendo a transformação da segunda imagem uma rotação de 0 a 50 graus sexagesimais em relação à primeira. ....	39
Figura 3.22. Correspondências para uma imagem com rúido gaussiano.....	40

Figura 3.23. Resultados para duas imagens, sendo a transformação a adição de ruído gaussiano na 2ª imagem em relação à primeira. ....	40
Figura 3.24. Correspondências para uma imagem com brilho. ....	41
Figura 3.25. A figura mostra os resultados para duas imagens, sendo a transformação de adição de brilho na segunda imagem com respeito à primeira.....	41
Figura 3.26. (a) Imagem A.(b) Imagem A com ruído gaussiano adicionado e rotação de 10 graus.....	43
Figura 3.27. Os quatro pontos escolhidos são representados por um círculo vermelho. ....	44
Figura 3.28. (a) Uma malha de 10x10 pontos é criada. (b) A imagem mostra a transformação que sofre os pontos de (a) usando a matriz de homografia. ....	44
Figura 3.29. A figura mostra os erros quadráticos médio para diferentes valores de $L$ . ....	45
Figura 3.30. Malha de pontos obtida pela matriz de homografia. ....	46
Figura 3.31. Correspondências usando o algoritmo SIFT com pós-processamento. ....	47
Figura 3.32. Os 4 pontos escolhidos para calcular a matriz da homografia é mostrado por um círculo azul. ....	48
Figura 3.33. Uma malha de pontos é criada na imagem . ....	48
Figura 3.34. Usando a matriz de homografia são obtidos os pontos resultantes. ....	49
Figura 4.1. (a) Imagem com o objeto na posição correta. (b) Imagem com o objeto na posição errada. ....	53
Figura 4.2. Usando o algoritmo SIFT para detectar o objeto. ....	54
Figura 4.3. Depois do pós-processamento as correspondências errôneas foram eliminadas.....	54
Figura 4.4. São selecionados os 4 pontos para calcular a matriz de homografia.....	55
Figura 4.5. (a) uma malha de pontos é criada sobre o objeto. (b) Usando a matriz de homografia é determinada a ubiqação dos pontos na segunda imagem. ....	56
Figura 4.6. Usando o algoritmo SIFT para detectar o objeto.....	56
Figura 4.7. Depois do pós-processamento as correspondências errôneas foram eliminadas.....	57
Figura 4.8. Os 4 pontos escolhidos para se calcular a matriz de homografia estão representados por um círculo azul.....	57
Figura 4.9. (a) uma malha de pontos é criada sobre o objeto. (b) Usando a matriz de homografia é determinada a ubiqação dos pontos na 2ª imagem. ....	58

# Lista de Tabelas

Tabela 3.1 Sigmas na primeira oitava.....	29
Tabela 3.2 Sigmas na 2 <sup>a</sup> oitava.....	30
Tabela 3.3 Valores de $E_{rms}$ da Fig. 3.29.....	46
Tabela 4.1 Calculo do ângulo para o primeiro objeto.....	59
Tabela 4.2 Calculo do ângulo para o segundo objeto.....	59
Tabela 4.3 Calculo do ângulo para o terceiro objeto.....	60
Tabela 4.4 Calculo do ângulo para o quarto objeto.....	60

# Abreviaturas

CSIFT	<i>Color and Scale Invariant Feature Transform</i>
DOG	<i>Difference of Gaussians</i>
RANSAC	<i>Random Sample Consensus</i>
SIFT	<i>Scale Invariant Feature Transform</i>
SURF	<i>Speed Up Robust Feature</i>

# Publicações

Listagem das publicações e participação em projetos e eventos acadêmicos durante o curso de mestrado na Faculdade de Engenharia Elétrica e de Computação da Unicamp (2010-2012).

1. CABELLO , F. A. C. ; IANO, Y.;Method of feature detection in images based on the algorithm SIFT.The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vancouver, Canada,2013.(submetido)
2. CABELLO , F. A. C. ; IANO, Y.; Desenvolvimento de uma interface virtual tátil para controle de acesso em áreas restritas. I Simpósio de Processamento de Sinais da UNICAMP (SPS-UNICAMP 2010) , Brazil, Outubro, 2010.

# Capítulo 1

## Introdução

Existem diferentes tipos de falhas mecânicas em circuitos impressos, como por exemplo, falta de componentes, falta de soldadura, má colocação de dispositivos, etc. Este trabalho dará uma contribuição para solucionar o problema de falta de componente e má colocação de dispositivos nas placas de circuito impresso. Para isto serão usadas imagens de placas de circuito impresso fornecidas pela empresa CadService<sup>1</sup> e também um algoritmo de detecção de características chamado SIFT o qual será modificado para nosso propósito.

Os problemas da detecção de características esbarram nas mesmas dificuldades vistas na detecção de objetos. Detectar uma característica consiste em: dada uma cena encontrar onde está a face, ou várias faces, bem como identificar seus componentes como olhos, boca, nariz.

### Principais trabalhos na área

Em [1] Lowe apresentou o método SIFT e ensina a como utiliza-lo em detecção de objetos, em [2] ele utilizou o método SIFT para detecção de objetos em 3D usando métodos probabilísticos e em [3] explicou mais detalhadamente o método SIFT com os parâmetros recomendáveis para a utilização do método.

Em [4] é desenvolvido um novo método de extração de características chamado SURF e esta baseado no método SIFT.

---

<sup>1</sup> <http://www.cadservice.com.br/>

Em [5] é utilizado o algoritmo PCA para diminuir o tempo de processamento embora não seja robusto e em [6] é testado a precisão do PCA-SIFT e foi comparado com o método SIFT. Em [7] é utilizado o método PCA-SIFT em faces para identificar homens e mulheres em [8] é utilizado o método SIFT para reconhecer alguns sinais da mão, por exemplo, diferencia se a mão esta aberta o fechada.

Em [9] é utilizado o método SIFT para diferenciar cenas de outras como, por exemplo, uma oficina de um parque. Em [10] é desenvolvido um método chamado CSIFT que é a utilização do método SIFT, mas usando a invariância em color.

Em [11] é utilizado o método CSIFT para detectar características em cenas de desenhos animados. Em [12] é utilizado o método CSIFT para detectar características em diferentes posições de um objeto 3D. Em [13] é modificado o método SIFT usando a normalização do vetor descritor. Em [14] é desenvolvido um novo método de compressão de imagens baseado no método SIFT. Em [15] são extraídos características usando o método SIFT para identificar faces.

## **Objetivos**

Os objetivos deste trabalho estão voltados principalmente para melhorias de um algoritmo de detecção de características.

Dessa forma, serão enfocados os seguintes itens:

- Melhor algoritmo de detecção de características para variações de escala, rotação, ruído gaussiano e brilho.
- Proposta de um algoritmo de detecção de características.
- Testes sobre o funcionamento do algoritmo proposto.
- Aplicação do algoritmo para resolver o problema de má colocação de dispositivos em placas de circuito impresso.

## **Estrutura da dissertação**

Todas as informações envolvidas no desenvolvimento deste trabalho estão descritas em cada um dos capítulos que compõem a presente pesquisa.

- Capítulo 2: definições gerais sobre detecção de características, em especial o algoritmo SIFT que será usado neste trabalho, e a proposta de um esquema de pós-processamento do SIFT para eliminar as correspondências errôneas.
- Capítulo 3: comparações entre algoritmos de detecção de características para variações em escala, rotação, ruído gaussiano e brilho; e uso da matriz de homografia.
- Capítulo 4: aplicação do algoritmo proposto para detectar má ubiquação de componentes em placas de circuito impresso.
- Capítulo 5: conclusões e trabalhos futuros.

# Capítulo 2

## Revisão teórica

### 2.1 Introdução

A tarefa de encontrar correspondências de pontos entre duas imagens da mesma cena ou objeto faz parte de muitas aplicações de visão por computador. Registro de imagens, calibração de câmera, reconhecimento de objetos e recuperação de imagens são apenas alguns exemplos.

Para qualquer objeto em uma imagem, pontos característicos podem ser extraídos para fornecer uma descrição característica desse objeto. Essa descrição é extraída a partir de uma imagem de treinamento, que então pode ser usada para identificar o objeto na tentativa de localizá-lo em uma imagem de teste contendo muitos outros objetos.

Para executar o reconhecimento confiável é importante que as características extraídas a partir da imagem de formação sejam detectáveis mesmo com mudanças na escala, rotação, ruído e brilho sobre a imagem. Esses pontos encontram-se geralmente em regiões de alto contraste da imagem, tais como as bordas do objeto.

### 2.2 Extração de características

A extração de informação de imagens através de processamento digital constitui na atualidade um imenso campo de estudo e investigação em diversas áreas, com uma grande quantidade de aplicações. Foi na área de visão computacional onde foi obtido um maior sucesso como, por exemplo, na detecção automática de características sobre imagens, que conta atualmente com uma variedade de métodos para esse propósito. Infelizmente, mesmo com a vasta quantidade de métodos, não existe um “método universal” para a detecção automática de características, por isso para cada problema é desenvolvido um algoritmo dependendo das necessidades do problema [16].

Uma imagem contém uma grande quantidade de dados onde a maior parte proporciona muito pouca informação para interpretar a cena. Um algoritmo de extração de características deve, em um primeiro passo, extrair da forma mais eficaz e robusta possível, determinadas características que proporcionem a máxima informação possível. Essas características devem cumprir as seguintes condições:

- O custo computacional para a extração de características não deve ser excessivo.
- A localização deve ser muito precisa.
- Devem ser robustas e estáveis.
- Conterão a maior informação possível da cena, isso significa que as características devem permitir a extração de informação de tipo geométrico.

As características mais importantes para serem extraídas são: os pontos, as linhas e os círculos, como geometrias básicas na extração de características.

## **2.3 Matching**

A técnica de *Matching* é um ramo importante de processamento de imagens. É amplamente utilizada em muitos domínios, tais como análise de imagens aéreas, seguimento de objetos, reconhecimento de padrões e análise de movimento [17]. Os algoritmos *Matching* de imagens podem ser classificados em dois tipos: método *Template Matching* baseado em filtros no nível de cinzas e o *Feature Matching* baseado na extração de características onde é possível usar imagem coloridas ou de cinzas.

### **2.3.1 Template Matching**

O *Template Matching* é uma das técnicas mais comuns usadas em processamento de sinais e processamento de imagens. Aplicações relacionadas a *Template Matching* incluem a recuperação de imagens, reconhecimento de imagens, registro de imagens, detecção de objetos e estéreo *matching* [18]. Um exemplo desse método é mostrado na Fig.2.1.

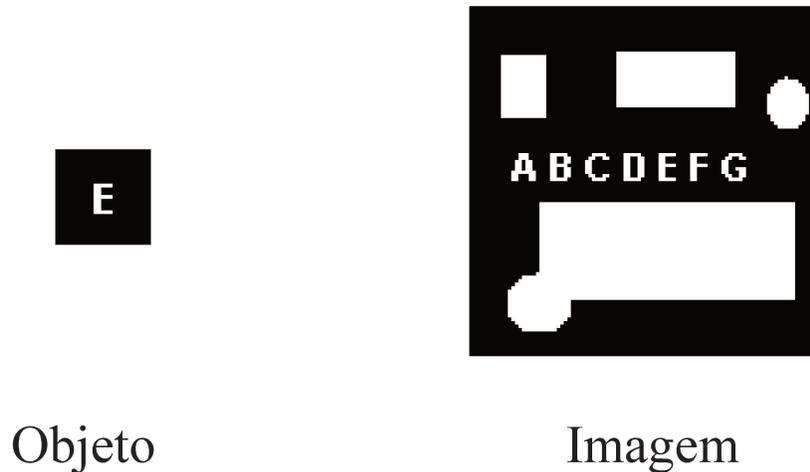


Fig. 2.1: Exemplo do *Template Matching*, no lado esquerdo está o objeto que será buscado na imagem.

### 2.3.2 *Feature Matching*

*Feature Matching* pode ser definido como um método essencial para o registro de imagens baseado em extração de características. A Fig. 2.2 é mostrada duas imagens de testes que o autor do algoritmo CSIFT [10] (O método SIFT adicionando características invariantes em color) utilizou para testar em imagens com variação de iluminação. Um algoritmo conhecido que usa o *Feature Matching* é o CSIFT e um exemplo de resultado do algoritmo é mostrado na Fig.2.3, onde pontos em comum nas duas imagens são identificados utilizando características semelhantes entre elas.



Fig. 2.2: Duas imagens do mesmo objeto.

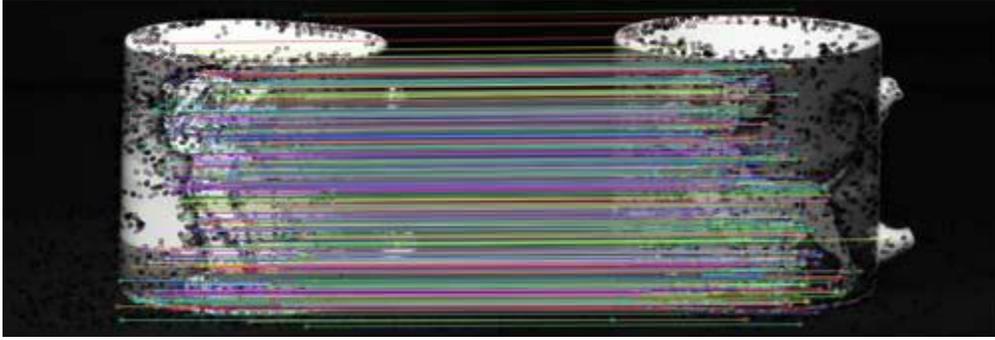


Fig.2.3: *Matching* entre as duas imagens para obter as características parecidas usando o algoritmo CSIFT.

## 2.4 O algoritmo SIFT

O algoritmo SIFT [2] que foi desenvolvido por Lowe, fornece em uma imagem um conjunto de características que não sofrem de muitas das complicações vividas em outros métodos, como mudanças de escala ou rotação porque utiliza métodos que procuram invariâncias como o espaço de escalas que proporciona invariância na escala.

O SIFT transforma uma imagem em uma grande coleção de vetores de características locais. Cada um desses vetores característicos é invariante para qualquer mudança de escala ou rotação da imagem [19].

### 2.4.1 Detecção de extremos no espaço de escalas

Primeiramente se deve buscar pontos que sejam invariantes a mudanças de escala da imagem. A função chamada espaço de escalas foi utilizada para conseguir esse objetivo. Ela é construída usando funções gaussianas.

#### 2.4.1.1 Espaço de escalas gaussiano

Dada uma imagem contínua  $f_0: R^2 \rightarrow R$ , define-se o espaço de escala Gaussiano de  $f_0$  como a função  $F: R^2 \times R_+ \rightarrow R$  (representada por  $F_t(x, y)$  onde  $t \in R_+$ ) que é a solução da equação bidimensional

$$\frac{\partial F_t(x, y)}{\partial t} = \nabla^2 F_t(x, y) = \frac{\partial^2 F_t(x, y)}{\partial x^2} + \frac{\partial^2 F_t(x, y)}{\partial y^2} \quad (2.1)$$

$$F_0(x, y) = f_0(x, y)$$

A solução pode ser expressa como uma convolução com Gaussianas Bidimensionais

$$F_t(x, y) = G_t(x, y) * f_0(x, y) \quad (2.2)$$

Onde \* é a operação convolução em  $x$  e  $y$ , e também:

$$G_\sigma(x, y) = G_\sigma(x)G_\sigma(y) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(x^2+y^2)} \quad (2.3)$$

Chama-se  $L_\sigma(x, y)$  à função criada no espaço de escalas a partir da imagem  $I(x, y)$  convoluída com a função gaussiana  $G_\sigma(x, y)$ .

$$L_\sigma(x, y) = G_\sigma(x, y) * I(x, y) \quad (2.4)$$

Para gerar imagens do Laplaciano do Gaussiano de forma rápida, usou-se o espaço de escala. Calcula-se, assim, a diferença entre duas escalas consecutivas. Também é calculada a diferença de gaussianas formadas. Nesse caso, obtém-se a diferença entre imagens vizinhas da pirâmide gaussiana separadas por uma constante  $K$ , conforme Fig. 2.4.

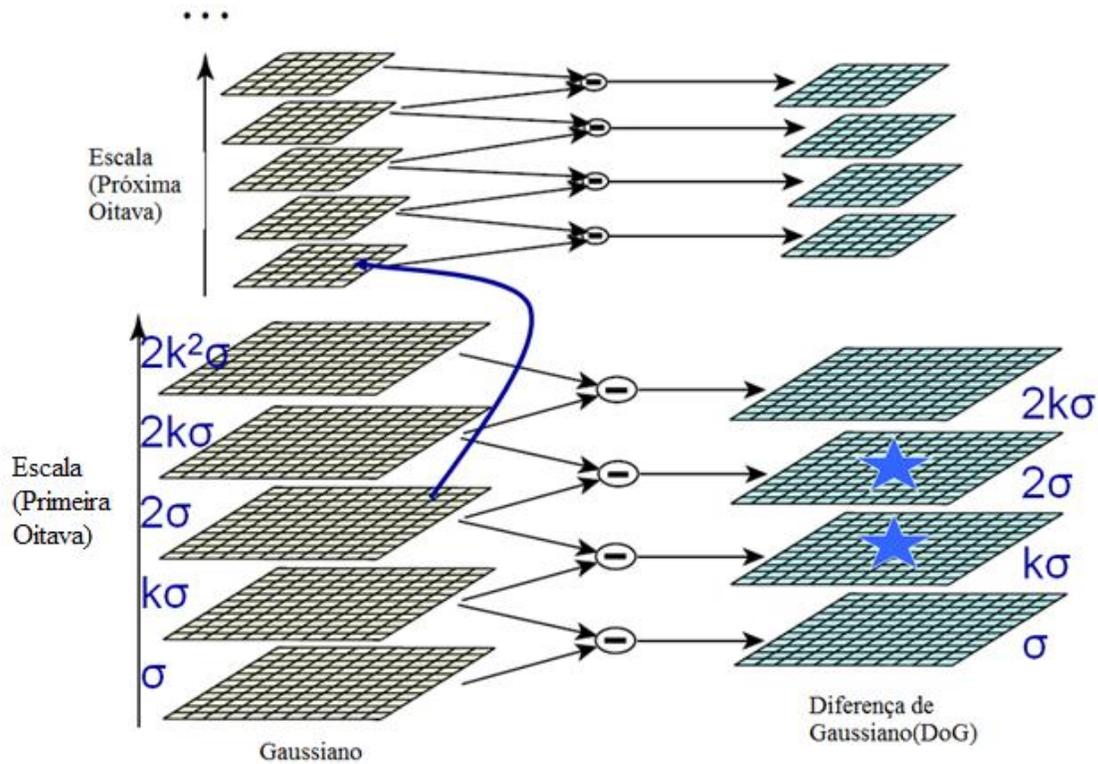


Fig. 2.4. Representação do procedimento de obtenção das Diferenças de Gaussianas (*DoG*) para diversas oitavas de uma imagem partindo da pirâmide gaussiana[3].

Definindo a função *DoG* como:

$$DoG = G_{k\sigma}(x, y) - G_{\sigma}(x, y) \quad (2.5)$$

Seja  $D_{\sigma}(x, y)$  a diferença entre duas imagens vizinhas no espaço de escala.

$$D_{\sigma}(x, y) = (G_{k\sigma}(x, y) - G_{\sigma}(x, y)) * I(x, y) \quad (2.6)$$

$$D_{\sigma}(x, y) = L_{k\sigma}(x, y) - L_{\sigma}(x, y) \quad (2.7)$$

A equação (2.7) significa que *DoG* pode ser calculada subtraindo imagens suavizadas por um Filtro Gaussiano em escalas  $\sigma$  e  $k\sigma$ .

