CRIAÇÃO DE UM PACOTE EM JULIA VOLTADO AO ENSINO DE PROCESSAMENTO DIGITAL DE IMAGENS ORBITAIS

Bruno Menini Matosak [1], *Nilcilene das Graças Medeiros* [2]

[1]Universidade Federal de Viçosa, bruno.matosak@ufv.br; [2]Universidade Federal de Viçosa, nilcilene.medeiros@ufv.br

RESUMO

O Processamento Digital de Imagens (PDI) orbitais requer comumente grande capacidade de processamento computacional, além de rotinas de programação robustas. Tais fatores acabam por encarecer ferramentas pagas de PDI, o que inviabiliza seu uso, inclusive para instituições com verba limitada. Uma saída é o uso de softwares *open source*, devido à sua proposta de disponibilização gratuita, porém, poucos destes pacotes possuem o seu enfoque principal numa interface amigável e de cunho para o ensino. Neste sentido, este trabalho visou a criação de um pacote com 11 funções implementadas visando a prática de ensino, em ambiente de programação Julia, que é uma linguagem gratuita e proeminente no cenário acadêmico quanto ao tempo de execução dos algoritmos. Para auxiliar as práticas de PDI que utilizam como dados imagens orbitais, foi escrito um texto didático para acompanhar o uso do pacote.

Palavras-chave — Processamento de Imagens, Linguagem Julia, Imagens Orbitais.

ABSTRACT

The digital image processing of satellite's images often requires great execution capacities of computer machines, as well as robust algorithms. This factors end up increasing the price of paid digital image processing tools, what hinders its use, mostly by institutions with low budgets. As an alternative, open source software can be used, due to their free distribution, however, just a few focus on friendly interfaces for teaching processes. In this regard, this work aimed the creation of a package with 11 functions, implemented aiming its use at teaching occasions, in Julia, a free programming language prominent in the academic field, when comparing the total execution time of algorithms. To help the practice of digital image processing that use orbital images, a didactic text was also created to be part of the package as well.

Key words — *Image's Processing, Julia Language, Satellite's Images*.

1. INTRODUÇÃO

O Processamento Digital de Imagens (PDI) é um ramo da computação definido como todo tipo de processamento cujas entradas ou saídas são imagens, envolvendo extração de atributos ou o reconhecimento de objetos individuais, por exemplo [1].

Exemplos de funções comumente usadas em PDI são filtros Passa-Baixa, que visam a redução de ruídos nas imagens através da suavização da cena, em contrapartida os filtros Passa-Alta visam o realce das feições contidas nas imagens, o que pode resultar também no realce dos ruídos [1]. Outro ramo do PDI estuda a aplicação de processos da Morfologia Matemática em imagens, com uso das operações base denominadas Erosão e Dilatação, podendo ser empregadas separadamente ou de forma combinada para os mais diversos fins, como a extração de bordas (Gradiente Morfológico) ou a separação de feições adjuntas (Abertura e Fechamento Morfológico) [1]. Tais ferramentas podem ser aplicadas em imagens orbitais, das quais normalmente deseja-se extrair informações espaciais.

Quando são utilizadas imagens orbitais como dados de entrada nos processamentos, visa-se eficiência nos processos computacionais e nas linguagens de programação, a fim de obter bons resultados nesse processo, com baixo tempo de execução, porém, devido à grande quantidade de informação que essas imagens possuem, uma vez que em suas aquisições são representadas grandes áreas territorias, tais dados demandam grande capacidade de processamento e armazenamento.

Softwares robustos, capazes de processar cenas de grandes dimensões, possuem normalmente um alto custo atrelado à sua aquisição, o que é justificável dado o trabalho envolvido na criação de ferramentas computacionais estáveis e de alto desempenho. Tal fator, porém, se mostra um limitante a alguns usuários, visto que tais ferramentas, normalmente, são caras e protegidas pelos criadores, impedindo os usuários de identificar o seu princípio de funcionamento.

