PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

ESTUDO DO ALGORITMO SIFT (SCALE INVARIANT FEATURE TRANSFORM)

GOIÂNIA,
2024

RÍVERSON DA COSTA SOUZA

ESTUDO DO ALGORITMO SIFT (SCALE INVARIANT FEATURE TRANSFORM)

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação

Orientador: Prof. Me. Gustavo Siqueira Vinhal.

GOIÂNIA,
2024

RÍVERSON DA COSTA SOUZA

ESTUDO DO ALGORITMO SIFT (SCALE INVARIANT FEATURE TRANSFORM)

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Ciência da Computação, em _____/_____/_____.

_____

Orientador(a): Prof. Me. Gustavo Siqueira Vinhal.

_____

Prof. Me. Fernando Gonçalves Abadia.

_____

Prof. Me. Rafael Leal Martins.

GOIÂNIA,
2024

DEDICATÓRIA

Dedico esta monografia à minha família pela fé e confiança demonstrada.

Aos meus amigos, pelo apoio incondicional.

Aos professores, pelo simples fato de estarem dispostos a ensinar.

À meu orientador, pela paciência, demonstrada no decorrer deste trabalho.

Enfim, a todos, que de alguma forma tornaram este caminho mais fácil de ser percorrido.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

RESUMO

Este trabalho apresenta um estudo do algoritmo SIFT *(Scale Invariant Feature Transform)*, que se propõe a detectar pontos de interesse de uma imagem em outra e fazer comparações entre elas. É realizado em quatro etapas principais: detecção de pontos finais, localização de pontos-chave, definição de orientação e descrição de pontos-chave. Os dois primeiros definem a parte do detector e os dois seguintes definem a geração do descritor. Para analisar o comportamento do algoritmo, experimentos foram realizados em imagens com diferentes propriedades, e algumas distorções como redução de escala, rotação, borramento, escurecimento e adição de ruído foram realizados nas imagens. O tempo de processamento e o número de valores internos foram subtraídos para analisar as combinações propostas. Esse trabalho contribui para aplicações que busquem utilizar algoritmos descritores de pontos chave para serem usados na detecção e reconhecimento de objetos em imagens diferentes.

Palavras-chave: Descrição de pontos-chave; Detecção de pontos finais; Chave; Localização de Pontos Chave.

ABSTRACT

This work presents a study of the SIFT (Scale Invariant Feature Transform) algorithm, which proposes to detect points of interest from one image to another and make comparisons between them. It is carried out in four main steps: endpoint detection, keypoint location, orientation definition, and keypoint description. The first two define the detector part and the next two define the descriptor generation. To analyze the behavior of the algorithm, experiments were performed on images with different properties, and some distortions such as scaling, rotation, blurring, darkening and adding noise were performed on the images. The processing time and the number of internal values were subtracted to analyze the proposed combinations. This work contributes to applications that seek to use keypoint descriptor algorithms to be used in the detection and recognition of objects in different images.

Keywords: Description of key points; Endpoint detection; Key; Location of Key Points.

1. Introdução

A base para estudos relacionados à computação de imagens é baseada nas propriedades fundamentais das imagens (DAWSON-HOWE, 2014). Essas informações podem ser apresentadas e obtidas digitalmente por meio de dispositivos fotográficos, como câmeras. À medida que uma imagem é adquirida e sofre conversão, podem ocorrer erros que diminuem a capacidade de compreensão da imagem, mais comumente ruídos, pontos que não representam a imagem real. No entanto, existem maneiras de remover esses ruídos usando técnicas de processamento de imagem (DAWSON-HOWE, 2014).

Para realizar o processamento da imagem, é necessário capturá-la. Através de aparelhos eletrônicos é possível obter imagens, dentre esses aparelhos um dos mais utilizados atualmente são smartphones e webcams. Esses aparelhos possuem um hardware extremamente desenvolvido que possibilita a fotografia. Anteriormente, esse processo só era possível com câmeras específicas para esse fim. O mecanismo que permite a captura da imagem é chamado de painel fotossensível localizado dentro da câmera, que interpreta a quantidade de luz que passa pela lente da câmera. Além desse painel, há também um revestimento que evita a dispersão da luz. Este revestimento contém uma lente que permite focalizar a luminância que representa o plano fotossensível (DAWSON-HOWE, 2014).

No âmbito da computação visual, as imagens são de grande importância. Sua identificação pode ser realizada por um painel fotossensível e essas imagens podem ser projetas em duas dimensões (2D) ou em três dimensões (3D) (DAWSON-HOWE, 2014).

Detectar pontos de interesse em uma imagem é uma tarefa muito comum no processamento de imagens. Existem vários algoritmos para resolver esse problema, mas um bom algoritmo deve ser robusto a vários fatores: iluminação, rotação, escala e muito mais. Este trabalho centra-se na implementação e experimentação do algoritmo *Scale Invariant Feature Transform, SIFT*, que pretende ser um método robusto conforme explicado.

Após detectar esses pontos-chave ou pontos-chave na imagem, foi criado um descritor para cada ponto e podemos então realizar comparações entre os pontos. Desta forma é possível criar um casamento, um pareamento, entre diferentes imagens contendo o mesmo objeto. Realizamos testes verificando o desempenho da implementação combinando imagens com diferentes efeitos.

Uma solução óbvia e precisa para o reconhecimento de objetos sujeitos a transformações geométricas é o método da força bruta que consiste em fazer uma série de operações de casamento de padrões entre a imagem analisada e a imagem do objeto, considerando todas as posições possíveis que o padrão possa aparecer na imagem analisada e todas as transformações que possa estar sujeito dentro de um intervalo estabelecido (TSAI; TSAI, 2002).

Claramente, esta solução é inviável já que demanda muito tempo de processamento. Nos experimentos realizados neste trabalho, constatou-se que o algoritmo SIFT *(Scale Invariant Feature Transform),* foi capaz de obter resultados similares aos resultados do método de força bruta, com tempo de processamento bem menor. Daí, a importância de se estudar esse método.

1.1 Objetivos

1.1.1 Geral

Estudo do Algoritmo SIFT para encontro de similaridade entre imagens.

1.1.2 Específicos

Entender conceitos de imagens;
Explorar o algoritmo SIFT;
Verificar aplicações para o algoritmo SIFT;

1.2 Justificativa

A visão computacional é crucial para muitos setores, como segurança, automação e tecnologia de consumo. O algoritmo Scale Invariant Feature

Transform (SIFT), desenvolvido por David Lowe, é reconhecido exclusivamente por sua eficácia na detecção e descrição de características locais em imagens, sendo invariante à escala e à rotação e robusto à variação.

Este estudo justifica-se pela necessidade de uma compreensão detalhada do SIFT, suas vantagens, limitações e aplicações práticas. À medida que a demanda por sistemas precisos de visão computacional continua a aumentar, o conhecimento profundo do SIFT pode aprimorar as soluções existentes e inspirar novas abordagens. Portanto, a pesquisa SIFT é fundamental tanto para o avanço teórico quanto para aplicações práticas, ajudando a desenvolver sistemas de visão mais eficientes e precisos, o estudo detalhado do SIFT é essencial para aprimorar as soluções existentes e desenvolver novas abordagens.

Portanto, a investigação do algoritmo SIFT não apenas enriquece o conhecimento teórico, mas também tem implicações práticas significativas, permitindo o desenvolvimento de sistemas mais sofisticados e capazes de enfrentar desafios reais na análise de imagens.

