MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

DESENVOLVIMENTO DE ROTINAS EM AMBIENTE PYTHON PARA O PROCESSAMENTO DE IMAGENS E DADOS ÓPTICOS COLETADOS EM AMBIENTES AQUÁTICOS PARA MONITORAMENTO DE SISTEMAS AQUÁTICOS CONTINENTAIS POR SENSORIAMENTO REMOTO.

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA
(PIBIC/INPE/CNPq)

Felipe Menino Carlos (Fatec “Jessen Vidal”, Bolsista PIBIC/CNPq)
E-mail: felipe.carlos@fatec.sp.gov.br

Cláudio Clemente Faria Barbosa (INPE, Orientador)
E-mail: claudio.barbosa@inpe.br

Julho de 2018

“Eu posso não ter ido onde gostaria, mas acho que cheguei onde deveria”.

Douglas Adams

RESUMO

A utilização de técnicas de sensoriamento remoto, tem sido amplamente utilizada em diversas áreas, por conta da riqueza de dados que os produtos (Imagens) do uso desta técnica podem trazer. O Laboratório de Instrumentação de Sistemas Aquáticos (LabISA), faz a utilização destes produtos para a realização de estudos de águas interiores. Porém, para que estes produtos sejam utilizados, são necessárias algumas etapas de pré-processamento, para a remoção de ruídos que podem estar presentes nos produtos. Desta forma as rotinas desenvolvidas neste trabalho, em ambiente Python e Matlab, almejam facilitar a aplicação de algumas correções em imagens oriundas de sensoriamento remoto. Ao final do trabalho, todos os sistemas foram validados pelos usuários.

Palavras-chave: Correções. Python. Matlab.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

MATLAB	Matrix LABoratory
GEE	Google Earth Engine
LabISA	Laboratório de Instrumentação de Sistemas Aquáticos
REM	Radiação eletromagnética
OSGEO	Open Source Geospatial Foundation
GDAL	Geospatial Data Abstraction Library
UML	Unified Modeling Language
MSI	Multiespectral Instrument
OLI	Operational Land Imager

SUMÁRIO

Pág.

1 INTRODUÇÃO

A utilização de produtos de sensoriamento remoto é feita pelas mais diversas áreas, para as mais variadas aplicações, podendo ir desde o controle e proteção ambiental (MASCARENHAS et al, 2008), até o estudo e verificação de meios aquáticos (CAIRO, 2015).

Estes produtos, neste caso, imagens, de sensoriamento remoto são gerados através de sensores instalados em aeronaves e satélites (NOVO, 2010), como por exemplo os sensores, OLI e MSI, presentes nas plataformas Landsat-8 e Sentinel-2, respectivamente.

Porém, no momento da aquisição das imagens, diversos ruídos podem ser gerados, como *sunglint* e até mesmo interferências atmosféricas, o que dificulta o uso dos produtos, fazendo com que seja necessário o pré-processamento, para a remoção dos ruídos.

Existem diversos métodos para a remoção e correção destes ruídos nas imagens, mas, nem sempre a aplicação destes acaba sendo uma atividade rápida e fácil, por conta das diversas etapas que alguns métodos apresentam.

O LabISA, que faz a utilização de imagens de sensoriamento remoto e de produtos derivados destas imagens para estudos da caracterização de ambientes de águas interiores, utiliza diversos métodos de correção das imagens, porém, estes, todos aplicados manualmente, etapa a etapa.

Desta forma foi levantado pelos pesquisadores que para o aumento da eficiência da aplicação dos métodos de correção, é necessário o desenvolvimento de ferramentas que auxiliem a execução destas etapas, encapsulando o máximo possível os diversos passos necessários nos métodos de correção.

1.1 Objetivo geral

Desenvolver ferramentas que facilitem a seleção, aquisição e pré-processamento de imagens e dados ópticos envolvidos nas pesquisas desenvolvidas pelos pesquisadores do Laboratório de Instrumentação de Sistemas Aquáticos.

1.2 Objetivos específicos

- Desenvolver rotinas (*softwares*) para automatizar os processos de consulta, seleção e aquisição, de imagens multi-temporais nas bases de dados da NASA/USGS, ESA e Amazon AWS;
- Desenvolver rotinas para otimizar as etapas de pré-processamento das imagens utilizadas no LabISA;
- Desenvolver rotinas para otimizar a análise de dados ópticos do LabISA.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo revisa todos os conceitos que foram utilizados no desenvolvimento das ferramentas de aquisição e pré-processamento de imagens e de dados ópticos.

2.1 Sensoriamento remoto

Segundo Novo (2010), sensoriamento remoto é a junção do trabalho de sensores, equipamentos para processamento e transmissão de dados, que são colocados a bordo de aeronaves, espaçonaves e diversas outras plataformas, com o objetivo de estudar a superfície do planeta Terra.

2.2 Sensores

Sensores, neste caso, são equipamentos capazes de coletar energia refletida de objetos, e convertê-la em sinais que possam ser registrados e apresentados de forma a possibilitar à extração de informações (SPRING, 2006).

2.2.1 MSI

O *Multiespectral Instrument é* um sensor que está a bordo do satélite Sentinel-2, que foi desenvolvido com o objetivo de realizar aquisições, via imagens multiespectrais, de grandes áreas da superfície terrestre (ESA, 2018).

2.2.2 OLI

Operational Land Imager é o sensor presente no satélite Landsat-8, capaz de realizar aquisições de imagens com resoluções que possibilitam a distinção dos mais diversos alvos da superfície terrestre (NASA, 2018).

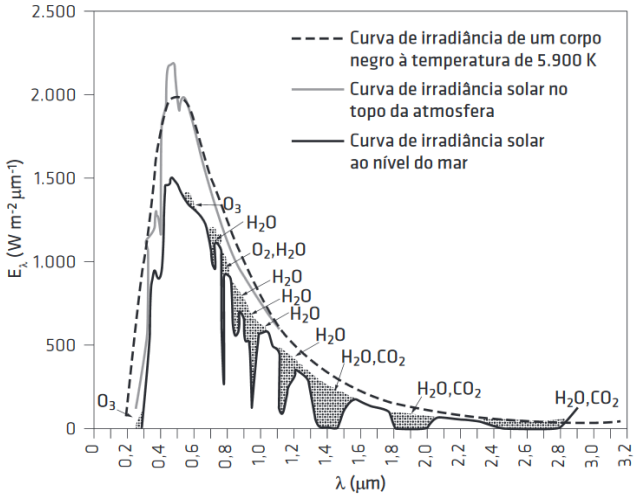
2.3 Correção atmosférica

O sensoriamento remoto, fundamenta-se na utilização da REM registrada por sensores para inferir características de superfícies ou objetos terrestres, podendo variar de acordo com a atuação do sensor.

Entretanto a correlação dos dados sensor-superfície apresenta limitações uma vez que entre eles existe a atmosfera, um meio extremamente dinâmico, que ao interagir com a REM, provoca alterações nos fluxos radiantes proveniente dos alvos (M. LATORRE et al).

Esta interação, como demonstrado na Figura 1, causa uma interferência espectral sobre o fluxo incidente na superfície terrestre (PONZONI et al, 2007).

Figura 1 - Interferência atmosférica



Fonte: Ponzoni et al (2007).

Assim, esta interação da REM com a atmosfera, pode causar incertezas nas análises, uma vez que, os dados presentes na imagem representam a soma dos efeitos atmosféricos e da realidade dos alvos capturados. Com isso torna-se necessário a remoção da interferência atmosférica nos valores presentes na imagem.

Para a realização desta remoção existem diversos métodos. Os subtópicos a seguir apresenta alguns de seus tipos.

2.3.1 Métodos alternativos

Métodos alternativos, são métodos que para a realização da correção utilizam como base somente a imagem, buscando identificar os valores dos pixels escuros, para quantificar a interferência dos efeitos atmosféricos nos demais valores da imagem (SOUZA, 2008).

2.3.2 Métodos físicos

Denomina-se, métodos físicos, os métodos que são fundamentados na teoria da transferência radiativa. Estes levam em consideração as propriedades físicas da atmosfera para a realização da correção (SOUZA, 2008).

2.3.2.1 6SV

O modelo 6SV é um código de transferência radiativa para a correção atmosférica de dados de diferentes satélites para uma grande variedade de condições climatológicas (VERMOTE et al., 1997).

2.4 Sunglint

De acordo com Streher (2013) o *sunglint*, pode ser definido como sendo a luz do sol que é refletida diretamente pela superfície da água, sem interagir com a coluna d'água, e atinge o campo de visada do sensor.

A figura 2 representa a maneira com que o *sunglint* é gerado.

Figura 2 - Geração do sunglint

Fonte: Adaptado de ESA (2012)

2.5 Requisitos

Leite (2001), define requisitos como sendo, objetivos e restrições definidas por clientes e usuários do sistema que modelam as características e propriedades que o software terá.

Veja que, ainda segundo Leite (2001), um conjunto de requisitos pode ser considerado uma capacidade que o software deve possuir para atender as

necessidades do usuário e ainda ajudá-lo a solucionar um determinado problema.

2.5.1 Requisitos funcionais e não funcionais

Sommerville (2011), especifica que os requisitos de um sistema podem ser classificados em duas categorias, os requisitos funcionais e não-funcionais.

Nos requisitos funcionais, tem-se a definição do comportamento do sistema, dos formatos de entrada e saída. De maneira geral é possível entender os requisitos funcionais, como sendo aquilo que o sistema deve fazer.

Já nos requisitos não funcionais, é especificado a maneira com que o sistema irá tratar as entradas e saídas, os limites do sistema, além de diversas outras características, como confiabilidade e desempenho. De maneira direta pode-se entender requisitos não funcionais como a forma com que o sistema realiza suas operações.

2.6 UML

Criada por Grady Booch, Jim Rumbaugh e Ivar Jacobson, a UML, é uma linguagem-padrão para descrever e documentar projetos de *software*, podendo ser utilizada para visualizar, especificar, construir e documentar os artefatos de um sistema de *software* (PRESSMAN, 2009).

Costa (2001), afirma ainda que, a UML define um conjunto de diagramas e notações, que permitem representar as múltiplas perspectivas do sistema, sendo algumas delas; diagramas de classe, caso de uso e sequência.

2.6.1 Diagrama de caso de uso

Casos de uso representam a interação entre os atores, podendo ser estes, usuário e até mesmo outros sistemas, e o sistema que será desenvolvido. Este é um diagrama que traz uma representação de “alto nível”, ou seja, faz a demonstração das interações com o sistema sem a preocupação com a estrutura interna do projeto.

O diagrama de caso de uso ajuda a definir as características do ponto de vista do usuário (PRESSMAN, 2009).

2.7 Framework

Um *framework* pode ser entendido como um conjunto de classes interligadas, que permitem um reuso abrangente de código (ALVIM, 2008).

Este conjunto de classes facilita o processo de desenvolvimento de outros *softwares*, uma vez que, possui funções a facilitar as mais diversas atividades.

2.8 GDAL

GDAL é uma biblioteca para a abstração e tradução de formados de dados geográficos, distribuído pela OSGEO como projeto livre (GDAL, 2018).

Possui uma grande quantidade de funções para tratamento de imagens e uma comunidade de usuários bastante ativa, e por conta disto é aplicado em trabalhos de sensoriamento remoto, como por exemplo em Parente e Ferreira (2018) e Canty (2014).

2.9 Qt

Qt é um *framework* escrito em C++ para o desenvolvimento de interfaces gráficas de forma facilitada, criado pela empresa Trolltech, tem um ecossistema muito bem desenvolvido, com ferramentas como Qt Design que ajudam e aceleram o processo de desenvolvimento de interfaces gráficas.

2.9.1 PyQt

PyQt é um porte do *framework* Qt para a linguagem Python. Isto permite que, códigos Python sejam escritos utilizando as funções presentes no *framework*.

2.10 Flask

Flask é um “micro” framework, para o desenvolvimento de aplicações web, tendo como principal característica, a simplicidade (FLASK, 2018).

Tem a definição de “micro” por conta de sua simplicidade, onde a base do framework vem apenas com o necessário para o funcionamento do software, deixando a cargo do desenvolvedor a escolha por todas as demais características que o framework irá agregar. Isto permite um controle e gerência maior das dependências que estão sendo aplicados no código do software que está sendo desenvolvido.