Foi usada a função gaussiana porque elas suavizam as imagens. Isso significa que são eliminados detalhes indesejados e ruídos e, além disso, tem uma baixa complexidade computacional. Com a Diferença de Gaussiano, as características fortes serão realçadas.

A Diferença de Gaussiano é esquematizada na Fig.2.4 e também será explicado nos quatro passos seguintes.

A imagem inicial é convoluída consecutivamente com filtros gaussianos para produzir imagens separadas com um fator de escala  $k$ , conforme os passos seguintes.

1. Para cada oitava do espaço de escala, a imagem inicial é convoluída repetitivamente com Gaussianos para produzir o conjunto de imagens de espaço de escala que é mostrado à esquerda. Lowe considera necessário fazer a convolução da imagem até  $2\sigma$ , para ser possível a construção de descritores invariantes à escala.
2. São subtraídas imagens Gaussianas adjacentes para produzir as imagens de Diferenças de Gaussiano que é mostrada na direita.
3. Para criar a oitava seguinte, a resolução da imagem é reduzida (*downsampling*), tomando-se cada segundo pixel em cada linha e coluna, gerando-se uma nova oitava, e voltando-se ao passo 1.

Nas Figuras 2.5 e 2.6 é mostrado um esquema onde é exemplificada a pirâmide gaussiana para uma imagem de um olho humano, o exemplo foi extraído de [20].

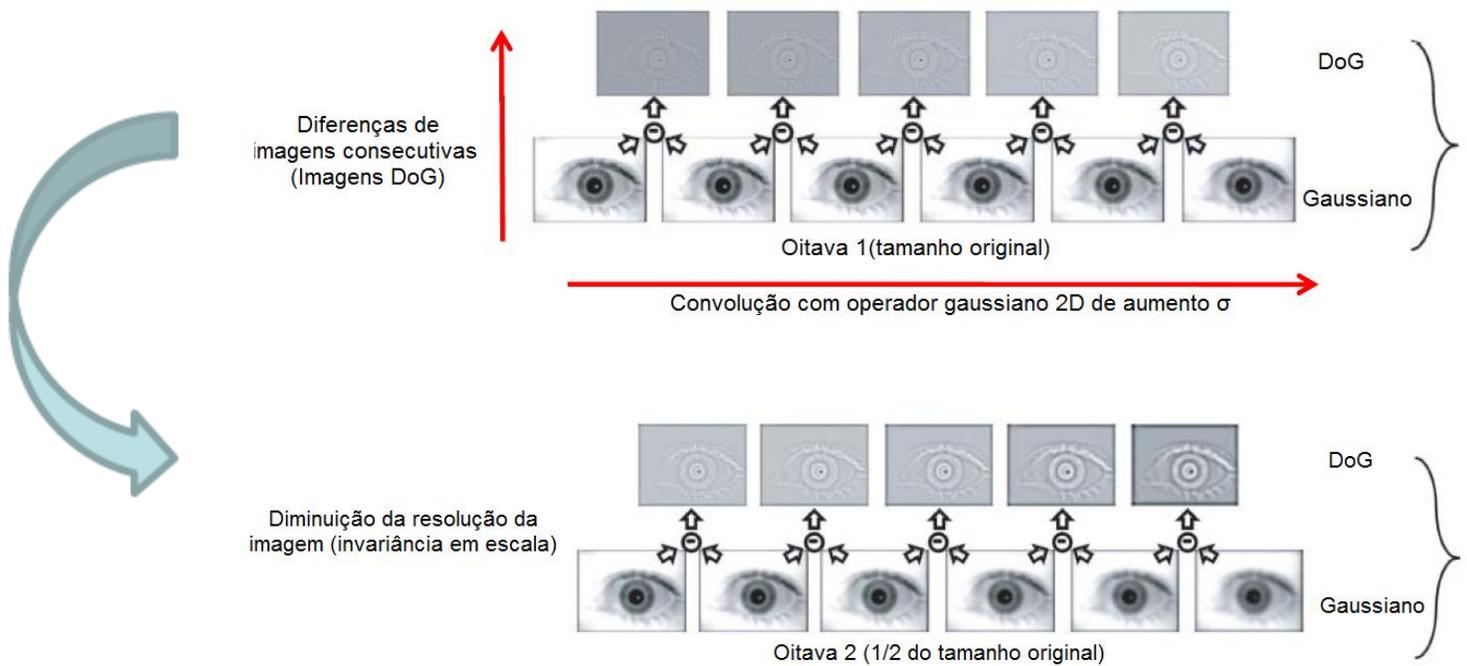


Fig. 2.5. Obtendo a pirâmide gaussiana e a Diferença de Gaussiano para uma imagem de olho humano.

Em cada localização do candidato, um modelo detalhado está apto a determinar a localização e escala. Pontos característicos são selecionados com base em medidas de sua estabilidade.

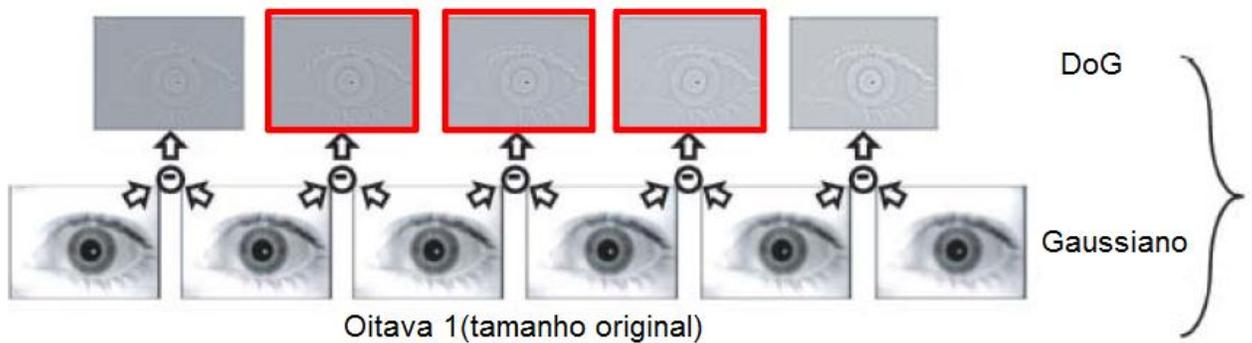


Fig. 2.6. Localização dos máximos e mínimos do DoG.

## 2.4.2 Localização de extremos

Para detectar os máximos e mínimos locais de  $D_{\sigma}(x, y)$ , cada ponto é comparado com seus oito vizinhos de sua escala, sendo nove vizinhos na escala superior e nove vizinhos na escala inferior, como é apresentado na Fig. 2.7. O ponto é selecionado se é o maior ou menor de todos os seus vizinhos. Na figura, o ponto marcado com “X” é comparado com seus vizinhos marcados como “O”. As 3 imagens *DoG* apresentadas na figura correspondem à diferença entre imagens adjacentes da pirâmide gaussiana.

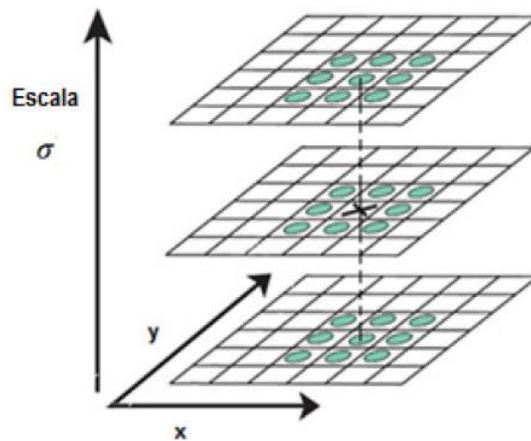


Fig.2.7. Detecção de extremos no espaço-escala.

O custo computacional para esse procedimento é baixo devido ao fato que a maior parte de pontos será eliminada nas primeiras buscas. A partir de então, será feita a detecção de extremos em cada intervalo de cada oitava. Um extremo é definido como qualquer valor no *DoG* maior que todos os seus vizinhos no espaço-escala. A próxima etapa é definir a localização dos pontos característicos e fazer o descarte de pontos instáveis.

### 2.4.3 Localização precisa de pontos característicos

Nem todos os pontos encontrados são pontos característicos da imagem. Nesta etapa serão eliminados os pontos instáveis que podem ser de dois tipos: os pontos que tenham baixo contraste (sensível ao ruído) e os pontos que estejam localizados ao longo de arestas (sensível ao ponto de vista ou variação de iluminação).

#### 2.4.3.1 Remoção de pontos com baixo contraste

Uma vez que um ponto característico candidato é encontrado por meio da comparação de um pixel com seus vizinhos, o passo seguinte é realizar um ajuste detalhado para os dados de localização, escala e proporção de curvaturas principais. Essa informação permite encontrar os pontos que têm baixo contraste para serem rejeitados (sensível ao ruído) ou estão mal localizados ao longo de uma aresta (sensível ao ponto de vista ou variação de iluminação).

O método desenvolvido neste trabalho consiste em ajustar uma função quadrática  $3D$  do ponto de amostragem local de modo a determinar uma localização interpolada do máximo. Isto é feito utilizando uma expansão de Taylor (acima dos termos quadráticos) da função Diferença de Gaussiano aplicada à imagem,  $D_\sigma(x, y)$ , deslocada de modo que a origem dessa expansão esteja localizada no ponto de amostragem [21]:

$$\begin{aligned} X_i &= (x_i, y_i, \sigma_i) \\ \Delta X &= (x - x_i, y - y_i, \sigma - \sigma_i) \\ X &\leftarrow X_i + \Delta X \end{aligned}$$

Utilizando a expansão de Taylor para aproximar  $D_\sigma(x, y)$  e estimar  $\Delta X$

$$D(\Delta X) = D(X_i) + \frac{\partial D^T(X_i)}{\partial X} \Delta X + \frac{1}{2} X^T \frac{\partial^2 D(X_i)}{\partial X^2} \Delta X \quad (2.8)$$

Procurando o extremo  $D(\Delta X)$  derivando a equação (2.8) e igualando a zero.

$$\frac{d}{dX} \left( D(X_i) + \frac{\partial D^T(X_i)}{\partial X} \Delta X \right) = 0$$

$$\frac{\partial D(X_i)}{\partial X} + \frac{\partial^2 D(X_i)}{\partial X^2} \Delta X = 0$$

$$\frac{\partial^2 D(X_i)}{\partial X^2} \Delta X = - \frac{\partial D(X_i)}{\partial X} \quad (2.9)$$

$$\Delta X = - \frac{\partial^2 D^{-1}(X_i)}{\partial X^2} \cdot \frac{\partial D(X_i)}{\partial X} \quad (2.10)$$

Da equação (2.9),  $\Delta X$  pode ser calculada solucionando um sistema linear 3x3,

$$\frac{\partial D(X_i)}{\partial X} = \begin{bmatrix} \frac{\partial D}{\partial \sigma} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial x} \end{bmatrix} \quad (2.11)$$

$$\frac{\partial^2 D(X_i)}{\partial X^2} = \frac{\partial}{\partial X} \left( \frac{\partial D(X_i)}{\partial X} \right) = \frac{\partial}{\partial X} \left( \begin{bmatrix} \frac{\partial D}{\partial \sigma} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial x} \end{bmatrix} \right)$$

$$\frac{\partial^2 D(X_i)}{\partial X^2} = \begin{bmatrix} \frac{\partial}{\partial \sigma} \cdot \frac{\partial D}{\partial \sigma} & \frac{\partial}{\partial \sigma} \cdot \frac{\partial D}{\partial y} & \frac{\partial}{\partial \sigma} \cdot \frac{\partial D}{\partial x} \\ \frac{\partial}{\partial y} \cdot \frac{\partial D}{\partial \sigma} & \frac{\partial}{\partial y} \cdot \frac{\partial D}{\partial y} & \frac{\partial}{\partial y} \cdot \frac{\partial D}{\partial x} \\ \frac{\partial}{\partial x} \cdot \frac{\partial D}{\partial \sigma} & \frac{\partial}{\partial x} \cdot \frac{\partial D}{\partial y} & \frac{\partial}{\partial x} \cdot \frac{\partial D}{\partial x} \end{bmatrix}$$

$$\frac{\partial^2 D(X_i)}{\partial X^2} = \begin{bmatrix} \frac{\partial^2 D}{\partial \sigma^2} & \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial \sigma x} \\ \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial y^2} & \frac{\partial^2 D}{\partial xy} \\ \frac{\partial^2 D}{\partial \sigma x} & \frac{\partial^2 D}{\partial xy} & \frac{\partial^2 D}{\partial x^2} \end{bmatrix} \quad (2.12)$$

Substituindo as equações (2.11) e (2.12) em (2.9)

$$\begin{bmatrix} \frac{\partial^2 D}{\partial \sigma^2} & \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial \sigma x} \\ \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial y^2} & \frac{\partial^2 D}{\partial xy} \\ \frac{\partial^2 D}{\partial \sigma x} & \frac{\partial^2 D}{\partial xy} & \frac{\partial^2 D}{\partial x^2} \end{bmatrix} \cdot \begin{bmatrix} \Delta \sigma \\ \Delta y \\ \Delta x \end{bmatrix} = - \begin{bmatrix} \frac{\partial D}{\partial \sigma} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial x} \end{bmatrix} \quad (2.13)$$

Onde:

$$\frac{\partial D}{\partial \sigma} = \frac{D_{K+1}^{i,j} - D_{K-1}^{i,j}}{2} \quad (2.14)$$

$$\frac{\partial D}{\partial y} = \frac{D_K^{i+1,j} - D_K^{i-1,j}}{2} \quad (2.15)$$

$$\frac{\partial D}{\partial x} = \frac{D_K^{i,j+1} - D_K^{i,j-1}}{2} \quad (2.16)$$

$$\frac{\partial^2 D}{\partial \sigma^2} = \frac{\partial}{\partial \sigma} \frac{\partial D}{\partial \sigma} = (D_{K+1}^{i,j} - D_K^{i,j}) - (D_K^{i,j} - D_{K-1}^{i,j})$$

$$\frac{\partial^2 D}{\partial \sigma^2} = D_{K+1}^{i,j} - 2D_K^{i,j} + D_{K-1}^{i,j} \quad (2.17)$$

$$\frac{\partial^2 D}{\partial \sigma x} = \frac{\partial}{\partial x} \frac{\partial D}{\partial \sigma} = \frac{\partial}{\partial x} \left( \frac{D_{K+1}^{i,j} - D_{K-1}^{i,j}}{2} \right)$$

$$\frac{\partial^2 D}{\partial \sigma x} = \frac{1}{2} \left( \frac{\partial}{\partial x} D_{K+1}^{i,j} - \frac{\partial}{\partial x} D_{K-1}^{i,j} \right) = \frac{1}{2} \left( \frac{D_{K+1}^{i,j+1} - D_{K+1}^{i,j-1}}{2} - \frac{D_{K-1}^{i,j+1} - D_{K-1}^{i,j-1}}{2} \right)$$

$$\frac{\partial^2 D}{\partial \sigma x} = \frac{(D_{K+1}^{i,j+1} - D_{K+1}^{i,j-1}) - (D_{K-1}^{i,j+1} - D_{K-1}^{i,j-1})}{4} \quad (2.18)$$

$$\frac{\partial^2 D}{\partial \sigma y} = \frac{(D_{K+1}^{i+1,j} - D_{K+1}^{i-1,j}) - (D_{K-1}^{i+1,j} - D_{K-1}^{i-1,j})}{4} \quad (2.19)$$

Substituindo as equações (2.14), (2.15), (2.16), (2.17), (2.18) e (2.19) na equação (2.13) pode-se obter os valores de  $\Delta\sigma$ ,  $\Delta y$  e  $\Delta x$ . Se algum desses valores é maior que 0,5, o processo será repetido.

É aconselhável, segundo Lowe, que se rejeitem valores de  $|D(X_i + \Delta X)|$  inferiores a um determinado limiar. Em [2], é aconselhado trabalhar com o valor 0,03 para esse limiar, assumindo-se que os tons de cinza dos *pixels* da imagem estejam normalizados em valores entre 0 e 1.

### 2.4.3.2 Remoção de pontos ao longo de uma aresta

Lowe percebeu que a função *DoG* possui uma resposta com altas intensidades ao longo de arestas, isto significa que vários pontos chaves obtidos nos máximos e mínimos do *DoG* estarão localizados sobre as arestas. Para eliminar aqueles pontos será usada a matriz Hessiana  $H$  2x2, calculada para a localização e escala dos pontos chaves na função  $D$ .

Harris [22] usou a matriz de auto-correlação:

$$A_W(x, y) = \sum_{x \in W, y \in W} \begin{bmatrix} f_x^2 & f_x f_y \\ f_x f_y & f_y^2 \end{bmatrix}$$

$$R(A_W) = \text{Det}(A_W) - \alpha \cdot \text{Trace}^2(A_W)$$

$$R(A_W) = \lambda_1 \lambda_2 - \alpha \cdot (\lambda_1 + \lambda_2)^2 \quad (2.20)$$

O algoritmo *SIFT* utiliza a matriz Hessiana por sua eficiência

$$H(x, y) = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \quad (2.21)$$

A matriz Hessiana representa as magnitudes das curvaturas de  $D$  nos eixos  $x$  e  $y$ . As derivadas são calculadas através de aproximações de diferenças entre pontos vizinhos à localização e escala estabelecida como segue:

$$D_{xx} = D_{\sigma}^{x+1,y} - 2D_{\sigma}^{x,y} + D_{\sigma}^{x-1,y}$$

$$D_{yy} = D_{\sigma}^{x,y+1} - 2D_{\sigma}^{x,y} + D_{\sigma}^{x,y-1}$$

$$D_{xy} = \frac{(D_{\sigma}^{x+1,y-1} - D_{\sigma}^{x-1,y-1}) - (D_{\sigma}^{x+1,y+1} - D_{\sigma}^{x-1,y+1})}{4}$$

Sejam:

$\alpha$  : maior autovalor ( $\lambda_{max}$ )

$\beta$  : menor autovalor ( $\lambda_{min}$ )

$$r = \frac{\alpha}{\beta}$$

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta$$

$$Tr(H) = \beta(r + 1) \tag{2.22}$$

$$Det(H) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta$$

$$Det(H) = \beta^2 r \tag{2.23}$$

Será criado um novo parâmetro que só dependa da variável  $r$ , para isto serão usadas as equações (2.22) e (2.23).

Elevando ao quadrado a equação (2.22)

$$Tr^2(H) = \beta^2(r + 1)^2 \tag{2.24}$$

Das equações (2.23) e (2.24) elimina-se o parâmetro  $\beta$ :

$$\frac{Tr^2(H)}{Det(H)} = \frac{\beta^2(r+1)^2}{\beta^2 r}$$

$$\frac{Tr^2(H)}{Det(H)} = \frac{(r + 1)^2}{r} \quad (2.25)$$

$$\frac{Tr^2(H)}{Det(H)} = r + \frac{1}{r} + 2 \quad (2.26)$$

É conhecido que para todo  $r \in R$  que:

$$r + \frac{1}{r} \geq 2 \quad (2.27)$$

Então substituindo a desigualdade (2.27) na equação (2.26)

$$\frac{Tr^2(H)}{Det(H)} \geq 4 \quad (2.28)$$

Onde a igualdade acontece quando  $r = 1$ , significa que  $\alpha = \beta$ . Segundo o algoritmo de detecção de cantos de Harris[22] para detecção de bordas, se  $\lambda_1 \gg \lambda_2$  ou  $\lambda_2 \gg \lambda_1$ , aquele ponto está sobre uma borda(ver Fig. 2.8). Serão rejeitados aqueles pontos da seguinte forma: se  $r$  é grande então é provável que o ponto chave esteja localizado em uma borda, porque  $\alpha = r\beta$ . Lowe considerou que para  $r > 10$ , o ponto estava localizado sobre uma borda e esse ponto será rejeitado. Para rejeitar aqueles pontos será usada a equação (2.25).

