Processamento de Imagens - Universidade Federal de Rondônia.

Jáder Louis de Souza Gonçalves[1]

Nataly Bonfim Tobias[2]

Orientador:Dr. Lucas Marques de Cunha[3]

INTRODUÇÃO

Na era atual da tecnologia digital, o processamento de imagens desempenha um papel crucial em uma variedade de campos, variando do entretenimento à ciência e tecnologia. Reconhecendo essa necessidade, o projeto "Lumina Pixel Studio" foi desenvolvido como um software educacional de testes para processamento de imagens. O objetivo principal deste trabalho é desenvolver o aplicativo Lumina Pixel Studio, que contém a integração de várias funcionalidades essenciais para o processamento de imagens. Entre estas funcionalidades, incluem-se operações algébricas como dissolve cruzado e dissolve cruzado não-uniforme; transformações de intensidade, que abrangem técnicas como negativo, alargamento de contraste, limiarização, transformação de potência e transformação logarítmica; a manipulação de histogramas através de expansão e equalização; e o controle de contraste adaptativo com parâmetros ajustáveis. Adicionalmente, o software incorpora transformações geométricas como mudança de escala, cisalhamento, rebatimento, rotação, e outras técnicas mais avançadas. Filtragem linear e não-linear, detecção de bordas usando o Gradiente de Sobel, aprimoramento de bordas e convolução com ajuste de offset também são aspectos fundamentais deste projeto. Estas melhorias visam equipar o Lumina Pixel Studio para atender às demandas

variadas do processamento de imagens digitais em diversos contextos, proporcionando uma ferramenta robusta e versátil para acadêmicos e entusiastas do campo.

REVISÃO DE LITERATURA

2.1 O que é Processamento de Imagens

O Processamento de Imagens é um campo que se dedica à modificação e análise de imagens através do uso de computadores. Conforme explicado por Chavéz (2013), este campo envolve a alteração das informações contidas nas imagens sob diversos aspectos. O objetivo pode ser tanto a produção de uma nova imagem quanto a extração de informações específicas a partir da imagem original. Este processo é realizado através de um conjunto de técnicas que permitem capturar, representar e transformar imagens com o auxílio de tecnologias computacionais.

O emprego destas técnicas no processamento de imagens facilita a percepção humana e a interpretação automática, melhorando a qualidade visual das imagens e permitindo a extração e identificação de informações valiosas contidas nela. Essas técnicas são aplicadas em diversas áreas, incluindo a visão computacional e o reconhecimento de padrões, que são exemplos de análise de imagens. Dentro do Processamento de Imagens, pode-se distinguir entre três principais categorias com base no tipo de entrada e saída. A primeira é o próprio processamento de imagens, onde a entrada e a saída são ambas imagens. A segunda categoria é a síntese de imagens, na qual a entrada é uma representação tridimensional e a saída é uma imagem. Por último, a análise de imagens, que começa com uma imagem como entrada e produz como saída informações sobre formas e outras características tridimensionais.

2.1.2 Transformação de Intensidade

A Transformação de Intensidade, segundo Ponti Jr. (2013), é um aspecto fundamental do processamento de imagens digitais. Essa técnica envolve a manipulação dos valores de intensidade (ou níveis de cinza) em uma

imagem para alcançar resultados específicos, como realçar características, corrigir contraste, ou modificar a aparência geral da imagem.

No contexto do processamento de imagens, a transformação de intensidade opera diretamente sobre os pixels da imagem. Cada pixel de uma imagem em escala de cinza tem um valor de intensidade, que geralmente varia de 0 (preto) a 255 (branco). Ao aplicar uma transformação de intensidade, esses valores são ajustados de acordo com uma função matemática específica.

2.1.3 Detecção de bordas

A detecção de bordas, conforme descrito por Seara e Elizandro (2013), é uma técnica crucial no processamento de imagens digitais. Esta técnica é focada na identificação de pontos em uma imagem onde ocorre uma mudança brusca ou uma descontinuidade nas intensidades. As bordas identificadas em uma imagem são importantes para a compreensão da estrutura das formas presentes na cena, sendo fundamentais para a análise e interpretação de imagens em diversas aplicações. No processo de detecção de bordas, algoritmos são utilizados para localizar e marcar as mudanças de intensidade que caracterizam as bordas de objetos em uma imagem. Essas mudanças são geralmente indicativas de limites ou contornos de objetos, variações de textura, ou outras informações visuais importantes.

