Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Análise de linhas e falhas em plantios de cana

Vinícius Toshiyuki Menezes Sugimoto

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Marcelo Ladeira

Brasília
2021

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Análise de linhas e falhas em plantios de cana

Vinícius Toshiyuki Menezes Sugimoto

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Marcelo Ladeira (Orientador)
CIC/UnB

Dibio Leandro Borges

Flavio Barros Vidal

Prof. Dr. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 3 de novembro de 2021

Dedicatória

Eu dedico este trabalho à minha família, que sempre me apoiou durante minha graduação, e principalmente à minha mãe, Francisca, que por anos se levantou de madrugada para preparar café da manhã para mim antes de eu sair para a universidade.

Também dedico à Bruna Mayra, que me incentivava a fazer minhas tarefas quando eu ficava com preguiça.

Agradecimentos

Agradeço ao Prof. Dr. Luís Paulo Faina Garcia e a Roberson Borges Rodrigues pelo fornecimento dos dados usados neste trabalho e pela disposição em ajudar a sanar qualquer dúvida sobre eles.

Agradeço também ao Prof. Dr. Marcelo Ladeira, meu orientador, por me acompanhar durante os dois anos de desenvolvimento deste projeto.

Sumário

Lista de Figuras

Lista de Abreviaturas e Siglas

API Application Programming Interface.

BIL Binary Interleaved.

BMP Bitmap.

CRISP-DM Cross-Industry Standard Process for Data Mining.

CSV Comma Separeted Values.

ECW Enhanced Compression Wavelet.

GDAL Geospatial Data Abstraction Library.

GeoTIFF Geo Tagged Image File Format.

GIMP GNU Image Manipulation Program.

GSD Ground Sample Distance.

GTK GIMP ToolKit.

HDR High Dynamic Range.

IOU Intersection Over Union.

IV Índice de Vegetação.

KML Keyhole Markup Language.

QGIS Quantum Geographic Information System.

RGB Red-Green-Blue.

SHP ESRI Shapefile.

TIF Tagged Image File.

UTM Universal Transversa de Mercator.

VANT Veículo Aéreo Não Tripulado.

Capítulo 1

Introdução

A agricultura no Brasil é uma das principais bases da economia, servindo não só o Brasil como também exportando para diversos outros países. Para suprir toda a demanda da agricultura brasileira, maquinários e técnicas cada vez mais avançadas e um uso eficiente da terra são necessários. Por exemplo, é importante poder prever o desenvolvimento do plantio, como observar e tratar plantas doentes, que não nasceram ou não estão se desenvolvendo corretamente. Entretanto, com o tamanho enorme de plantações de grande escala, conduzir esse trabalho manualmente é inviável. Com isso, o auxílio de ferramentas modernas é essencial.

O Brasil é um grande produtor mundial de cana, chegando a ser o maior produtor de cana-de-açúcar e o segundo maior de etanol do mundo. Com a produção em larga escala de cana, muitos problemas surgem, como, por exemplo, a identificação de falhas no plantio, seja por falha mecânica do maquinário que semeia os pés de cana ou por pés que acabam morrendo ou não se desenvolvendo corretamente. Essa categoria de problema, dada a escala das plantações, é inviável de ser tratada manualmente. Por isso, soluções mais tecnológicas precisam ser desenvolvidas.

O uso de um Veículo Aéreo Não Tripulado (VANT) para a capta de imagens de plantações e obtenção de informações sobre os objetos e ambientes capturados, conhecido como fotogrametria, é uma técnica que pode ser utilizada para obter os dados necessários para diversas análises que podem ser feitas sobre uma plantação [1]. Uma prática comum do plantio de cana-de-açúcar é o plantio em fileiras paralelas, portanto, o problema de identificação da localização dessas fileiras em imagens capturadas por VANT é a base para diversas análises que podem ser feitas sobre uma plantação. Por exemplo, a identificação do percentual de falhas no plantio nessas fileiras, que pode ajudar a estimar o lucro a ser obtido pela produção e na aplicação adequada de tratamento do solo e das plantas. Este trabalho parte da hipótese de que é possível automatizar o processo de identificação de linhas de plantio com aprendizado de máquina e visão computacional dada uma imagem