Se  $\frac{Tr^2(H)}{Det(H)} > \frac{(r'+1)^2}{r'}$  o ponto será rejeitado,  $r' = 10$ .

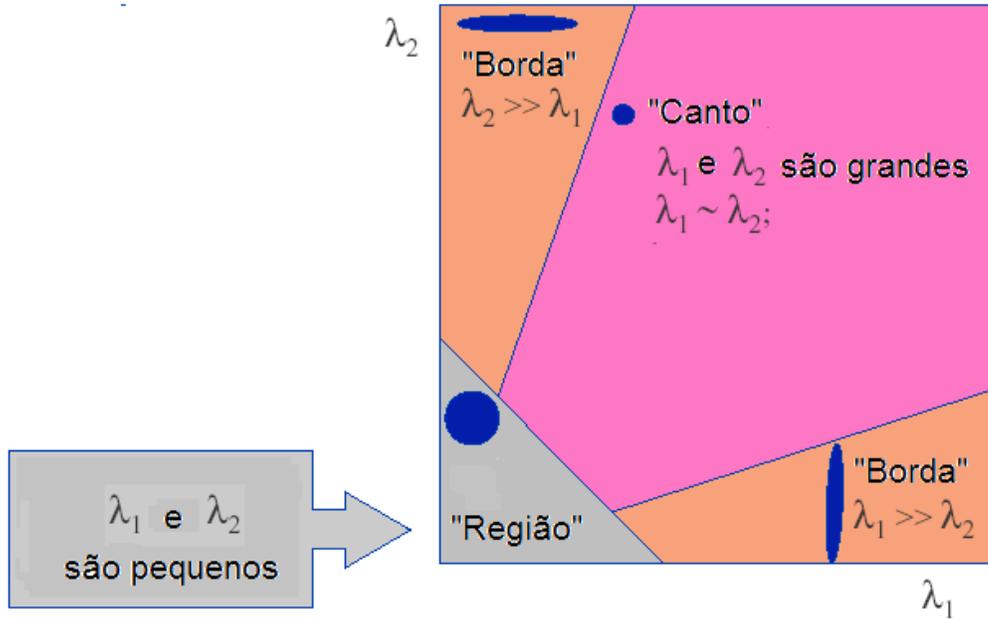


Fig.2.8. Detecção de canto, borda ou região segundo os valores de  $\lambda_1$  e  $\lambda_2$ .

#### 2.4.4 Atribuição de orientações dos descritores

Cada ponto chave terá sua localização e escala, neste passo será calculada a orientação da característica. É necessário saber a orientação para se construir o descritor e seja definido se o ponto é invariante à rotação. Essa orientação será calculada para cada  $L_\sigma(x, y)$ , obtida na equação (2.4), a magnitude  $m(x, y)$  e orientação  $\theta(x, y)$  do gradiente.

$$m(x, y) = \sqrt{(L_\sigma(x + 1, y) - L_\sigma(x - 1, y))^2 + (L_\sigma(x, y + 1) - L_\sigma(x, y - 1))^2} \quad (2.29)$$

$$\theta(x, y) = \tan^{-1}\left(\frac{L_\sigma(x, y + 1) - L_\sigma(x, y - 1)}{L_\sigma(x + 1, y) - L_\sigma(x - 1, y)}\right) \quad (2.30)$$

Com os valores da magnitude e orientação do gradiente, obtidas das equações (2.29) e (2.30), é criado um histograma de orientações para *pixels* em uma região vizinha ao redor do ponto chave [23]. O histograma (de  $0$  a  $2\pi$ ) será dividido em 36 regiões, onde cada região cobrirá 10 graus sexagesimais assim como é mostrado em Fig. 2.9.

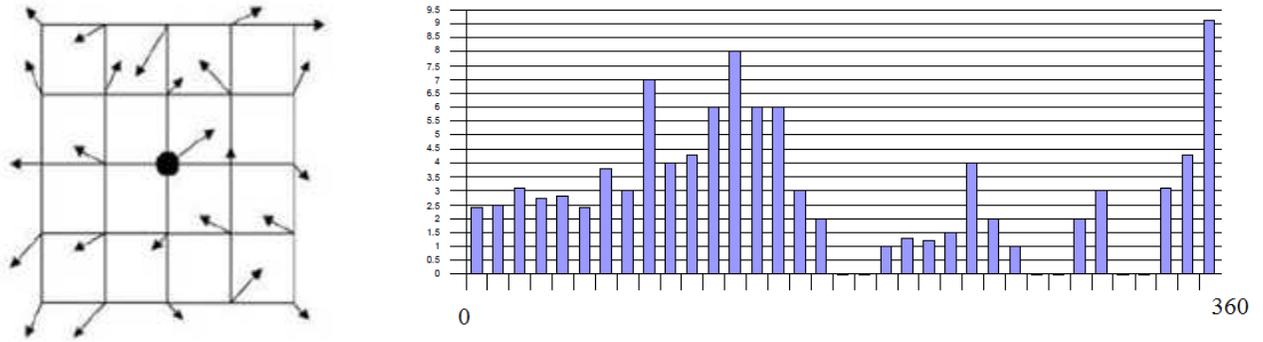


Fig.2.9. Histograma de orientações de um ponto chave.

Os pontos que pertencem à vizinhança do ponto chave terão pesos que serão adicionados ao histograma.

O primeiro peso será a magnitude que é obtida na equação (2.29), o segundo peso será uma janela Gaussiana com  $\sigma$  igual a 1,5 vezes maior que a escala do ponto chave. A janela é representada pela seguinte equação:

$$G_{\sigma}(\Delta x, \Delta y) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(\Delta x^2 + \Delta y^2)} \quad (2.31)$$

Onde  $\Delta x$  e  $\Delta y$  são as distâncias entre cada ponto vizinho e o ponto chave. O valor dos pesos calculados para cada ponto na vizinhança em  $(x, y)$  é atualizado na expressão:

$$W'_{\theta} = W_{\theta} + \alpha \cdot m(x, y) \cdot G_{\sigma}(\Delta x, \Delta y) \quad (2.32)$$

$$\alpha = \begin{cases} \frac{d}{i}, & d < i \\ 0, & d > i \end{cases}$$

Onde  $W'_{\theta}$  é a atualização de  $W_{\theta}$ ,  $d$  é a distância absoluta em graus entre a orientação do ponto e o  $\theta$  discretizado, e  $i$  é o intervalo em graus entre os  $\theta$ 's discretizados (nesse caso  $i$  é igual a 10 graus) [24].

O valor máximo do histograma é o valor dominante dos gradientes locais. Também são considerados os valores acima de 80% do valor máximo e, por isso, um ponto chave poderá ter mais de uma orientação.

Para uma melhor exatidão é criada uma parábola para interpolar os três valores do histograma mais próximos ao ponto máximo. Logo, cada ponto chave tem as quatro dimensões: posição (x e y), magnitude e orientação.

A Fig.2.10 representa o algoritmo que realiza o cálculo da orientação. Na esquerda são representadas as magnitudes e orientações dos gradientes de cada vizinho, o círculo é o gaussiano e na direita é representada a orientação do ponto chave.

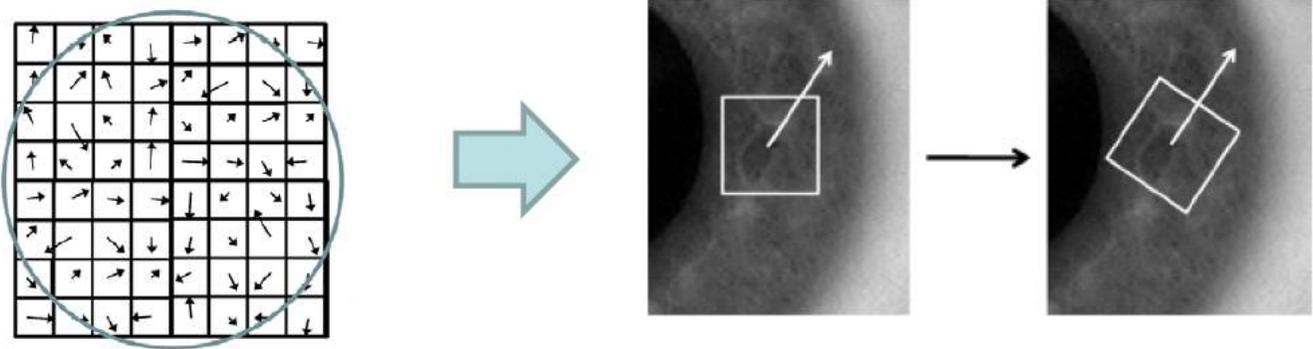


Fig.2.10. Cálculo da orientação do ponto chave.

### 2.4.5 Construção do Descritor Local

Os gradientes locais da imagem são medidos na escala selecionada na região em torno de cada ponto chave. Esses são transformados em uma representação que permite a localização mesmo com níveis significativos de distorção da forma local e mudança de iluminação.

Um descritor é calculado para a região de imagem local, que é tão característico quanto possível em cada ponto chave candidato. As magnitudes imagem gradiente e orientações são mostradas ao redor do local ponto chave. Estes valores encontram-se ilustrados com pequenas setas em cada local da amostra na primeira imagem da Fig.2.11.

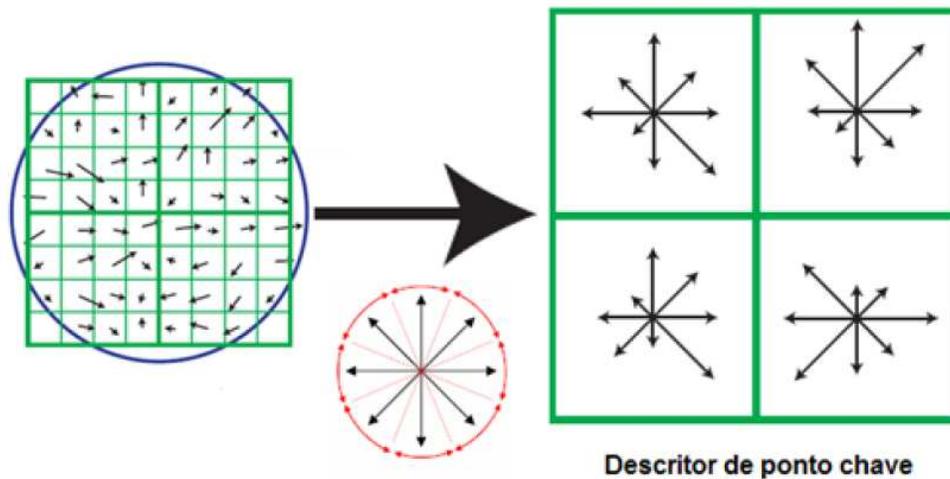


Fig.2.11. Descritor do ponto chave para uma janela de 8x8.

A função de ponderação gaussiana com ' $\sigma$ ' relacionada com a dimensão do ponto chave é utilizada para atribuir um peso à magnitude. Utilizou-se aqui um  $\sigma$  igual a metade da largura da janela do descritor nesta implementação. A fim de alcançar a invariância de orientação, as coordenadas do descritor e as orientações de gradiente são giradas em relação à orientação do ponto chave. Esse processo é indicado na Figura 2.12. Na aplicação deste trabalho, uma amostra de matriz de 16x16 (ver Fig. 2.12) é calculada e é utilizado um histograma com 8 posições.

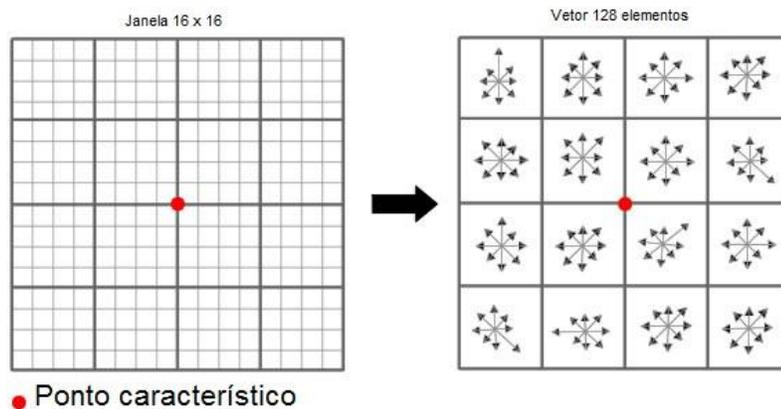


Fig.2.12. Descritor do ponto chave para uma janela de 16x16.

O descritor consiste de um vetor contendo os valores do histograma. A janela de  $16 \times 16$  pixels é convertida em uma janela de  $4 \times 4$  pixels, mas com cada pixel com um histograma de 8 posições, assim o vetor característico terá 128 elementos. O vetor característico é normalizado para se obter invariância à iluminação.

Cada imagem será representada como um conjunto de descritores robustos onde cada um pertence a um ponto chave, e podem ser usados para fazer a correspondência da imagem em outra imagem.

#### **2.4.6 Matching do SIFT**

O objetivo do *matching* é comparar descritores dos pontos chave entre duas imagens para saber quantos pontos chave tem em comum. Essa comparação pode ser feita usando a distância euclidiana, *Nearest Neighbour*, etc. Lowe utilizou o método de *Best-Bin-First* (BBF) [25], que pode identificar os vizinhos mais próximos com baixo custo computacional.

Nesse procedimento serão obtidas boas correspondências e falsas correspondências. Para a eliminação de falsas correspondências será usado o método de *distance ratio*, que realiza uma divisão entre a menor distância com a segunda menor distância.

Lowe rejeitou todas as correspondências cujo valor de *distance ratio* era superior a 0.8, o que eliminou 90% das falsas correspondências, mas com 5 % das boas correspondências. O objetivo agora será eliminar a maior quantidade de falsas correspondências restantes.

## 2.5 Algoritmo SIFT proposto

O esquema apresentado neste trabalho tem como objetivo eliminar a maior quantidade de falsas correspondências. O algoritmo de Lowe para eliminar falsas correspondências descarta 90% delas. O algoritmo aqui apresentado eliminará a maior quantidade dos 10% restantes utilizando métodos geométricos.

Primeiramente serão estabelecidos os parâmetros do algoritmo SIFT que será usado e depois o algoritmo para eliminar as falsas correspondências.

### 2.5.1 Parâmetros do SIFT

O algoritmo base para programar em Matlab o algoritmo SIFT foi desenvolvido por Yan Tao Zhen [26], onde o programa segue todos os passos do algoritmo desenvolvido por Lowe, incluindo o algoritmo que calcula as correspondências corretas usando a relação de distâncias entre o melhor e o segundo melhor.

Os parâmetros usados no algoritmo SIFT para obter as correspondências entre duas imagens são: sigma inicial = 1.6, número de oitavas = 4, número de escalas = 3, *distance ratio* = 75%, os demais parâmetros são por defeito segundo o algoritmo que desenvolveu Yan Tao Zhen.

### 2.5.2 Algoritmo para eliminar falsas correspondências

Usando só o algoritmo SIFT foi observado (por exemplo, na Fig. 2.13) que existem falsas correspondências. Por essa razão foi desenvolvido um algoritmo para eliminar a maior quantidade de elas.

A ideia do algoritmo consiste em encontrar três correspondências entre as imagens  $A$  e  $B$ . Isso significa que o triângulo formado por três pontos chaves de  $A$  deve ser semelhante ao triângulo formado por três pontos chaves na imagem  $B$  (onde esses pontos chaves pertencem às três correspondências). Logo foi estabelecida uma relação entre os ângulos dos dois triângulos.

Se as três correspondências são corretas, então os ângulos na imagem  $A$  e na imagem  $B$  são similares para variações de escala e rotação. Em seguida é explicado o esquema do algoritmo para as duas imagens  $A$  e  $B$  depois de ter realizado o algoritmo SIFT.

- Passo 1: Escolher três pontos chave na imagem  $A$  e três pontos chave na imagem  $B$  que são correspondentes (as correspondências são obtidas usando o algoritmo SIFT),  $X_a, Y_a$  e  $Z_a$  para a imagem  $A$  e  $X_b, Y_b$  e  $Z_b$  para a imagem  $B$ .
- Passo 2: Calcular os ângulos dos pontos escolhidos nas imagens  $A$  e  $B$ , chama-se  $\angle X_a$  ao ângulo formado pelos pontos  $Y_a, X_a$  e  $Z_a$  e será realizado o mesmo procedimento para os 5 ângulos restantes.
- Passo 3: Será calculada a diferença dos ângulos dos pontos correspondidos,  $D_x = |\angle X_a - \angle X_b|$ ,  $D_y = |\angle Y_a - \angle Y_b|$  e  $D_z = |\angle Z_a - \angle Z_b|$ . Será calculada a soma dessas diferenças e se o valor é menor que um limiar  $K$ , a quantidade de vezes que o ponto está entre os pontos pertencentes a essa soma, será armazenado em um vetor  $V$ .

Se  $D_x + D_y + D_z < K$ , então o vetor  $V$  será:  $V(x) = V(x) + 1$ ,  $V(y) = V(y) + 1$ ,  $V(z) = V(z) + 1$ .

Esse procedimento será realizado para todas as combinações de 3 pontos pertencentes a todas as correspondências.

- Passo 4: É normalizado o vetor  $V$ , e todos os pontos que sejam maiores que um limiar  $L$  serão os novos pontos de correspondência. Os outros pontos serão rejeitados.

O valor da constante  $K$  foi de 15 graus sexagesimais, mas esse valor pode variar entre 10 e 25 onde o algoritmo trabalha de forma aceitável. Se o valor de  $K$  é grande então existirá maior número de correspondências errôneas e se é pequeno, eliminará muitas correspondências corretas.

O valor da constante  $L$  foi de 75%, mas a faixa onde se trabalha de forma aceitável é de 40% a 90%. Se esse valor é grande, então rejeitará muitas correspondências corretas e se esse valor é pequeno existirá maior número de correspondências errôneas.

A Fig. 2.14 mostra as correspondências após a aplicação do algoritmo proposto utilizando a imagem Lena com resolução 512 x 512 e ela girada 10 graus sexagesimais.



Fig.2.13. Correspondências entre duas imagens usando o algoritmo SIFT.



Fig.2.14. Correspondências depois de usar o algoritmo para eliminar as correspondências erradas.

# Capítulo 3

## Testando a invariância e uso da Matriz de Homografia

### 3.1 Introdução

Neste capítulo será testado o algoritmo proposto e será comparada sua invariância em escala, rotação, ruído gaussiano e brilho com outros dois métodos de detecção de características chamados SURF e RANSAC. Além disso, será explicado o uso da Matriz de Homografia para poder saber que transformação aconteceu.

### 3.2 Extração de características usando o algoritmo SIFT

Será usada a imagem Lena com 240 x 320 (Ver Fig. 3.1) para testar o espaço de escala. O espaço e escala da imagem é mostrado em Fig. 3.2, Fig. 3.3, Fig. 3.4 e Fig. 3.5.



