
MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÕES

PÁTRIA AMADA
BRASIL
GOVERNO FEDERAL

# REPRODUTIBILIDADE EM PROCESSAMENTO DE IMAGENS DE SATÉLITES

Murilo Ferreira Alves Batista,

Universidade de São Paulo

Relatório de Iniciação Científica do programa PIBIC, orientado pelo Dr. Thales Sehn Körting e pelo Dr. Laércio Massaru Namikawa.

INPE
São José dos Campos
2022

## RESUMO

Uma das características fundamentais da pesquisa científica atualmente é a capacidade de reproduzir os resultados encontrados utilizando os mesmos métodos. Por meio disso, cientistas do mundo todo podem atestar a veracidade de uma descoberta, bem como entender com maior profundidade os resultados. Entretanto, na área de processamento de imagens, a reprodutibilidade é uma dificuldade encontrada por muitos pesquisadores. Grande parte dos resultados encontrados são arquivos que foram submetidos a vários algoritmos diferentes com parâmetros distintos. Por isso, o presente projeto tem como objetivo permitir a reprodutibilidade do processamento de imagens de satélites na biblioteca TerraLib e no ambiente TerraView por meio da inserção de metadados nos arquivos finais.



Palavras-chave: Sensoriamento Remoto. Documentação. Algoritmos de Processamento de Imagens.

## LISTA DE TABELAS

**Pág**.

**SUMÁRIO**

# 1 INTRODUÇÃO

Um dos preceitos fundamentais da ciência moderna é a reprodutibilidade dos resultados. Uma pesquisa precisa ter a capacidade de ser reproduzida usando-se os mesmos métodos e chegando aos mesmos resultados. Dessa forma, outros cientistas podem atestar a veracidade do estudo realizado e entender melhor o seus resultados. Essa característica também é importante para garantir o que Karl Popper chamava de falseabilidade, ou seja, a possibilidade de tentar provar que o resultado é inverídico. Todas essas características são extremamente importantes na produção moderna de conhecimento.

Entretanto, a reprodutibilidade é uma característica que encontra dificuldades na área de processamento de imagens, sobretudo de satélites. Nesse campo, parte significativa dos resultados encontrados são os arquivos finais gerados pela sequência de algoritmos utilizados. Cada um desses algoritmos tem um conjunto de parâmetros que mudam o resultado final e precisam ser executados na ordem correta. Também, os próprios arquivos utilizados na entrada dos algoritmos possuem uma série de variáveis que são relevantes para o processo.

Por esses fatos, existe uma grande dificuldade em reproduzir um resultado nesse setor. Dessa forma, o presente trabalho tem o objetivo de solucionar esse problema. Para isso, a biblioteca TerraLib, que serve de base para softwares importantes como o TerraView, foi modificada para incluir um conjunto de metadados nos arquivos (rasters) produzidos ao final do processamento.

## 2 LEVANTAMENTO DE DADOS

Antes de iniciar as modificações, foram feitas algumas etapas de preparação. A primeira delas consistia em entender o funcionamento individual de cada algoritmo de processamento de rasters disponível na TerraLib e catalogar os seus parâmetros de entrada e saída. Em seguida, foi realizado o download e compilação do código fonte da TerraView para que fosse criado um ambiente de desenvolvimento.

Tabela 2.1 - Algoritmos de processamento de Raster.

| Algoritmo | Entrada | Parâmetros | Saída |
|---|---|---|---|
| Operação Aritmética | Raster | Equação; Normalizar | Raster |
| Classificador | Raster | Tipo; Bandas; Paleta; Parâmetros específicos do tipo | Raster |
| Recorte | Raster | Tipo; Parâmetros específicos do tipo | Raster |
| Detecção de Nuvens | Raster | Banda; Operador morfológico; Nuvem; Sombra | Vetor |
| Transformação de Cor | Raster | Tipo; RGB; Matiz; Luminância; Saturação; Bandas; Bits por pixel; Decomposição | Raster |
| Compor e Decompor Bandas | Raster | Banda; Camada; Interpolador; Normalizar; Allow no-data pixels | Raster |
| Contraste | Raster | Bandas; Área do histograma; Intervalos do tipo de dados; Tipo | Raster |
| Filtro | Raster | Tipo; Iterações; Definição de máscaras de usuário (Tamanho e Valor padrão); Bandas | Raster |

| | | | |
|---|---|---|---|
| Fusão | Rasters | Tipo; Nome da camada; Bandas; Sensor; Interpolador; Wavelet | Raster |
| Majority Filter | Raster | Tipo; Número de pixels; Número de vizinhos; Limiar de substituição; | Raster |
| Modelo de mistura | Raster | Tipo; Curvas espectrais; Modelo de mistura; Banda; Componentes; Dummy; Normalizar; Imagem de erro; Decompor imagem de saída | Raster |
| Mosaico | Raster | Tipo; Interpolador; Método de mistura; Dummy | Raster |
| PCA | Raster | Tipo de dado de saída; Bandas; Inverter; Matriz PCA | Raster |
| Pós-classificação | Raster | Peso; Limite | Raster |
| Raster edit | Raster | Regra de substituição; Região de interesse; Banda | Raster |
| Fatiamento de Imagem | Raster | Bandas; Valores mínimo e máximo; Passos; Precisão; Equalizar; Usar somente área visível | Raster |
| Rasterização | Geometria | Atributos; Precisão; Barra de cor; Matriz; Legendas | Raster |
| Registro | Raster | Informações; Pontos de controle | Raster |
| Segmentação | Raster | Tipo; Bandas; Peso; Valor nulo; Tamanho | Raster e Vetor |