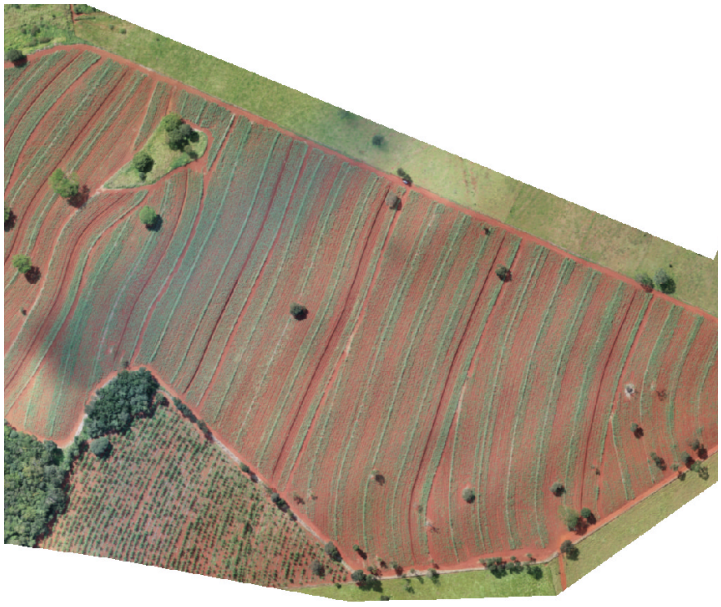

Figura 1.1: Exemplo de fazenda de plantio de cana em linhas

aérea de uma plantação de cana, como na Figura 1.1.

As plantações das quais foram obtidos os dados para este trabalho são de uma única empresa, que já tem um fluxo de trabalho que trata esses dados. Nessa empresa, o problema da identificação de linhas de cana e falhas é tratado utilizando as imagens capturadas pelo VANT e um programa proprietário para o processamento e análise dos dados gerados, chamado *INFOROW*. Entretanto, o INFOROW não provê uma solução totalmente automatizada para o problema de identificação de linhas de cana e falhas, necessitando da presença de um humano auxiliando todo o processamento dos dados. Além de que a solução do INFOROW também comente erros durante a detecção de linhas frequentemente, como os mostrados na Figura 1.2.

Este trabalho propõe uma abordagem de aprendizado de máquina para os problemas de identificação de linhas de cana em plantações. Essa abordagem visa reduzir a presença de um humano durante o processamento e análise de imagens de plantações de cana e marcação de falhas nas linhas de plantio. Adicionalmente, deve reduzir a quantidade de erros de marcação em comparação com o INFOROW, além de gerar resultados compatíveis com os formatos utilizados pelo INFOROW.

Este trabalho segue o padrão de indústria para mineração de dados especificado pela metodologia Cross-Industry Standard Process for Data Mining (CRISP-DM) [2]. A metodologia CRISP-DM representa o ciclo de vida do projeto com seis etapas, que serão exploradas em mais detalhes nos capítulos seguintes. A primeira etapa do CRISP-DM é chamada **entendimento do negócio**, que consiste em entender a fundo o problema, buscando detalhes sobre seu impacto no meio onde ele aparece e quais os objetivos do pro-

Figura 1.2: Exemplos de linhas detectadas com erros.

jeto, a segunda etapa é chamada **entendimento dos dados**, que consiste em organizar todos os dados que se relacionam ao problema, tentando identificar quais são mais importantes para o objetivo do projeto, a terceira etapa é chamada **preparação dos dados**, que se resume em definir quais dados serão utilizados, seus formatos e como se relacionam entre si e com o problema, a quarta etapa é chamada **modelagem**, sendo a aplicação dos conhecimentos adquiridos nas etapas anteriores na construção de um modelo que cumpre o objetivo estabelecido, a quinta etapa é chamada **avaliação**, que avalia o trabalho feito até o momento e determina se as técnicas e o modelo estão sendo adequados e efetivos, se os dados estão preparados para possibilitar e facilitar atingir o objetivo do projeto e se os dados estão bem entendidos, a sexta etapa é chamada **implementação**, sendo a implementação da solução desenvolvido no projeto caso ela seja satisfatória. Essa última não será abordada neste projeto, pois somente será desenvolvido até a etapa anterior, de modelagem.