Fig. 3.1. Imagem de teste Lena.

Para a criação do espaço de escala foi utilizado 4 oitavas e 3 escalas, conforme recomendação de Lowe. Seja ‘o’ a oitava e ‘s’ a escala da imagem, com sigma inicial igual a 1,6. O valor da constante multiplicadora  $k$  do espaço de escalas é calculado usando a seguinte equação[2]:

$$k = 2^{1/s} \quad (3.1)$$

Para esse caso o valor de  $k = 2^{1/3} = 1.2599$ , então os sigmas para a primeira oitava serão:

Tabela 3.1: Sigmas na primeira oitava.

Escala	Sigma
-1	1,6
0	2,016
1	2,539
2	3,2
3	4,0317

Para a oitava seguinte o valor inicial de sigma será duas vezes o valor do sigma inicial da oitava anterior. O valor do sigma inicial na 2ª oitava será  $2 \times 1,6 = 3,2$ .

Tabela 3.2: Sigmas na 2<sup>a</sup> oitava.

Escala	Sigma
-1	3,2
0	4,032
1	5,079
2	6,4
3	8,0635

O mesmo procedimento será realizado até cobrir todas as oitavas que nesse caso são 4 oitavas.



Fig. 3.2. Oitava 1.



Fig. 3.3. Oitava 2.

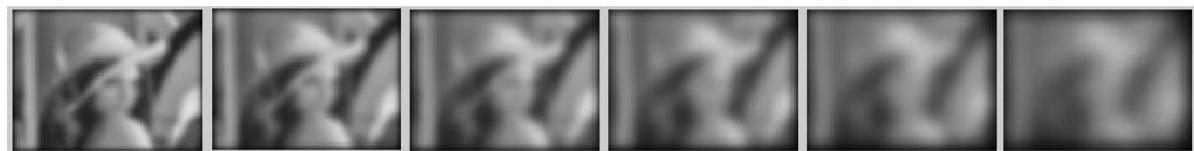


Fig. 3.4. Oitava 3.

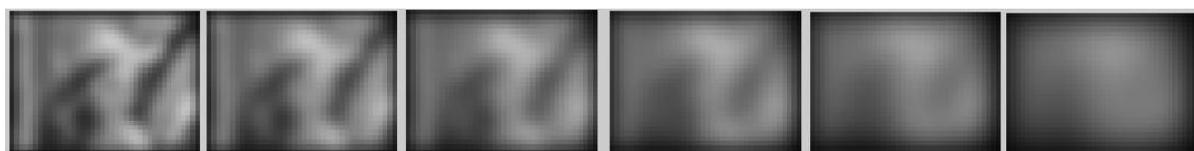


Fig. 3.5. Oitava 4.

A Fig.3.6 é apresentada a *DoG* usando a pirâmide gaussiana formada pelas 4 oitavas anteriores. Como cada oitava tem 6 imagens, a *DoG* terá 5 imagens por cada oitava.

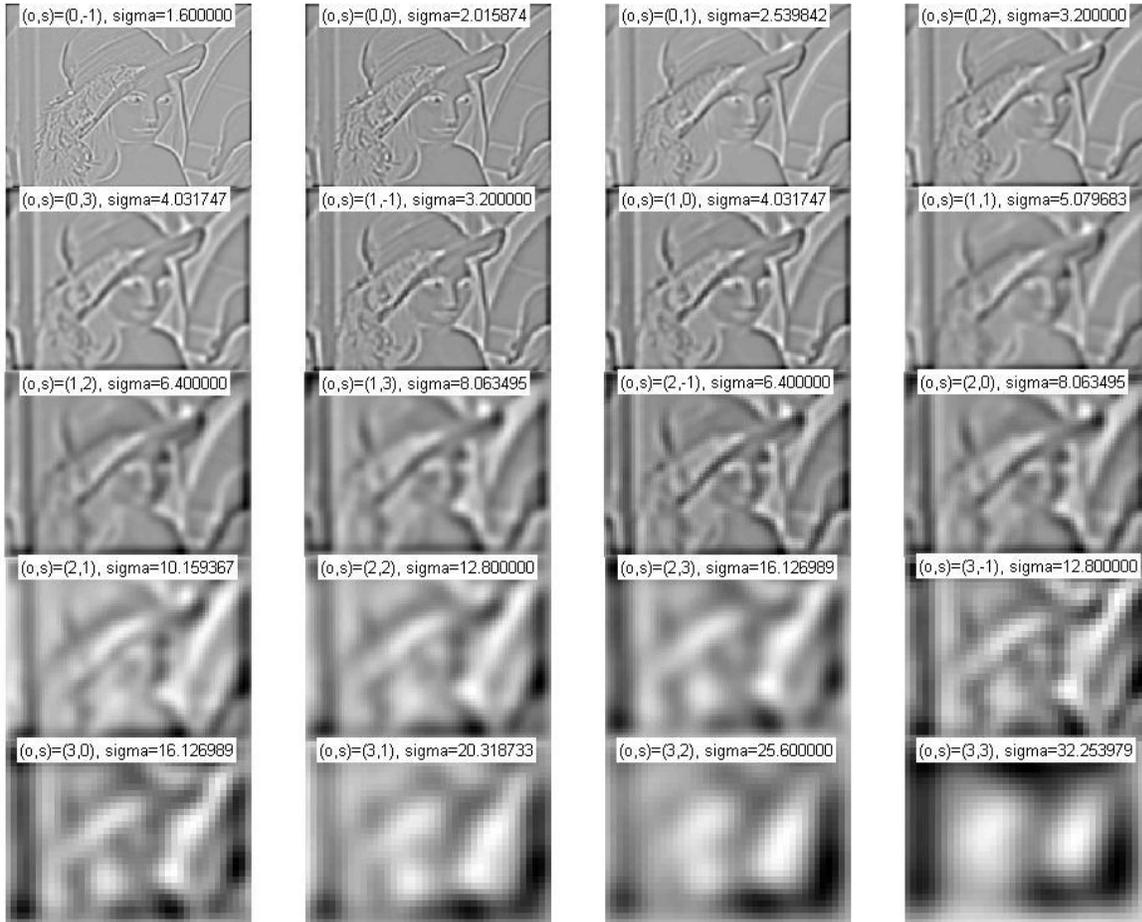


Fig. 3.6. Diferença de Gaussiano da imagem Lena.

Para calcular os pontos máximos e mínimos em cada oitava, serão escolhidas de 3 em 3 imagens do *DoG*. Nesse caso, os máximos e mínimos em cada oitava serão definidos por 3 imagens. Em seguida serão mostrados todos os pontos em uma única imagem, conforme figuras de 3.7 a 3.9.

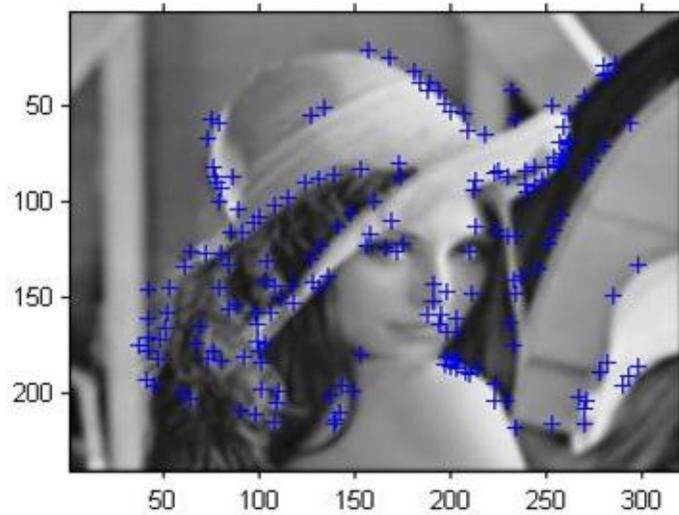


Fig. 3.7. Máximos e mínimos na primeira oitava.

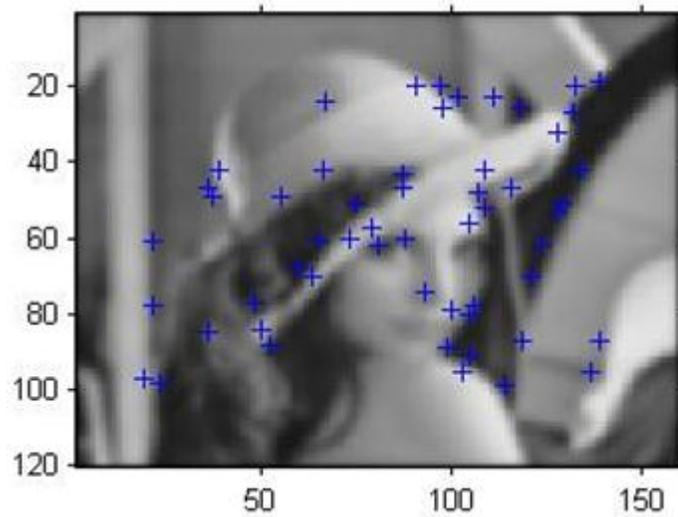


Fig. 3.8. Máximos e mínimos na segunda oitava.

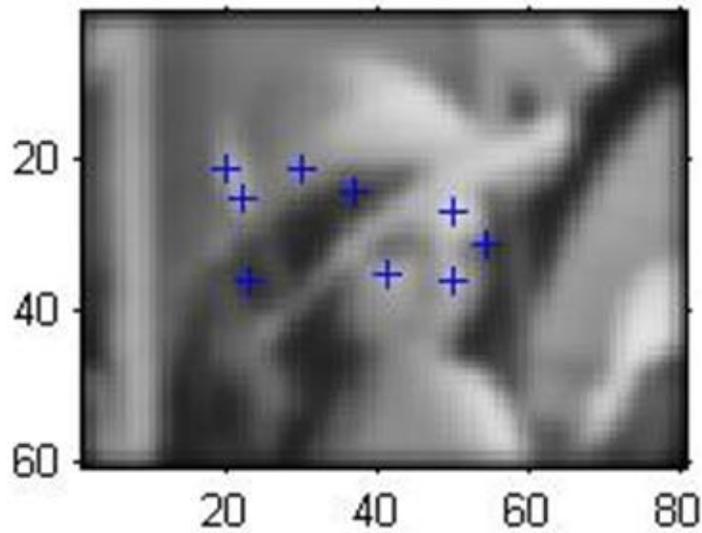


Fig. 3.9. Máximos e mínimos na terceira oitava.

Depois de se obter os máximos e mínimos em todas as oitavas, também foram obtidas as escalas e localizações de todos os pontos chave. Seguindo os passos do algoritmo SIFT explicado no Capítulo 2, serão calculados os pontos chave com suas orientações. O resultado é mostrado na Fig.3.10.

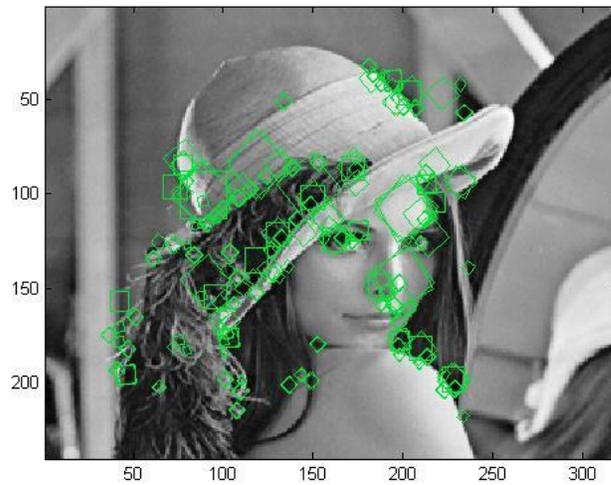


Fig. 3.10. Características da imagem Lena 240 x 320.

### 3.3 Testando o algoritmo SIFT para comparar características entre duas imagens

Será usada a imagem Lena de  $512 \times 512$  e a rotação em 10 graus sexagesimais para comparar as características. Na Fig. 3.11 e Fig. 3.12 podem ser observadas as características da imagem Lena e ela girada 10 graus sexagesimais, respectivamente.

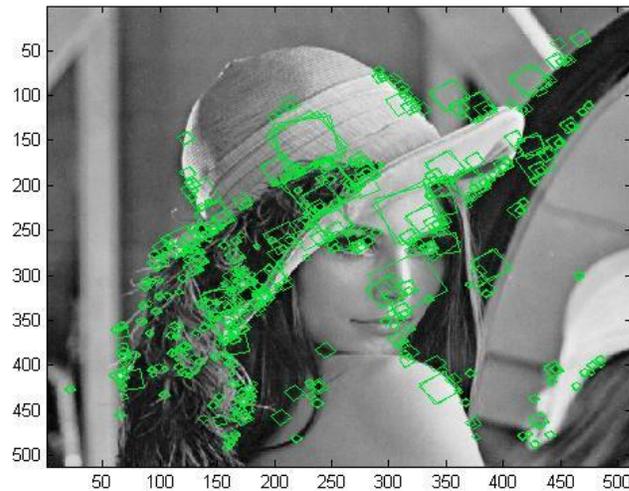


Fig. 3.11. Características da imagem Lena  $512 \times 512$ .

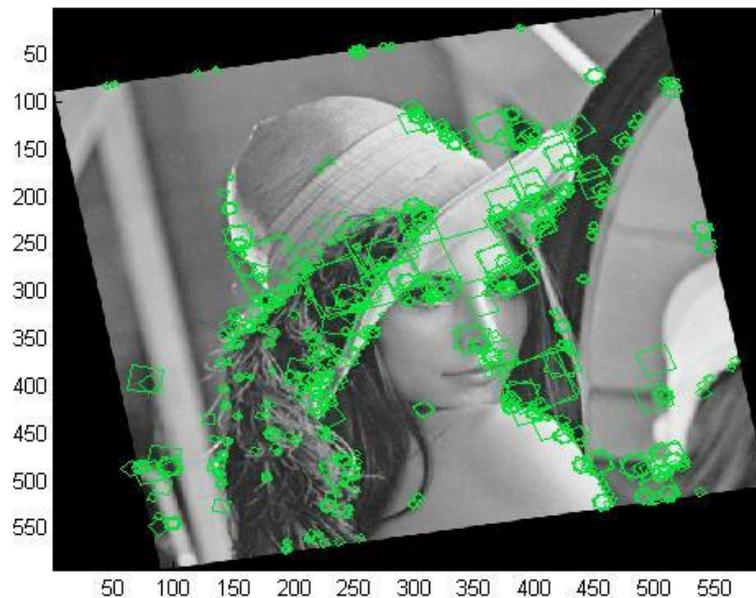


Fig. 3.12. Características da imagem Lena girada 10 graus.

Usando o algoritmo SIFT são obtidas as correspondências entre as duas imagens e o resultado é mostrado em Fig. 3.13.

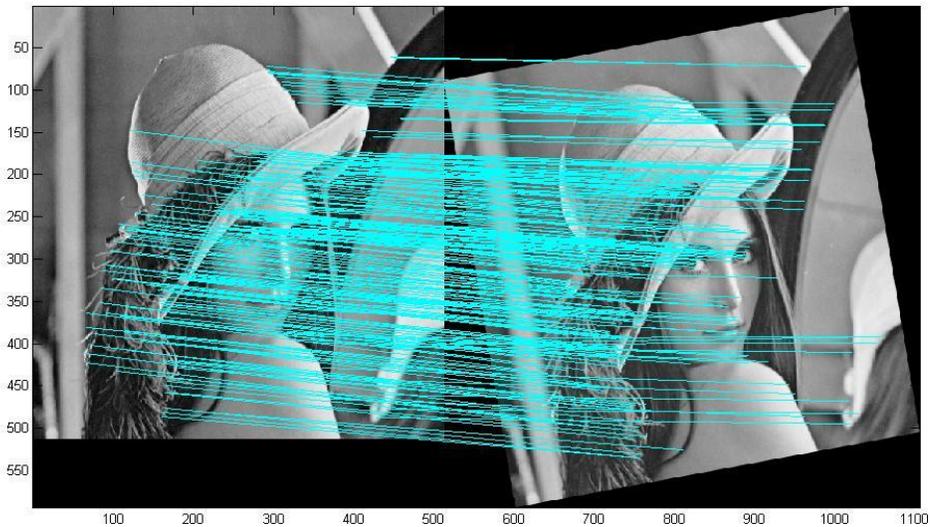


Fig. 3.13. Correspondências das duas imagens anteriores.

Se a imagem original Lena é girada em 60 graus e são calculadas as correspondências, como observado em Fig. 3.14, pode-se claramente perceber as correspondências errôneas.

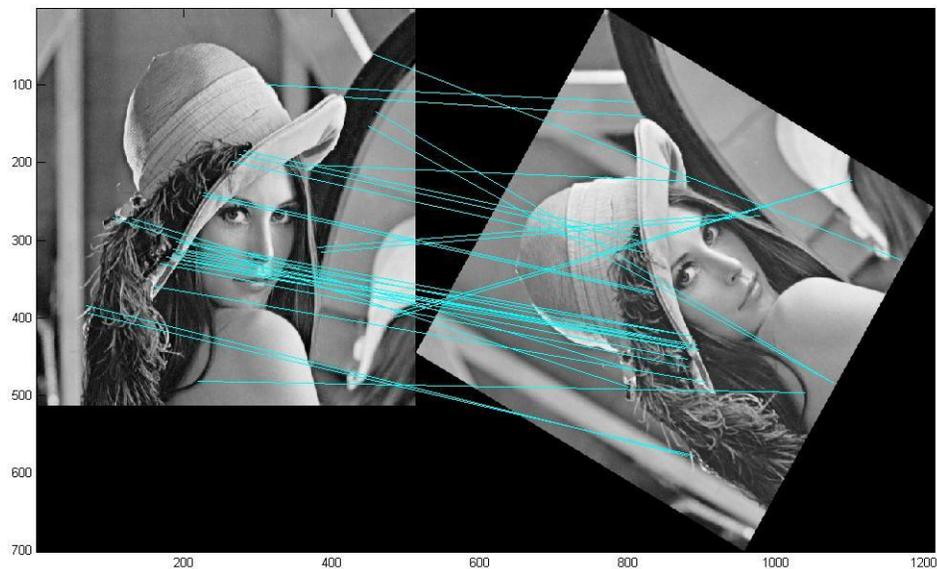


Fig. 3.14. Correspondências das duas imagens giradas 60 graus sexagesimais.

O algoritmo explicado na Seção 2.5, tem a tarefa de eliminar as correspondências errôneas. Aplicando esse algoritmo, obtém-se o resultado mostrado na Fig. 3.15 (foram eliminadas as correspondências errôneas).



Fig. 3.15. Eliminação das correspondências errôneas pela aplicação do algoritmo proposto.

Em seguida, é mostrado um exemplo de duas fotos que foram tiradas de ângulos diferentes, representando um caso mais real já que a segunda imagem não foi obtida por computador como nos casos anteriores.