2. Fundamentação Teórica

A visão computacional é um campo que se beneficia muito com os avanços nos algoritmos de detecção de similaridade de imagens. Esses algoritmos são essenciais para tarefas como reconhecimento de objetos, habitação 3D, rastreamento de movimento e muitas outras aplicações que permitem a identificação e correspondência de pontos-chave em diferentes imagens. Os algoritmos mais conhecidos e comumente usados incluem SIFT (Scale Invariant Feature Transform), SURF (Accelerated Robust Features), ORB (Oriented FAST and Rotated Brief), BRIEF (Binary Robust Independent Basic Features) e AKAZE (Accelerated KAZE).

Desenvolvido por David Lowe, o SIFT é conhecido por sua invariância ao dimensionamento e rotação e por sua alta robustez a mudanças de iluminação e ruído. Criado por Herbert Bay e colegas, SURF é uma versão otimizada do SIFT que oferece maior velocidade e maiores resultados. Introduzido por Ethan Rublee e sua equipe, o ORB combina a velocidade da detecção de pontos-chave FAST com a eficiência descrita por BRIEF, tornando-o ideal para dispositivos com recursos limitados. BRIEF foca na simplicidade e usa descritores binários para operações rápidas, mas é menos robusto. Finalmente, AKAZE utiliza difusão não linear para detecção de características baseadas em KAZE, proporcionando um bom equilíbrio entre precisão e eficiência.

Esses algoritmos possuem características e vantagens próprias e são adequados para diferentes aplicações.

Apesar das vantagens de algoritmos como SURF, ORB, BRIEF e AKAZE em termos de velocidade e simplicidade, o SIFT continua a ser a escolha preferida quando a precisão e a robustez são priorizadas. A capacidade do SIFT de detectar e descrever características locais de maneira invariável em escala e rotação e ser robusto a mudanças na iluminação torna o SIFT uma ferramenta indispensável.

SIFT é um algoritmo de visão computacional publicado por David Lowe, em 1999 (Lowe, 1999) e patenteado nos EUA pela *University of British Columbia.*

SIFT pode ser usado para obter dados sobre uma imagem. Esses dados são muitas vezes referidos como descritores. Uma vez que foi encontrado descritores de recursos que nos interessam, podemos usá-los para comparar

imagens, detectar imagens dentro de outras imagens ou até mesmo salvá-las como nossa própria assinatura.

Ao comparar imagens diferentes, podemos comparar pontos-chave, e quando as assinaturas de um determinado conjunto de pontos têm valores muito próximos, podemos dizer que esses pontos representam o mesmo objeto mostrado nas duas imagens.

A ideia do algoritmo SIFT segue a teoria do espaço de escala para procurar pontos-chave em várias dimensões da imagem para criar várias instâncias da imagem original e tornar a escala invariante. O algoritmo começa construindo uma pirâmide de imagens divididas em oitavas, que por sua vez são divididas em intervalos. Cada oitavo consiste em uma série de imagens do mesmo tamanho e, à medida que você passa de um oitavo para o outro, o tamanho das imagens é reduzido pela metade. Dentro da mesma oitava, temos o espaçamento da imagem, cada imagem suavizada por um filtro gaussiano $G(x, y, \sigma)$. A diferença entre uma imagem e a próxima na mesma oitava é o fator k do filtro, ou seja, se a imagem i é suavizada por um filtro $G(x, y, \sigma)$, então a imagem $i + 1$ é suavizada por um filtro $G(x, y, k\sigma)$.

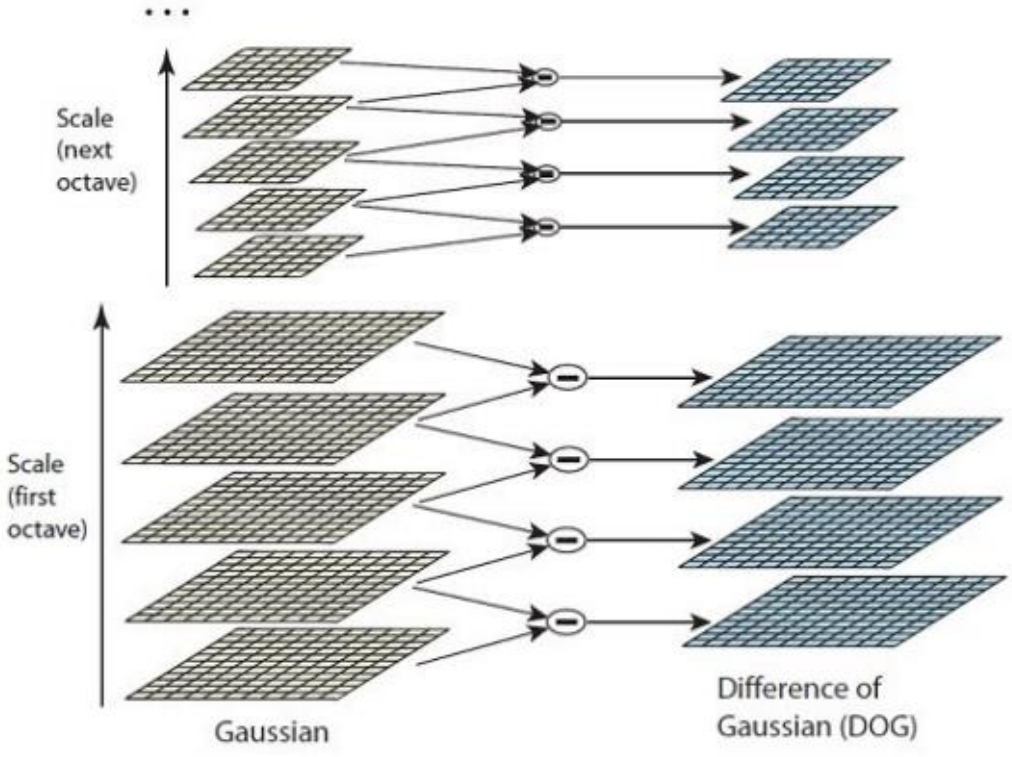


Figura 1: Representação do procedimento de obtenção das Diferenças de Gaussianas DoG para diversas oitavas de uma imagem. (Lowe 2004).

Uma maneira eficiente de criar uma Diferença de Gaussiana é mostrada na Figura 1, cujos 4 passos são descritos abaixo.

1. A imagem inicial é convoluída de forma incremental com filtros gaussianos para produzir imagens separadas pelo fator de escala k no espaço de escala, mostrado na coluna da esquerda.

2. Lowe acredita que é necessário convoluir a imagem até $2\,\sigma$ para construir descritores invariantes em escala. Assim, para geração em intervalos s, o fator de escala k é definido como $k = 2^{\frac{1}{s}}$, produzindo $s + 3$ imagens em uma oitava, de forma que a detecção extrema cobre toda a oitava.

3. Imagens em escalas adjacentes são subtraídas para produzir as imagens da Diferença do Gaussiano mostradas à direita (na Figura 1)

4. Uma vez processada a oitava, é reduzida a resolução da imagem (downsample) tomando-se cada segundo pixel da imagem no centro da oitava, gerando-se uma nova oitava (alteração da frequência de muestreo por um fator de dois), e voltando-se ao passo número 1.

A partir daí, será realizada a detecção dos extremos em cada intervalo de cada oitava. Um extremo é definido como qualquer valor no DoG que é maior do que todos os seus vizinhos de escala espacial.

Os extremos são dados por valores máximos ou mínimos locais para cada $D(x, y, \sigma)$, que podem ser obtidos comparando a intensidade de cada ponto com as intensidades de seus oito vizinhos em sua escala, com os nove pontos vizinhos na escala superior, e o nove vizinhos na escala inferior, mostrados na Figura 2. Na figura, o ponto marcado com um "X" é comparado aos seus vizinhos marcados com um "O". As 3 imagens DoG apresentadas na figura correspondem à diferença entre as imagens adjacentes da pirâmide gaussiana.