Existem várias técnicas e operadores para a detecção de bordas. Alguns dos mais comuns incluem o operador de Sobel, o operador de Prewitt e o detector de bordas de Canny. Cada um desses métodos utiliza diferentes abordagens matemáticas para identificar as bordas, variando em sensibilidade e precisão.

2.1.4 Transformação geométrica

A Transformação Geométrica, segundo Mezadri e Pissini (2016), é um aspecto importante no campo do processamento de imagens. Esta técnica envolve a alteração da geometria de uma imagem, incluindo operações como rotação, escala, translação, e distorção. O objetivo dessas transformações é modificar a posição ou a orientação dos objetos dentro da imagem, sem alterar suas propriedades intrínsecas, como cor ou textura. No processamento de imagens, a transformação geométrica é frequentemente utilizada para corrigir distorções, alinhar imagens, ou para mudar a perspectiva de onde uma imagem é vista. Essas transformações são fundamentais em uma variedade de

aplicações, desde a edição de imagens digitais até a visão computacional e a análise de imagens médicas.

Existem diferentes tipos de transformações geométricas:

- **Transformações Lineares:** Incluem rotação, escala e cisalhamento. Por exemplo, a rotação gira a imagem em torno de um ponto central, a escala aumenta ou diminui o tamanho da imagem, e o cisalhamento inclina a imagem em um eixo específico.
- **Transformações Não Lineares:** Incluem operações mais complexas como distorções curvilíneas, que podem ser usadas para simular efeitos de lente ou correções de perspectiva.

Mezadri e Pissini (2016) destacam que as transformações geométricas devem ser aplicadas cuidadosamente, pois podem introduzir artefatos ou distorções indesejadas se não forem bem controladas. Além disso, essas transformações frequentemente requerem uma interpolação de pixels, pois a imagem transformada pode necessitar de valores de intensidade para novas posições de pixel que não estavam presentes na imagem original.

2.1.5 Operações Aritméticas

As operações aritméticas no processamento de imagens, conforme descrito por Chávez (2013), são procedimentos fundamentais que envolvem a manipulação de imagens através de operações algébricas básicas como soma, subtração, multiplicação e divisão, aplicadas pixel a pixel. Estas operações permitem a criação de uma nova imagem C(x,y) a partir da combinação de duas imagens de entrada A(x,y) e B(x,y).

- Adição de Imagens: A operação de adição, onde $C(x,y)=A(x,y)+B(x,y)$, é frequentemente usada para obter a média de múltiplas imagens da mesma cena. Isso é útil para reduzir ruídos aleatórios e pode ser empregado para sobrepor conteúdos de uma imagem à outra.
- Subtração de Imagens: Na subtração, $C(x,y)=A(x,y)-B(x,y)$, esta operação é útil para remover padrões indesejáveis, detectar mudanças entre duas imagens da mesma cena e pode ser utilizada para calcular gradientes, o que é importante na detecção de bordas.

- Multiplicação e Divisão de Imagens: Na multiplicação, $C(x,y)=A(x,y)*B(x,y)$, e na divisão, $C(x,y)=A(x,y)/B(x,y)$, essas operações são usadas para corrigir defeito ou para aplicar máscaras que realçam ou ocultam partes específicas da imagem.

2.1.5.1 Operações Locais

Chávez (2013) descreve as operações locais, como uma vizinhança de pixels para modificar o valor de um ponto específico. Estas operações são cruciais para a filtragem espacial e para alterar a própria estrutura da imagem. Podem ser usadas para aguçar a imagem, acentuando mudanças de intensidade com filtros passa-altas, ou para suavizá-la, tornando as mudanças de intensidade menos abruptas com filtros passa-baixas. Essas operações também são fundamentais para procurar formas específicas na imagem através de padrões de busca ou para definir bordas e remover ruídos.

2.1.5.2 Convolução

Esta é uma operação matemática que combina duas funções para produzir uma terceira função. No contexto da pesquisa de Chávez (2013), processamento de imagens, a convolução é utilizada para aplicar um filtro sobre uma imagem. Isso é feito "convoluindo" a imagem com um kernel ou máscara, que é uma pequena matriz usada para aplicar efeitos como desfoque, nitidez, realce de bordas, entre outros. A convolução é uma ferramenta poderosa pois permite a aplicação de diversas operações de filtragem espacial de maneira eficiente e eficaz, sendo fundamental em muitas aplicações de processamento de imagens.

METODOLOGIA

[1] Jáder Louis de Souza Gonçalves, Ciência da Computação, UNIR, jaderlouis@proton.me.