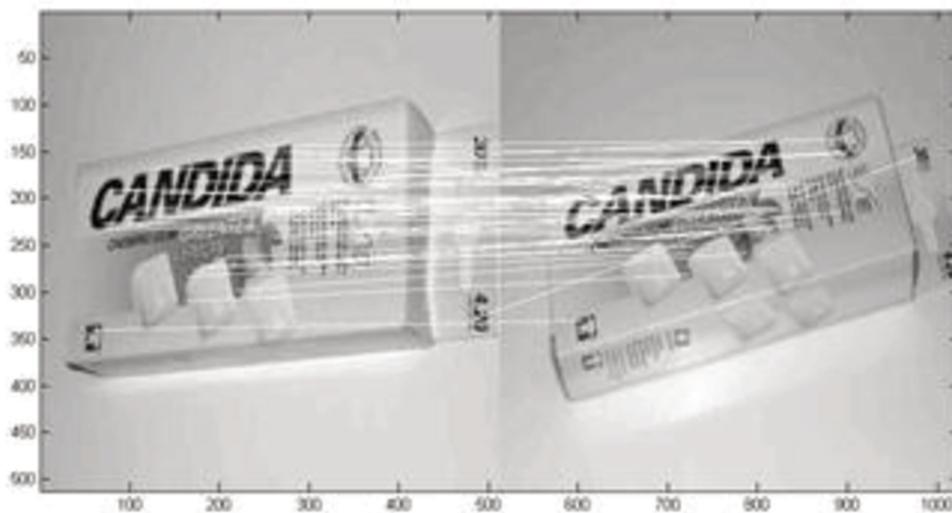


Fig. 3.16. Correspondências de duas imagens que contem o mesmo objeto.

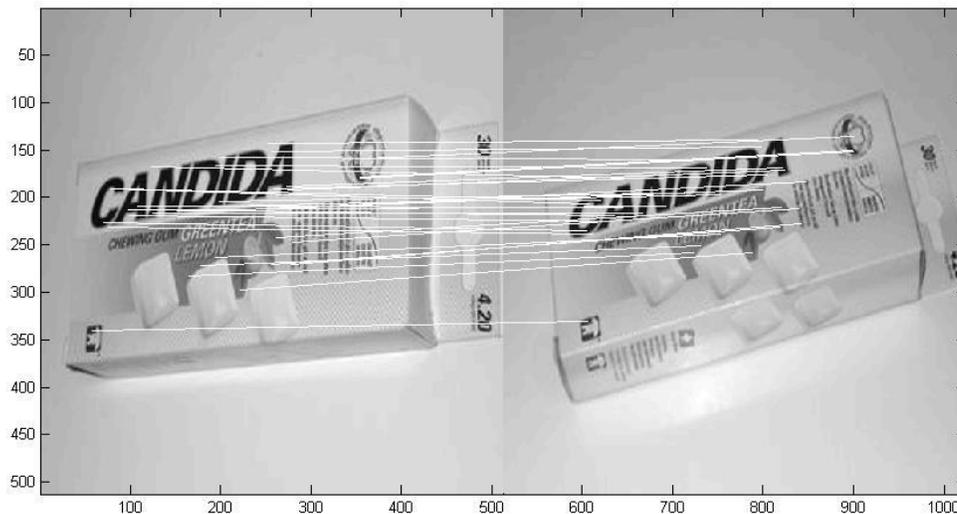


Fig. 3.17. Eliminação das correspondências errôneas.

### 3.4 Testando para variações em escala, orientação, ruído gaussiano e brilho

Para testes com o algoritmo SIFT, SIFT com pós-processamento, SURF e RANSAC, a escala, rotação e brilho das imagens são variados. Foram usadas 20 imagens [27] para todos os algoritmos, todas as imagens de resolução 512 x 512.

Seja %P a porcentagem de correspondências corretas sobre todas as correspondências, ou também chamado precisão. Para todos os testes, a quantidade de correspondências foi constante usando todos os algoritmos.

#### 3.4.1 Variações em escala

Foi variada a escala da imagem Lena alterando o tamanho dos eixos  $x$  e  $y$  de 100% a 300% de seu valor, com intervalos de 20%. A função de *Matlab* usada para essas variações foi o *imresize*, o método utilizado foi o *bilinear*. Para variar a escala  $E$  serão variados os eixos  $x$  e  $y$ . Se  $E$  é igual a 1,6 então os novos  $x'$  e  $y'$  serão  $x'=1,6x$  e  $y'=1,6y$ , conforme mostrado na Fig. 3.18. Podemos observar na Fig. 3.19 que o algoritmo SIFT com pós-processamento é mais eficiente que o SIFT e RANSAC.



Fig. 3.18 Correspondências para imagens de diferente escala.

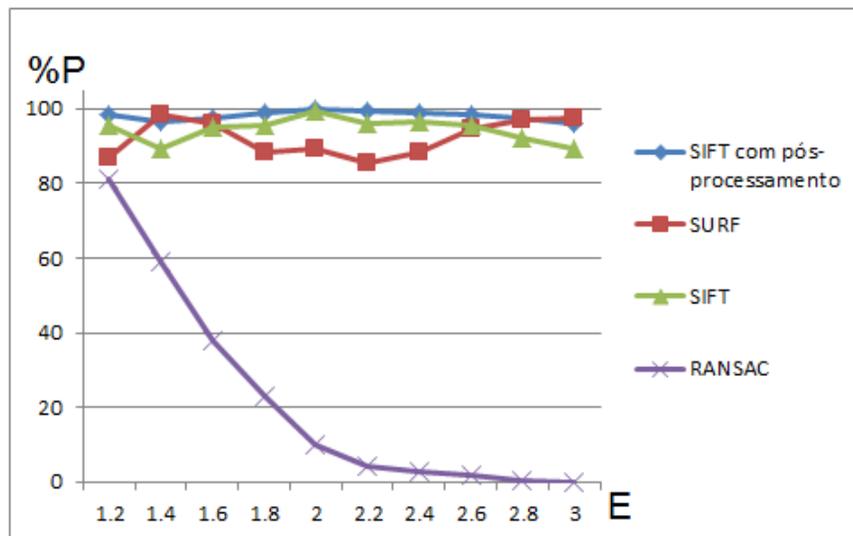


Fig. 3.19 Resultados de testes para duas imagens,  $E$  é a escala que serão variados os eixos  $x$  e  $y$  da imagem.

### 3.4.2 Variações em rotação

A imagem de teste foi rotacionada de 10 em 10 graus e comparada com a original até chegar a 50 graus de rotação. Foi escolhido o valor final de 50 graus porque acima desse valor a quantidade de correspondências foi zero, dificultando a comparação com outros métodos a partir desse limiar. A Fig. 3.20 mostra as correspondências com a imagem rotacionada. Pode-se

observar a partir dos resultados da Fig. 3.21, que o algoritmo SIFT com pós-processamento é mais eficiente que o SIFT, SURF e RANSAC.

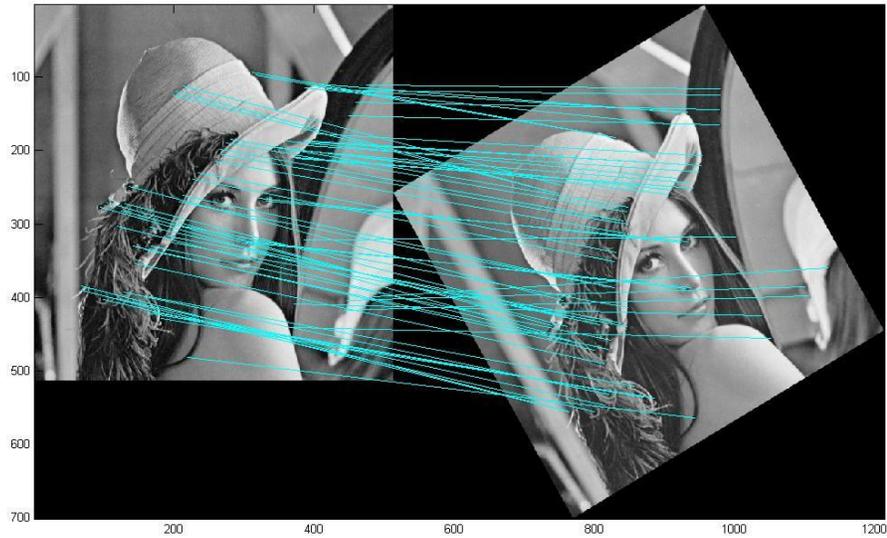


Fig. 3.20 Correspondências para imagens de ângulos diferentes.

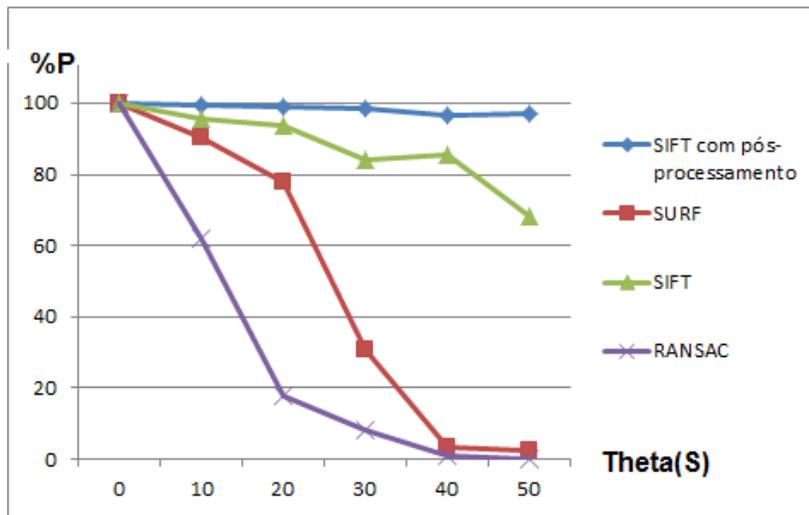


Fig. 3.21 Resultados para duas imagens, sendo a transformação da segunda imagem uma rotação de 0 a 50 graus sexagesimais em relação à primeira.

### 3.4.3 Variações em ruído gaussiano

Foram realizados testes adicionando ruído gaussiano tipo ‘blur’ usando a função Matlab  $H = fspecial('gaussian', hsize, sigma)$ , que retorna um filtro passa baixas Gaussiano rotacionalmente simétrico de tamanho  $hsize$ , com desvio padrão  $sigma$  (positivo). O valor de  $hsize$  pode ser um vetor especificando o número de linhas e colunas de  $H$  ou um escalar, caso em que  $H$  é uma matriz quadrada. Para os testes foi usado  $hsize = 9 * sigma$ , a Fig. 3.22 mostra as correspondências para uma imagem com ruído gaussiano. Pode-se observar a partir dos resultados mostrados na Fig. 3.23 que o algoritmo SIFT com pós-processamento teve o melhor desempenho.

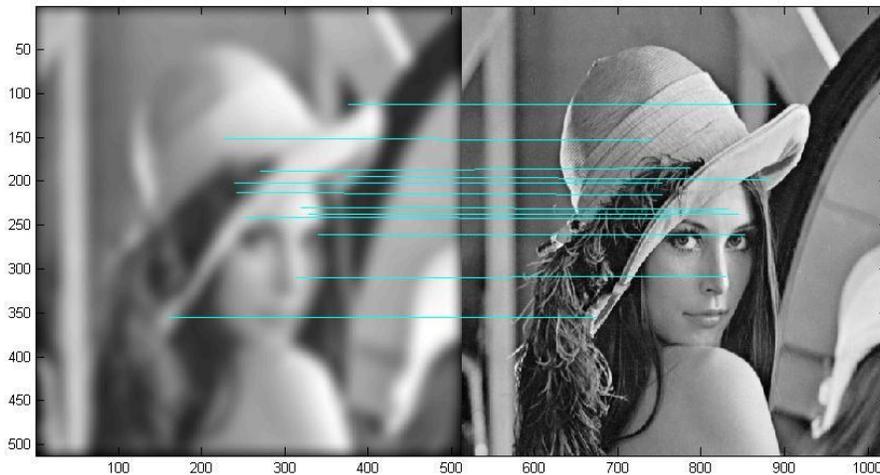


Fig. 3.22 Correspondências para uma imagem com ruído gaussiano.

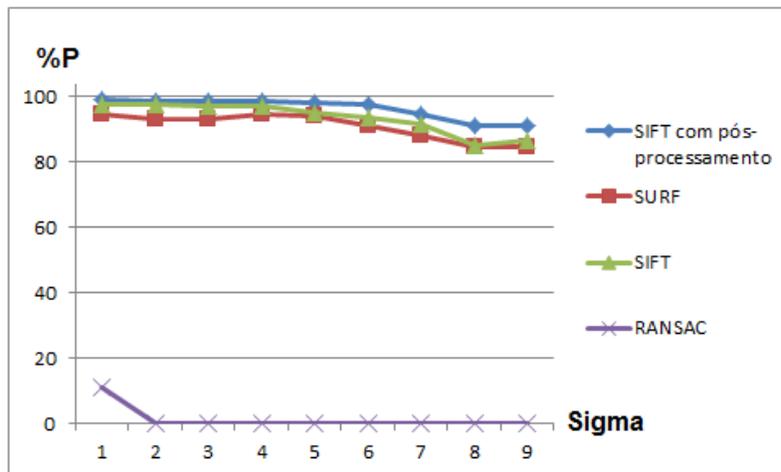


Fig. 3.23. Resultados para duas imagens, sendo a transformação a adição de ruído gaussiano na 2ª imagem em relação à primeira.

### 3.4.4 Variações em brilho

As imagens em escala cinza e tipo *uint8* variam de 0 a 255 o valor de cada pixel. Para variar o brilho, foi somada uma intensidade representada aqui pela variável  $B$ . A Fig 3.24 mostra um exemplo de correspondências para uma imagem com variação de brilho. A partir da Fig. 3.25 pode-se observar que o algoritmo SIFT com pós-processamento teve o melhor desempenho.

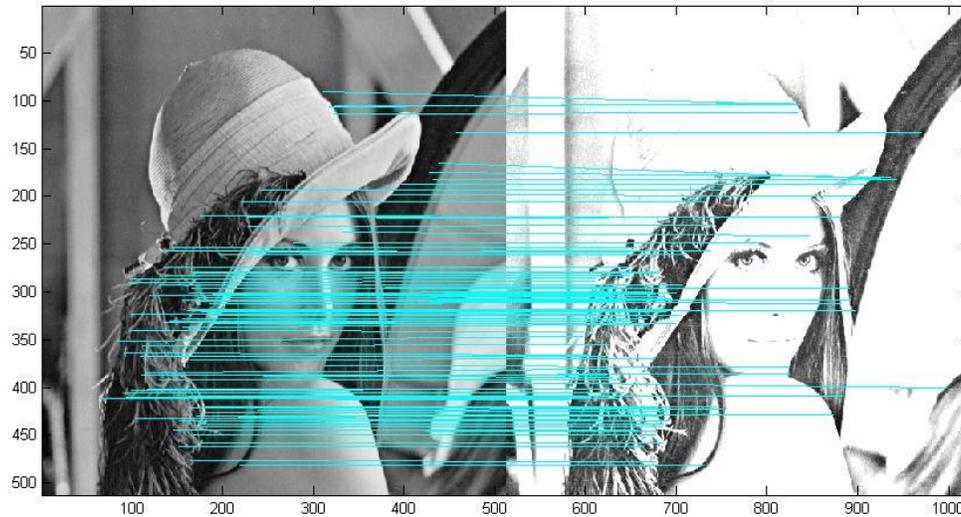


Fig. 3.24. Correspondências para uma imagem com brilho.

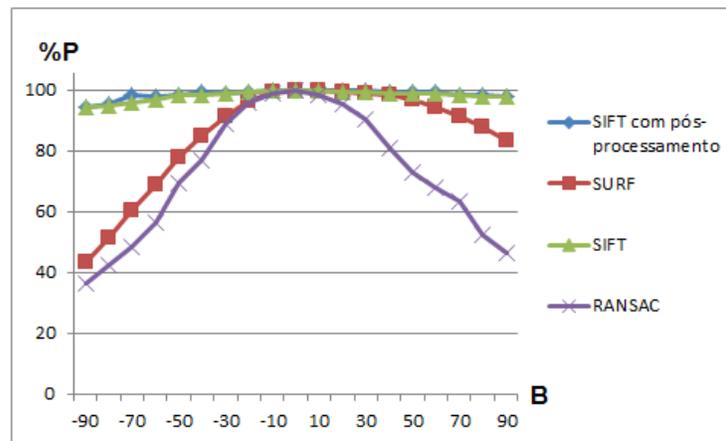


Fig. 3.25. A figura mostra os resultados para duas imagens, sendo a transformação de adição de brilho na segunda imagem com respeito à primeira.

### 3.5 Matriz de Homografia

Dado um conjunto de pontos  $p_i$  e um conjunto de pontos  $p'_i$ , se quer calcular a matriz que faz com que a transformação projetiva de  $p'_i$  coincida com  $p_i$ . A matriz não singular 3x3 que permite a correspondência entre os pontos das duas imagens é a matriz de homografia  $H$ . A relação é definida da seguinte forma.

$$p'_i = H \cdot p_i \quad (3.2)$$

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \quad (3.3)$$

$$\begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad (3.4)$$

$$X'_i = \frac{x'_i}{z'_i}, Y'_i = \frac{y'_i}{z'_i}$$

A primeira questão é considerar quantos pontos são necessários para se estimar a matriz  $H$ . Cada correspondência de pontos fixa duas restrições para a matriz  $H$ . A matriz  $H$  tem 8 parâmetros independentes, então serão necessárias 8 restrições que serão resultado de 4 pontos de correspondência. Para o cálculo da matriz de Homografia, é usado o algoritmo DLT normalizado [28].

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3.5)$$

$$u = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \quad (3.6)$$

$$v = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \quad (3.7)$$

$$\begin{cases} u_i x_i \cdot h_{31} + u_i y_i \cdot h_{32} + u_i = x_i \cdot h_{11} + y_i \cdot h_{12} + h_{13} \\ v_i x_i \cdot h_{31} + v_i y_i \cdot h_{32} + v_i = x_i \cdot h_{21} + y_i \cdot h_{22} + h_{23} \end{cases} \quad (3.8)$$

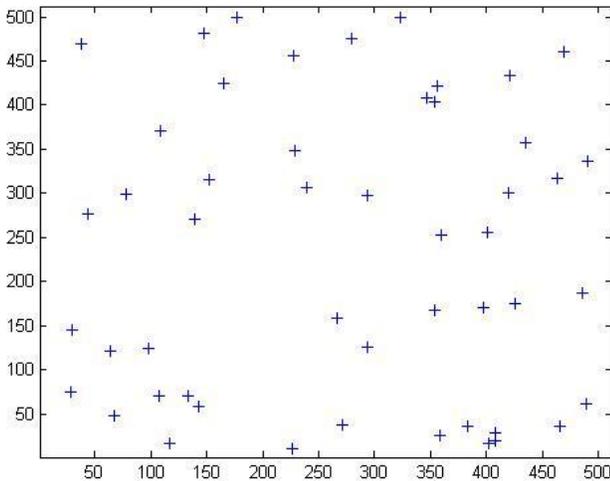
$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -u_1 x_1 & -u_1 y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -v_1 x_1 & -v_1 y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -u_2 x_2 & -u_2 y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -v_2 x_2 & -v_2 y_2 \\ \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -u_n x_n & -u_n y_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -v_n x_n & -v_n y_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \Leftrightarrow Ax = b \quad (3.9)$$

$$x = (A^T A)^{-1} A^T b \quad (3.10)$$

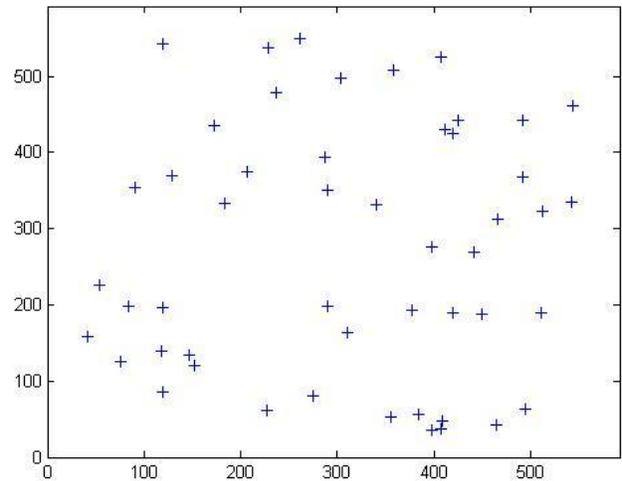
### 3.6 Eleição de pontos para calcular a Matriz de Homografia

São criados  $N$  pontos aleatórios em um espaço de  $L \times L$ . Esses pontos serão somados a um valor aleatório com valor máximo 3, logo os novos pontos serão girados em um ângulo  $\theta$ .