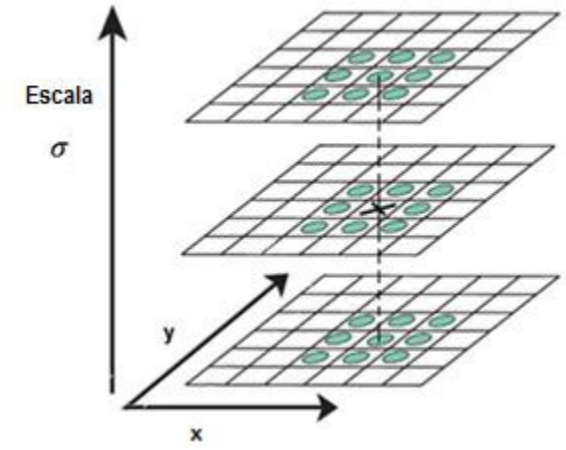


Figura 2 Detecção de extremos no espaço-escala. Fonte: (Lowe, 2004).

A próxima etapa é definir a localização dos pontos-chave e fazer o descarte de pontos instáveis.

2.1 Localização Precisa de Pontos Chaves

Todos os pontos identificados como extremos são candidatos a pontos chave. Agora queremos calcular a localização exata desses pontos-chave.

O método consiste em construir uma função quadrática 3D do ponto de amostragem local para determinar a posição interpolada do máximo.

Isto é feito utilizando uma expansão de Taylor da função Diferença de Gaussiano aplicada à imagem, $D(x, y, \sigma)$, deslocada de modo que a origem desta expansão esteja localizada no ponto de amostragem (Brown and Lowe, 2002):

$$D(\bar{x}) = D + \frac{\partial D^T}{\partial \bar{x}}\bar{x} + \frac{1}{2}\bar{x}^T\frac{\partial^2 D}{\partial x^2}\bar{x} \ldots \qquad (1)$$

$$\bar{X} = (x, y, \sigma)^T \qquad (2)$$

Onde o valor de D, sua primeira e segunda derivadas são calculadas no ponto amostral e $\bar{x}$, representa o deslocamento daquele ponto.

A localização em sub-pixels do ponto de interesse é dada pelo extremo da função apresentada na equação (1). Esta localização $\hat{X}$, é determinada ao se calcular a derivada de $D(x)$ em relação a x , e igualando o resultado a zero:

$$\frac{\partial D}{\partial \bar{x}} + \frac{\partial^2 D}{\partial \bar{x}^2}\hat{X} = 0 \qquad (3)$$

Tem-se então a posição do extremo, dada por:

$$\hat{X} = -\frac{\partial^2 D^{T-1}}{\partial \bar{x}^2}\frac{\partial^2 D}{\partial \bar{x}} \qquad (4)$$

O valor da função no extremo $D(\bar{x})$, é útil para suprimir extremos instáveis de baixo contraste que seriam sensíveis ao ruído.

Substituindo-se a equação (4) na equação (1) obtém-se:

$$D\hat{X} = D + \frac{1}{2}\frac{\partial D^T}{\partial \bar{x}}\hat{X} \qquad (5)$$

De acordo com Lowe, um valor $|D(\hat{X})|$ abaixo de um certo limite deve ser rejeitado. Em Brown e Lowe (2002), recomenda-se trabalhar com um valor de 0,03 para este limite (assumindo-se que os tons de cinza dos *pixels* da imagem estejam normalizados em valores entre 0 e 1).

Além do procedimento apresentado para rejeitar pontos, Lowe também aponta que a função DoG tem uma resposta "forte" ao longo das arestas, mesmo que a localização ao longo da aresta seja mal determinada, i.e., pontos nas arestas poderiam ser escolhidos como pontos de interesse, o que não é desejável. Mas estes pontos podem ser detectados e removidos, conforme mostrado abaixo.

A eliminação dos pontos-chave próximos às arestas é feita usando uma matriz Hessiana 2x2, H, calculada na localização e escala dos pontos-chave na função D.

$$H(x,y) = \begin{bmatrix} D_{xx}\ D_{xy} \\ D_{x,y} D_{yy} \end{bmatrix} \quad (6)$$

onde $D_{x,y}$ é a derivada de $D(x,y,\sigma)$ na localização e escala em relação a x e y; $D_{x,x}$ é a derivada segunda em relação a x; e $D_{y,y}$ é a derivada segunda em relação a y.

Assim, a Hessiana representa a segunda derivada, permitindo que a magnitude da curvatura de D seja medida a partir de seus autovalores.

As derivadas são estimadas através das diferenças entre pontos vizinhos à localização e escala definida, e pode ser aproximada por

$$D_{xx} = D(x+1,y,\sigma) - 2D(x,y,\sigma) + D(x-1,y,\sigma) \quad (7)$$

$$D_{yy} = D(x,y+1,\sigma) - 2D(x,y,\sigma) + D(x,y-1,\sigma) \quad (8)$$

$$D_{xy} = \begin{pmatrix} D(x-1,y+1,\sigma) - D(x+1,y+1,\sigma) \\ +D(x+1,y-1,\sigma) - D(x-1,y-1,\sigma) \end{pmatrix} /4 \quad (9)$$

Determina-se α, o autovalor com maior magnitude, e β, o de menor. Pode-se, então, calcular a soma dos autovalores pelo traço de H e o produto pelo seu determinante:

$$T_r(H) = D_{xx} + D_{yy} = \alpha + \beta \quad (10)$$

$$Det(H) = D_{xx} D_{yy} - \left(D_{xy}\right)^2 = \alpha\beta \quad (11)$$

Caso o determinante seja negativo, as curvaturas tenham sinais diferentes e o ponto seja descartado, não é considerado um extremo. Sendo r

a razão entre o autovalor de maior magnitude e o de menor, de modo que $\alpha = r\beta$, então

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r} \qquad (12)$$

A equação (12) depende apenas da razão entre os autovalores, independentemente de seus valores individuais. O valor /r fornece uma medida de quão diferentes são os autovalores, significando que quando são idênticos, é mínimo e cresce com o valor de r. Assim, os pontos próximos às extremidades são eliminados descartando pontos abaixo de um determinado limite (r):

$$\frac{T_r(H)^2}{Det(H)} < \frac{(r+1)^2}{r} \qquad (13)$$

A equação (13) é altamente eficiente para o cálculo. Lowe sugere o uso de $r = 10$, o que elimina os pontos-chave que não são estáveis apesar de estarem próximos de extremidades.

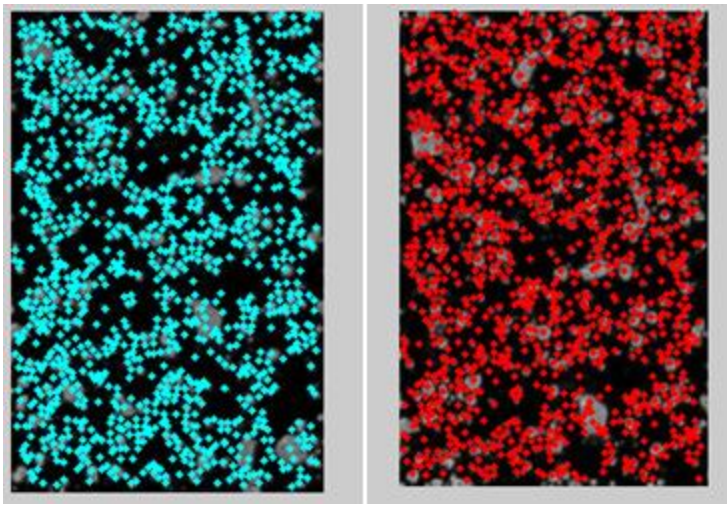


Figura 3 Pontos-chaves localizados em duas imagens, antes e após uma deformação trativa na direção vertical. Fonte: Autoria Própria.