Uma alternativa gratuita são os softwares *open source*, que possuem como característica principal a facilidade de acesso aos usuários, ou mesmo instituições de orçamento limitado, sendo este modelo de criação e publicação de software o mais apto a vencer as barreiras criadas pelas disparidades sociais [2]. A difusão do modelo de software livre ajudaria no uso e aplicabilidade de processamento de várias ferramentas, inclusive as de PDI, sendo o software livre detentor de alto potencial educativo para salas de aula, ensino a distância e colaborar na capacitação de professores.

Uma linguagem de programação *open source* que vem demonstrando alto potencial para o ganho de eficiência em

algoritmos de PDI é a Julia. Testes realizados indicam que sua velocidade de execução é equiparável ao C, com casos onde a linguagem mostrou-se até mais eficiente. Quando comparada a outras opções disponíveis, Julia demonstrou uma velocidade média de execução mais rápida do que opções gratuitas como o R, Scilab e Pithon, ou mesmo ferramentas pagas, como o MATLAB [3]. Além disso, Julia possui uma sintaxe simples, o que facilita o seu uso, e foi criada a fim de ser uma opção as linguagens mais conhecidas [4].

Todo o potencial desta linguagem de programação pode ser usado para o processamento digital de imagens orbitais, uma vez que bibliotecas necessárias para o carregamento e criação de novas imagens já são disponibilizadas gratuitamente, sendo possível a sua conversão em arranjos correspondentes aos níveis de cinza componentes das cenas – foi então criado um pacote para tal, visando o seu uso em sala de aula. Vantagens como a facilidade na análise do código fonte e a eficiência na execução dos mesmos são de grande interesse para o aprendizado. Um texto explicativo das funções se mostra uma opção importante aos alunos, para servir de apoio aos conceitos introduzidos e discutidos em sala de aula.

2. MATERIAIS E MÉTODOS

Para a criação de um pacote a ser usado no processamento digital de imagens orbitais, optou-se em utilizar o sistema operacional Ubuntu 18.04, visto que este é largamente usado em institutos de ensino, dada sua gratuidade, porém, o pacote pode ser usado em qualquer sistema operacional que seja possível a instalação da linguagem Julia.

Foram realizados estudos relacionados às funções de realce de imagens orbitais mais utilizadas, a fim de que o pacote fosse de interesse aos usuários da área, de modo a ser útil às atividades de ensino. Ao selecionar as funções a serem inseridas no pacote, visou-se oferecer diferentes opções, tais como: Filtros Passa-Baixa, além de uma abordagem adaptativa, Filtros Passa-Alta, Morfologia Matemática e processos de Limiarização.

Buscou-se realizar uma implementação que visasse manter a simplicidade do algoritmo, de forma que caso seja necessário analisar o código, este seja compreendido de forma mais didática e prática.

As funções componentes do pacote foram escritas com auxílio do software *Atom* – um editor de texto gratuito de código aberto. Por apresentar um sistema de compilação JIT (*Just in Time*), os arquivos salvos como texto no formato Julia (*.jl*) são compilados a medida que são executados, o que explica sua eficiência [5], principalmente com relação a outras linguagens que são interpretadas.

Os componentes do pacote, uma vez implementados, foram testados com recorte de uma imagem orbital, proveniente do satélite RESOURCESAT-1, sensor LISS-3, para averiguar erros na implementação.

Após completada a etapa de implementação das funções, o pacote foi disponibilizado no portal *Github*, para que fosse possível sua utilização por demais programadores. Através deste portal, é possível também sugerir correções e otimizações ao código, podendo este ser adequado a possíveis mudanças em versões mais recentes da Julia que, por ventura, sejam criadas.

Foi criado um texto explicativo de conceitos básicos de PDI e das funções implementadas, com exercícios para a fixação do conteúdo discutido. Tal texto se encontra junto da biblioteca no *Github* e pode ser acessado facilmente através de um *link* disponível na página do pacote.

3. RESULTADOS E DISCUSSÃO

Foram implementadas ao todo 11 funções, as quais estão descritas na Tabela 1, inclusive com seus respectivos parametros de entrada.