Para  $N = 50, L = 512$  e  $\theta = 10$  graus.



(a)



(b)

Fig. 3.26. (a) Imagem A. (b) Imagem A com ruído gaussiano adicionado e rotação de 10 graus.

Serão escolhidos 4 pontos que fiquem afastados um do outro, assim como mostrado em Fig. 3.27. A partir dos 4 pontos, será obtida a matriz de homografia e assim será possível saber quais transformações todos os pontos sofreram.

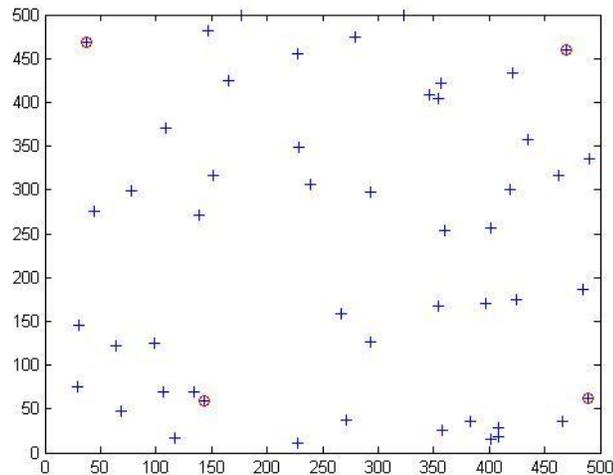


Fig. 3.27. Os quatro pontos escolhidos são representados por um círculo vermelho.

Será criada uma malha de pontos para logo transformá-los segundo a matriz de homografia calculada pelos 4 pontos escolhidos.

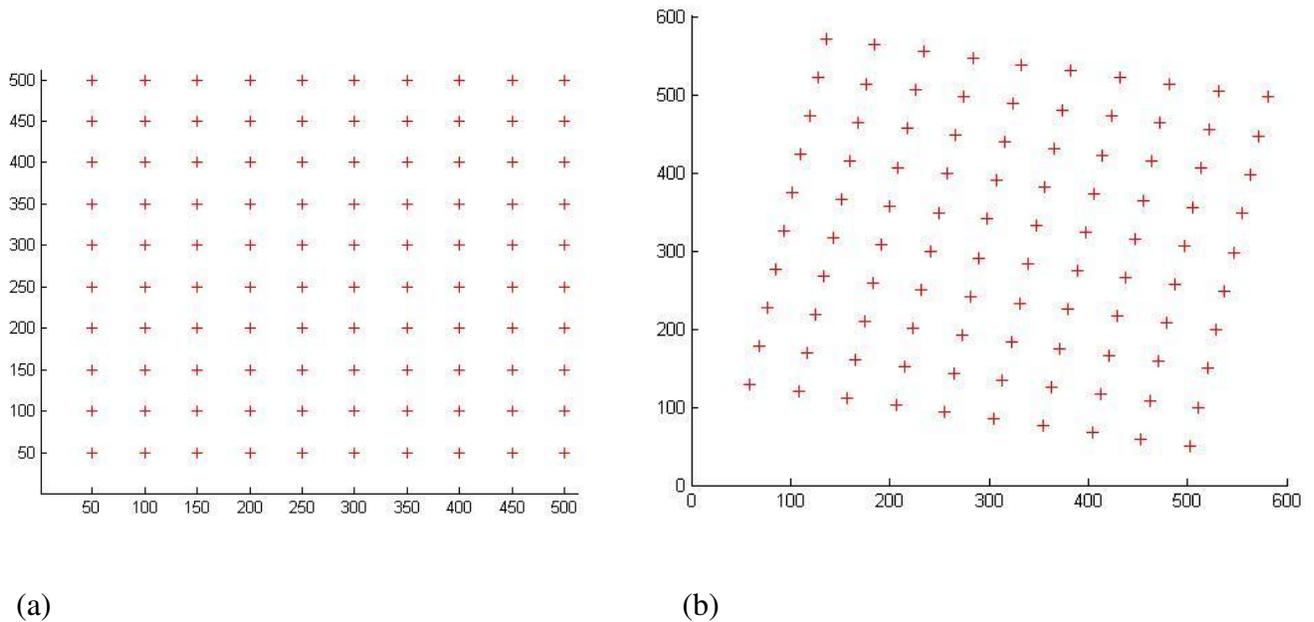


Fig. 3.28. (a) Uma malha de 10x10 pontos é criada. (b) A imagem mostra a transformação que sofre os pontos de (a) usando a matriz de homografia.

Será calculada a posição real dos pontos sabendo que foi rotacionada 10 graus. Com isso, pode-se calcular o erro quadrático médio dos 100 pontos. Para se gerar o gráfico apresentado na Fig. 3.29, foi usada a média dos erros quadráticos médios de 10 variações.

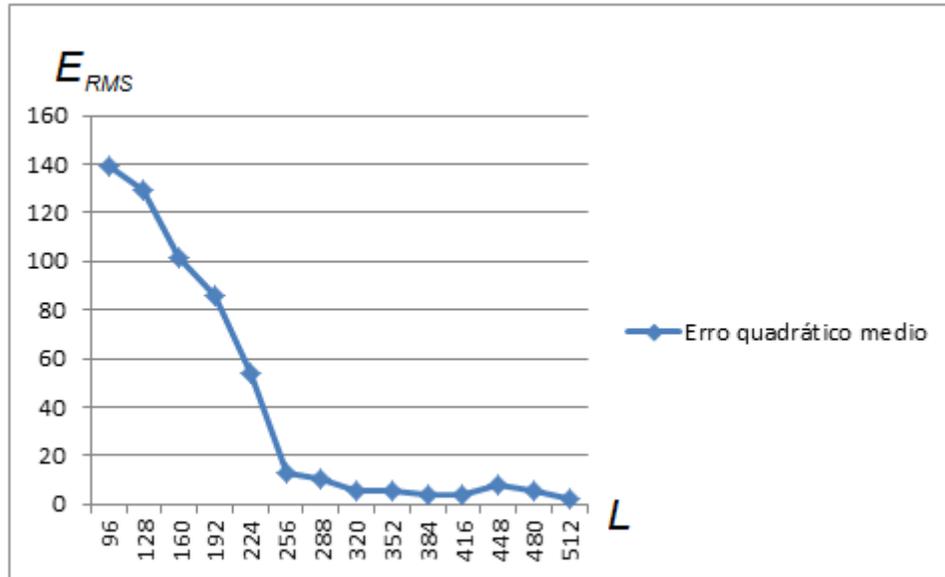


Fig. 3.29. A figura mostra os erros quadráticos médio para diferentes valores de  $L$ .

Na Fig.3.29 pode-se observar que a partir de  $L$  maior que 256 o  $E_{RMS}$  tem um valor baixo, isso significa que enquanto os 4 pontos escolhidos para calcular a matriz de homografia encontram-se mais afastados uns dos outros o  $E_{RMS}$  é menor.

Na Tabela 3.3, enquanto mais espaçados estão os pontos, é menor o erro ao utilizar a matriz de homografia para calcular os pontos originais. As imagens utilizadas para se determinar as invariâncias definidas na Seção 2.2 foram imagens de resolução 512 x 512.

Será aplicado um algoritmo para calcular 4 pontos que estejam espaçados na maior distância possível uma de outra, para assim diminuir o  $E_{RMS}$  no momento de escolher os quatro pontos da matriz de homografia.

Em seguida, será mostrada a transformação que a malha de pontos sofreu usando 4 pontos que ficam próximos.

Tabela 3.3: Valores de  $E_{rms}$  da Fig. 3.29.

L	Erms
32	9705,1
64	3532,3
96	139,31
128	128,99
160	101,83
192	85,816
224	54,04
256	12,835
288	10,302
320	5,5223
352	5,592
384	4,0118
416	3,866
448	7,9554
480	5,282
512	2,4622

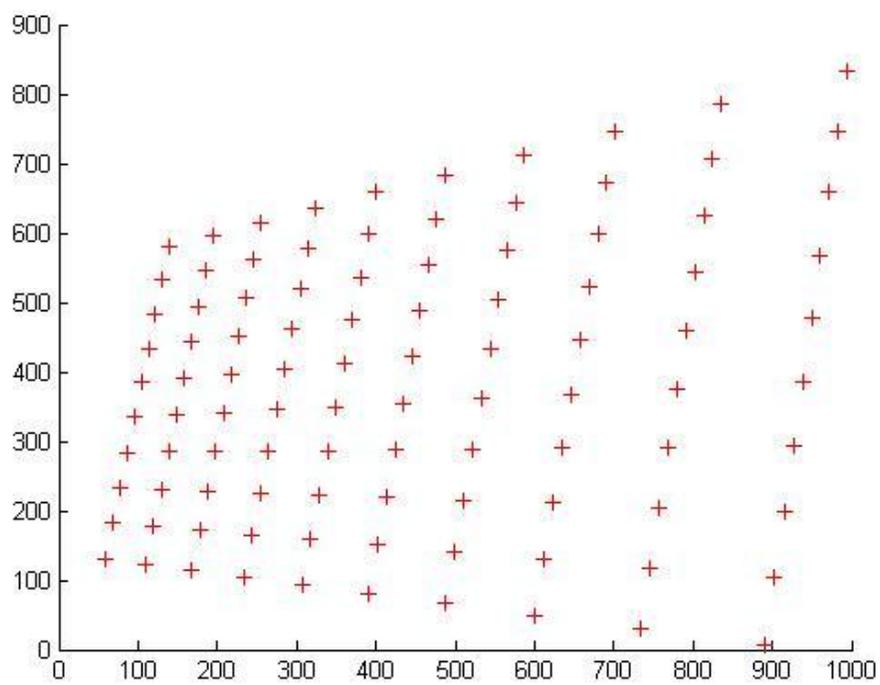


Fig. 3.30. Malha de pontos obtida pela matriz de homografia.

O valor do  $E_{RMS}$  dos 100 pontos foi:

$$E_{RMS} = 212,0427$$

Na Fig. 3.30 pode-se observar que não é eficiente rotacionar em 10 graus a malha de pontos da Fig. 3.28.

A seguir é apresentado um exemplo usando a imagem Lena 512x512 rotacionada de 30 graus. A Fig. 3.31 mostra os pontos do algoritmo SIFT com pós-processamento. Serão escolhidos 4 pontos que ficam o mais afastado possível um do outro assim como é observado em Fig. 3.32.

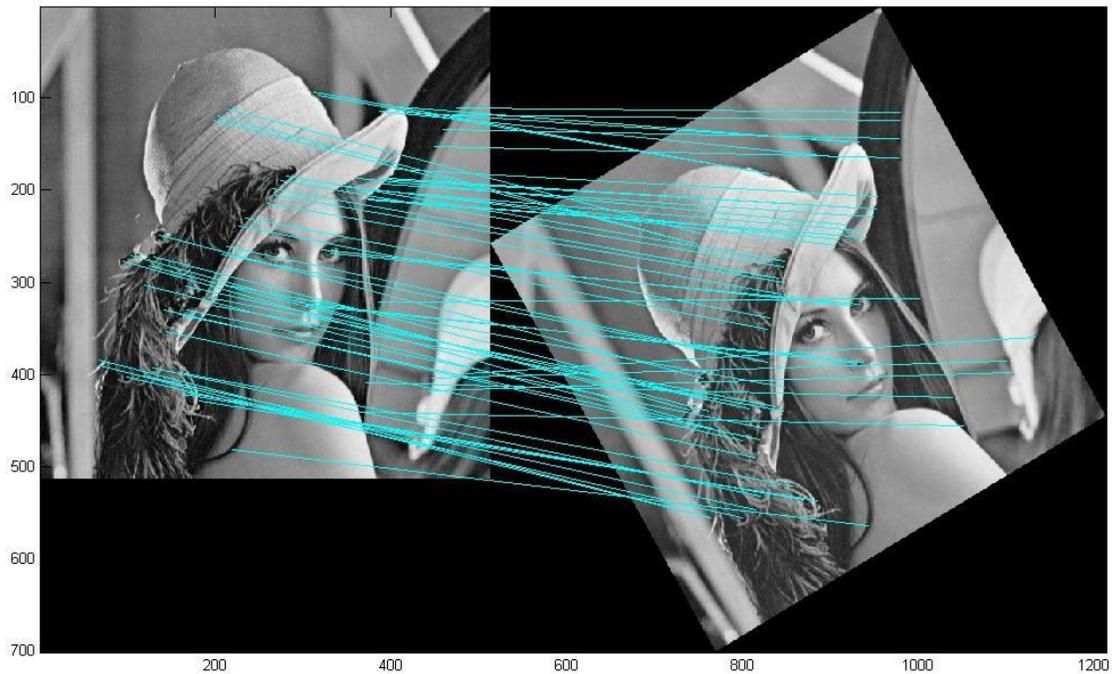


Fig. 3.31. Correspondências usando o algoritmo SIFT com pós-processamento.

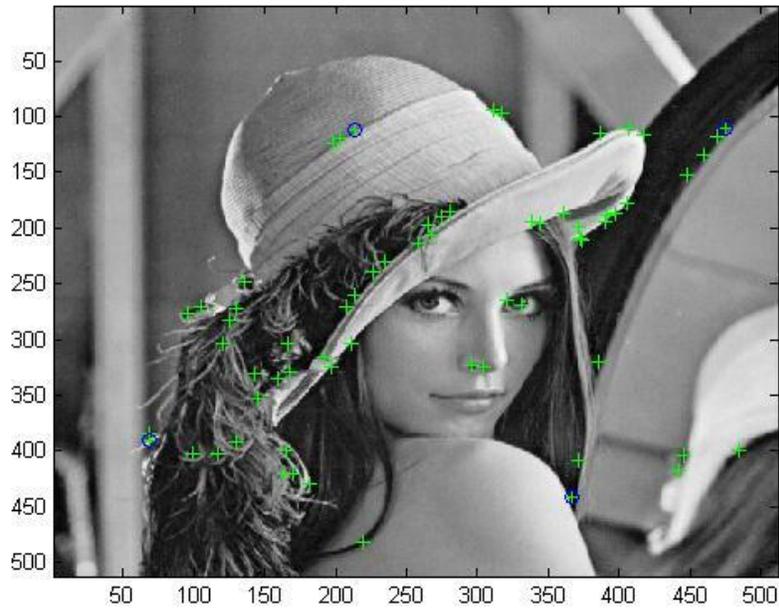


Fig. 3.32. Os 4 pontos escolhidos para calcular a matriz da homografia é mostrado por um círculo azul.

Uma malha de pontos é criada sobre a imagem original (Ver Fig. 3.33). Usando a matriz de homografia será calculada a nova posição dos pontos e o resultado é apresentado na Fig. 3.34.

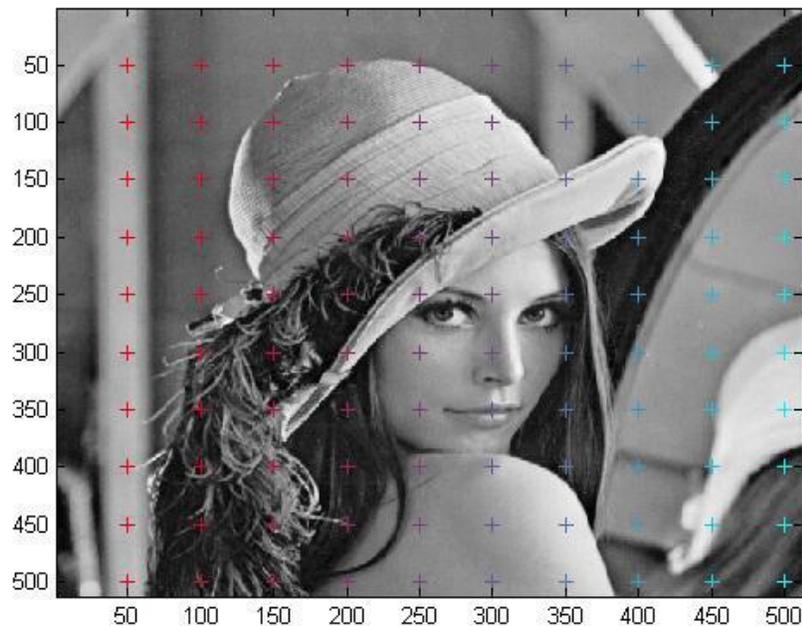


Fig. 3.33. Uma malha de pontos é criada na imagem.

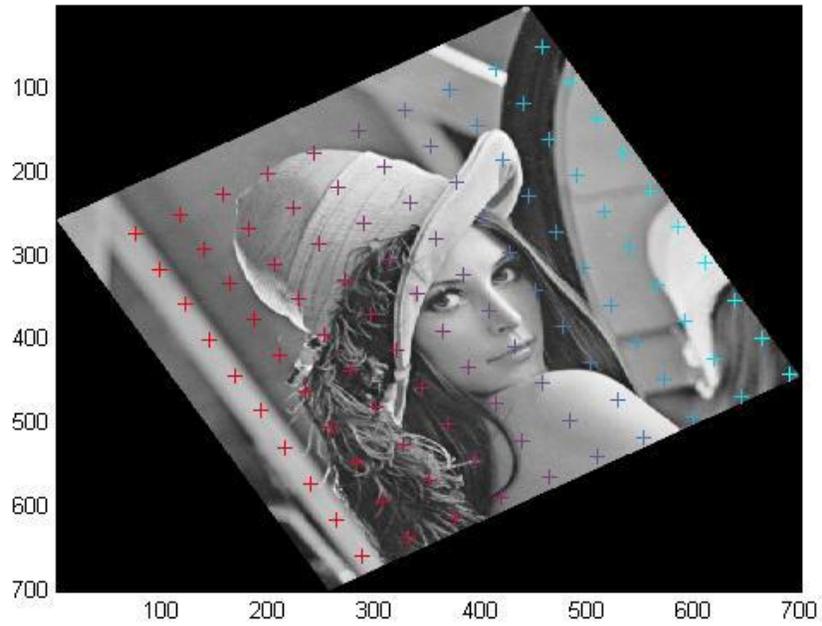


Fig. 3.34. Usando a matriz de homografia são obtidos os pontos resultantes.

O erro quadrático médio para os 100 pontos criados foi:

$$E_{RMS} = 3,440$$

# Capítulo 4

## Uso do SIFT proposto sobre placas de circuitos impressos

### 4.1 Introdução

Foram recebidas fotos de problemas em placas de circuito impresso da empresa CadService, que tinha em montagem de placas a má colocação de componentes (um exemplo é visualizado em Fig. 4.1). Foi testado nosso algoritmo em 4 objetos diferentes e em 11 fotos para cada objeto sendo a primeira a imagem em posição correta. Como as únicas variações nesses casos foram de ubiquação, rotação e escala, a matriz de homografia será aproximada a uma matriz cujos elementos sejam o valor da escala em  $x$  e em  $y$ , rotação e translação. Será detectada a falha por rotação de componente, se o componente esta rotacionado um ângulo maior de 10 graus sexagesimais o componente esta colocado erroneamente, caso contrario o componente esta colocado corretamente.