2.2 Atribuição da Orientação dos Descritores

A cada ponto-chave é atribuída uma orientação que será posteriormente usada para construir descritores invariantes de rotação. Esta invariância é obtida através das características locais da imagem.

Calcula-se para cada amostragem da imagem na escala, $L(x, y, \sigma)$, a magnitude $m(x, y)$ e orientação $\theta(x, y)$ do gradiente usando as diferenças de pixels:

$$m(x,y) = \sqrt{\left(\frac{\left(L(x+1,y) - L(x-1,y)\right)^2}{(L(x,y+1) - L(x,y-1))^2}\right)} + \tag{14}$$

$$\theta(x,y) = tan^{-1}\left(\frac{(L(x,y+1) - L(x,y-1))}{(L(x+1,y) - L(x-1,y))}\right) \tag{15}$$

O histograma de orientação para o pixel é plotado na vizinhança ao redor do ponto-chave. O histograma $(0\ a\ 2\pi)$ tem 36 regiões, abrangendo todas as direções possíveis, veja a Figura 4 (Lowe, 2004).

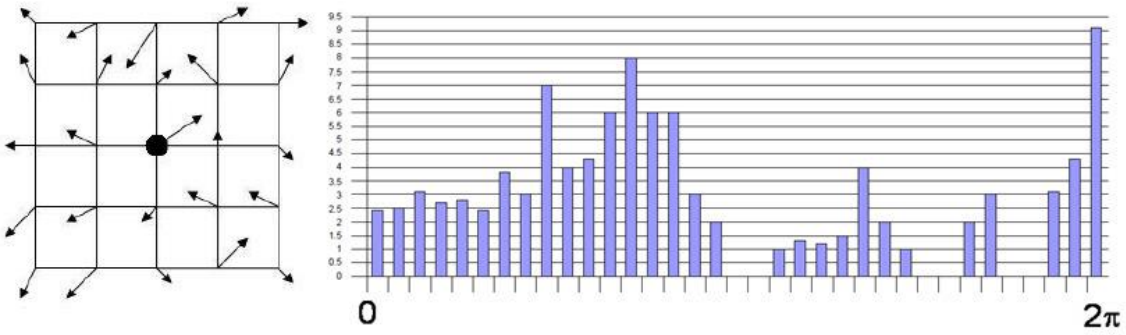


Figura 4 Histograma de orientações de um ponto-chave. (Lowe, 2004).

Cada ponto nas proximidades do ponto-chave é adicionado ao gráfico com um determinado valor de peso. O primeiro peso é o valor da magnitude $m(x, y)$ de cada ponto adicionado. O segundo peso é dado por uma janela Gaussiana circular com σ' igual a 1,5 vezes maior que a escala primária. Esta janela é definida pela equação Gaussiana

$$g(\triangle x,\triangle y,\sigma') = \frac{1}{2\pi\sigma'^2} e^{-(\triangle x^2+\triangle y^2)/2\sigma'^2} \quad (16)$$

Onde $\triangle x \; e \; \triangle y$ são as distâncias entre cada ponto verificado e o ponto-chave.

O valor dos pesos calculados para cada ponto na vizinhança em (x, y) é atualizado na expressão:

$$h'_\theta = h_\theta + \alpha m(x,y).\, g(\triangle x,\triangle y,\sigma') \quad (17)$$

com

$$\alpha = \begin{cases} d/\, i, d < i \\ \overline{0, d > i} \end{cases}$$

Onde h'_θ é a atualização de h_θ , e d é a distância absoluta em graus entre a orientação do ponto e o θ discretizado, e i é o intervalo em graus entre os θ' discretizados.

Os picos no gráfico de orientação correspondem às direções dominantes da inclinação local. Além do valor máximo, também são considerados valores máximos correspondentes a pelo menos 80% do valor desse valor máximo. Assim, um mesmo ponto-chave pode ter mais de uma direção associada.

O pico deste histograma é usado para definir sua orientação. No caso de múltiplos picos com alta amplitude, o ponto-chave adquirirá múltiplas orientações, tornando-o ainda mais estável para futuras identificações. No final, a parábola é usada para interpolar os três valores do histograma mais próximos do pico para obter uma melhor precisão de sua posição.

A Figura 5 apresenta vários pontos-chave identificados em uma imagem de uma superfície metálica, cujas magnitudes e orientações são representadas por vetores.

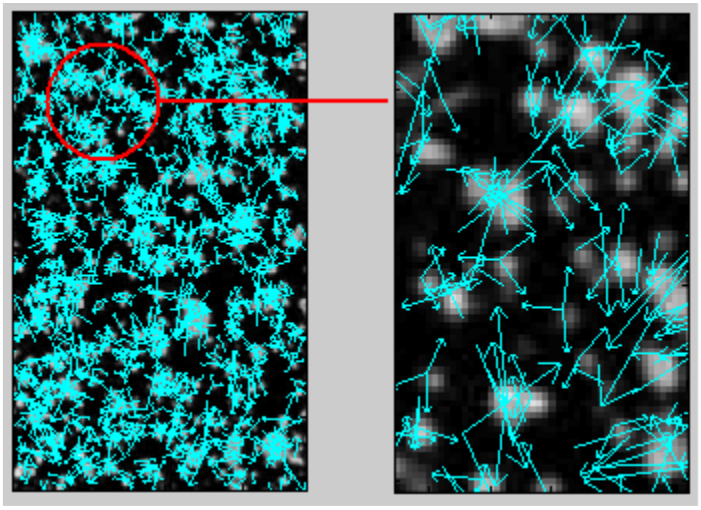


Figura 5 Atribuição de orientação e magnitude a cada ponto-chave. Fonte: Autoria Própria.

Cada ponto-chave tem agora quatro dimensões: posição x e y; magnitude; orientação.

2.3 Construção do Descritor Local

Nesta seção, cada ponto-chave será atribuído a um descritor invariante para iluminação e ponto de vista 3D, o que os tornará bem distinguíveis. É importante lembrar que os procedimentos a seguir serão realizados com valores normalizados contra a direção e magnitude do gradiente definido na seção anterior para cada ponto-chave.

Para que os identificadores tenham invariância rotacional, as orientações de gradiente desses pontos são rotacionadas por um ângulo correspondente à orientação do ponto-chave definida na seção anterior.

Um descritor de ponto-chave é então criado calculando as magnitudes e orientações dos gradientes que são amostrados em torno do local do ponto-chave. Este procedimento é ilustrado na Figura 4.7, onde os gradientes são mostrados por pequenas setas em cada local da amostra. As regiões de amostragem Nxn com kxk pixels ao redor da localização do ponto-chave são definidas.