Toda a forma de agregação de códigos ao Flask é feita através de extensões (FLASK, 2018), estes que são conjuntos de códigos, cada um contendo uma funcionalidade diferente. Com isto é possível manter o software trabalhando somente com o que será utilizado.

2.11 Google Earth Engine

O Google Earth Engine (GEE) é uma plataforma de análise global de grandes conjuntos de dados geoespaciais (Gorelick et al, 2017).

Esta plataforma permite que, análises de grandes conjuntos de dados sejam feitas de forma rápida e facilitada, isso por conta dos diversos algoritmos de processamento de dados geoespaciais já implementados, e vários conjuntos de dados (Landsat-8, Sentinel-2, entre outros) disponíveis dentro da própria plataforma.

O GEE ainda fornece uma interface de comunicação com seus serviços, o que permite a aplicativos externos a plataforma, utilizar o poder computacional e algoritmos disponíveis.

2.12 Web scraping

O *web scraping* é uma técnica destinada a extrair dados de sites da *web*. Para a realização desta atividade, normalmente é aplicado técnicas que simulam a navegação humana em determinado site. Esta é uma técnica amplamente utilizada para a recuperação de dados com o objetivo de centralização e análise (VARGIU. E; URRU. M, 2012).

3 MATERIAIS E MÉTODOS UTILIZADOS

Abaixo é apresentado os materiais e métodos utilizados no desenvolvimento deste trabalho.

3.1 Materiais

Os materiais utilizados no desenvolvimento deste trabalho podem ser divididos em dois grupos: (i) Dados, (ii) Tecnologias

Nos dados tem-se imagens multiespectrais dos sensores OLI/Landsat-8, MSI/Sentinel-2 e imagens hiperespectrais.

Nas tecnologias, a linguagem de programação Python, por ser muito versátil além de possuir diversas bibliotecas que facilitam o desenvolvimento e análise de dados e imagens. A linguagem de programação Matlab, utilizada nas rotinas onde a necessidade não foi suprida com as bibliotecas presentes em Python.

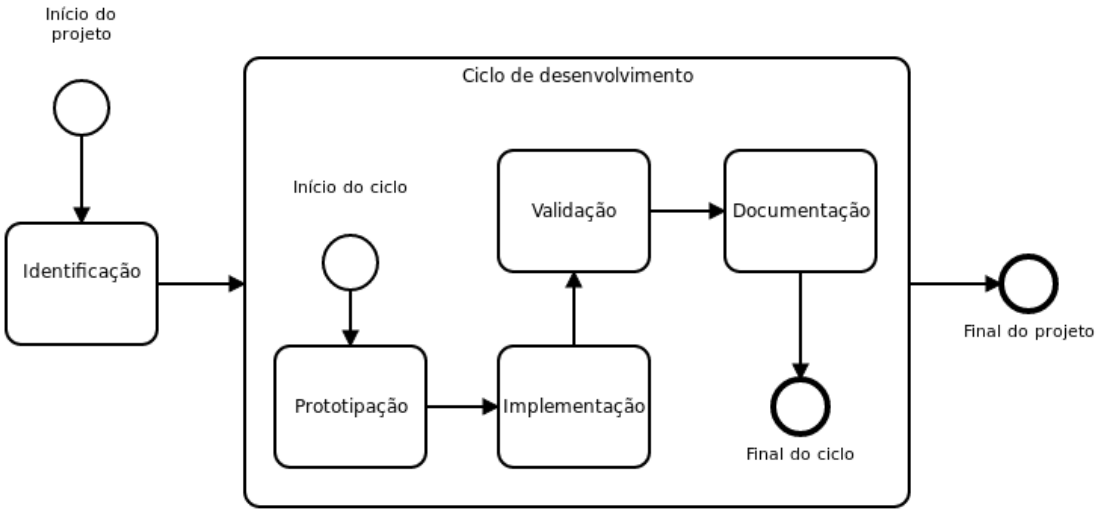
Foi utilizado também os *frameworks* Flask e Qt, por conta da facilidade e velocidade de desenvolvimento que estas trazem, e a plataforma Google Earth Engine, que possibilitou a busca automatizada de parâmetros atmosféricos.

3.2 Métodos

O método utilizado para o desenvolvimento deste trabalho foi dividido nas seguintes etapas: (i) Identificação, (ii) Prototipação, (iii) Implementação, (iv) Validação e (v) Documentação.

A Figura abaixo demonstra a ligação entre cada uma destas etapas, e em seguida a descrição de cada uma delas.

Figura 3 - Metodologia



Fonte: Produção do autor

A etapa de identificação foi realizada junto os pesquisadores e alunos de pós-graduação do LabISA, com o intuito de identificar os principais processos realizados no laboratório para que fosse possível implementar melhorias destes. Neste momento basicamente foi realizado o acompanhamento aos pesquisadores e alunos para entender os principais procedimentos realizados. Esta etapa inicial foi executada uma vez, já que, após o mapeamento dos procedimentos que necessitavam de melhoria e automatização, o ciclo de implementações e verificações, explicados abaixo, começou a ser feito com cada um dos procedimentos identificados.

Com a identificação dos procedimentos a serem melhorados, que neste caso identificou a necessidade de melhoria na busca e aquisição de imagens multiespectrais, bem como os processos de correção atmosférica e remoção de *sunglint* das imagens, um ciclo de prototipação, implementação e documentação (Figura 3) foi iniciado, onde cada uma das etapas citadas foi realizada com todas as rotinas que foram desenvolvidas, veja que, para iniciar o ciclo de desenvolvimento de uma nova rotina, todo o ciclo da rotina anterior teve de ser finalizado.

Abaixo, o descritivo de cada uma das etapas feitas no ciclo é realizado.

Na etapa de prototipação, foi feito junto aos usuários o levantamento de requisitos funcionais, como interfaces de usuário, e principais funcionalidades. Também foi feito a identificação dos casos de uso de cada *software* desenvolvido, além da identificação dos requisitos não funcionais.

Com a prototipação finalizada, as etapas de implementação foram iniciadas, aqui as tecnologias foram escolhidas, variando de acordo com as necessidades dos requisitos e então o desenvolvimento foi feito.

Após a implementação, às validações foram realizadas pelos alunos e pesquisadores do LabISA.

Por fim, a documentação das rotinas foi feita, para que a utilização e possível continuação do desenvolvimento seja feita de forma mais simples.

4 DESENVOLVIMENTO E RESULTADOS

Este tópico irá demonstrar as rotinas desenvolvidas, suas etapas de prototipação, implementação e resultados obtidos.

4.1 Desenvolvimento

Esta seção demonstra as etapas de desenvolvimento de cada uma das rotinas que foram implementadas para a melhoria nos processos do LabISA, os quais foram identificados como passíveis a melhoria no início do desenvolvimento do projeto.

4.1.1 Requisitos funcionais e não funcionais

A primeira etapa realizada no ciclo de desenvolvimento, é a prototipação, que neste caso teve início com o levantamento dos requisitos funcionais e não funcionais.

Para isto foi realizado, para cada rotina desenvolvida, a identificação dos requisitos funcionais e não funcionais.

4.1.1.1 Satellitepy

O primeiro processo identificado com a necessidade de melhorias foi o de busca e aquisição de imagens dos sensores Sentinel-2/MSI e Landsat-8/OLI, para a melhoria deste processo, foi desenvolvido o **Satellitepy**.

A tabela abaixo apresenta os requisitos funcionais e não funcionais levantados para esta rotina.

Tabela 1 – Requisitos funcionais e não funcionais: Satellitepy

REQUISITOS FUNCIONAIS	REQUISITOS NÃO FUNCIONAIS
Fazer a busca por imagens dos sensores Landsat-8/OLI e Sentinel-2/MSI	Realizar desenvolvimento em Python
Realizar a seleção de múltiplas imagens para *download*	Apresentar facilidade no uso
Evitar necessidades de *logins* no sistema	Velocidade nas pesquisas das imagens
Apresentar um *quick-look* das imagens no momento da seleção de imagens	
Permitir a inserção do path/row (orbita/ponto) como forma de busca (No caso do Landsat-8)	
Permitir a inserção do tile id como forma de busca (No caso do Sentinel-2)	
Permitir a busca de path/row; tile id em um mapa	
Permitir a seleção das bandas que se deseja baixar em cada imagem	

Fonte: Produção do autor

4.1.1.2 Atmospy

As correções atmosféricas que fazem a aplicação dos códigos 6SV, apresentam a dificuldade de aquisição dos parâmetros atmosféricos, ao identificar este processo, a criação do **Atmospy** foi proposta para facilitar todo o processo.

A tabela 2 apresenta os requisitos funcionais e não funcionais levantados para esta rotina.

Tabela 2 – Requisitos funcionais e não funcionais: Atmospy

REQUISITOS FUNCIONAIS	REQUISITOS NÃO FUNCIONAIS
Busca automática dos constituintes atmosféricos	Desenvolvimento em Python
Facilidade de utilização	Desempenho nas correções
Permitir que os resultados da simulação 6S sejam salvos	Facilidade de utilização
Informar ao usuário sobre as etapas que estão sendo realizadas	
Permitir inserção manual dos constituintes atmosféricos	

Fonte: Produção do autor

4.1.1.3 rGlint

Os processos de remoção de sunglint em imagens hiperespectrais podem apresentar diversas etapas, desta forma o **rGlint** foi uma rotina desenvolvida para agilizar o processo de remoção deste fenômeno de imagens hiperespectrais com a aplicação de diversos métodos diferentes.

Abaixo é apresentado seus requisitos funcionais e não funcionais.

Tabela 3 – Requisitos funcionais e não funcionais: rGlint

REQUISITOS FUNCIONAIS	REQUISITOS NÃO FUNCIONAIS
Permitir a seleção de bandas que serão trabalhadas	Desempenho nas correções
Visualização da imagem carregada	Facilidade de uso
Realizar correção Kutser (2009)	Condicionar o programa a trabalhar com quantidades variadas de banda
Realizar correção Kutser (2013)	Salvar resultados para serem utilizados no ENVI
Realizar correção Goodman (2008)	
Realizar correção Lyzenga (2006)	
Realizar correção Hedley (2005)	
Plot dos resultados	
Salvar os resultados	

Fonte: Produção do autor

4.1.2 Casos de uso

Os casos de uso, como citado, representam as interações que os atores, neste caso usuários, terão com o sistema.

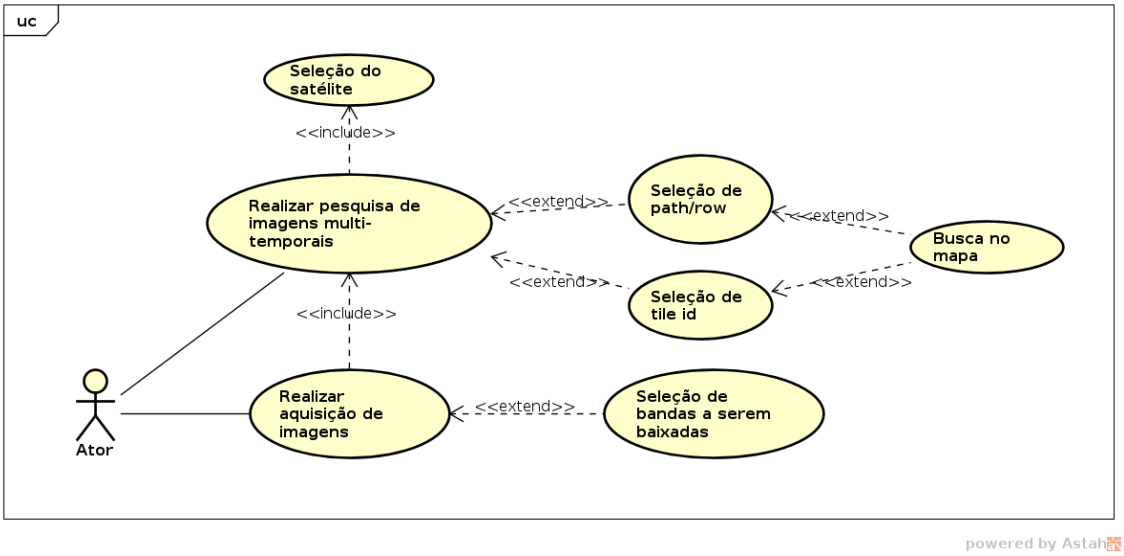
Para o total entendimento, na etapa de prototipação os casos de uso também foram identificados, para cada uma das rotinas desenvolvidas.