[2] Nataly Bonfim Tobias, Ciência da Computação, UNIR. nataly.tobias11@gmail.com

[3] Orientador: Dr. Lucas Marques de Cunha, UNIR, lucas.marques@unir.br.

Neste projeto, houve a combinação das pesquisas Aplicada e Bibliográfica, iniciando-se com estudos fundamentais para estabelecer uma base sólida que permitisse o trabalho com as funções relevantes. Inicialmente, realizou-se um estudo aprofundado sobre as bibliotecas a serem utilizadas e o funcionamento das mesmas. Posteriormente, procedeu-se ao processamento

das funções sem o auxílio dessas bibliotecas (Numpy, OpenCV, PIL…), empregando-as apenas para aprimorar a performance e o funcionamento da aplicação.

O desenvolvimento das funções base ocorreu na linguagem **Octave**, com a realização de testes necessários para a posterior transposição para Python. Uma vez criadas todas as funções, partiu-se para o desenvolvimento da versão inicial da interface de usuário (front-end) do aplicativo, utilizando-se inicialmente a biblioteca **TKinter** devido à sua facilidade de uso. Contudo, essa escolha revelou-se problemática, pois o resultado final da interface mostrou-se insatisfatório e pouco atraente para o usuário. Diante disso, optou-se por refazer completamente o front-end na extensão da mesma biblioteca, chamada **Tkinter Custom** (https://customtkinter.tomschimansky.com/), alcançando-se um resultado final mais satisfatório, por mais que ainda não totalmente refinado.

O processo de desenvolvimento foi documentado e atualizado constantemente na plataforma **GitHub** (https://github.com/Prism411/luminapixelstudio-project), permitindo a verificação das etapas de criação da aplicação na plataforma. As funções de processamento de imagens foram desenvolvidas no período de aproximadamente dois meses, entre o início de dezembro e o final de janeiro. Já o desenvolvimento do front-end foi concluído em um mês, restando etapas de aprimoramento, verificação de erros e correção de problemas de desempenho. Fica evidente que há espaço para melhorias, as quais não foram totalmente exploradas devido às limitações de prazo.

4. Materiais e Métodos

No projeto Lumina Pixel Studio, o módulo imageProcesser.py é o núcleo das operações de processamento de imagem, apresentando uma gama de funções detalhadamente elaboradas para manipulação e transformação de imagens. Estas funções foram inicialmente criadas sem recorrer a bibliotecas externas como NumPy ou OpenCV, enfatizando a construção manual dos algoritmos, antes de integrar tais bibliotecas para aprimoramento de

desempenho.

- **“redimensionar_imagem”**: Modifica as dimensões espaciais de uma imagem, empregando algoritmos de redimensionamento para alterar sua resolução.
- **“dissolve_cruzado e dissolve_cruzado_nao_uniforme”:** Aplicam uma técnica de mesclagem que combina duas imagens de maneira proporcional, controlada pelo parâmetro alfa, que regula a transparência e sobreposição.
- **“negativo”:** Inverte o espectro de cores, convertendo cada pixel para seu complemento dentro do espaço de cores.
- **“alargamento_contraste”:** Ajusta o contraste da imagem ao modificar a escala de intensidade dos pixels, focando em faixas específicas de luminosidade.
- **“limiarizacao”:** Converte a imagem em uma representação binária, separando pixels com base em um limiar de intensidade pré-definido.
- **“transformacao_potencia”:** Realiza a correção gama, ajustando a luminosidade global da imagem através de uma transformação não-linear,
- **“transformacao_logaritmica”:** Aplica uma transformação logarítmica para mapear um intervalo maior de valores de brilho em um intervalo menor, realçando detalhes em áreas mais escuras.
- **“expand_histogram_auto” e “histogram_equalization”:** São métodos de ajuste do histograma para melhorar a distribuição de intensidades e aumentar o contraste da imagem. A primeira amplia o alcance dinâmico das intensidades, enquanto a segunda redistribui uniformemente as intensidades ao longo de todo o espectro disponível.
- **Transformações Geométricas:** As funções “scale_image”,

"shear_image" e "reflect_image" aplicam alterações matriciais para modificar a estrutura espacial da imagem, incluindo escalonamento, cisalhamento e reflexão.