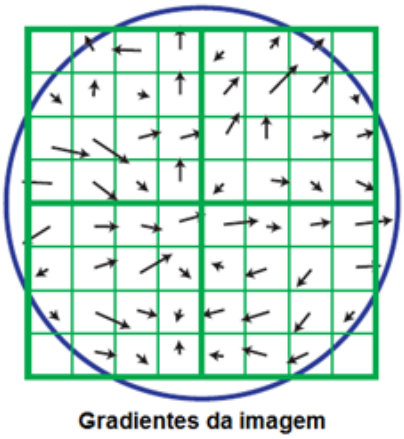
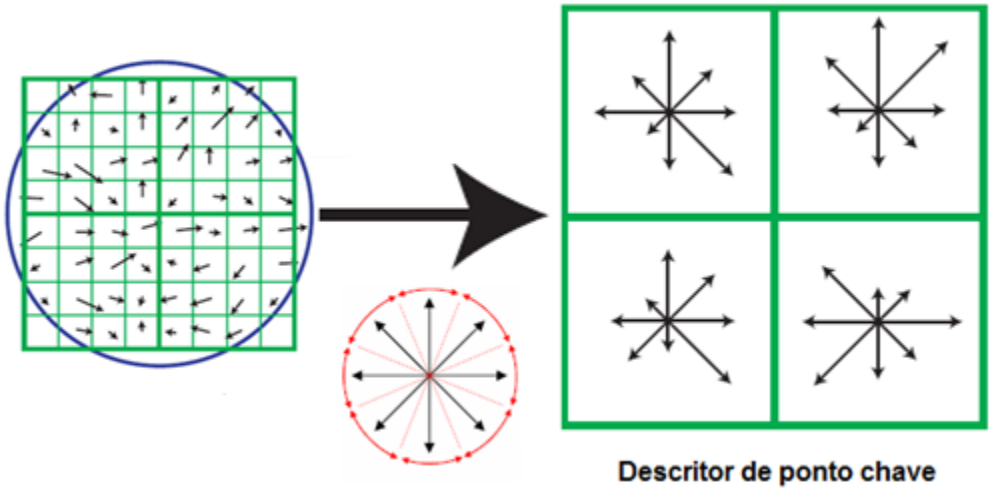


Figura 6 Mapa de gradientes para $n = 2$ regiões e $k = 4$ pixels. (Lowe, 2004).

A função gaussiana é usada para ponderar a magnitude do gradiente em cada ponto nas proximidades do ponto-chave, com uma janela de suavização Gaussiana dimensionada para metade da largura da janela de descrição. Este gaussiano evita mudanças abruptas do descritor para pequenas mudanças na posição da janela e também reduz a ênfase em gradientes distantes do centro do descritor, que são mais afetados por erros.

Uma vez realizada a suavização dos gradientes, o descritor consiste em um vetor contendo os valores do histograma. No exemplo da Figura 7, o histograma tem 8 valores de orientação, cada um criado ao longo de uma janela de fundo de $4x4$ pixels. O vetor de recursos resultante possui 128 elementos com uma janela de suporte total de $16x16$ pixels.



.

Figura 7 Construção do descritor para um ponto-chave de $2x2$ com 48 elementos. (Lowe, 2004).

No entanto, duas imagens do mesmo objeto podem ter variações de brilho que alteram significativamente os descritores resultantes. Assim, normaliza-se que o identificador é imutável contra a iluminação.

Os descritores são invariantes a mudanças homogêneas no brilho da imagem porque essa mudança representa a adição de uma constante a todos os pixels da imagem e os descritores são calculados a partir das diferenças de pixel. Já as mudanças de contraste homogêneas, representadas pela multiplicação de todos os pixels por uma constante, são corrigidas pela normalização dos descritores.

As variações não lineares causadas pela saturação da câmera ou pelo efeito de iluminação de superfícies tridimensionais em diferentes orientações podem ter um grande efeito no tamanho dos descritores, mas com pouco efeito na orientação. Este efeito é reduzido armazenando um valor máximo no tamanho. Após a normalização, todos os valores acima de um determinado limite são ajustados para esse limite. Isso é feito para que as direções com magnitudes muito grandes não dominem a representação do descritor. Lowe sugere usar um limite de 0,2. Isso significa que a correspondência para grandes tamanhos de gradiente não é tão importante em comparação com a distribuição de orientações.

Para cada imagem, vários descritores são gerados, cada um referente a um ponto-chave. O resultado é, portanto, um conjunto de descritores robustos que podem ser usados para casar a imagem com outra imagem, conforme detalhado na próxima seção. Mais detalhes sobre a estrutura dos identificadores SIFT podem ser encontrados em Lowe (2004).

2.4 *Matching*: Encontrando os Pontos em Comum

A ideia da *matching* é extrair pontos-chave de duas imagens e procurar os pontos correspondentes em cada imagem, conforme ilustrado na Figura 8. Compare as pontuações com base na semelhança das respectivas descrições.

Ter uma solução poderosa para o problema de localização de similaridade pode ser considerado um elemento-chave na automatização de tarefas de fotografia (Schenk, 1999). Muitos aplicativos de Visão Computacional requerem a identificação de elementos repetitivos entre duas imagens.

Figura 8 Processo de correspondência entre duas imagens através da técnica SIFT. Fonte: Autoria Própria.

Ao trabalhar com SIFT, os pontos de interesse são detectados pelo método e representados por identificadores. Os descritores são vetores que podem ser comparados, por exemplo, usando a distância euclidiana. Normalmente, os melhores candidatos de correspondência são pontos próximos, portanto, o melhor candidato é o ponto com a menor distância euclidiana.

Lowe utilizou uma modificação do algoritmo Árvore k-d chamado de método de Best-Bin-First (BBF) (Beis & Lowe, 1997), que pode identificar os vizinhos mais próximos com elevada probabilidade, utilizando apenas uma quantidade limitada de esforço computacional.

O problema de emparelhamento é assim reduzido a encontrar o vizinho mais próximo. No entanto, alguns pontos instáveis são detectados ao longo do caminho, resultando em falsas correspondências. Para eliminar esse problema, um método para comparar a distância mais curta com a segunda melhor distância é usado, selecionando apenas correspondências próximas por um limite. Lowe rejeitou todas as correspondências em que a razão de distância era

maior que 0,8, o que eliminou 90% das correspondências falsas, mas rejeitou apenas menos de 5% das correspondências corretas. Portanto, as correspondências são efetivamente refinadas e as correspondências falsas são descartadas.

2.5 Aplicações do SIFT

2.5.1 Reconhecimento de objetos

O SIFT é usado para detectar e considerar objetos em imagens, independentemente de sua escala e orientação. Os exemplos incluem sistemas de reconhecimento de imagem e aplicações de realidade aumentada.

2.5.2 Reconstituição 3D

O SIFT utiliza correspondências de pontos-chave entre múltiplas imagens, facilita a visualização tridimensional de cenas e objetos e é amplamente utilizado em fotogrametria e modelagem 3D.

2.5.3 Visão Robótica

Na robótica, o SIFT é usado em navegação e mapeamento, permitindo que robôs identifiquem e naveguem em ambientes complexos usando visão computacional.

2.5.4 Registro de imagens médicas

Usado para alinhar diferentes imagens médicas (como ressonância magnética e tomografia computadorizada) para diagnóstico.

2.5.5 Realidade Aumentada (AR)

O SIFT detecta e rastreia pontos-chave em ambientes reais, sobrepondo informações digitais com precisão.

2.5.6 Mosaico de imagens

Usado para múltiplas imagens em uma única imagem panorâmica, detectando e combinando características comuns.

2.5.7 Monitoramento de tráfego

Nos sistemas de vigilância, o SIFT ajuda a identificar e rastrear veículos e pedestres sob diferentes condições de iluminação e clima.

2.5.8 Recuperar imagens por conteúdo

Facilita a busca e recuperação de imagens em grandes bases de dados e compara características das imagens consultadas.

2.5.9 Detecção e correspondência de pontos de interesse em imagens de satélite

Usado para mapear e monitorar mudanças nas imagens de satélite ao longo do tempo.

2.5.10 Processamento de Documentos

SIFT é usado para alinhar e comparar diferentes versões de documentos.

2.5.11 Reconhecimento de Faces

Embora também seja comumente associado ao reconhecimento de objetos genéricos, o SIFT pode ser aplicado no reconhecimento de faces.

2.5.12 Segurança e Vigilância

Sistemas de segurança utilizam SIFT para identificar e rastrear intrusos ou atividades suspeitas em áreas monitoradas por câmeras.

2.5.13 Outras Aplicações

O SIFT também tem aplicações em áreas como a análise forense de imagens, onde a robustez contra a manipulação é crucial, bem como em sistemas de vigilância e monitoramento.