4.1.2.1 Satellitepy

Abaixo é apresentado os casos de uso que foram mapeados para o **Satellitepy**.

A Figura 4 apresenta os casos de uso do **Satellitepy**, e em seguida a ela é realizado um detalhamento de cada um dos casos de uso apresentados.

Figura 4 - Casos de uso: Satellitepy



Fonte: Produção do autor

Abaixo são listados os casos de uso de forma detalhada:

Realizar pesquisa de imagens multi-temporais: A partir da janela de pesquisa, o usuário pode inserir um intervalo de datas, as quais o *software* deve buscar.

Seleção do satélite: Na janela de pesquisa, o usuário define o satélite o qual a pesquisa levará em consideração.

Seleção de path/row: Na janela de pesquisa, após a seleção do satélite, a partir do sensor escolhido pelo usuário, ele insere o path/row do local de onde ele está buscando imagens.

Seleção de tile id: Na janela de pesquisa, após a Seleção do satélite, a partir do sensor escolhido pelo usuário, ele insere o tile id do local de onde ele está buscando imagens.

Busca no mapa: O usuário no momento da Seleção do tile id ou Seleção de path/row, pode fazer a seleção do local desejado utilizando uma janela com um mapa.

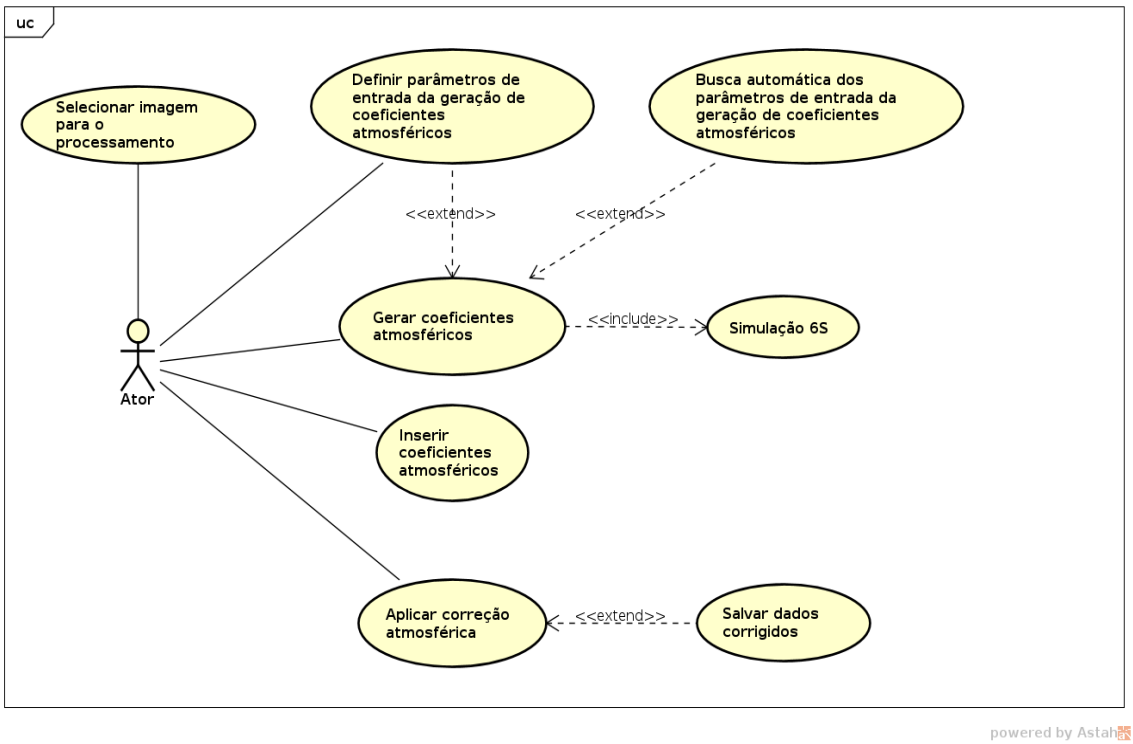
Realizar aquisição de imagens: Na janela de resultados de pesquisa, o usuário pode escolher as imagens que serão baixadas.

Seleção de bandas a serem baixadas: Na janela de *download*, o usuário define a banda que será baixada em cada uma das imagens.

4.1.2.2 Atmospy

A seguir é feito a apresentação dos casos de uso que foram mapeados para o **Atmospy**, e em seguida é realizado uma descrição de cada caso de uso.

Figura 5 - Casos de uso: Atmospy



Fonte: Produção do autor

Abaixo são listados os casos de uso de forma detalhada:

Selecionar imagem para o processamento: A partir da tela inicial o usuário define definir qual imagem será corrigida.

Gerar coeficientes atmosféricos: Na guia de geração de coeficientes atmosféricos, o usuário pode escolher iniciar o processo de simulação atmosférica.

Definir parâmetros de entrada da geração de coeficientes atmosféricos: Na guia de geração de coeficientes atmosféricos, no momento de gerar coeficientes atmosféricos, o usuário define os parâmetros que serão utilizados no momento da simulação atmosférica.

Busca automática dos parâmetros de entrada da geração de coeficientes atmosféricos: Na guia de geração de coeficientes atmosféricos, no momento de gerar coeficientes atmosféricos, o usuário pode não definir os parâmetros que serão utilizados na simulação, e sim definir que o *software* deve realizar a busca por estes.

Simulação 6S: Sempre que a ação gerar coeficientes atmosféricos é iniciada, a simulação 6S é feita, é neste ponto que a simulação da atmosfera é realizada.

Inserir coeficientes atmosféricos: Na guia de correção atmosférica, o usuário escolhe o arquivo de coeficientes atmosféricos que será utilizado na correção.

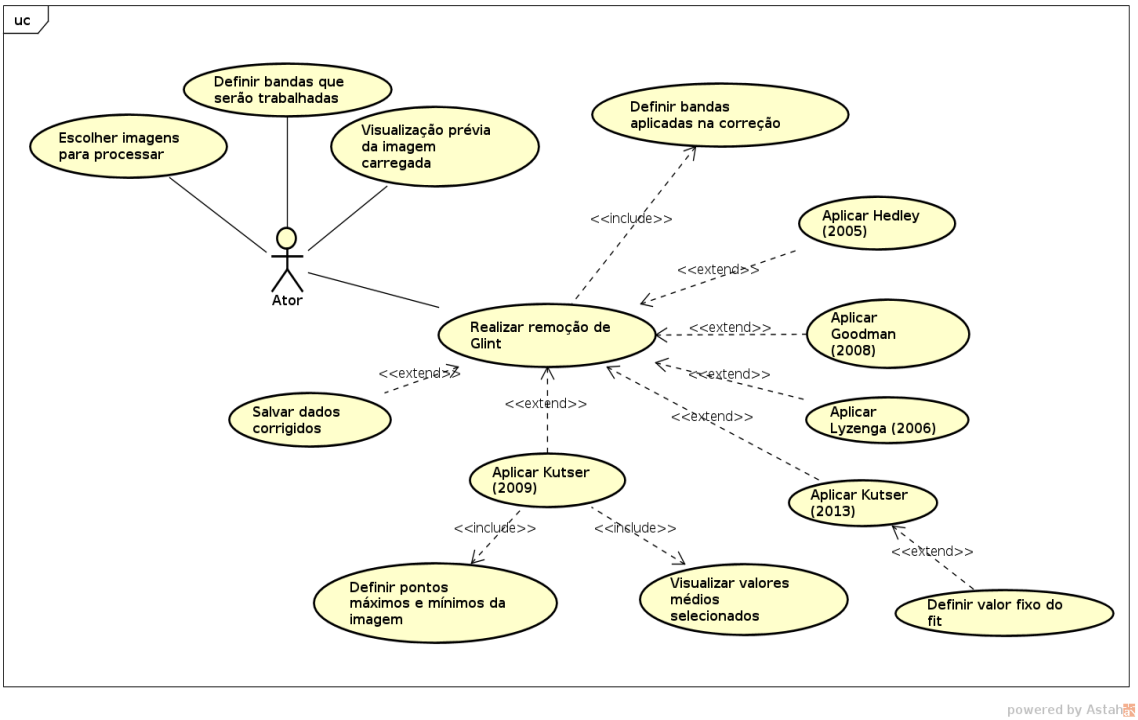
Aplicar correção atmosférica: A partir da guia de correção atmosférica, o usuário pode aplicar a correção na imagem definida.

Salvar dados corrigidos: Na guia de correção atmosférica, o usuário pode definir a intenção de salvar os resultados da correção, para que o *software* após aplicar as correções na imagem, salve os resultados.

4.1.2.3 rGlint

Abaixo a apresentação dos casos de uso mapeados e em seguida o detalhamento de cada um deles é feito.

Figura 6 - Casos de uso: rGlint



Fonte: Produção do autor

Abaixo são listados os casos de uso de forma detalhada:

Escolher imagens para processar: A partir da tela inicial o usuário define a imagem que será corrigida.

Definir bandas que serão trabalhadas: Na tela inicial, após a definição da imagem a ser trabalhada, o usuário escolhe o intervalo de bandas que ele deseja trabalhar ao decorrer das correções na rotina.

Visualização prévia da imagem carregada: Na tela inicial, na guia de visualização, o usuário pode visualizar em cores reais a imagem carregada no programa.

Realizar remoção de Glint: A partir da tela inicial, na guia de correção, o usuário pode fazer a remoção de sunglint na imagem carregada.

Aplicar Hedley (2005): Quando o usuário escolher a remoção de *sunglint*, ele poderá optar por realizar a correção de Hedley (2005).

Aplicar Goodman (2008): Quando o usuário escolher a remoção de *sunglint*, ele poderá optar por realizar a correção de Goodman (2008).

Aplicar Lyzenga (2006): Quando o usuário escolher a remoção de *sunglint*, ele poderá optar por poder realizar a correção de Lyzenga (2006).

Aplicar Kutser (2013): Quando o usuário escolher a remoção de *sunglint*, ele poderá optar por realizar a correção de Kutser (2013).

Definir valor fixo do fit: Na janela de aplicação de Kutser (2013), o usuário pode optar por escolher um valor fixo ao qual será utilizado no *fit*.

Aplicar Kutser (2009): Quando o usuário escolher a remoção de *sunglint*, ele pode optar por realizar a correção de Kutser (2009).

Definir pontos máximos e mínimos da imagem: Na janela de aplicação de Kutser (2009), o usuário define os pontos máximos e mínimos da imagem.

Visualizar valores médios selecionados: O usuário visualiza os valores médios selecionados pelo *software* para a aplicação da correção.

Salvar dados corrigidos: Na tela inicial, antes de inicializar as janelas de remoção de *sunglint*, o usuário pode escolher salvar os resultados que são gerados nas correções.

4.1.3 Prototipação

Após a identificação dos requisitos de desenvolvimento, foi realizado a prototipação das interfaces de usuário, para que assim, no momento do desenvolvimento o processo fosse mais assertivo e direto com o que o usuário espera do sistema.

Os subtópicos a seguir apresentam a prototipação realizada em cada um dos sistemas desenvolvidos, estes aprovados pelos usuários.

4.1.3.1 Satellitepy

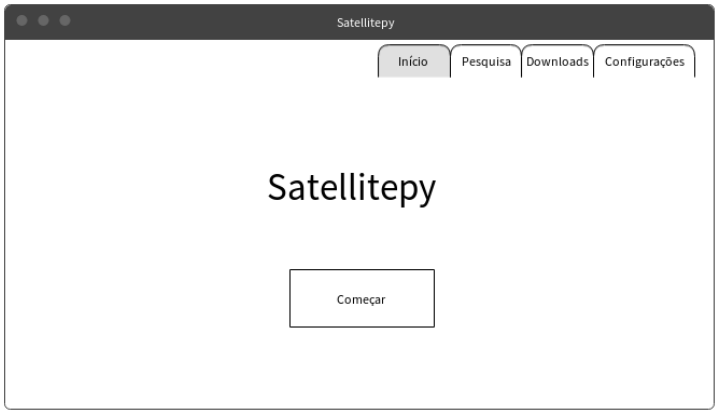
Abaixo a prototipação das janelas do **Satellitepy**.

4.1.3.1.1 Tela inicial

A tela inicial do **Satellitepy** é simples, e fornece acesso a todos os demais recursos presentes no *software*.