| | | | |
|---|---|---|---|
| | | mínimo dos segmentos; Limiar de similaridade ; Best fitting | |
| Vetorização | Raster | Bandas; Número de geometrias máximas | Vetor |
| Extração de limites | Raster | Banda; Valor dummy | Vetor |
| Raster statistics | Raster | Bandas; ROI; Target label; Domínio | - |
| Raster transform | Raster | Bandas; Ordem das bandas; Interpolador; Raio; Tipo de dado; No-data; Resample; SRS | Raster |
| Raste restoration | Raster | Sensor; Sampling factor; Raster resolution; Ordem das bandas | Raster |

A partir desse levantamento de algoritmos e parâmetros, foi possível entender o funcionamento individual de cada um e o resultado na saída. Assim, decidiu-se gerar metadados apenas para os algoritmos que produziam rasters. Também, optou-se por deixar de fora, em um primeiro momento, a inserção de metadados em algoritmos que utilizem parâmetros pouco práticos de serem serializados.

Logo, foi necessário realizar a instalação do TerraView por meio da compilação do seu código fonte. Para isso, foi utilizado um ambiente Linux Ubuntu 20.04 e o tutorial disponível na documentação oficial da TerraView. Para realizar a instalação das dependências foi utilizado o gerenciador de pacotes do sistema operacional. Por fim, o desenvolvimento do código foi realizado na IDE Code::Blocks.

# 3 MÉTODO

Com o intuito garantir a reprodutibilidade do processamento de imagens, optou-se por inserir metadados nos rasters criados na saída dos algoritmos. Isso foi feito modificando-se o construtor da matriz de saída, que gera o resultado por meio da biblioteca GDAL.

Para isso, foi criado um método chamado serialize na classe de parâmetros de entrada de cada um dos algoritmos. Quando este método é executado, ele recolhe os metadados de cada um dos rasters de entrada, caso houver, e insere no raster de saída. Também, é responsável por inserir uma nova entrada com o nome do algoritmo utilizado e seus respectivos parâmetros.

Cada metadado é criado utilizando-se a estrutura de containers associativos em C++. Nela, cada atributo recebe uma chave, que no nosso caso é o nome do algoritmo utilizado e a data e hora em que o processamento foi realizado, e um valor, que para esse projeto serão os parâmetros de entrada utilizados.

Dessa forma, criou-se uma estrutura que permite identificar quais processos foram realizados. Graças a presença da hora e data, é possível saber a ordem em que ocorreram. Por fim, temos uma lista completa de todos os parâmetros utilizados. Assim, a reprodutibilidade é garantida.

# 4 RESULTADOS

A partir das mudanças realizadas na TerraLib, pudemos inserir metadados nos rasters gerados e garantir a reprodutibilidade do processamento de imagens.

Para atestar o resultado, utilizou-se o módulo de testes da referida biblioteca. Em seguida, cada um dos arquivos gerados foi analisado para verificar que os metadados foram inseridos corretamente. Abaixo, temos alguns exemplos deles.

Tabela 4.1 - Resultado da execução dos algoritmos.

| Chave | Valor |
|---|---|
| 2022 09 01 17 56 37 Filter | inRaster: cbers_rgb342_crop1.tif; Filter type: Dilation Filter; iterations: 4; Height of the convolution window: 3; Width of the convolution window: 3; Output bands: 0 1 2 |
| 2022 09 01 18 02 21 Contrast | inRaster: cbers2b_rgb342_crop.tif; outRageMin: -1.79769e+308; outRageMax: +1.79769e+308; Sampled: 0; inRasterBands: Band 0 0; Banda 0 1; Band 0 2; ContrastType: DecorrelationEnhancement |

É fácil identificar nesses exemplos quais foram os algoritmos utilizados, o momento em que foram realizados e seus parâmetros. É uma boa demonstração de como a reprodutibilidade pode ser garantida.

Por fim, é bom citar algumas limitações do que foi implementado. A inserção de metadados na Fusão pelo método IHS não foi realizada por dificuldades técnicas. E algoritmos que são utilizados como sub rotina por outros algoritmos não geram metadados para enviar que eles representem algo que não ocorreu.

# 5 CONCLUSÃO

Os primeiros meses foram dedicados a entender o funcionamento da TerraLib e as minúcias dos algoritmos de processamento presentes. Em seguida, realizamos a inserção gradual de códigos na biblioteca. Por fim, o último mês foi importante para realizar testes, encontrar potenciais defeitos e corrigi-los.

Assim, foi possível garantir a reprodutibilidade do processamento de imagens de satélites e ampliar a qualidade das pesquisas realizadas. Entretanto, ainda é importante melhorar alguns aspectos. Por exemplo, incluir os algoritmos que não foram aprimorados pelos motivos citados anteriormente e permitir a exportação dos metadados em algum formato desejado, como JSON.

Por fim, os resultados encontrados foram extremamente satisfatórios e passaram em todos os testes realizados.

## REFERÊNCIAS BIBLIOGRÁFICAS

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS. TerraLib and TerraView Wiki Page. 2021. Disponível em: http://www.dpi.inpe.br/terralib5/wiki/doku.php. Acesso em: 30 set 2022.