3. Resultados

Para a realização dos testes desse trabalho foi utilizada uma máquina Acer com processador i5, 6,00 GB de memória RAM, disco SSD 445 GB. As imagens foram obtidas da internet utilizando o site do Google.

A configuração dos parâmetros do algoritmo está apresentada na tabela 1:

Número de oitavos	4
Número de intervalos	5
k	$\sqrt{2}$.
σ	1.6
Limiar de constrate	7.65
Limiar de curvatura	10

Tabela 1: Parâmetros. Fonte: Autoria Própria.

- Imagem recortada
- Imagem rotacionada 10^0
- Imagem reduzida pela metade
- Imagem com ruído HSV
- Imagem artística com estilo cartoon

Os resultados são expostos na seção seguinte.

3.1 Imagem Recortada

Observa-se que neste caso, figura 9, ele faz uma boa ligação entre os pontos originais e os pontos restantes. No entanto, liga pontos sem correspondência.

Os resultados do primeiro teste estão apresentados na tabela 2:

Keypoints primeira imagem	1789
Keypoints segunda imagem	959
Número de matches	233
Porcentagem de matches	24.3%

Tabela 2: Resultados 1^0 teste. Fonte: Autoria Própria.

3.2 Imagem Rotacionada

Com um ângulo de rotação pequeno, percebe-se que faz uma associação razoável entre alguns pontos, mas também há muitas associações incorretas.

Os resultados do segundo teste estão apresentados na tabela 3:

Keypoints primeira imagem	1789
Keypoints segunda imagem	2093
Número de matches	271
Porcentagem de matches	15.1%

Tabela 3: Resultados 2^0 teste. Fonte: Autoria Própria.

3.3 Imagem Reduzida

O ponto forte do algoritmo é que ele é invariante a mudanças de escala e o teste mostra que houve forte associação correta, com poucos pontos errados presentes.

Os resultados do terceiro teste estão apresentados na tabela 4:

Keypoints primeira imagem	1789
Keypoints segunda imagem	870
Número de matches	176
Porcentagem de matches	20.2%

Tabela 4: Resultados 3^0 teste. Fonte: Autoria Própria.

3.4 Imagem com ruído

Neste caso, um ruído HSV foi usado para distorcer a imagem. Percebe-se que o algoritmo faz boas correspondências.

Os resultados do quarto teste estão apresentados na tabela 5:

Keypoints primeira imagem	1789
Keypoints segunda imagem	17509
Número de matches	315
Porcentagem de matches	17.6%

Tabela 5: Resultados 4^0 teste. Fonte: Autoria Própria.

3.5 Imagem artística

No caso de uma imagem em estilo cartoon, o desempenho mais fraco do algoritmo será notado com a menor porcentagem de correspondência possível, mas ainda assim algumas associações corretas podem ser observadas. Veja a figura 9.

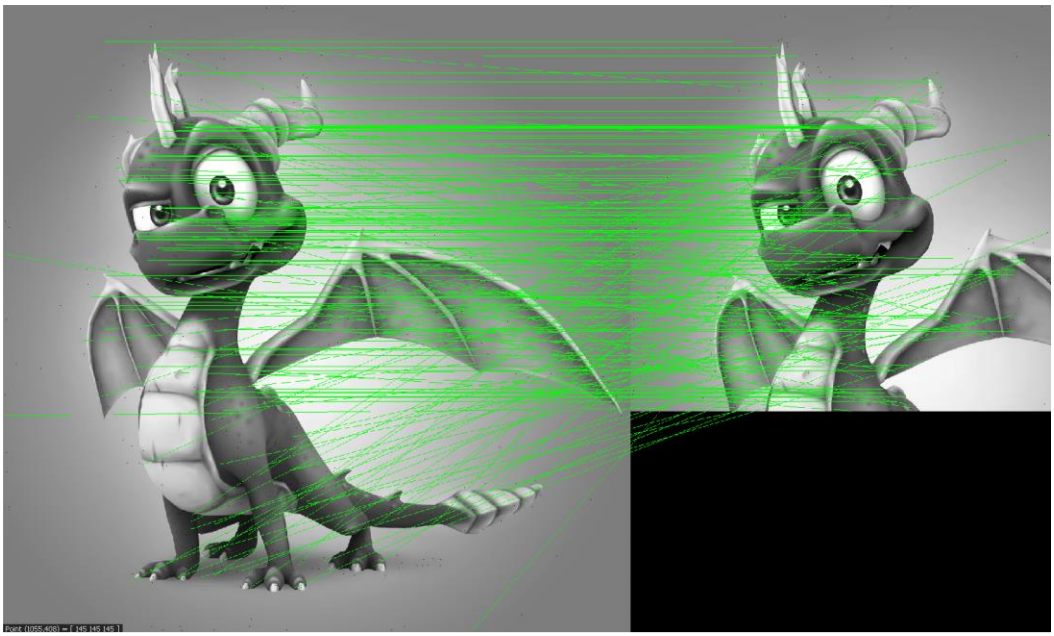


Figura 9: Imagem recortada. Fonte: Autoria Própria.

Na próxima etapa, essa mesma imagem passa pelo processo de rotação. Veja a figura 10.

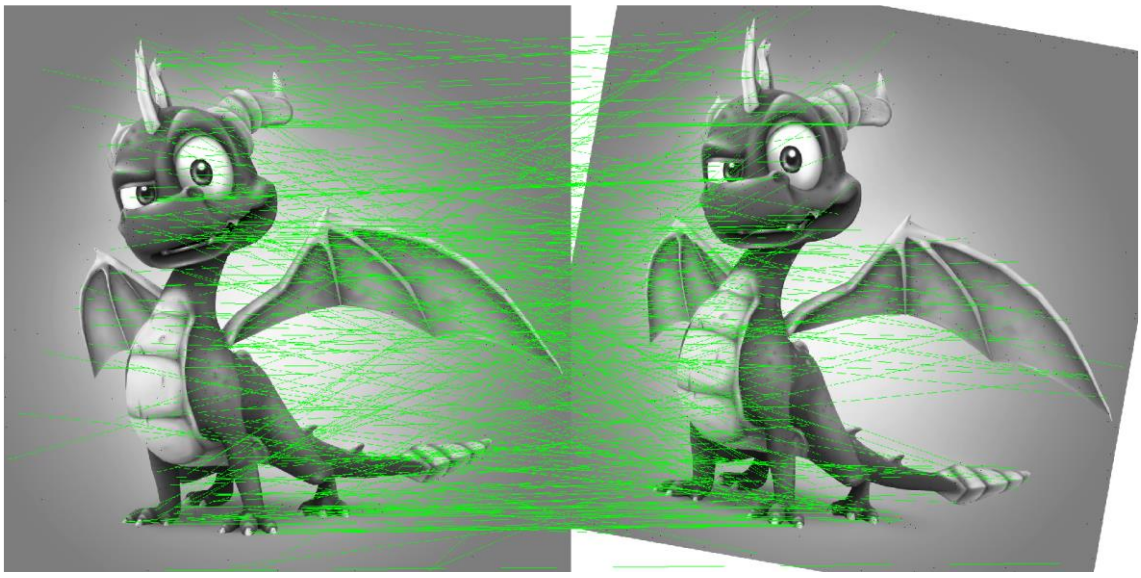


Figura 10: Imagem rotacionada. Fonte: Autoria Própria.

Em sequência, acontece o processo de redução. Veja a figura 11.