A Figura 7 apresenta a prototipação da tela inicial.

Figura 7 - Tela inicial do Satellitepy



Fonte: Produção do autor

4.1.3.1.2 Tela de pesquisa de imagens

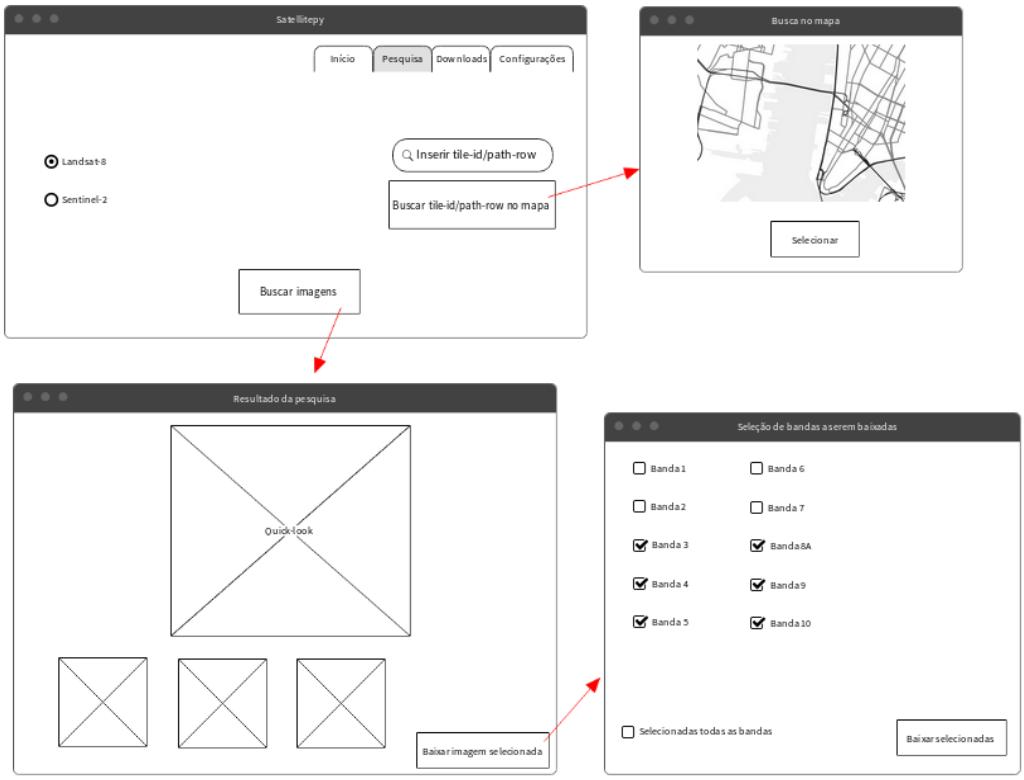
A tela de pesquisa de imagens é dividida em algumas sub telas.

Na tela principal de pesquisa, é possível realizar a definição de qual sensor deve ser pesquisado, bem como um campo para a inserção do identificador da cena que está sendo pesquisada.

Para facilitar o processo de identificação da cena, um botão para acessar um mapa e definir o local pesquisado foi adicionado.

No momento que o usuário realiza a pesquisa uma nova janela é exibida com os resultados, nesta há um campo de *quick-look* onde pode-se ir navegando por cada uma das imagens encontradas. Quando o usuário decide escolher uma imagem, é possível clicar no botão “Baixar imagem selecionada”, ao clicar, uma janela é exibida, nesta pode-se escolher as bandas que deseja fazer a aquisição.

Figura 8 - Tela de pesquisa de imagens



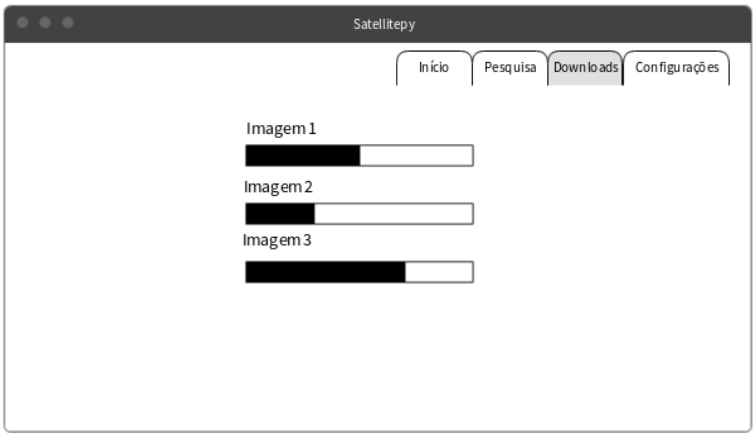
Fonte: Produção do autor

4.1.3.1.3 Tela de download

A tela de *download*, irá apresentar ao usuário o progresso dos *downloads* que estão sendo realizados.

Nesta basicamente há uma barra de progresso para cada conjunto de imagens que está sendo baixado.

Figura 9 - Tela de download



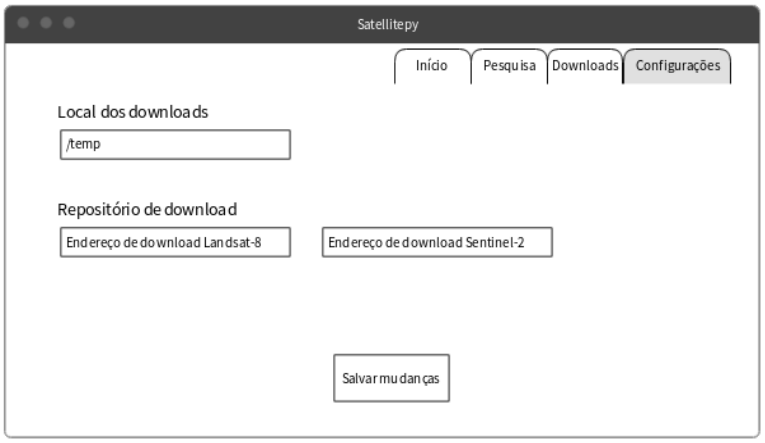
Fonte: Produção do autor

4.1.3.1.4 Tela de configurações

A tela de configurações fornece ao usuário algumas configurações do programa.

Nesta é disponibilizado primeiro um campo para o usuário definir o local onde o *software* deve salvar as imagens que serão baixadas. Há também campos para indicar o servidor de onde as imagens estão sendo retiradas, isto para que caso haja necessidade de mudança em algum serviço, esta seja feita sem grandes problemas.

Figura 10 - Tela de configuração



Fonte: Produção do autor

4.1.3.2 Atmospy

Os subtópicos abaixo apresentam a prototipação do **Atmospy**. Nesta a tela principal é dividida em duas guias, que são apresentadas abaixo.

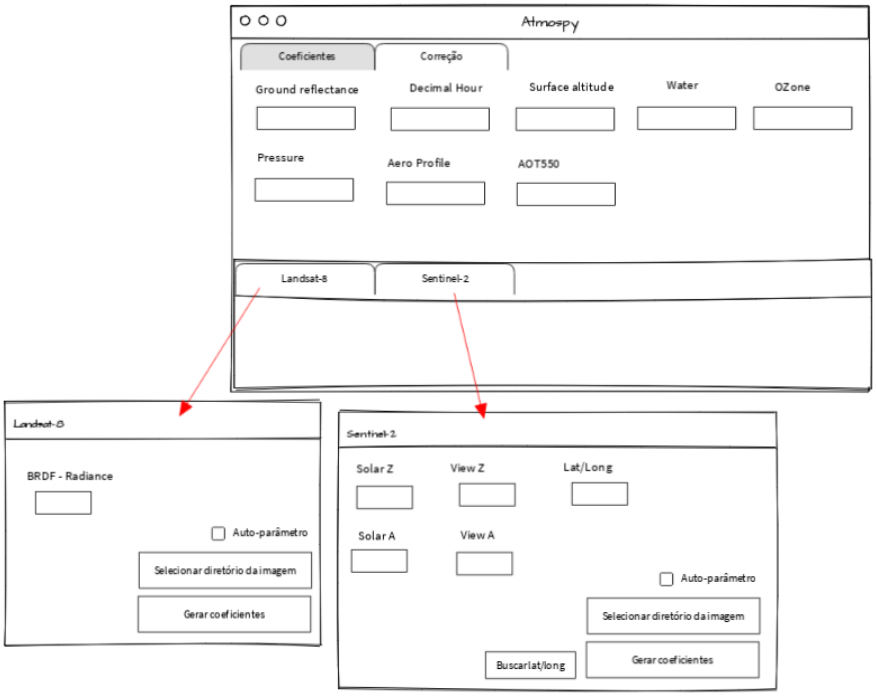
4.1.3.2.1 Guia de geração de coeficientes

A primeira guia da tela principal é a de geração de coeficientes, nesta é disponibilizado os campos para a inserção das características atmosféricas.

Além disto é disponibilizado também dois painéis, este que pode ser alterado de acordo com o sensor a ser utilizado. Cada um dos painéis apresenta campos para a inserção de características atmosféricas utilizadas especificamente em cada sensor.

Há ainda o seletor “Auto-parâmetros”, este utilizado para indicar que o programa deve pesquisar as características atmosféricas utilizadas na geração de coeficientes, evitando assim que o usuário tenha que explicitar cada um deles.

Figura 11 - Tela principal do Atmospy: Guia de geração de coeficientes atmosféricos



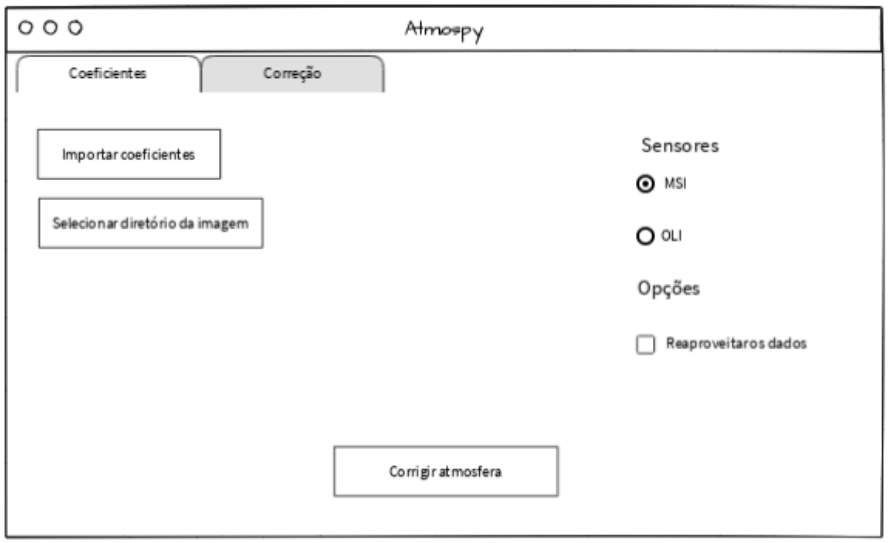
Fonte: Produção do autor

4.1.3.2.2 Guia de aplicação de correção atmosférica

Ainda na janela principal do **Atmospy**, há a guia de correção atmosférica. Nesta guia fica disponível botões para importar coeficientes já gerados pelo programa, assim como botão para selecionar o diretório da imagem a ser corrigida.

Há também alguns seletores para a definição do sensor que será corrigido, além de uma opção extra “Reaproveitar dados”, que indica ao programa que os dados, coeficiente e diretório da imagem, devem ser os mesmos utilizados na última geração de coeficientes bem-sucedida.

Figura 12 - Tela principal do Atmospy: Guia de correção atmosférica



Fonte: Produção do autor

4.1.3.3 rGlint

A seguir tem-se a demonstração dos protótipos das janelas do **rGlint**.

4.1.3.3.1 Tela inicial

A tela inicial do **rGlint** é segmentada em quatro partes, descritas abaixo.

A primeira parte da interface é reservada para carregar a imagem que será trabalhada, além de permitir um filtro das bandas que serão utilizadas por todas as demais etapas da rotina.