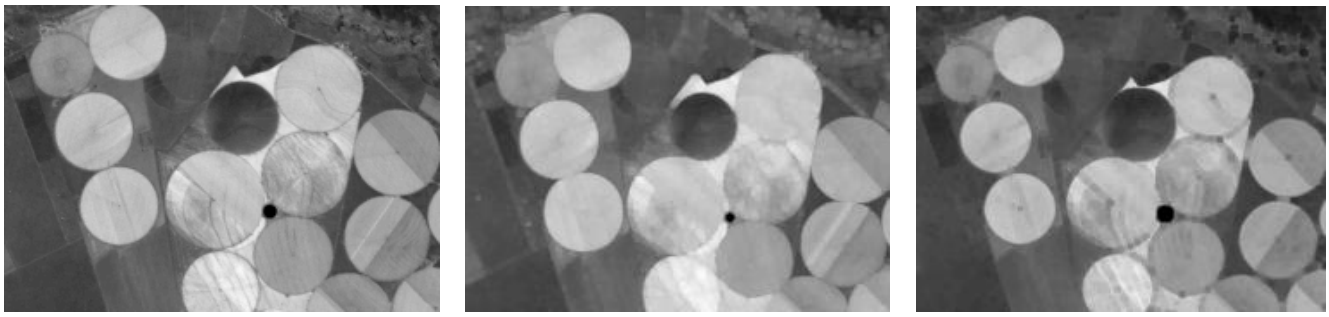
Tabela 1. Funções implementadas e parâmetros de entrada.

Nome	Função	Parâmetros de Entrada
Dilatação	*dilation_( )*	imagem, ordem da máscara (opcional)
Erosão	*erosion_( )*	imagem, ordem da máscara (opcional)
Filtro Gaussiano Adaptativo	*filter_gauss_localsigma( )*	imagem, ordem da máscara (opcional), c (opcional)
Filtro Gaussiano	*filter_gauss( )*	imagem, sigma (opcional), ordem da máscara (opcional)
Filtro de Média	*filter_mean( )*	imagem, ordem da máscara (opcional)
Filtro de Mediana	*filter_median( )*	imagem, ordem da máscara (opcional)
Operador de Sobel	*filter_sobel( )*	imagem, direção da borda (opcional)
Gradiente Morfológico	*grad_morf( )*	imagem, ordem da máscara (opcional)
Limiarização pela Mediana	*median_threshold( )*	imagem
Afinamento Morfológico	*morph_thinning( )*	imagem
Limiarização Método de Otsu	*otsu_threshold_( )*	imagem, "image" ou "lvl" (opcional)

Durante o processo de desenvolvimento, nomeou-se a biblioteca como IMGedu, que representa a junção de “IMG” caracterizando o processamento digital de imagens, e “edu” relativo ao caráter educacional do código escrito, assim como o texto de apoio. Exemplos da aplicação de algumas das funções implementadas (Tabela 1) podem ser vistos nas Figuras 1, 2, 3 e 4.

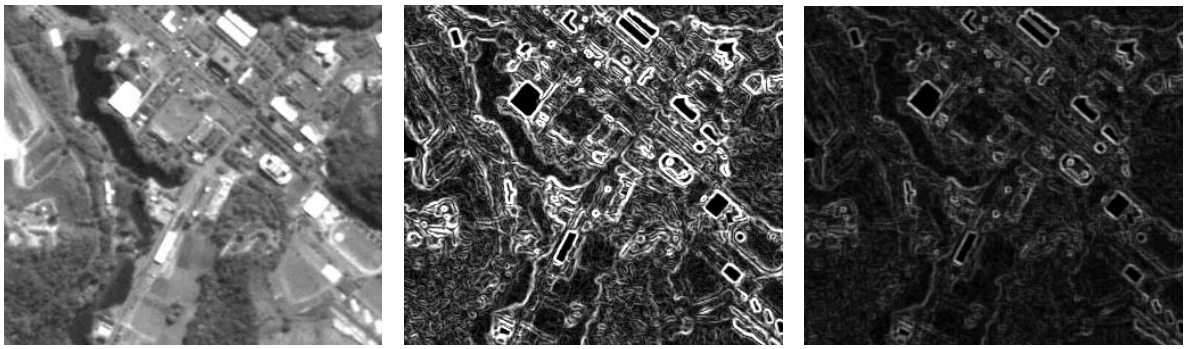
(a) (b) (c)

Figura 1. Recorte de imagem proveniente do satélite RESOURCESAT-1 sensor LISS-3 de uma região de pivôs centrais (a), recorte com aplicação do Filtro Gaussiano convencional (b) e recorte com aplicação de Filtro Gaussiano Adaptativo (c).