Nos capítulos seguintes será mostrado como cada uma das etapas do CRISP-DM foi utilizada neste projeto. No Capítulo 2 serão apresentadas as etapas de entendimento do negócio e entendimento dos dados do CRISP-DM seguidas da revisão da literatura, no Capítulo 3 serão apresentadas as etapas de preparação dos dados e de modelagem do CRISP-DM, que apresentam os modelos propostos para a solução do problema e finaliza escolhendo um modelo final para avaliação, no Capítulo 4 será feita uma avaliação do modelo final desenvolvido, comentando seus erros, acertos, pontos que podem ser melhorados e possíveis trabalhos futuros, compreendendo a etapa de avaliação do CRISP-DM.

Capítulo 2

Revisão da literatura

Neste capítulo o problema trabalhado neste projeto será desenvolvido mais a fundo na Seção 2.1, que também definirá o objetivo deste trabalho, depois os dados disponíveis e utilizados serão explorados na Seção 2.2, e por fim trabalhos relacionados e relevantes para este projeto serão apresentados em uma revisão da literatura na Seção 2.3.

2.1 Entendimento do negócio

Nesta seção será apresentada a etapa de entendimento do negócio do CRISP-DM, que consiste em entender os objetivos do projeto e os seus requisitos de uma perspectiva de negócio e então converter esse entendimento em um problema computacional. Este projeto usa dados de uma empresa que trabalha com o plantio de algumas variedades de cana com plantações espalhadas ao longo de diversas fazendas. Na Subseção 2.1.1 será apresentado uma formação de base sobre o modo de operação da empresa antes deste trabalho, na Subseção 2.1.2 serão apresentados os requisitos para a completude deste projeto, suas restrições e suposições efetuadas e na Subseção 2.1.3 serão apresentados os objetivos deste projeto bem como seus critérios de sucesso.

2.1.1 Formação de base

Atualmente, a empresa capta imagens de partes de suas fazendas com um VANT que contém coordenadas geográficas de seus limites, monta imagens completas de cada fazenda a partir das partes e utiliza o *software* proprietário INFOROW para identificar linhas de cana e falhas no plantio e apresentar orientações de replantio. O VANT tem GPS integrado para georreferenciar as imagens que captura com Ground Sample Distance (GSD) mínimo de 4 cm/píxel. Cada fazenda tem um ou mais talhões, a unidade mínima de plantio, sendo

uma área na fazenda que contém alguma plantação, que também tem seus limites definidos com coordenadas geográficas adicionados posteriormente manualmente.

Cada imagem de fazenda montada é processada pelo INFOROW, um *software* pago desenvolvido pela empresa SOMO. O INFOROW consegue identificar linhas de cana, distinguindo entre cana e ervas daninhas, quantificar e identificar falhas no plantio, monitorando a saúde do canavial e conduzindo análise de decisão para renovação da área de plantio, e analisar o espaçamento entre linhas de cana, controlando a organização do canavial, indicando áreas que estão idealmente espaçadas ou não. Os dados passam sequencialmente por cada uma das etapas supracitadas na mesma ordem listada. A Figura 2.1 mostra resumidamente as etapas principais de processamento do INFOROW. Apesar do INFOROW apresentar soluções para os problemas apresentados neste trabalho, ele não apresenta soluções totalmente automatizadas. O INFOROW é composto de cinco sub-*softwares* que têm seu funcionamento apresentado no Anexo I.

2.1.2 Requisitos, suposições e restrições

Nesta seção serão apresentados requisitos, suposições e restrições que impactam decisões sobre este projeto, tal como a decisão de seu objetivo e a viabilidade de implementação.