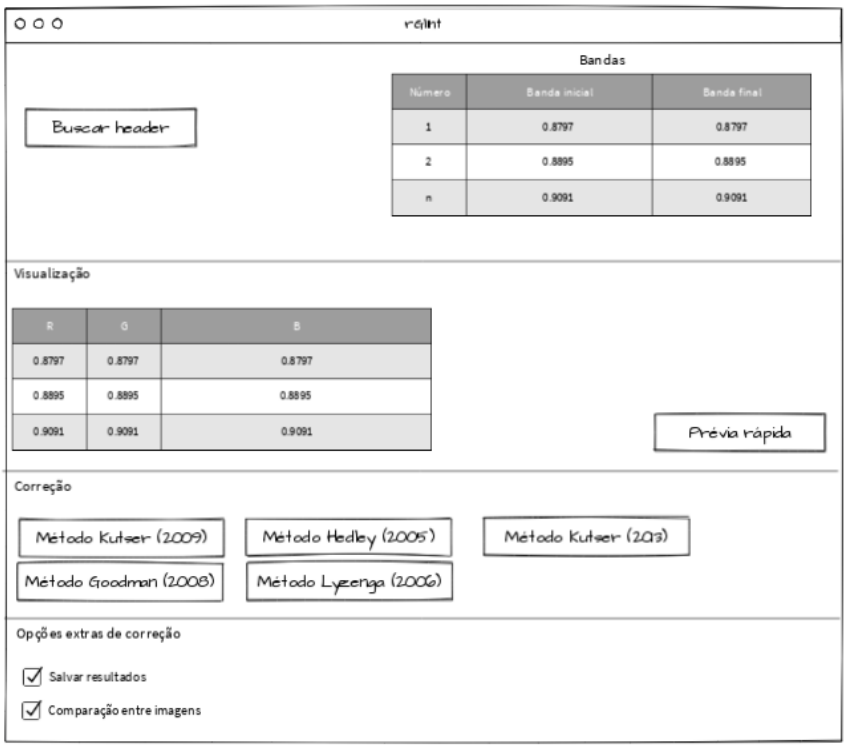
Na segunda parte é utilizada para visualização dos dados, onde o usuário pode definir as bandas que serão empregadas no RGB e então visualizar a imagem que ele carregou no **rGlint**.

A terceira parte é a relacionada as correções disponíveis no sistema, nesta o usuário pode escolher qual correção será aplicada em sua imagem, e a cada

seleção uma janela, com os campos de entradas necessários para o método selecionado, é exibida.

A quarta e última parte da tela inicial, há opções extras para o usuário, nesta pode-se definir se o resultado será ou não salvo, e definir se uma comparação da imagem original e a corrigida deve ser exibida.

Figura 13 - Tela inicial do rGlint



Fonte: Produção do autor

4.1.3.3.2 Telas de correções

O **rGlint** possui cinco telas de correção, uma para cada método de correção.

A primeira tela, é a de correção pelo método Kutser (2009), nesta há campos para definir as bandas que serão utilizadas no método, um campo para inserir o tamanho da janela de correção que será gerada, além de botões para a definição

dos pontos máximos e mínimos da imagem, e também um botão para visualizar os valores médios definidos pela imagem para a realização da correção.

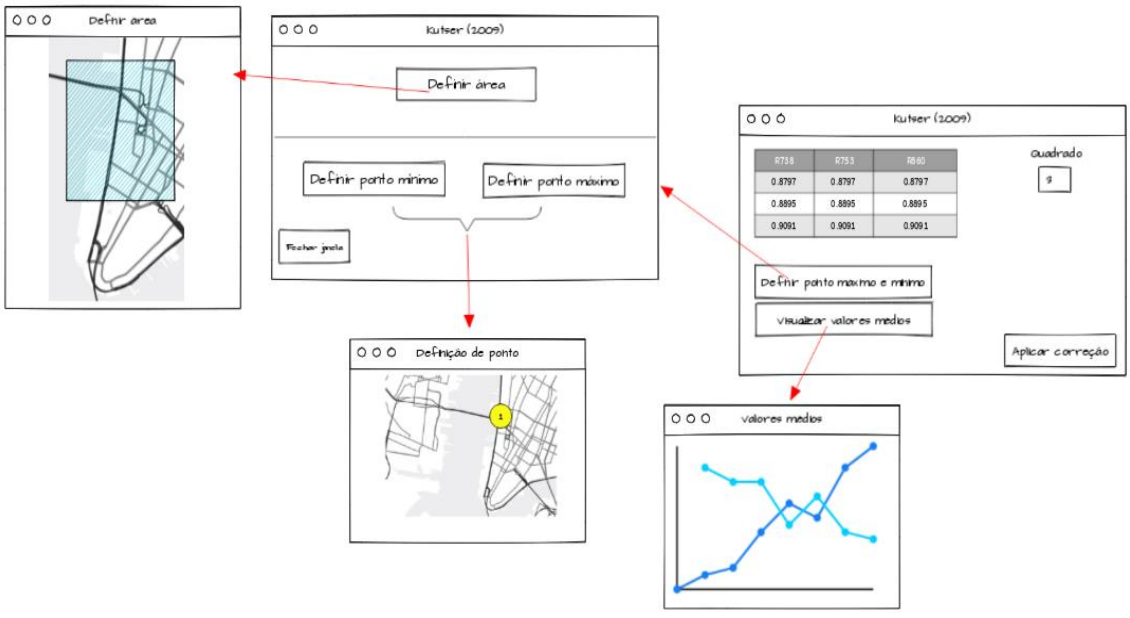
O botão “Definir ponto máximo e mínimo”, exibe uma nova janela, está com as opções para filtrar a área de seleção dos pontos e fazer as seleções de ponto máximo e mínima.

Para definir a área de seleção dos pontos, há o botão “Definir área”, neste a imagem toda é exibida e o usuário pode definir uma área que será utilizada no momento da definição dos pontos.

Há também os botões “Definir ponto mínimo” e “Definir ponto máximo” que exibem a imagem cortada nas dimensões definidas pelo usuário no momento da definição de área, e nesta o usuário pode escolher o ponto que ele considera máximo e mínimo.

Após fazer a definição dos pontos, pode-se verificar os valores médios dos pontos escolhidos, em cada uma das bandas, isto permite ao usuário verificar se os valores escolhidos estão dentro de algum padrão já conhecido.

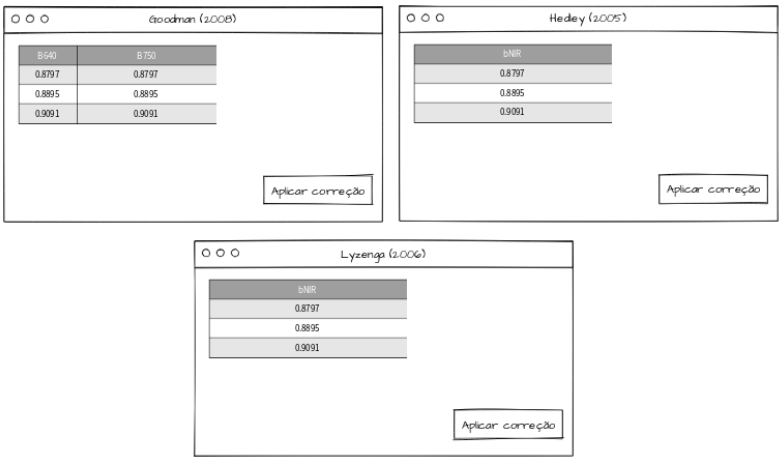
Figura 14 – Janelas da correção pelo método Kutser (2009)



Fonte: Produção do autor

A segunda, terceira e quarta telas representam respectivamente as correções Goodman (2008), Hedley (2005) e Lyzenga (2006), que apresentam as mesmas formas de entrada, nestas sendo necessário definir apenas um comprimento de onda, e então iniciar a correção.

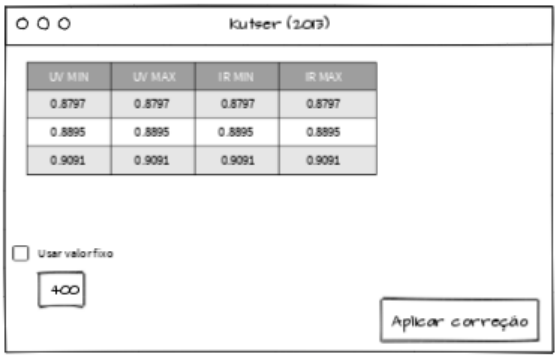
Figura 15 - Telas de correção Goodman (2008), Hedley (2005) e Lyzenga (2006)



Fonte: Produção do autor

A quinta e última tela, que representa a correção pelo método de Kutser (2013), tem uma janela, onde há uma tabela para a definição dos valores que serão utilizados na correção, além disto há também um campo para fixar um valor de *fit* (Ajuste) que será utilizado na correção.

Figura 16 – Tela de correção de Kutser (2013)



Fonte: Produção do autor

4.1.4 Implementação

As principais etapas de desenvolvimento de cada uma das rotinas são apresentadas nos subtópicos abaixo.

4.1.4.1 Satellitepy

Com o objetivo de facilitar a seleção e aquisição de imagens dos sensores Sentinel-2/MSI e Landsat-8/OLI, o **Satellitepy**, assim como citado anteriormente, foi desenvolvido.

Ao analisar os requisitos e interações identificados com os usuários, decidiu-se criar um sistema *web*, para que as formas de interação fossem melhores feitas.

Desta forma o sistema foi criado utilizando a linguagem Python junto ao *framework* Flask.

Para o desenvolvimento da aplicação, primeiro fez-se a criação das funções de busca do sistema. Estas fazem a consultas às bases de dados da Amazon, e da Copernicus Open Access Hub, as quais fornecem imagens de vários sensores dentre eles o Sentinel-2/MSI e Landsat-8/OLI. Em seguida o desenvolvimento das funções de aquisição também foram realizados.

A busca utiliza técnicas de *web scraping* para realizar as consultas aos índices de imagens que estão disponíveis em cada um dos serviços, assim como as formas de aquisição, que também utilizam da técnica para realizar o download da imagem que foi selecionada pelo usuário após a pesquisa.

O emprego de dois serviços distintos se dá pelo fato da necessidade da aquisição de múltiplos formatos da imagem, no caso da Amazon as imagens são fornecidas em arquivos separados, o que permite ao usuário escolher quais dados serão baixados, sendo útil em casos onde bandas específicas são necessárias. Porém os usuários podem escolher também um arquivo compactado, com todas as arquivos e informações da imagem. Com isto

empregado o usuário tem a liberdade de escolher os formatos que melhor cabe a suas necessidades.

Ainda sobre os serviços que são consultados, estes foram escolhidos, para evitar que autenticações por partes do usuário sejam feitas.

Após o desenvolvimento das funções de busca, criou-se a interface de usuário, esta que seguiu os protótipos que foram desenvolvidos junto aos usuários.

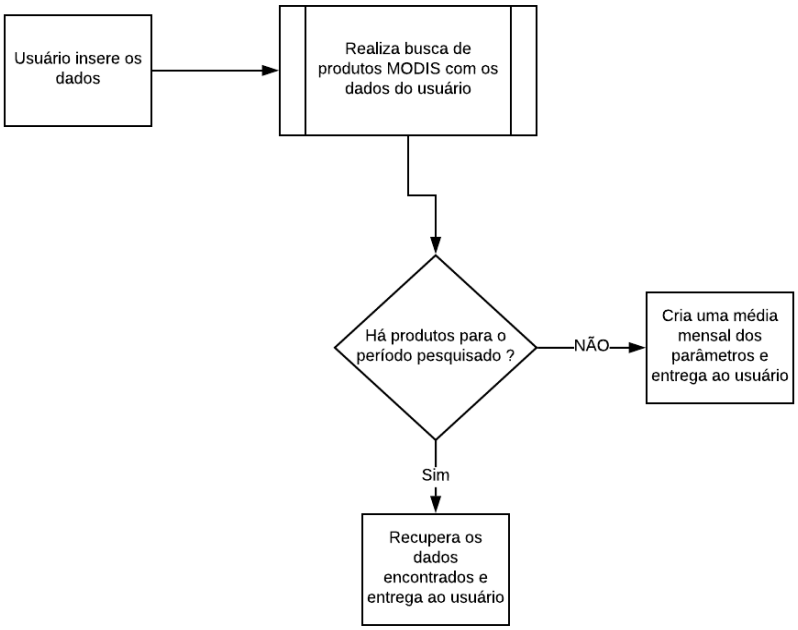
4.1.4.2 Atmospy

Sendo a remoção da interferência atmosférica algo importante para a análise de imagens de sensoriamento remoto, criou-se o **Atmospy**, uma rotina para facilitar todo o processo de remoção da interferência atmosférica.