- **"rotate_image" e "rotate_image2":** Implementam a rotação da imagem, aplicando cálculos trigonométricos e interpolação bilinear para manter a integridade dos pixels após a rotação.
- **"vertical_pinch" e "edge_pinch":** Introduzem efeitos de distorção, manipulando coordenadas de pixel para criar efeitos como compressão ou expansão em áreas específicas. "field_based_warping" executa transformações de warping baseadas em um campo de deslocamento, permitindo alterações complexas e personalizadas na topologia da imagem.
- **"aplicar_filtro_mediana" e "aplicar_filtro_media":** Implementam filtragem espacial para suavizar a imagem, atenuando ruídos através da média ou mediana dos valores de pixel em uma janela local. "aplicar_sobel" utiliza convolução para aplicar o operador de Sobel, realçando bordas horizontais e verticais por meio de gradientes de intensidade.
- **Amplificação e realce:** Funções como "agucamento_bordas", "high_boost" e "convolucao_com_offset" concentram-se no realce e na amplificação de detalhes da imagem, empregando técnicas de convolução com kernels específicos para intensificar bordas e texturas. As funções "sharpness_filter", "emboss_filter_one", "emboss_filter_two", e "deteccao_bordas" são uma extensão da função "convulacao_com_offset" servindo como máscaras (kernels) pré definidas para situações específicas.

5. Resultados e Discussões

Devido ao extenso período de desenvolvimento deste projeto, é possível categorizar os desafios enfrentados em três tipos: problemas crônicos de desenvolvimento, questões relacionadas ao processamento de imagens e dificuldades de portabilidade.

5.1 Problemas Crônicos de Desenvolvimento

Dentre os desafios crônicos de desenvolvimento, destacam-se as dificuldades inerentes ao próprio desenvolvimento do aplicativo. Para definir o ambiente de operação do aplicativo (desktop, mobile, web). Devido à ampla gama de funcionalidades que o aplicativo deveria oferecer, optou-se pelo desenvolvimento para desktop, principalmente por questões de performance e capacidade de processamento.

O desenvolvimento do projeto ocorreu simultaneamente aos desafios relacionados ao processamento de imagens. No início, certas deficiências no núcleo do aplicativo tornaram-se evidentes, como problemas de desempenho. Durante esse período, ainda não havíamos realizado a portabilidade para um modo independente de bibliotecas externas. A falta de atenção na resolução desses problemas acarretou uma crescente dívida técnica, resultando em um aplicativo com desempenho insatisfatório, interface visual pouco atraente e recorrentes falhas de funcionamento.

Após a conclusão da primeira fase do projeto e a realização de testes de estresse, ficaram evidentes essas deficiências. Decidiu-se então migrar o front-end do aplicativo para uma biblioteca mais moderna, que oferecesse uma interface nativa mais agradável. Esta decisão, no entanto, foi adiada para após a implementação completa das funcionalidades com o uso de bibliotecas como NumPy, OpenCV e PIL. A portabilidade para outro front-end e a execução manual das funções sem o uso de bibliotecas externas seriam realizadas em uma fase posterior.

5. 2 Problemas de Processamento de Imagens

Paralelamente aos problemas crônicos de desenvolvimento, emergiram desafios significativos relacionados ao processamento de imagens. As funções originalmente desenvolvidas em Octave apresentavam complexidades na transposição para Python, especialmente aquelas que requerem a passagem de matrizes como parâmetros, como nas operações de convolução ou ‘warp’ baseado em campos. A implementação dessas funções exigiu estratégias cuidadosas, considerando as diferenças entre as linguagens. Um desafio particular foi observado nas funções de rotação de imagens. A incidência de 'dead pixels', ou pontos pretos, tornou-se evidente durante a rotação das

imagens, indicando falhas na interpolação. Para resolver isso foi necessário desenvolver um método de rotação com interpolação bilinear partindo do zero, o que por sua vez, impactou negativamente o desempenho dessas funções. A complexidade computacional aumentou consideravelmente, frequentemente envolvendo múltiplos loops aninhados para processar todos os canais de cores.
Após a conclusão da portabilidade das funções de Octave para Python, iniciou-se a etapa de reescrita das funções em Python, sem o auxílio de bibliotecas de processamento de imagens prontas, como NumPy e OpenCV.

Essa abordagem resultou em uma adição no desempenho do aplicativo, com operações de processamento de imagens consumindo, em alguns casos, mais de 10 minutos para serem concluídas.

5.3 Resolução dos Problemas

Após a implementação das funções sem o uso de bibliotecas externas, iniciou-se o desenvolvimento do novo front-end do aplicativo, utilizando uma versão atualizada da mesma biblioteca previamente empregada. Este processo foi concluído sem maiores dificuldades. Contudo, desafios significativos relacionados ao desempenho ainda precisavam ser abordados. Um estudo aprofundado foi realizado para identificar estratégias eficazes de otimização de desempenho. Entre as medidas consideradas, estavam o uso de threading para paralelização de processos e a implementação de algoritmos de cache para dados frequentemente acessados.