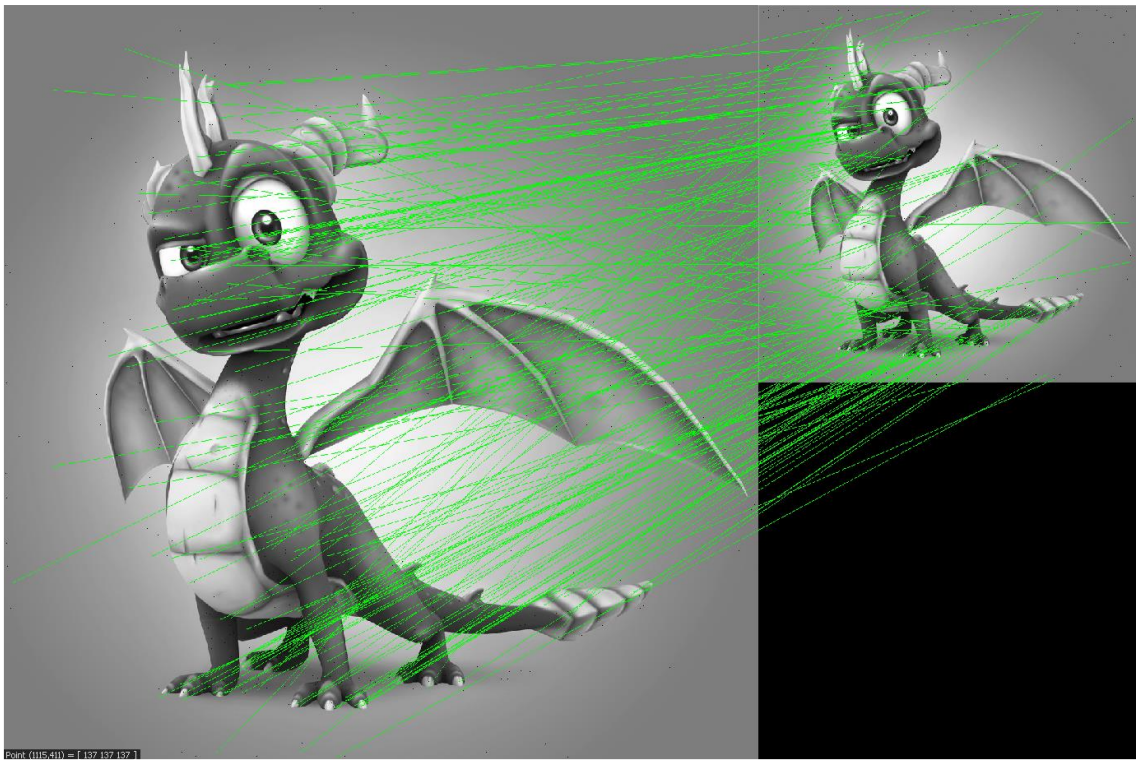


Figura 11: Imagem reduzida. Fonte: Autoria Própria.

Abaixo temos como resultado uma imagem com ruído. Veja a figura 12.

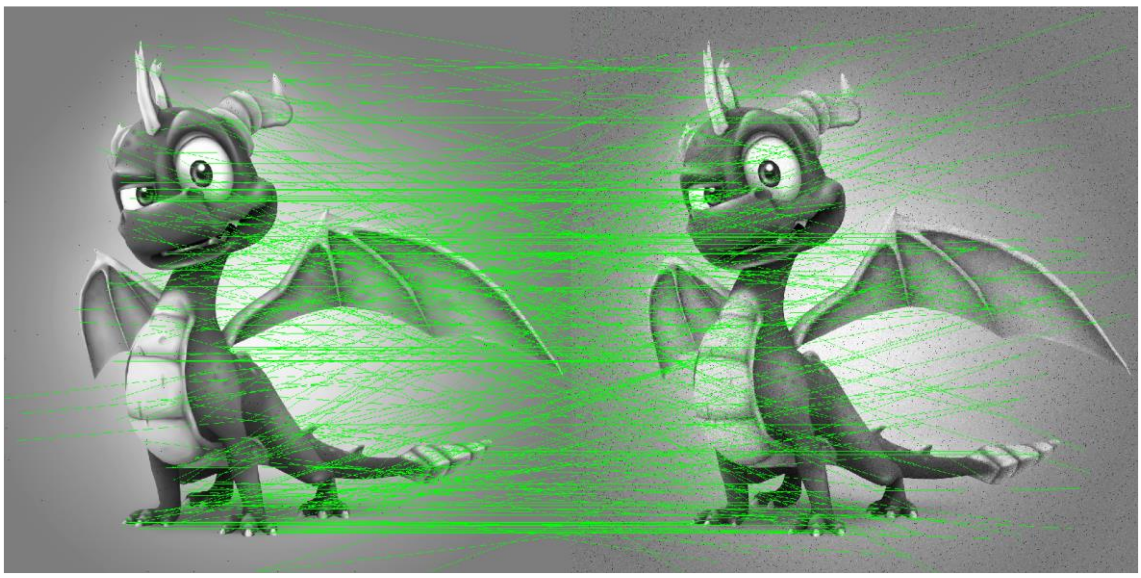


Figura 12: Imagem com ruído. Fonte: Autoria Própria.

Os resultados do quinto teste estão apresentados na tabela 6:

Keypoints primeira imagem	1789
Keypoints segunda imagem	2935
Número de matches	224
Porcentagem de matches	12.5%

Tabela 6: Resultados 5^0 teste. Fonte: Autoria Própria.

Finalizando o processo, temos uma imagem artística. Veja a figura 13.

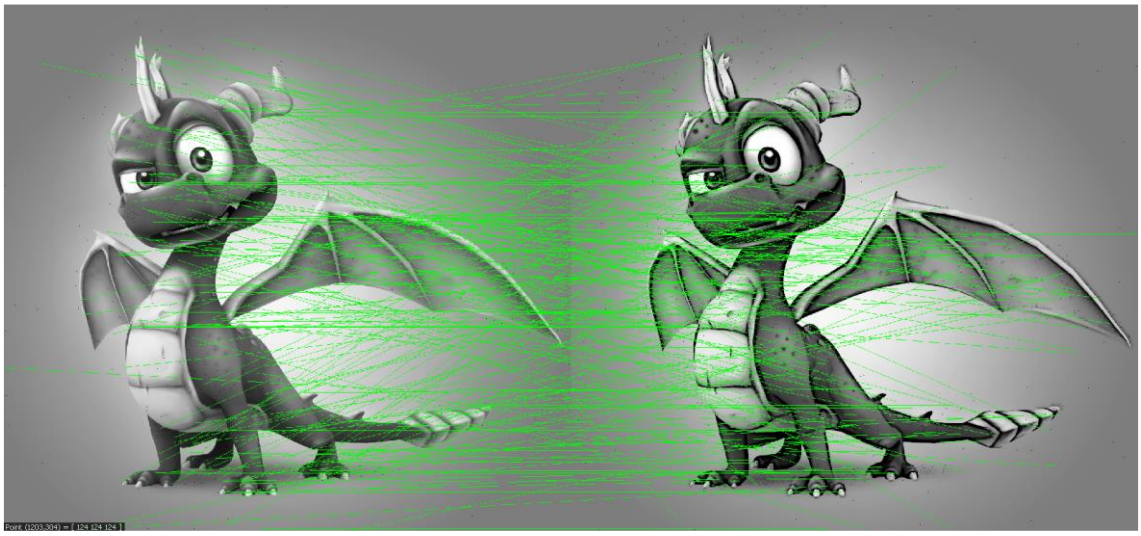


Figura 13: Imagem artística. Fonte: Autoria Própria.

A seguir, será mostrado o desempenho do algoritmo com outras imagens, sendo realistas ou não.

Observou-se com os testes que imagens geradas por computador apresentam menos ruído e, portanto, não geram tantos keypoints ruins. De qualquer modo, esse é um aspecto a ser melhorado na implementação. Veja a figura 14.

Figura 14: Lena com escala ampliada. Fonte: Autoria Própria.