Para seu desenvolvimento, inicialmente fez-se as funções de simulação atmosférica, as quais dependem de diversos parâmetros. Neste ponto uma função de busca destes parâmetros, adaptada de Murphy (2018), que faz o uso do Google Earth Engine, foi empregada no código do **Atmospy**.

A forma com que a busca é feita no GEE, pode ser vista na Figura a seguir:

Figura 17 - Funcionamento da busca de parâmetros

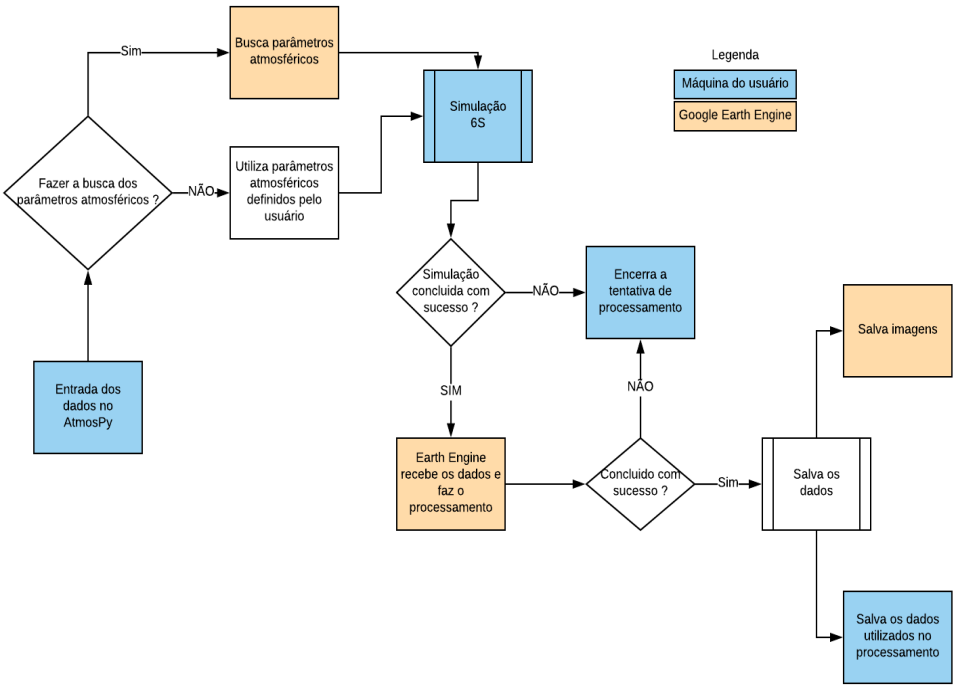


Fonte: Produção do autor

Após a etapa de busca e simulação as funções de remoção atmosférica foram desenvolvidas, nesta o GDAL foi empregado para a leitura e escrita dos arquivos. O fluxo de funcionamento do sistema é demonstrado na Figura 18.

Com as funções principais desenvolvidas, a interface do usuário foi criada utilizando PyQt, o que tornou o desenvolvimento da interface algo simples e rápido.

Figura 18 - Fluxo de funcionamento do Atmospy



Fonte: Produção do autor

4.1.4.3 rGlint

Assim como citado, o **rGlint**, foi desenvolvida para facilitar a aplicação de diferentes formas de correção de *sunglint* em imagens hiperespectrais.

Seu desenvolvimento teve início nas funções de leitura e escrita, criando-se assim formas diferentes de carregar e disponibilizar os dados para o usuário. Após o desenvolvimento das formas de correção, as funções de visualização dos dados foram desenvolvidas.

Com estas funções prontas as etapas de tratamento dos dados começaram a ser criadas, junto às interfaces de usuário, isto porque, cada etapa da correção ficou vinculada com uma determinada interface de usuário.

Por fim as funções de salvar os dados foram adicionadas na interface principal do **rGlint**.

4.2 Resultados

Esta seção apresenta os resultados obtidos a partir do desenvolvimento de cada uma das rotinas.

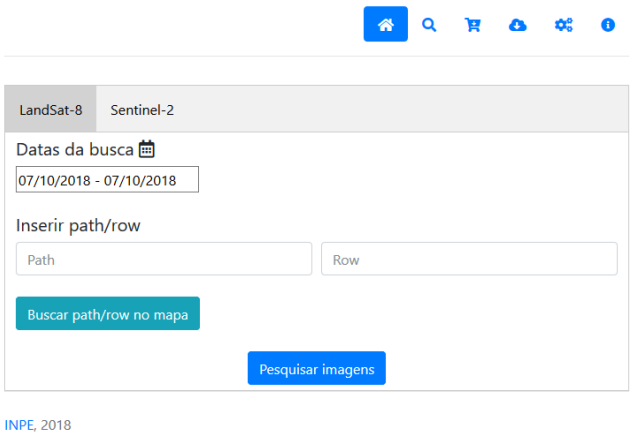
4.2.1 Satellitepy

Os principais resultados obtidos com o desenvolvimento do **Satellitepy** podem ser visualizados nos subtópicos abaixo.

4.2.1.1 Tela de pesquisa

A tela de pesquisa desenvolvida centraliza as opções que o usuário pode utilizar para pesquisar as imagens.

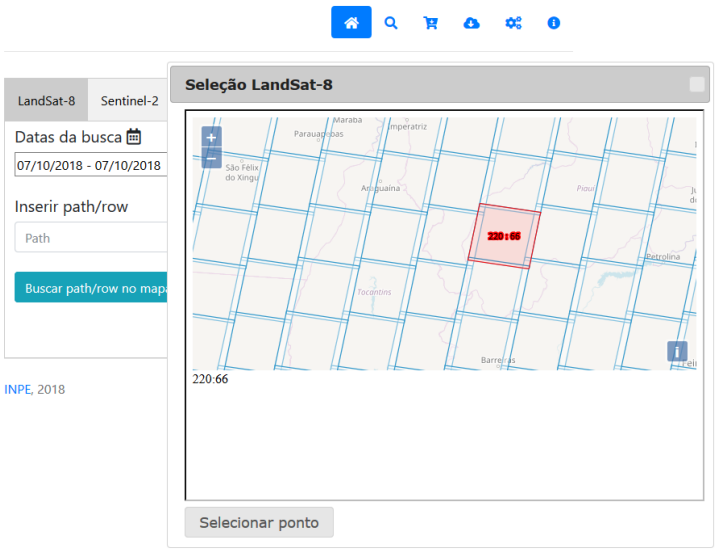
Figura 19 - Tela de pesquisa do Satellitepy



Fonte: Produção do autor

Veja que é possível escolher as datas de pesquisa, o sensor e a localização da cena desejada. Caso o usuário não saiba a numeração da cena pode fazer a busca do mesmo utilizando um mapa (Figura 20).

Figura 20 - Pesquisa de cena

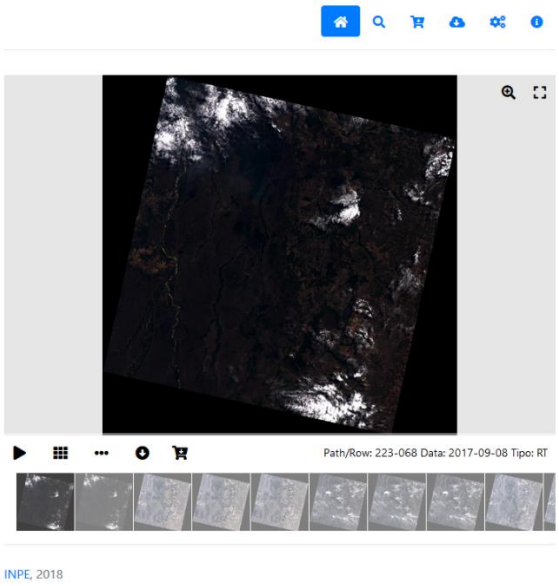


Fonte: Produção do autor

4.2.1.2 Tela de resultados da pesquisa e aquisição de imagens

Após inserir os parâmetros desejados e fazer a pesquisa, os resultados são exibidos em uma interface com o *quick-look* de cada imagem encontrada.

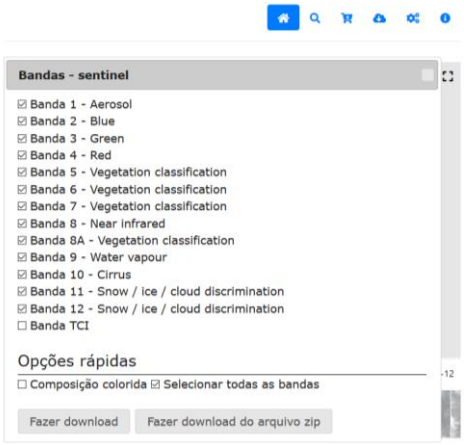
Figura 21 - Tela de resultados de pesquisa



Fonte: Produção do autor

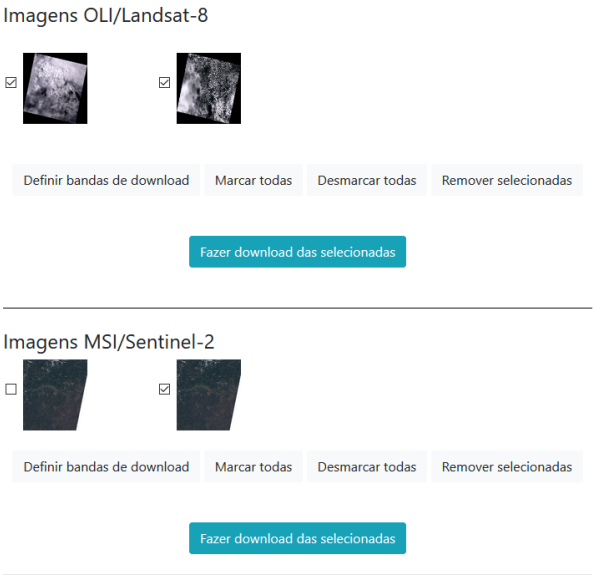
A partir desta interface o usuário pode escolher fazer a aquisição da imagem (Figura 22) ou adicionar o resultado em um carrinho (Figura 23), este onde as imagens podem ser centralizadas para que a aquisição de múltiplas imagens com as mesmas definições (Bandas a serem baixadas) possam ser feitas.

Figura 22 - Janela para aquisição de imagem única



Fonte: Produção do autor

Figura 23 - Carrinho de imagens



Fonte: Produção do autor

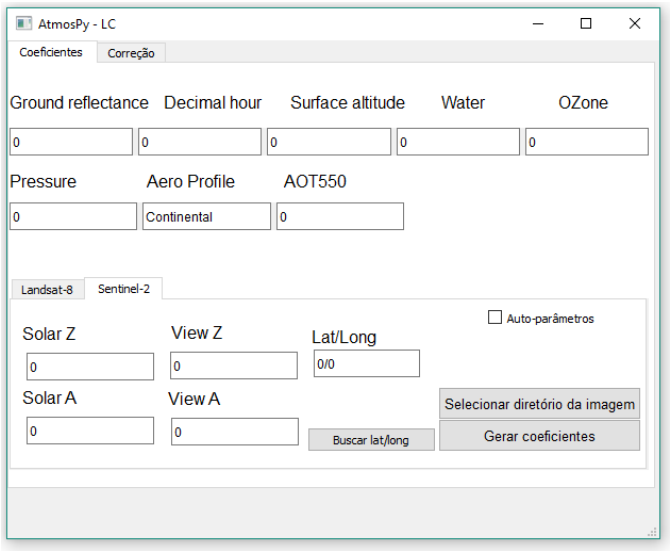
4.2.2 Atmospy

Os subtópicos abaixo apresentam os resultados do desenvolvimento do **Atmospy**.

4.2.2.1 Tela de simulação

Nesta tela, o usuário pode realizar a simulação 6SV, nela há campos para a definição dos parâmetros que deverão ser levados em consideração na simulação.