### 4.2 Aproximação da matriz de homografia

Seja a matriz de homografia definida na equação (4.3), que será aproximada a uma matriz que é resultado de multiplicar uma matriz escala nos eixos  $x$  e  $y$ , com uma matriz que tem as componentes de rotação e translação.

$$H = \begin{bmatrix} -f_x & 0 & 0 \\ 0 & -f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \quad (4.1)$$

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} -f_x & 0 & 0 \\ 0 & -f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad (4.2)$$

$$s\tilde{m} = A[R \ t]\tilde{M} \quad (4.3)$$

$$H = \begin{bmatrix} -f_x r_{11} & -f_x r_{12} & -f_x T_x \\ -f_y r_{21} & -f_y r_{22} & -f_y T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \quad (4.4)$$

$$H = \begin{bmatrix} -f_x \cos \theta & f_x \sin \theta & -f_x T_x \\ -f_y \sin \theta & -f_y \cos \theta & -f_y T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \quad (4.5)$$

Da equação (4.3) e (4.4):

$$-f_x \cos \theta = h_{11} \quad (4.6)$$

$$f_x \sin \theta = h_{12} \quad (4.7)$$

$$-f_y \sin \theta = h_{21} \quad (4.8)$$

$$-f_y \cos \theta = h_{22} \quad (4.9)$$

São 4 equações e 3 incógnitas, é calculado o ângulo usando as equações (4.6) e (4.7) e também é calculado o ângulo usando as equações (4.8) e (4.9).

$$\theta_1 = \tan^{-1}\left(-\frac{h_{12}}{h_{11}}\right) \quad (4.10)$$

$$\theta_2 = \tan^{-1}\left(\frac{h_{21}}{h_{22}}\right) \quad (4.11)$$

O ângulo que o objeto girou é obtido como uma média dos dois ângulos obtidos da matriz de homografia, conforme mostrado nas equações (4.10) e (4.11).

$$\theta = \theta_1 + \theta_2 \quad (4.12)$$

Usando as equações (4.6) e (4.7) são calculados os valores de  $f_x$ .

$$f_{x1} = -h_{11}/\cos \theta$$

$$f_{x2} = h_{12}/\sin \theta$$

$$f_x = \sqrt{f_{x1} \cdot f_{x2}} \quad (4.13)$$

Usando as equações (4.8) e (4.9) são calculados os valores de  $f_y$ .

$$f_{y1} = -h_{22}/\cos \theta$$

$$f_{y2} = -h_{21}/\sin \theta$$

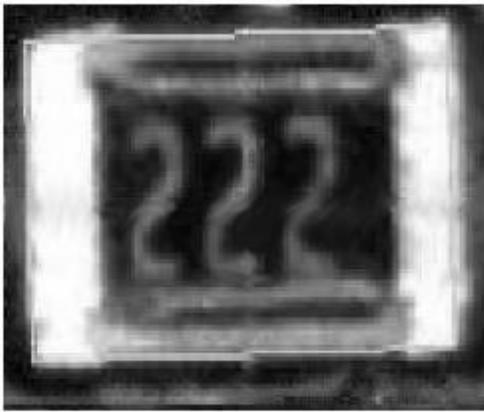
$$f_y = \sqrt{f_{y1} \cdot f_{y2}} \quad (4.14)$$

### 4.3 Testes usando o algoritmo SIFT com pós-processamento

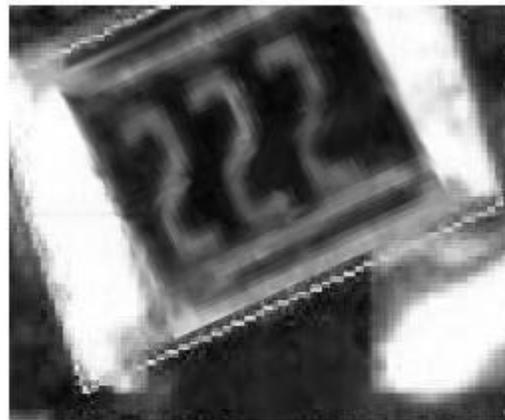
Será mostrada a aplicação do algoritmo SIFT com pós-processamento em um exemplo de colocação errônea de componente e em colocação correta de componente.

### 4.3.1 Exemplo de ubiquação errônea de componente

A seguir será mostrado um exemplo onde existe o problema de má ubiquação de componentes, onde o segundo componente está rotacionado em um determinado ângulo com o qual a placa impressa deveria passar por revisão.



(a)



(b)

Fig. 4.1.(a) Imagem com o objeto na posição correta. (b) Imagem com o objeto na posição errada.

Na Fig. 4.1(a) é mostrada uma imagem de boa ubiquação de componente, a partir dessa imagem será calculado em outras imagens se o componente está rotacionado ou deslocado, para assim saber se a placa impressa precisa de correção. Será usado o algoritmo SIFT nas imagens da Fig. 4.1(a) e Fig. 4.1(b), para identificar o objeto na Fig. 4.1(b). As correspondências são mostradas na Fig. 4.2.

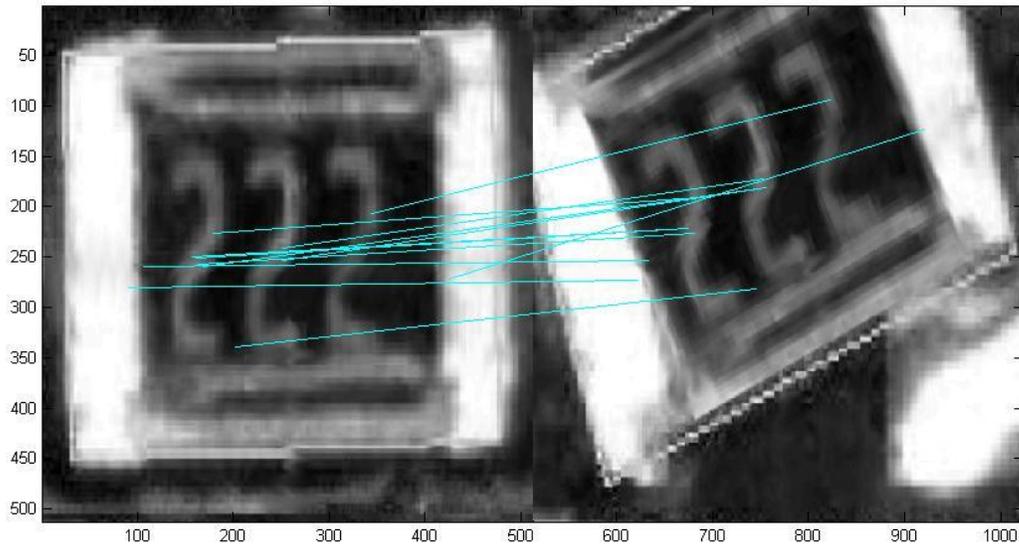


Fig. 4.2. Usando o algoritmo SIFT para detectar o objeto.

O algoritmo de pós-processamento é aplicado às correspondências obtidas em Fig. 4.2. , assim serão eliminadas as correspondências errôneas.

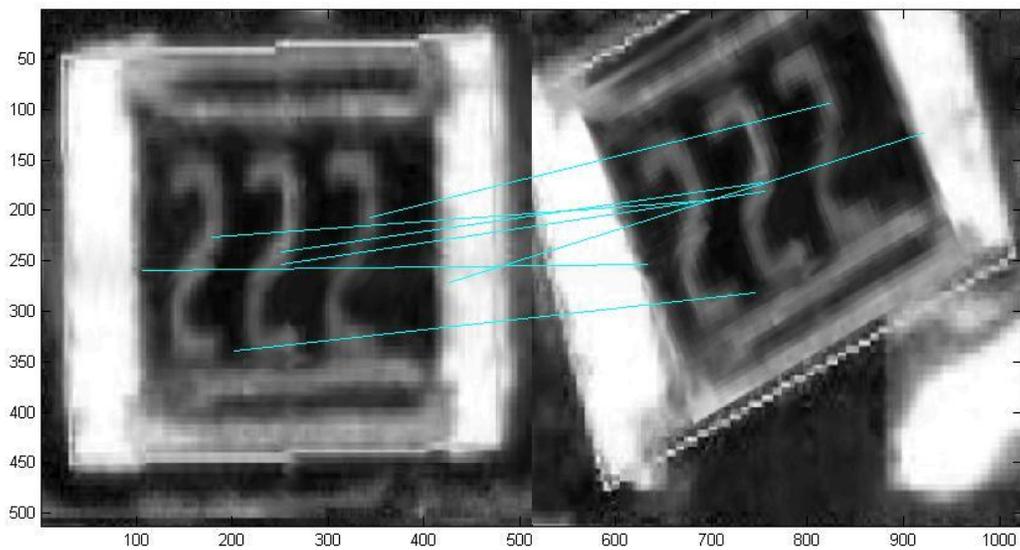


Fig. 4.3. Depois do pós-processamento as correspondências errôneas foram eliminadas.

Usando o algoritmo desenvolvido na Secção 3.6 são seleccionados os melhores pontos para se calcular a matriz de homografia. Na Fig. 4.4 são mostrados os 4 pontos seleccionados com um círculo azul.

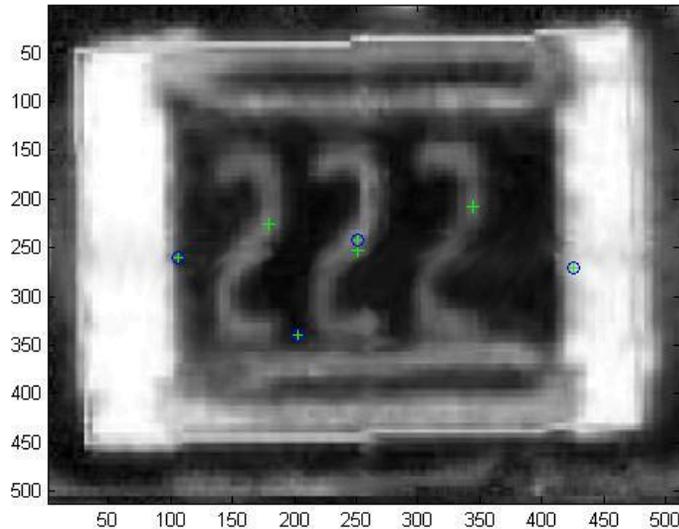


Fig. 4.4. São seleccionados os 4 pontos para calcular a matriz de homografia.

A matriz de homografia é calculada usando o algoritmo explicado no Capítulo 3, e o resultado para esse exemplo é mostrado a seguir.

$$H = \begin{bmatrix} 1,1591 & 0,8801 & -182,6552 \\ -0,6498 & 1,6706 & -10,6744 \\ -0,0002 & 0,0016 & 1,0000 \end{bmatrix}$$

Usando as equações explicadas na Secção 4.2, é calculado o ângulo aproximado que o objeto girou e as escalas nos eixos  $x$  e  $y$  em comparação do objeto original.

$$\theta = -29,2317$$

$$S_x = 1,4554$$

$$S_y = 1,5013$$

Uma malha de pontos é criada sobre o objeto, como mostrado na Fig. 4.5(a). Usando a matriz de homografia é calculada a transformação que sofreu a malha para ser calculada na 2ª imagem, a nova malha de pontos é mostrada na Fig. 4.5(b).

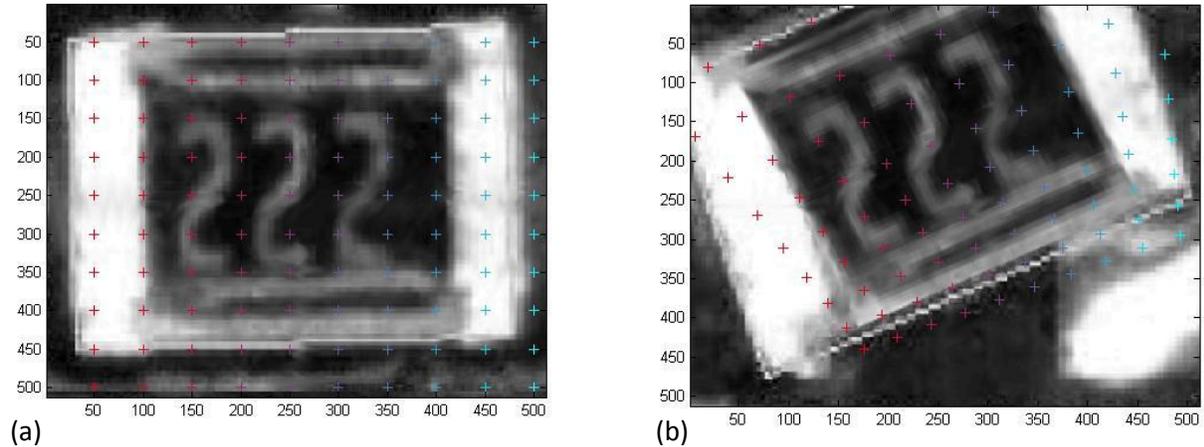


Fig. 4.5. (a) Uma malha de pontos é criada sobre o objeto. (b) Usando a matriz homografia é determinada a ubicação dos pontos na segunda imagem.

### 4.3.2 Exemplo de ubicação correta de componente

A seguir será mostrado um exemplo onde não existe o problema de má ubicação de componentes, onde será calculado o ângulo que o componente está girado em relação a um componente com ubicação correta

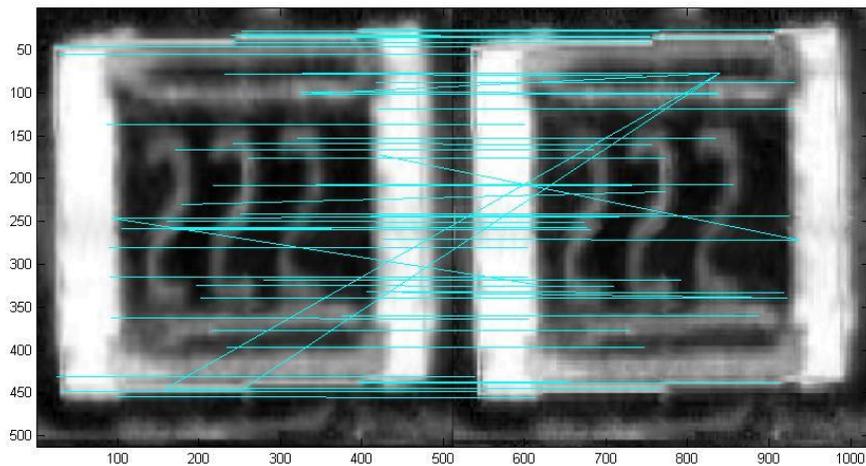


Fig. 4.6. Usando o algoritmo SIFT para detectar o objeto.

Na Fig. 4.6 são mostradas as correspondências para se detectar o componente usando o algoritmo SIFT.

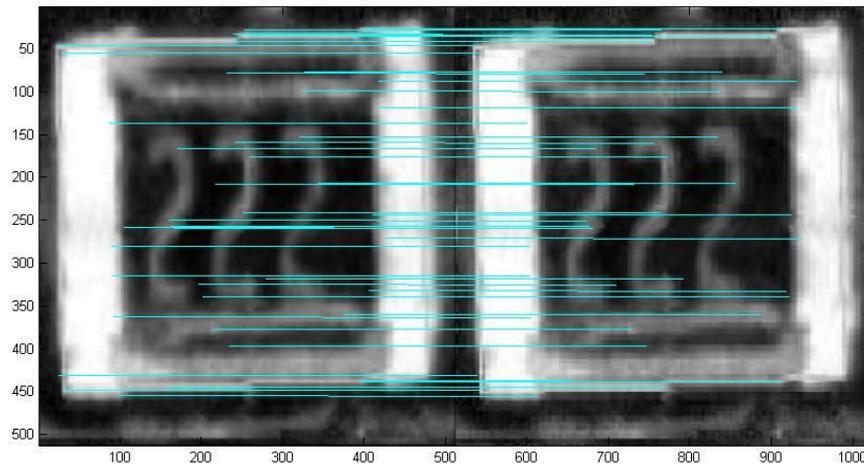


Fig. 4.7. Depois do pós-processamento as correspondências errôneas foram eliminadas.

A Fig. 4.7 mostra as correspondências que resultaram da eliminação das errôneas. Serão escolhidos os 4 pontos para se calcular a matriz de homografia (Ver Fig. 4.8).

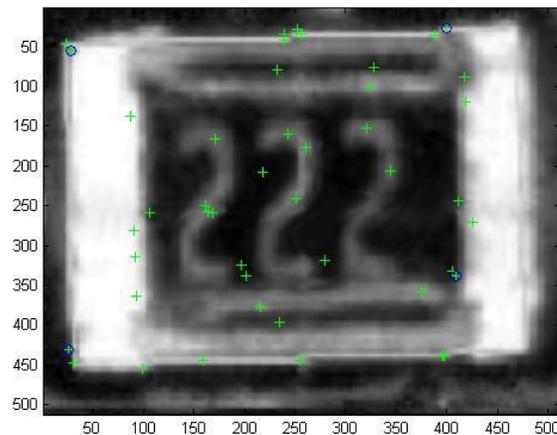


Fig. 4.8. Os 4 pontos escolhidos para se calcular a matriz de homografia estão representados por um círculo azul.

A matriz de homografia usando os 4 pontos escolhidos será:

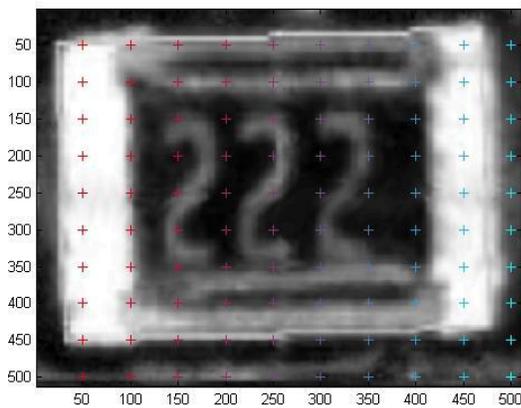
$$H = \begin{bmatrix} 1,0011 & 0,0002 & -0,2325 \\ -0,0002 & 1,0030 & 0,1467 \\ -0,0000 & 0,0000 & 1,0000 \end{bmatrix}$$

A rotação e escalas nos eixos x e y serão:

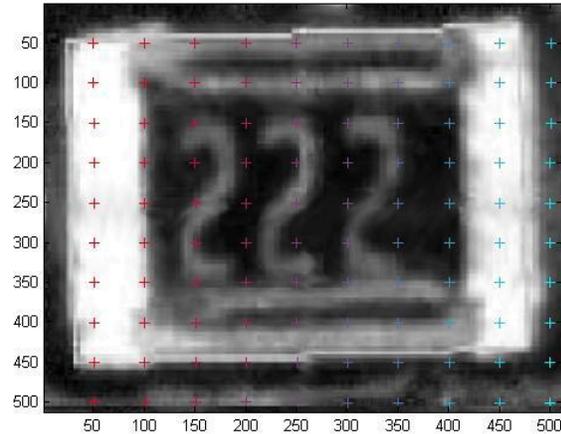
$$\theta = -0,0114$$

$$S_x = 1,0011$$

$$S_y = 1,0020$$



(a)



(b)

Fig. 4.9. (a) Uma malha de pontos é criada sobre o objeto. (b) Usando a matriz homografia é determinada a ubiquação dos pontos na 2ª imagem.