- Este projeto deve apresentar uma solução para o problema de identificação de linhas de cana e falhas proposto bem como implementar uma aplicação que utilize essa solução de maneira prática.
- Este projeto utiliza dados de uma única fonte, uma empresa brasileira que trabalha com o plantio de cana.
- Não foi possível acesso direto ao programa INFOROW durante o desenvolvimento deste projeto.
- Este projeto não tem vínculo algum com a SOMO, empresa responsável pelo desenvolvimento e distribuição do INFOROW.

  O uso de dados provenientes do INFOROW se deve à facilidade de sua obtenção. Apesar de existirem conjuntos de dados com imagens aéreas de diversas culturas de plantio, os dados de marcação de linhas e falhas não são tão facilmente encontrados.

2.1.3 Objetivos

Esta seção visa clarificar quais são os objetivos deste projeto sob uma perspectiva de negócio e os objetivos sob uma perspectiva de um problema computacional. Assim, um objetivo de negócio pode ser algo como “diminuir o prejuízo causado pela má identificação

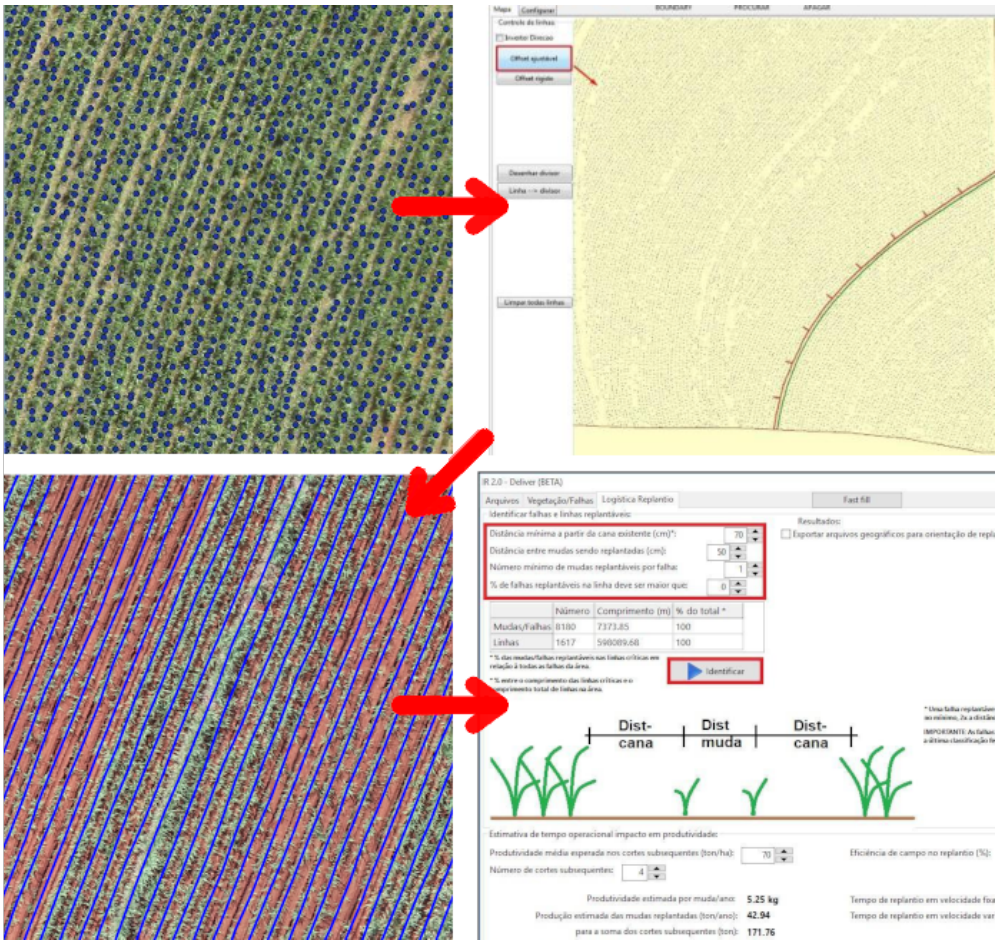

Figura 2.1: Etapas de processamento do INFOROW. Geração de nuvem de pontos sobre cada planta de cana (superior esquerdo), conexão de pontos para marcação de linhas (superior direito), linhas marcadas (inferior esquerdo) e análise das linhas e suas falhas encontradas (inferior direito).