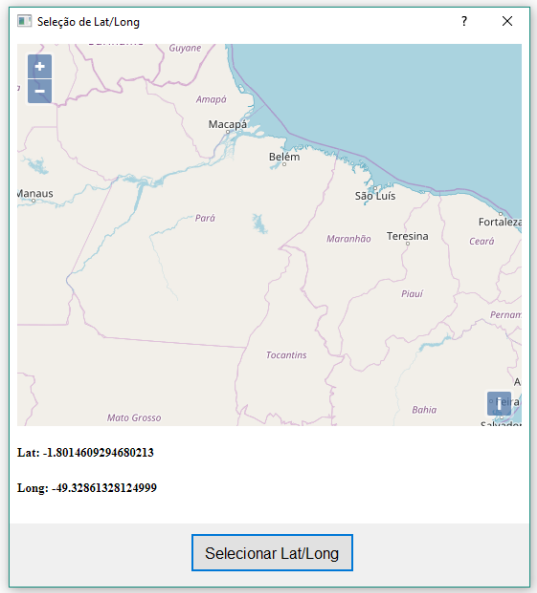
Figura 24 - Tela de simulação do Atmospy



Fonte: Produção do autor

Perceba que, o botão “Buscar lat/long” fornece um mapa, para que a seleção do lat/long seja feita.

Figura 25 - Mapa de busca de lat/long



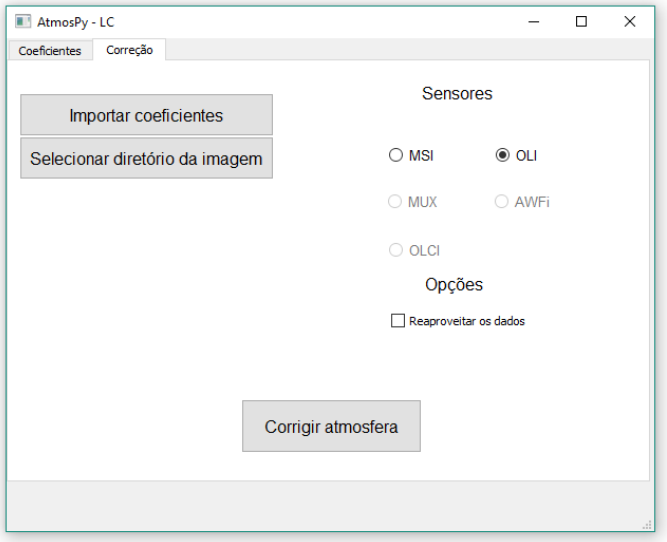
Fonte: Produção do autor

4.2.2.2 Tela de correção

A tela de correção permite basicamente que a correção, utilizando os resultados da simulação, seja realizada.

Nesta é possível importar os resultados da simulação, além da seleção do sensor e imagens que serão trabalhados.

Figura 26 - Tela de correção atmosférica



Fonte: Produção do autor

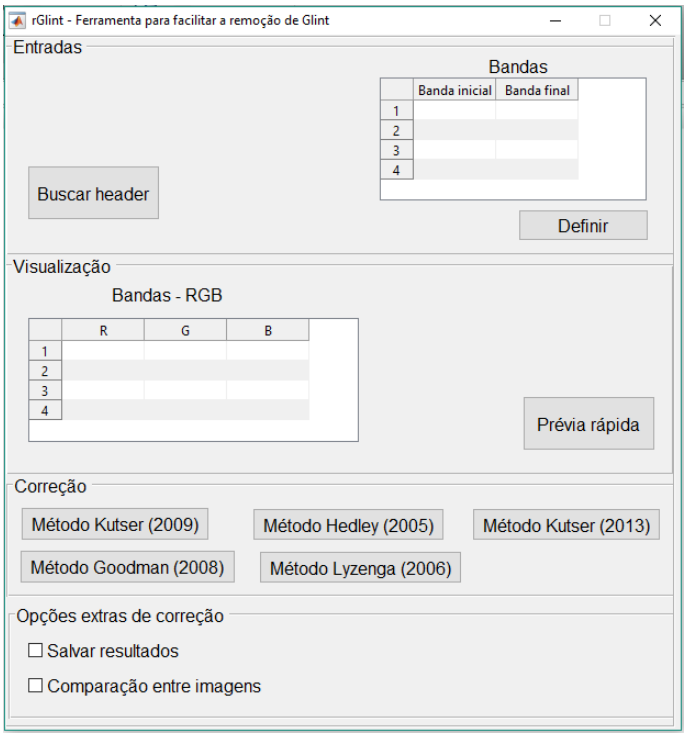
4.2.3 rGlint

Neste sub tópico são apresentados os resultados do desenvolvimento do **rGlint**.

4.2.3.1 Tela principal

A tela principal, como já demonstrado, permite o carregamento, visualização e correção de imagens.

Figura 27 - Tela inicial do rGlint



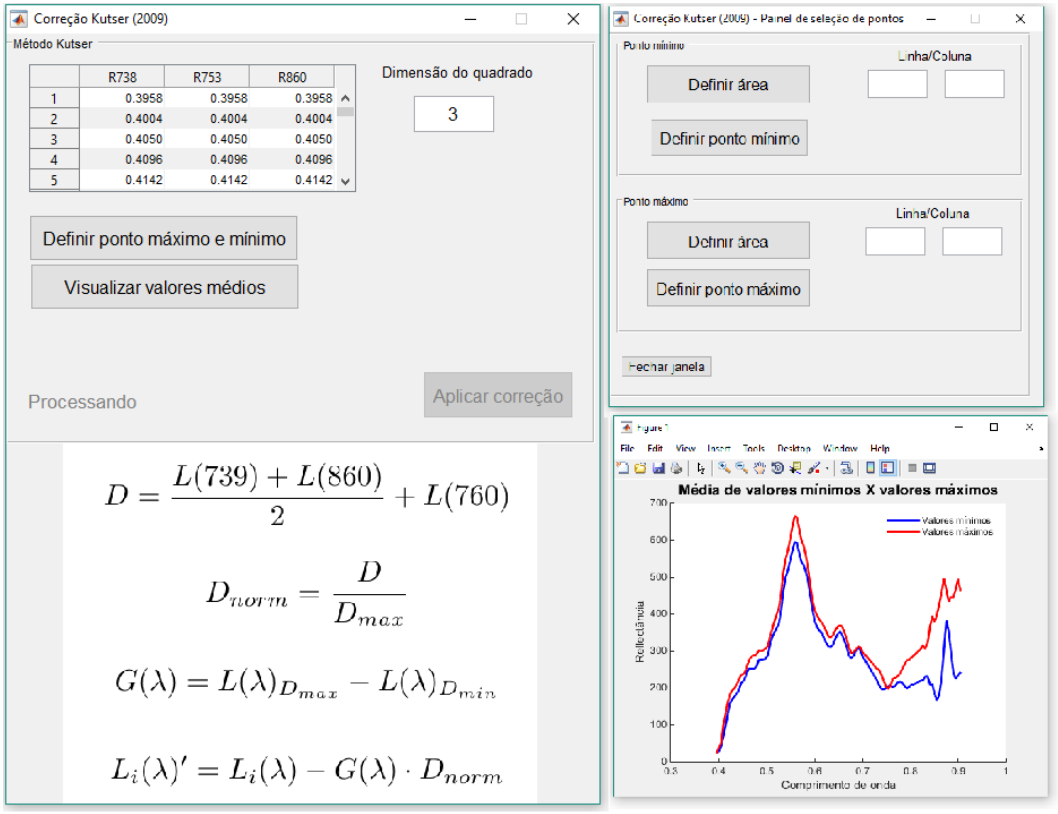
Fonte: Produção do autor

4.2.3.2 Telas de correção

Abaixo algumas telas de correções são apresentadas.

A correção Kutser (2009), por apresentar diversas etapas de entradas, tem no momento de sua correção diversas janelas, a Figura 28, demonstra as principais janelas desenvolvidas para esta correção.

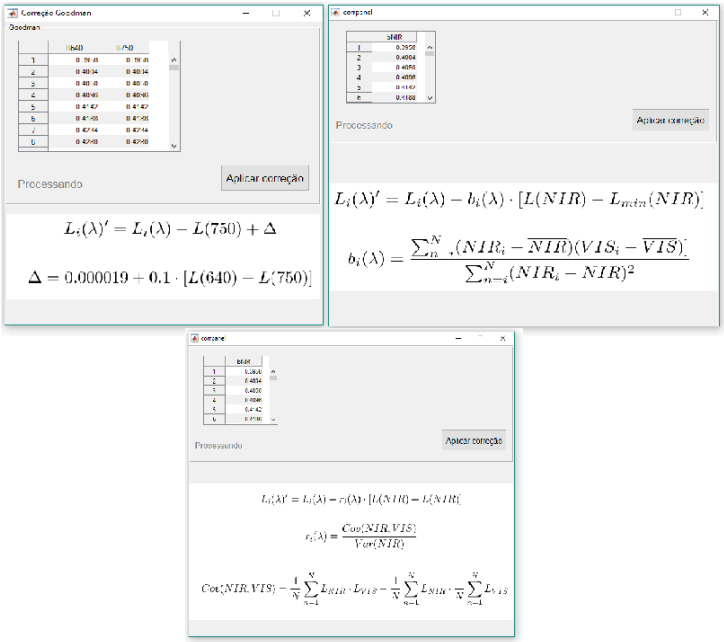
Figura 28 – Janelas de Kutser (2009)



Fonte: Produção do autor

As correções Goodman (2008), Hedley (2005) e Lyzenga (2006) apresentam poucas entradas, o que torna suas janelas de correção iguais, o resultado destas podem ser vistos na Figura a seguir.

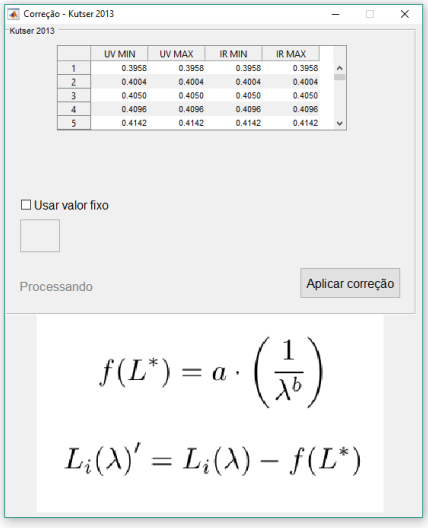
Figura 29 - Telas das correções Goodman, Hedley e Lyzenga



Fonte: Produção do autor

E por há também a tela da correção pelo método de Kutser (2009), que se difere somente nos campos de entrada, que neste caso faz a adição de um campo para a escolha de um valor fixo a ser utilizado no fit (ajuste).

Figura 30 - Tela de correção de Kutser (2013)



Fonte: Produção do autor

Em todas as telas de correção, fez-se a adição das fórmulas que são aplicadas, para que os usuários tenham a certeza das maneiras com que os dados são manipulados.

5 CONCLUSÃO

Este trabalho apresentou as etapas de desenvolvimento dos seguintes sistemas, **Satellitepy**, que foi desenvolvido para a busca e aquisição de imagens dos sensores MSI/Sentinel-2 e OLI/Landsat-8, **Atmospy**, que foi criado para facilitar a aplicação de correções atmosféricas e o **rGlint** que busca facilitar a aplicação de remoção de sunglint em imagens hiperespectrais.

5.1 Validação

Todos os resultados de desenvolvimento foram validados junto aos pesquisadores e alunos de pós-graduação do LabISA, as considerações sobre cada sistema estão descritas nos tópicos abaixo

5.1.1 Satellitepy

Neste sistema, os usuários realizaram a verificação da usabilidade do sistema, ou seja, sua forma de uso, velocidade e disponibilidade, além também da estrutura utilizada pelo software, como a maneira de configurar e salvar as imagens. As observações de cada usuário que testou o sistema, ao longo de seu desenvolvimento, foram sendo implementadas.