Percebe-se bem que os keypoints ruidosos atrapalham o matching, embora haja algumas associações corretas.

Abaixo temos mais um exemplo. Veja a figura 15.

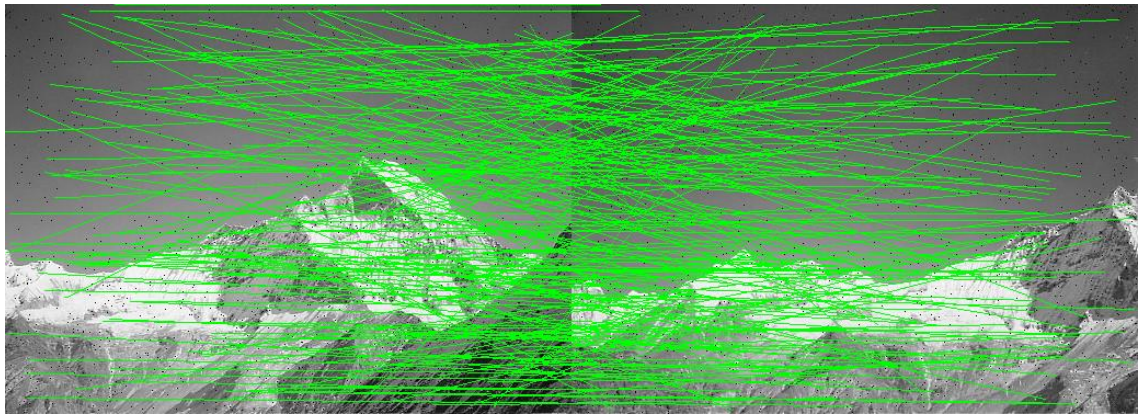


Figura 15: Duas fotos da mesma montanha, mas transladada. Fonte: Autoria Própria.

Pode-se ver, assim como na imagem da Lena, que a presença de keypoints ruidosos atrapalhou o matching. Sem eles, o desempenho do algoritmo seria melhor.

Observa-se ainda um link recortado. Veja a figura 16.

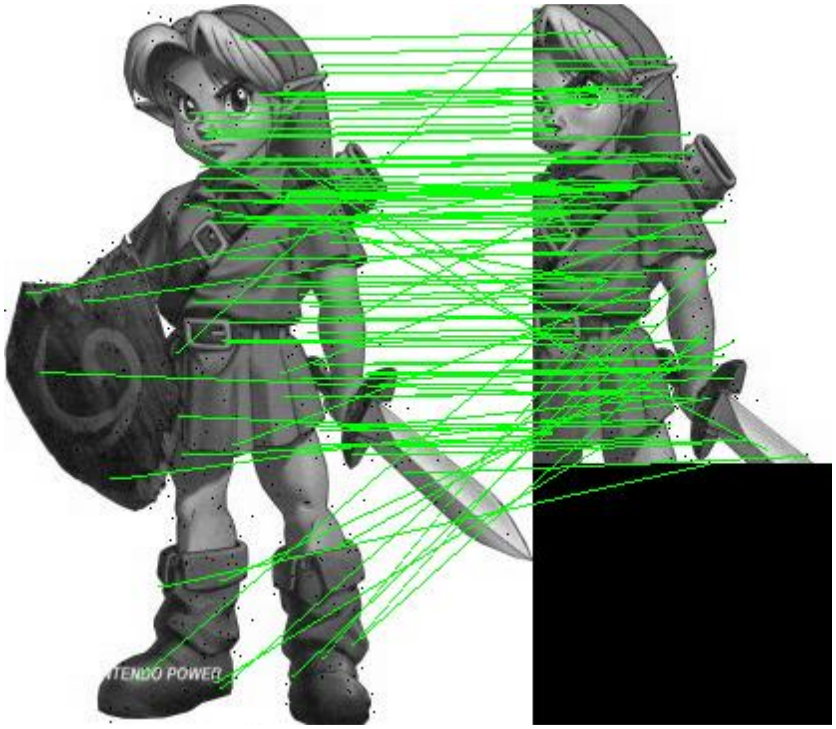


Figura 16: Link recortado. Fonte: Autoria Própria.

Podemos observar o mesmo comportamento apresentado na seção anterior, associação em sua maioria correta.

4. **Conclusões**

O objetivo deste trabalho foi apresentar o algoritmo SIFT para detecção de similaridade entre imagens. Para isso, uma breve introdução da sua aplicação foi apresentada. Em seguida, o funcionamento dele foi apresentado.

Neste estudo, o algoritmo SIFT é implementado e testado para diferentes perturbações da imagem. Apesar de alguns desencontros, podemos observar que pode fazer boas associações.

O ponto fraco da implementação é que ela atinge muitos pontos-chave que não são úteis.

Não foi realizada uma análise de tempo, mas fica claro que para imagens grandes como as utilizadas no experimento, que possuem tamanho de 1280 x 1280, o tempo de execução do algoritmo é proibitivo para execução em tempo real.

Percebeu-se que o algoritmo teve desempenho pior do que as imagens rotacionadas. Tem desempenho aceitável apenas quando o ângulo de rotação é pequeno. Como trabalho futuro, é necessário melhorar o desempenho do algoritmo nos seguintes aspectos:

- Diminuir número de keypoints ruidosos
- Melhorar desempenho de match
- Melhorar invariância à rotação.

É esperado também que esta pesquisa possa inspirar e incentivar o desenvolvimento de novas metodologias e técnicas, além disso, que contribua na promoção do ensino e aprendizagem e até mesmo desperte novas pesquisas e soluções.

Com isso recomenda-se, que seja mais bem explorado o tema aqui tratado; se busque alcançar resultados ainda melhores, compare resultados

utilizando técnicas diferentes, por exemplo, rede neural convolucional; implemente novas metodologias; e faça comparação com outras técnicas.

5. Referências

MARQUES FILHO, Ogê; NETO, Hugo Vieira. Processamento digital de imagens. Brasport, 1999.

Lowe, DG (2004). Recursos de imagem distintos de pontos-chave invariantes à escala. Jornal Internacional de Visão Computacional, 60(2), 91-110.

Bay, H., Tuytelaars, T. e Van Gool, L. (2006). SURF: Recursos robustos e acelerados. Na Conferência Europeia sobre Visão Computacional (pp. 404-417). Springer, Berlim, Heidelberg.

Rublee, E., Rabaud, V., Konolige, K. e Bradski, G. (2011). ORB: Uma alternativa eficiente ao SIFT ou SURF. Em 2011, Conferência Internacional sobre Visão Computacional (pp. 2564-2571). IEEE.

MOREIRA, Fabiano Cordeiro. **Reconhecimento e Classificação de Padrões de Imagens de Núcleos de Linfócitos do Sangue Periférico Humano com a Utilização de Redes Neurais artificiais.** 2002. 69 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Universidade Federal de Santa Catarina, Florianópolis, 2002. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/82305/188164.pdf?sequence=1>. Acesso em: 05 mai. 2024.

PEDRINI, H.; SCHWARTZ, W. R.; **Análise de imagens digitais**. **Princípio, algoritmos e aplicações**, Thomson, introdução P. 1-9, 2008.

GONZALES, Rafael C; WOODS, Richard E. Digital Image Processing Using MATLAB. 2. ed. New York: Gatesmark Publishing, 2009. 827 p.

M.S. MAILLET, M.Y. SHARAIHA, Binary Digital Image Processing, An Discrete Approach, 1th ed. Academic Press, London, 2000.

MATHWORKS. Color-based segmentation using k-means clustering. Disponível

em: <https://www.mathworks.com/help/images/color-based-segmentation-using-k- means-clustering.html;jsessionid=309fb8f6810bca075505d6fe629b> Acesso em: 05 mai. 2024.