Uma das soluções implementadas para reduzir a complexidade computacional foi a transformação de imagens em escala de cinza ou a binarização, o que contribuiu significativamente para a diminuição da carga de processamento. Adicionalmente, foi adotada a técnica de redução de resolução das imagens, permitindo um processamento mais ágil sem comprometer excessivamente a qualidade visual.

Embora outras estratégias de otimização fossem consideradas, como a utilização de algoritmos de compressão de imagens para reduzir o tamanho dos dados processados, decidiu-se priorizar o aprimoramento de outras funcionalidades do aplicativo. O desempenho alcançado foi considerado satisfatório para o escopo atual do projeto, com planos de retomar as otimizações adicionais em uma fase posterior.

6. CONCLUSÃO

Este projeto de desenvolvimento de software proporcionou uma compreensão abrangente e multifacetada dos desafios inerentes ao processamento de imagens digitais. Durante o processo, foram enfrentados e superados, obstáculos em três áreas principais: problemas crônicos de desenvolvimento, dificuldades específicas no processamento de imagens e questões de portabilidade.

Em termos de desenvolvimento, o projeto evidenciou a importância de escolhas estratégicas sobre o ambiente de desenvolvimento e a necessidade de balancear flexibilidade com desempenho. A experiência destacou particularmente a dívida técnica que pode acumular-se ao negligenciar aspectos fundamentais na fase inicial, como desempenho e estabilidade.

No que se refere ao processamento de imagens, a transposição de funções do Octave para Python revelou-se uma tarefa complexa, especialmente na adaptação de algoritmos para um novo ambiente de programação. A implementação de algoritmos de redimensionamento, mesclagem, transformações geométricas, ajustes de contraste, filtros e técnicas de realce de bordas forneceu *insights* valiosos sobre as nuances do processamento de imagens digitais. Esta experiência prática aprofundou o entendimento dos aspectos técnicos e teóricos envolvidos, desde a manipulação de pixels individuais até a aplicação de transformações complexas.

A experiência adquirida neste projeto será fundamental para o desenvolvimento na área de processamento de imagens. O conhecimento prático das limitações, desafios e soluções em processamento de imagens digitais proporciona uma base sólida para futuras pesquisas e projetos na área. Além disso, as habilidades adquiridas na solução de problemas complexos e na otimização de algoritmos são inestimáveis para qualquer empreendimento futuro em processamento de imagens ou em campos relacionados.

REFERÊNCIAS

CHÁVEZ, G. **Operações Algébricas e Lógicas.** 2013b. UFOP 2013. Disponível

em: http://www.decom.ufop.br/guillermo/BCC326/slides/Processamento-de-Imagens-Operacoes-Algebrica.pdf. Acesso em: 29 jan. 2024.

SEARA, D.; ELIZANDRO, G. **Encontrando a Linha Divisória: Detecção de Bordas**. 2013. - Inf, UFSC 2013. Disponível em: https://www.inf.ufsc.br/~aldo.vw/visao/bordas.pdf. Acesso em: 29 jan. 2024.

MEZADRI, F.; PISSINI, M. **TRANSFORMAÇÕS GEOMÉTRICAS** . 2016. UniCamp 2016. Disponível em: https://www.ime.unicamp.br/sites/default/files/lem/material/geometria_das_transformacoes.pdf. Acesso em: 29 jan. 2024.

CHÁVEZ, G. **Fundamentos** . 2013. decom.ufo. Disponível em: <http://www.decom.ufop.br/guillermo/BCC326/slides/Processamento-de-Imagens-Fundamentos.pdf>. Acesso em: 29 jan. 2024.

PONTI JR., M. **Realce de imagens parte1: operações pontuais**. Disponível em: <http://wiki.icmc.usp.br/images/5/5e/DIP_02_Realce_p1.pdf>. Acesso em: 29 jan. 2024.

Rodrigues da Silva, N., Pereira de Souza, F., & Romanini, E. (2018). **MÉTODO DE FATORAÇÃO LU PARA SOLUÇÃO DE SISTEMAS LINEARES**. Colloquium Exactarum. ISSN: 2178-8332, 9(4), 41–47. Recuperado de https://revistas.unoeste.br/index.php/ce/article