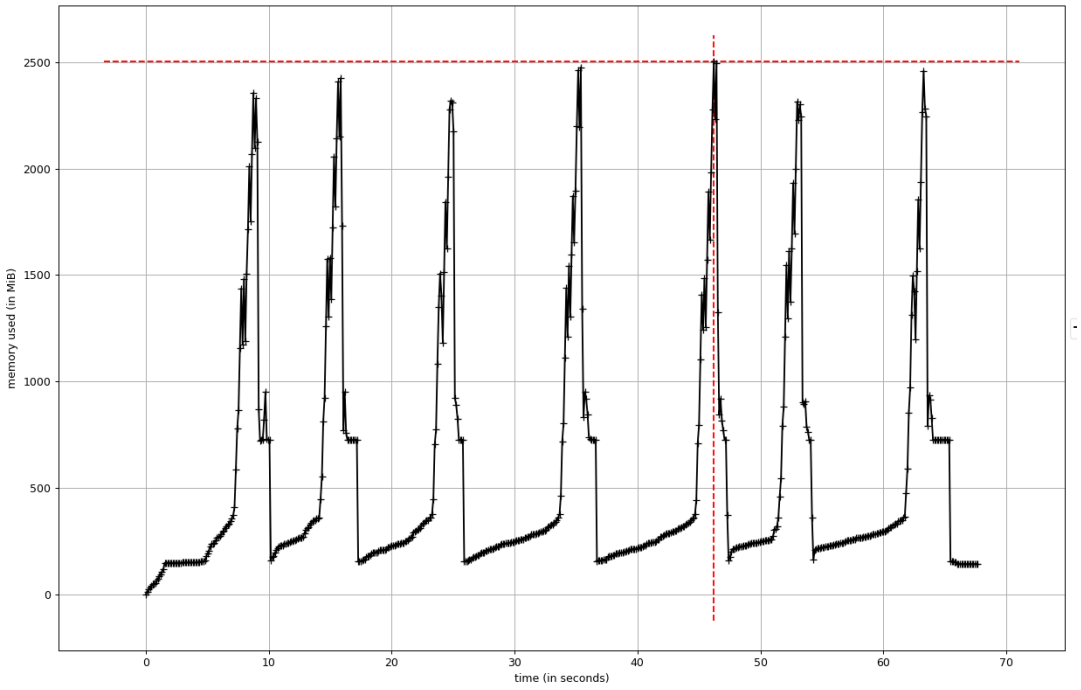
5.1.2 Atmospy

Para a verificação do **Atmospy**, duas etapas foram definidas, a de usabilidade e outra de verificação dos resultados das correções.

Na primeira etapa, a maneira com que a interface foi construída, foi verificada junto aos usuários, quando houve a necessidade de correção, esta foi realizada.

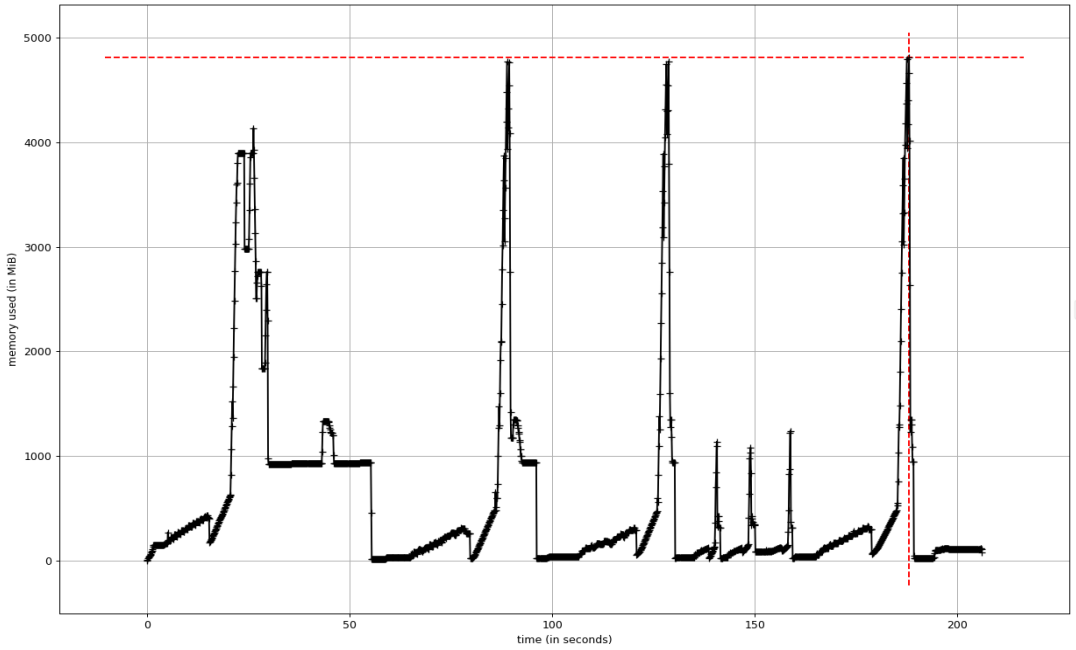
Na segunda etapa, fez-se a verificação dos resultados das correções, neste os erros identificados durante os testes foram corrigidos. Porém durante as diversas correções realizadas, percebeu-se que, a aplicação das correções fizera o uso excessivo de recursos computacionais, as Figuras 31 e 32 demonstram respectivamente, o decorrer do tempo de cada uma correção, em uma imagem do OLI/Landsat (7 bandas utilizadas) e em uma imagem MSI/Sentinel-2 (8 bandas utilizadas) o que torna necessário a verificação das etapas de correção e utilização de outros recursos computacionais, como placas de vídeo e até mesmo o teste de outras bibliotecas para a manipulação dos dados.

Figura 31 – Uso de memória X Tempo de execução (OLI/Landsat-8)



Fonte: Produção do autor

Figura 32 - Uso de memória X Tempo de execução (MSI/Sentinel-2)



Fonte: Produção do autor

5.1.3 rGlint

Da mesma maneira que o **Atmospy**, a verificação do **rGlint**, foi dividido em duas etapas, sendo a primeira para a verificação da usabilidade do sistema, e a segunda para validar as correções feitas.

Na etapa de verificação de usabilidade, diversas melhorias foram apontadas pelos usuários, em especial as correções com o método de Kutser (2009), que apresenta diversas etapas de input antes da correção. Cada melhoria apontada foi implementada no sistema.

Já na etapa de validação cada um dos métodos foi utilizado pelos pesquisadores e validados, em alguns casos, quando encontrou-se problemas nas aplicações de correção, foram feitas mudanças no código para resolver os problemas encontrados pelos usuários.

5.2 Trabalhos futuros

Para dar continuidade ao desenvolvimento das ferramentas apresentadas neste trabalho, os seguintes tópicos devem ser levados em consideração:

- Verificação e melhoria do desempenho das ferramentas de correção atmosférica;
- Ampliar os sensores aos quais as correções atmosféricas são aplicadas;
- Adicionar mais métodos de remoção de *Sunglint.*

REFERÊNCIAS BIBLIOGRÁFICAS

LATORRE, Marcelo et al. CORREÇÃO ATMOSFÉRICA: CONCEITOS E FUNDAMENTOS. **Revista Espaço e Geografia**, São José dos Campos, v. 5, n. 1, p.153-178, 14 mar. 2002. Disponível em: <http://www.lsie.unb.br/espacoegeografia/index.php/espacoegeografia/article/view/20>. Acesso em: 06 jul. 2018.

CAIRO, Carolline Tressmann. **CARACTERIZAÇÃO TEMPORAL DAS PROPRIEDADES BIO-ÓTICAS DO RESERVATÓRIO DE IBITINGA/SP.** 2015. 121 f. Dissertação (Mestrado) - Curso de Sensoriamento Remoto, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2015. Disponível em: <sid.inpe.br/mtc-m21b/2015/03.06.13.23-TDI>. Acesso em: 12 jul. 2018.

PONZONI, Flávio Jorge *et al*. **Calibração Absoluta de Sensores Orbitais:** Conceituação, Principais Procedimentos e Aplicação. São José dos Campos: Parêntese Editora, 2007.

GDAL.ORG. **GDAL - Biblioteca de Abstrações de Dados Geo-Espaciais.** Disponível em: <http://www.gdal.org/index_br.html>. Acesso em: 06 jul. 2018.

LEITE, Jair C. **Análise e Especificação de Requisitos.** Disponível em: <https://www.dimap.ufrn.br/~jair/ES/c4.html>. Acesso em: 07 jul. 2018.

ALVIM, Paulo. **Tirando o Máximo do Java EE 5 Open-source:** com jCompany Developer Suite. Belo Horizonte: Powerlogic Publishing, 2008.

COSTA, Carlos Alberto. A APLICAÇÃO DA LINGUAGEM DE MODELAGEM UNIFICADA (UML) PARA O SUPORTE AO PROJETO DE SISTEMAS COMPUTACIONAIS DENTRO DE UM MODELO DE REFERÊNCIA. **Gestão e Produção**, Caxias do Sul, v. 8, n. 1, p.19-36, abr. 2001. Disponível em: <http://www.scielo.br/pdf/gp/v8n1/v8n1a02>. Acesso em: 07 jul. 2018.

VARGIU, Eloisa; URRU, Mirko. Exploiting web scraping in a collaborative filtering- based approach to web advertising. **Artificial Intelligence Research**, [s.l.], v. 2, n. 1, 20 nov. 2012. Sciedu Press. http://dx.doi.org/10.5430/air.v2n1p44. Disponível em: <http://dx.doi.org/10.5430/air.v2n1p44>. Acesso em: 04 jul. 2018.

CANTY, Morton J.. **Google Play Image Analysis, Classification and Change Detection in Remote Sensing:** With Algorithms for ENVI/IDL and Python, Third Edition. New York: Crc Press, 2014. 576 p.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). **Google Earth Engine**: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment.

SOUZA, Juarez Dantas de. **MODELO FÍSICO-MATEMÁTICO DE CORREÇÃO ATMOSFÉRICA PARA IMAGENS TM - LANDSAT 5 E MODIS TERRA/AQUA.** 2008. 201 f. Tese (Doutorado) - Curso de Meteorologia, Universidade Federal de Campina Grande, Campina Grande, 2008. Disponível em: <http://www.dca.ufcg.edu.br/posgrad_met/teses/JuarezDSouza_2008.pdf>. Acesso em: 07 jul. 2018.

VERMOTE, E.f. et al. Second Simulation of the Satellite Signal in the Solar Spectrum, 6S: an overview. **Ieee Transactions On Geoscience And Remote Sensing**, [s.l.], v. 35, n. 3, p.675-686, maio 1997. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/36.581987.

PSF. **General Python FAQ.** Disponível em: <https://docs.python.org/3/faq/general.html>. Acesso em: 01 jul. 2018.

NASA. **Operational Land Imager (OLI).** Disponível em: <https://landsat.gsfc.nasa.gov/operational-land-imager-oli/>. Acesso em: 09 jul. 2018.

ESA. **MultiSpectral Instrument (MSI) Overview.** Disponível em: <https://earth.esa.int/web/sentinel/technical-guides/sentinel-2-msi/msi-instrument>. Acesso em: 09 jul. 2018.

SPRING. **Sensoriamento Remoto:** O que é sensoriamento remoto ?. Disponível em: <http://www.dpi.inpe.br/spring/portugues/tutorial/introducao_sen.html>. Acesso em: 09 jul. 2018.

STREHER, Annia Susin. **OCORRENCIA E REMOCÃO DOS EFEITOS DE ˜ SUNGLINT EM IMAGENS HIPERESPECTRAIS DE ALTA RESOLU ¸CAO ESPACIAL DO SENSOR SpecTIR.** 2013. 110 f. Dissertação (Mestrado) - Curso de Sensoriamento Remoto, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2013. Disponível em: <http://urlib.net/8JMKD3MGP7W/3DR968E>. Acesso em: 09 jul. 2018.

ESA. **TanSat (Chinese Carbon Dioxide Observation Satellite Mission).** Disponível em: <https://earth.esa.int/web/eoportal/satellite-missions/t/tansat>. Acesso em: 09 jul. 2018.

NOVO, Evlyn M. L. de Moraes. **Sensoriamento Remoto:** Princípios e Aplicações. 4. ed. São José dos Campos: Blucher, 2010. 388 p.

MASCARENHAS, Luciane Martins de Araújo et al. SENSORIAMENTO REMOTO COMO INSTRUMENTO DE CONTROLE E PROTEÇÃO AMBIENTAL: ANÁLISE DA COBERTURA VEGETAL REMANESCENTE NA BACIA DO RIO ARAGUAIA. **Sociedade e Natureza**, Uberlândia, v. 21, n. 1, p.5-18, abr. 2009. Disponível em: <http://www.scielo.br/pdf/sn/v21n1/v21n1a01.pdf>. Acesso em: 16 jul. 2018.

MURPHY, Sam**. Gee-atmcorr-S2**. Disponível em: <https://github.com/samsammurphy/gee-atmcorr-S2>. Acesso em: 21 fev. 2018.