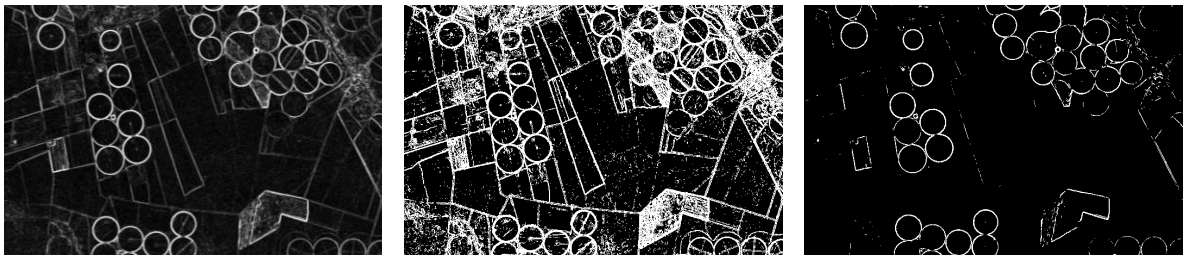
(a) (b) (c)

Figura 2. Recorte de imagem proveniente do satélite RESOURCESAT-1 sensor LISS-3 de uma região de pivôs centrais (a), aplicação da função de Dilatação (b), e aplicação da função de Erosão (c).



(a) (b) (c)

Figura 3. Recorte de uma imagem orbital da região do campus Viçosa da UFV (a), aplicação do Detector de Bordas de Sobel (b) e aplicação do operador Gradiente Morfológico (c).



(a) (b) (c)

Figura 4. Recorte de imagem orbital, no qual foi aplicado o Detector de Bordas de Sobel (a), aplicação da Limiarização pela Mediana em 4a (b) e aplicação da Limiarização pelo Método de Otsu em 4a (c).

Os resultados vistos nas imagens presentes nas Figuras 1, 2, 3, e 4 são relativos a aplicação das funções presentes no pacote em recortes de imagens orbitais. Na Figura 1 têm-se a aplicação de dois filtros em um recorte de imagem orbital visando a suavização da imagem e supressão de ruídos. Comparando os resultados obtidos pela aplicação de ambas funções, observa-se que as imagens suavizadas, porém a aplicação do Filtro Gaussiano Adaptativo (Figura

1c) demonstrou preservar mais a informação das bordas do que a aplicação do Filtro Gaussiano (Figura 1b).

A combinação das funções é usual em processos de PDI, como em processos morfológicos onde se combina as funções Erosão (Figura 2b) e Dilatação (Figura 2c) de diferentes formas para a definição de outros processos morfológicos, como uma ferramenta de detecção de bordas aplicando-se a subtração da operação de Dilatação do processo de Erosão, denominado Gradiente Morfológico (Figura 3c). Tal processo de detecção de bordas é comparável ao Operador de Sobel (Figura 3b), ambos visam a detecção de bordas de feições componentes da cena, porém os resultados são apresentados de forma diferente, pois, como pode ser visto, as bordas detectadas pelo Operador de Sobel possuem uma magnitude maior do que aquelas detectadas pelo Gradiente Morfológico.

No caso da Figura 4, foi inicialmente aplicada a função Detector de Bordas de Sobel na imagem original (Figura 4a), e em seguida duas versões de limiarização foram aplicadas: a Limiarização pela Mediana (Figura 4b) e pelo Método de Otsu (Figura 4c). Para o caso dessas funções de limiarização, enquanto a primeira apresentou um retorno maior das bordas contidas na cena, a segunda apresentou um melhor resultado quanto à diminuição de ruídos contabilizados como bordas.