A seguir são apresentadas as tabelas com os resultados de ângulo e escala nos eixos  $x$  e  $y$ . Das tabelas,  $\theta$  é o ângulo que o objeto está rotacionado,  $\theta_f$  é o ângulo obtido usando o algoritmo de aproximação da matriz de homografia,  $s_x$  e  $s_y$  são as escalas nos eixos  $x$  e  $y$  obtidos também da aproximação da matriz de homografia, erro é a diferença entre  $\theta$  e  $\theta_f$  para saber o erro do ângulo obtido manualmente e obtido pelo algoritmo proposto. Os valores de  $\theta$  e  $\theta_f$  estão medidos em graus sexagesimais.

Tabela 4.1: Calculo do ângulo para o primeiro objeto.

<b>Objeto 1</b>				
<b>theta</b>	<b>theta_f</b>	<b>sx</b>	<b>sy</b>	<b>erro</b>
-10	-9,8557	1,0122	0,9997	0,1443
-5	-5,1135	0,9969	1,0114	0,1135
8	8,0465	0,9995	0,9985	0,0465
2	2,1199	1,0126	1,0443	0,1199
6	6,0064	1,0002	1,001	0,0064
-45	-42,7476	1,099	1,2622	2,2524
-32	-32,6889	1,0105	1,0629	0,6889
-18	-18,0928	0,9976	1,0034	0,0928
23	22,0576	1,1598	1,3656	0,9424
35	35,4013	0,9847	0,9712	0,4013

Erro médio dos ângulos = 0,4808

Variância do erro dos ângulos = 0,6656

Tabela 4.2: Calculo do ângulo para o segundo objeto.

<b>Obejto 2</b>				
<b>theta</b>	<b>theta_f</b>	<b>sx</b>	<b>sy</b>	<b>erro</b>
-8	-7,9486	0,9982	1,0315	0,0514
-9	-8,4222	0,9768	1,05	0,5778
-4	-4,17	0,9973	0,9919	0,17
5	4,7531	0,9948	0,999	0,2469
7	6,9287	1,0065	1,0251	0,0713
21	21,8709	0,9243	0,8135	0,8709
25	25,1767	0,9758	0,9495	0,1767
-15	-14,577	1,1029	0,9842	0,423
-13	-12,3882	1,0238	1,0033	0,6118
-21	-21,5585	0,9654	0,9958	0,5585

Erro médio dos ângulos = 0,3758

Variância do erro dos ângulos = 0,2086

Tabela 4.3: Calculo do ângulo para o terceiro objeto.

<b>Objeto 3</b>				
<b>theta</b>	<b>theta_f</b>	<b>sx</b>	<b>sy</b>	<b>erro</b>
-30	-30,7745	1	1,0152	0,7745
-14	-14,5062	0,9936	0,9987	0,5062
-18	-17,0613	1,0152	0,9561	0,9387
-27	-28,6406	1,1065	1,1406	1,6406
-22	-21,8681	1,0001	1,0185	0,1319
4	3,9631	1,0005	1,0079	0,0369
8	7,8508	1,0965	1,0233	0,1492
-6	-5,9534	1,1294	1,1069	0,0466
-2	-2,005	1,1185	1,1324	0,005
-10	-9,9587	1,1207	1,1294	0,0413

Erro médio dos ângulos = 0,4271

Variância do erro dos ângulos = 0,4474

Tabela 4.4: Calculo do ângulo para o quarto objeto.

<b>Objeto 4</b>				
<b>theta</b>	<b>theta_f</b>	<b>sx</b>	<b>sy</b>	<b>erro</b>
-10	-9,8297	1,013	1,0202	0,1703
-5	-5,0603	1,009	1,0023	0,0603
10	9,5099	1,1177	1,5233	0,4901
7	6,7606	1,0142	1,0452	0,2394
3	3,0822	1,0105	1,0269	0,0822
30	30,0217	0,9807	0,9477	0,0217
18	18,1908	1,0124	1,0141	0,1908
-19	-17,6537	1,0931	1,0354	1,3463
-16	-16,4338	1,0451	1,109	0,4338
-27	-27,3091	1,0933	1,1117	0,3091

Erro médio dos ângulos = 0,3344

Variância do erro dos ângulos = 0,2470

Das tabelas anteriores o ângulo máximo de separação entre o ângulo real e o ângulo calculado por nosso algoritmo foi de 2,2524 graus sexagesimais, o que significa que o erro de ângulo é pequeno com o que pode-se estabelecer condições usando o *theta\_f*. Se o *theta\_f* é

maior que 10 graus sexagesimais, o componente tem ubiquação errônea. Caso contrário o componente tem ubiquação correta. Com isto pode-se observar que usando a matriz de homografia para calcular as escalas e ângulo, cada objeto tem 5 casos de ubiquação correta e 5 casos de ubiquação errônea.

# Conclusões

A principal contribuição deste trabalho foi desenvolver um método para a detecção de falhas em placas de circuito impresso. As falhas consideradas neste trabalho são por rotação dos componentes eletrônicos. O método proposto foi baseado na detecção das características relevantes que definem a imagem do circuito impresso. Com este objetivo foi usado o algoritmo SIFT com a implementação de um pós-processamento.

O algoritmo SIFT foi usado para detectar as características de uma imagem em outra. Na maior parte dos casos onde foi usado o algoritmo foi observada a existência de algumas correspondências errôneas, por este motivo foi necessário a implementação de um pós-processamento para eliminar uma grande parte delas.

O algoritmo proposto foi aplicado para a detecção de falha por rotação de quatro componentes eletrônicos situados na placa de circuito impresso. Para cada componente foi usado onze imagens com rotação correta ou errada, sendo a primeira a imagem de um componente situado corretamente em uma placa, e as outras dez as imagens dos componentes por detectar. O algoritmo conseguiu detectar todos os componentes com falhas das dez amostras analisadas para os quatro componentes eletrônicos tomados como referencia.

Como segunda contribuição do trabalho foi analisada as vantagens do algoritmo proposto com respeito a outros métodos existentes. O algoritmo proposto foi aplicado em 20 imagens onde suas características de escala, rotação, ruído gaussiano e brilho foram alteradas individualmente e os resultados da media foram comparados com os métodos SIFT, SURF e RANSAC. O algoritmo apresentou melhor resultado para variações de rotação, ruído gaussiano e brilho com respeito aos métodos SIFT, SURF e RANSAC. Na variação de escala o algoritmo só apresenta melhores resultados com respeito aos métodos SIFT e RANSAC.

O método manteve um comportamento bom e aceitável para as 4 variações aplicadas o qual pode ser uma boa alternativa de solução para problemas que apresentam alterações nas 4 variações conjuntas.

# Trabalhos futuros

Para diminuir o custo computacional do algoritmo desenvolvido em Matlab, é possível realizar o algoritmo usando programação em C, e pode ser usado o programa Visual Studio já que é muito utilizado em esses casos.

Também o algoritmo proposto pode ser desenvolvido para trabalhar em um FPGA e assim não será necessário o uso de um computador.

Pode-se desenvolver o algoritmo SIFT proposto para detecção de objetos usando uma base de dados grande das características de objetos e assim identificar um objeto, pertencente à base de dados, em outro ambiente.

# Referências bibliográficas

- [1] Lowe D.G., “Object Recognition from Local Scale-Invariant Features,” International Conference on Computer Vision, Corfu, Greece, pp. 1150-1157.
- [2] Lowe D.G., “Local feature view clustering for 3D object recognition,” IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, pp. 682-688.
- [3] Lowe D.G., “Distinctive Image Features from Scale-Invariant Keypoints,” International Journal of Computer Vision, 2004.
- [4] H. Bay, T. Tuytelaars, and L. Van Gool. “SURF: Speeded up robust features,” ECCV, 2006.
- [5] Y.Ke, R.Sukthankar, “PCA-SIFT: A more distinctive representation for local image descriptors,” Computer Vision and Pattern Recognition, 2004.
- [6] YongZhang, Kai-Bin Wei, “Research on Wide Baseline Stereo Matching Based on PCA-SIFT,” 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 2010.
- [7] Wang Yiding, Zhang Ning, “Gender Classification based on Enhanced PCA-SIFT Facial Features,” The 1st International Conference on Information Science and Engineering (ICISE2009).
- [8] Hailing Zhou, Lijun Xie, XuliangFang, “Visual Mouse: SIFT Detection and PCA Recognition,” International Conference on Computational Intelligence and Security Workshops, 2007, pp. 264-266.

- [9] SHI Dong-cheng, YAN Guo-qing, "Combining MSCR Detector and PCA-SIFT Descriptor for Scene Recognition," IEEE 2010, pp. 136-141.
- [10] Ping Zhou, Xiling Luo, "A Robust Feature Matching Algorithm Based on CSIFT Descriptors," Signal Processing, Communications and Computing (ICSPCC), 2011 IEEE International Conference, pp. 1-4. [Online]. Available:  
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6061763>
- [11] Liqiong Deng, Danwen Chen, Zhimin Yuan, Lingda Wu, "Cartoon Scene Matching System Using Color And CSIFT Feature," 3rd International Congress on Image and Signal Processing (CISP2010), 2010, pp. 2519-2524.
- [12] Qiuchuan Qin, Huaping Liu, "3D Model Registration Using CSIFT and POSIT," IEEE Symposium on Electrical & Electronics Engineering (EEESYM), 2012, pp. 662-665.
- [13] Jun Li, Kejian Yang, "Normalization Methods of SIFT Vector for Object Recognition," 10th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, 2011.
- [14] Huanjing Yue, Xiaoyan Sun, Feng Wu, Jingyu Yang, "SIFT-BASED IMAGE COMPRESSION," IEEE International Conference on Multimedia and Expo, 2012.
- [15] Cong Geng, Xudong Jiang, "Face Recognition using SIFT features", ICIP 2009, pp. 3313-3316.
- [16] Diego González Aguilera, "Procesamiento de Imágenes," Universidad de Salamanca, Master de geotecnologías cartográficas em ingeniería y arquitectura.
- [17] Xianjiu Guo, Wei Wang, "Image Matching Algorithm Based on Subdivision Wavelet and Local Projection Entropy," Proceedings of the 6th World Congress on Intelligent

- Control and Automation, June 21 - 23, 2006, Dalian, China, pp. 10380-10383. [Online]. Available: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1714036>
- [18] Shinichiro Omachi, Masako Omachi, "Fast Template Matching With Polynomials," IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 16, NO. 8, AUGUST 2007, pp. 2139-2149. [Online]. Available: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4271537>
- [19] Murray, "SIFT Image Features" . [Online]. Available: [http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\\_COPIES/AV0405/MURRAY/SIFT.html](http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/MURRAY/SIFT.html)
- [20] Fernando Alonso Fernandez, Pedro Tome Gonzalez, Virginia Ruiz-Albacete, Javier Ortega-Garcia, "Iris Recognition Based on SIFT Features," Biometric Recognition Group - ATVS, Escuela Politecnica Superior, Universidad Autonoma de Madrid. [Online]. Available: [http://atvs.ii.uam.es/files/2009\\_FAlonso\\_SIFT\\_Iris.pdf](http://atvs.ii.uam.es/files/2009_FAlonso_SIFT_Iris.pdf)
- [21] Matthew Brown, David Lowe, "Invariant Features from Interest Point Groups," BMVC 2002, pp. 253-262. [Online]. Available: <http://ece631web.groups.et.byu.net/References/Invariant%20features%20from%20interest%20point%20groups.pdf>
- [22] Chris Harris, Mike Stephens, "A COMBINED CORNER AND EDGE DETECTOR," Plessey Research Roke Manor, United Kingdom, pp. 147-152. [Online]. Available: <http://csce.uark.edu/~jgauch/library/Features/Harris.1988.pdf>
- [23] YU MENG, Bernard Tiddeman, "Implementing the Scale Invariant Feature Transform(SIFT) Method," Department of Computer Science University of St. Andrews. [Online]. Available: [http://www.cs.st-andrews.ac.uk/~yumeng/yumeng-SIFTreport-5.18\\_bpt.pdf](http://www.cs.st-andrews.ac.uk/~yumeng/yumeng-SIFTreport-5.18_bpt.pdf)

- [24] Giancarlo Luis Gomez Gonzales, “Aplicação da técnica SIFT para determinação de campos de deformações de materiais usando visão computacional,” Pontificia Universidade Catolica do Rio de Janeiro- PUC-RIO, 2011. [Online]. Available: [http://www.maxwell.lambda.ele.puc-rio.br/17050/17050\\_5.PDF](http://www.maxwell.lambda.ele.puc-rio.br/17050/17050_5.PDF)
- [25] Jeffrey S. Beis, David G. Lowe, “Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces,” Department of Computer Science University of British Columbia .[Online]. Available: <http://www.cs.ubc.ca/~lowe/papers/cvpr97.pdf>
- [26] Yan-Tao Zheng, Ming Zhao, Shi-Yong Neo, Tat-Seng Chua, Qi Tian, "Visual Synset: towards a Higher-level Visual Representation", in Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) 2008, Anchorage, Alaska, U.S., June 24- 26,2008
- [27] Berkeley Dataset. [Online]. Available: <http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/>
- [28] Montserrat González Ferreiro “Reconocimiento facial combinando 2D e 3D”, Universidad Politécnica de Cataluña. [Online]. Available: <http://es.scribd.com/doc/28778540/48/Calculo-de-la-homografia>

# Apêndice A:

## A Matriz Hessiana

Matriz Hessiana de grau 2:

$$Hf(x, y) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

**Exemplo 1:** Seja a função:  $f(x, y) = 5x^2y + y$

A função Hessiana é calculada da seguinte forma:

$$\begin{aligned} \frac{\partial^2 f}{\partial x^2} &= \frac{\partial}{\partial x} \left( \frac{\partial f}{\partial x} \right) = \frac{\partial}{\partial x} (10xy) = 10y \\ \frac{\partial^2 f}{\partial x \partial y} &= \frac{\partial}{\partial x} \left( \frac{\partial f}{\partial y} \right) = \frac{\partial}{\partial x} (5x^2 + 1) = 10x \\ \frac{\partial^2 f}{\partial y^2} &= \frac{\partial}{\partial y} \left( \frac{\partial f}{\partial y} \right) = \frac{\partial}{\partial y} (5x^2 + 1) = 0 \\ Hf(x, y) &= \begin{bmatrix} 10y & 10x \\ 10x & 0 \end{bmatrix} \end{aligned}$$

Máximos e Mínimos usando a função Hessiana

$f \in C^2(D)$  /  $D$  conjunto convexo de  $\mathbb{R}^n$ ,  $a \in D$  /  $f''(a) = \mathbf{0}_n$

- $f''(a) > 0$ ,  $a$  é um mínimo local de  $f$
- $f''(a) < 0$ ,  $a$  é um máximo local de  $f$
- $f''(a) \geq 0$ ,  $a$  não é extremo local de  $f$

# Apêndice B:

## Propriedades da Gaussiana

### Caso Unidimensional

A função Gaussiana de variância  $\sigma^2$  é a função dada por:

$$G_{\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}x^2}$$

Outra forma de representar essa função é utilizar  $\sigma^2 = 2t$ .

$$G_t(x) = \frac{1}{\sqrt{4\pi t}} e^{-\frac{1}{4t}x^2}$$

#### 1. Propriedade B.1:

$$\int_{-\infty}^{\infty} G(x) dx = 1$$

2. **Propriedade B.2:** A Transformada de Fourier de um filtro Gaussiano de variância  $\sigma^2$  é dada por

$$\begin{aligned} F\{G_t(x)\} &= \hat{G}(\omega) \\ \hat{G}(\omega) &= e^{-2\pi^2\sigma^2\omega^2} \end{aligned}$$

3. **Propriedade B.3:** A Transformada bilateral de Laplace de  $G_t(x)$  é

$$L\{G_t(x)\} = e^{\frac{1}{2}\sigma^2 s^2} = e^{s^2 t}$$

- 4. Propriedade B.4:** A convolução de dois núcleos Gaussianos de variâncias  $\sigma_1^2$  e  $\sigma_2^2$  produz um núcleo Gaussiano de variância  $\sigma_1^2 + \sigma_2^2$ , isto é

$$G_{t_1} * G_{t_2} = G_{t_1+t_2}$$

- 5. Propriedade B.5:** O núcleo gaussiano  $G_t(x)$  satisfaz

$$\frac{\partial G}{\partial t} = \frac{\partial^2 G}{\partial x^2}$$

$$\lim_{t \rightarrow 0} G_t(x) = \delta(x)$$

### Caso Bidimensional

$$G_t(x, y) = G_t(x)G_t(y) = \frac{1}{4\pi t} e^{-\frac{1}{4t}(x^2+y^2)}$$

- 6. Propriedade B.6:** O núcleo Gaussiano também tem a propriedade de semi-grupo

$$G_{t_1} * G_{t_2} = G_{t_1+t_2}$$

- 7. Propriedade B.7:** O núcleo Gaussiano  $G_t(x, y)$  satisfaz.

$$\frac{\partial G}{\partial t} = \nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$$

- 8. Propriedade B.8:** A solução  $F(x, y; t)$  da equação do calor bidimensional

$$\frac{\partial F}{\partial t} = \nabla^2 F$$

$$F(x, y; 0) = f_0(x, y)$$

é dada pela convolução  $F(x, y; t) = G_t(x, y) * f_0(x, y)$

# Apêndice C:

## Espaço de Escala Gaussiano

### Caso Unidimensional

Dado um sinal  $f_0: \mathbf{R} \rightarrow \mathbf{R}$ , define-se o espaço de escala Gaussiano de  $f$  como a função  $F: \mathbf{R} \times \mathbf{R}_+ \rightarrow \mathbf{R}$  (representada por  $F(\mathbf{x}, t) = F_t(\mathbf{x})$ ) que é a solução da equação do calor

$$\frac{\partial F_t(\mathbf{x})}{\partial t} = \frac{\partial^2 F_t(\mathbf{x})}{\partial x^2}$$

$$F_0(\mathbf{x}) = f_0(\mathbf{x})$$

O espaço de escala de  $f$  pode ser obtido através de convoluções com Gaussianas

$$F_t(\mathbf{x}) = G_t(\mathbf{x}) * f_0(\mathbf{x})$$

### Caso Bidimensional

Dado uma imagem contínua  $f_0: \mathbf{R}^2 \rightarrow \mathbf{R}$ , define-se o espaço de escala Gaussiano de  $f_0$  como a função  $F: \mathbf{R}^2 \times \mathbf{R}_+ \rightarrow \mathbf{R}$  (representada por  $F_t(\mathbf{x}, y)$  onde  $t \in \mathbf{R}_+$ ) que é a solução da equação do calor bidimensional

$$\frac{\partial F_t(\mathbf{x}, y)}{\partial t} = \nabla^2 F_t(\mathbf{x}, y) = \frac{\partial^2 F_t(\mathbf{x}, y)}{\partial x^2} + \frac{\partial^2 F_t(\mathbf{x}, y)}{\partial y^2}$$

$$F_0(\mathbf{x}, y) = f_0(\mathbf{x}, y)$$

A solução pode ser expressa como uma convolução com Gaussianas Bidimensionais

$$F_t(\mathbf{x}, y) = G_t(\mathbf{x}, y) * f_0(\mathbf{x}, y)$$