A capacidade de combinar as funções da biblioteca é de significativa importância para a criação de métodos de PDI derivados, a serem desenvolvidos pelos usuários durante a etapa do aprendizado. A variação dos parâmetros de entrada das funções é feita de forma simples e podem ser manipulados a fim de encontrar outros resultados, mais propícios a aplicações específicas, contribuindo assim com o aprendizado em relação ao propósito de cada função implementada.

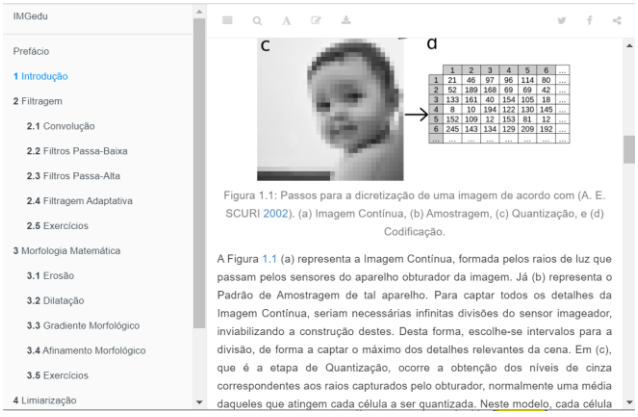


Figura 5. Captura de tela de parte do texto que acompanha o pacote.

Em testes realizados para uma imagem de 1000 x 700 pixels, exceto pelo Filtro Gaussiano Adaptativo, as funções apresentaram tempo de execução menor do que 0,5 segundos, tendo a maioria apresentado tempos consideravelmente menores do que este máximo. Os testes foram realizados no sistema operacional Ubuntu 18.04, em um computador com 8 Gb de memória RAM e processador Intel i5-5200U. A função referente ao Afinamento Morfológico não foi testada quanto ao seu tempo de execução, pois tal algoritmo depende também das feições contidas na imagem, não dependendo apenas das dimensões em pixels da cena analisada.

Um exemplo do texto didático que acompanha o pacote pode ser visto na Figura 5.

4. CONCLUSÕES

Por se tratar de uma implementação que visa a simplicidade da compreensão aos usuários, pode-se concluir que o código escrito ainda é passível de otimização, para que este possa ser executado de forma mais rápida. Tal mudança pode acarretar em um aumento da complexidade do algoritmo, com relação aos conhecimentos necessários em informática para compreendê-los, o que diminuiria sua didaticidade, portanto deve-se realizar tal otimização com cautela.

Atividades acadêmicas de aprendizado tendem a ser facilitadas com o auxílio do texto de apoio e das funções disponibilizadas, dada a natureza do pacote.

Para trabalhos futuros, recomenda-se a avaliação do tempo de execução das funções em comparação com funções de softwares livres e proprietários, para aferir se todo o potencial da linguagem Julia foi alcançado dentro do escopo deste trabalho. Para tanto, devem ser usadas funções equivalentes, onde as imagens de entrada das funções em todas as linguagens comparadas resultem exatamente nas mesmas imagens de saída.

REFERÊNCIAS

[1] Gonzalez, R. C. e Woods, R. E., “Processamento Digital de Imagens”, São Paulo: Pearson Prentice Hall, 3ª edição, 2010, 644 p.

[2] Kon, F., “O software aberto e a questão social”. São Paulo, Maio. 2001.

[3] Bezanson, J., Karpinski, S., Shah, V., Edelman, A., “Julia: A Fast Dynamic Language for Technical Computing”, *Society for Industrial and Applied Mathematics*, Vol. 59, No. 1, 2017, pp. 65–98.

[4] Bezanson, J., Karpinski, S., Shah, V., Edelman, A., “Why We Created Julia”, 14 de Fevereiro de 2012. Disponível em: <https://julialang.org/blog/2012/02/why-we-created-julia>, Acesso em: 26/04/2017.

[5] Pereira, J. M., “Introdução à linguagem de programação Julia: A ambiciosa linguagem de programação que quer substituir Python, R e Matlab.”, *III Escola Regional de Informática do Piauí. Livro Anais - Artigos e Minicursos*, v. 1, n. 1, p. 315-335, jun, 2017. ISBN: 978-85-7669-395-6.