de linhas de cana" e um objetivo de mineração de dados pode ser algo como "apresentar uma abordagem de aprendizado de máquina para identificar linhas de cana em imagens digitais". Com esses objetivos determinados, também devem ser determinados quais os critérios que devem ser cumpridos para que esses objetivos possam ser considerados cumpridos.

Com uma ideia da disposição dos dados, sua disponibilidade e do funcionamento do *software* INFOROW, é necessário estabelecer objetivos específicos para ser possível a conclusão deste projeto. Desta forma, deve-se analisar o que a empresa precisa de melhoras em relação ao plantio e à administração das plantações de cana, estabelecer objetivos a serem cumpridos para resolver algum dos problemas encontrados pela análise das necessidades da empresa e propôr possíveis abordagens de soluções para atingir estes objetivos.

Os dados gerados pela empresa com o uso do INFOROW, marcações de linhas de cana, marcações de falhas e logísticas de replantio, apesar de valiosos, contêm erros frequentes. Mais especificamente, a identificação errônea de linhas de cana nas plantações pode ser notada facilmente e é um erro que pode ser reduzido neste projeto. Esses erros são mostrados na Figura 2.2, em que marcações de linha de cana se estendem além do fim das linhas de cana, mesmo sobre áreas de coloração marrom, e na Figura 2.3, em que as marcações de linha de cana passam sobre árvores, de coloração verde similar a das linhas de cana, e não passam pelas linhas de cana sob a sombra da árvore, que estão em um tom mais escuro do que regiões ao seu redor.

Figura 2.2: Erro de marcação de linhas de cana sobre estrada de chão.

Os dados gerados pelo INFOROW contêm mais pontos que podem ser melhorados, além desses citados, de forma que um projeto que provesse uma solução completa para todas as frentes do INFOROW teria valor imenso, mas o tamanho e variedade de um

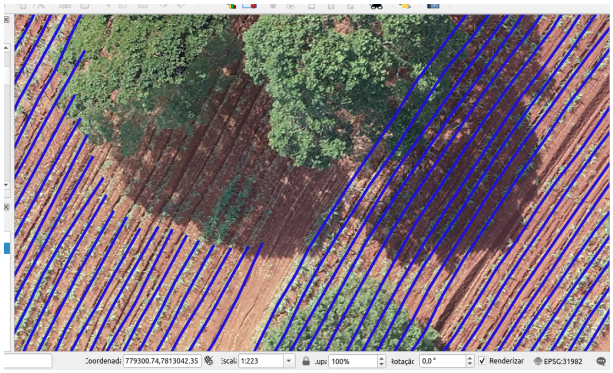

Figura 2.3: Erro de marcação de linhas de cana sobre árvore e falha de marcação em linha de cana sob a sombra da árvore.

projeto como esse foge do escopo desse trabalho. Assim, o problema nas marcações de linhas e falhas isoladamente é um desafio mais adequado para este projeto.

A empresa precisa decidir mais objetivamente como lidar com as falhas nas plantações, de modo a aumentar a produtividade. Para isso, ela usa as informações sobre as falhas nas plantações e as logísticas de replantio geradas pelo INFOROW como guia. Entretanto, o INFOROW em diversas partes necessita de ter seus parâmetros ajustados e precisa de auxílio humano para realizar a identificação das linhas de cana dentre outras tarefas, somado às falhas nos dados gerados, é definido o seguinte objeto para este projeto: trabalhar sobre os dados do INFOROW, substituindo seu módulo de detecção de linhas por uma solução guiada por aprendizado de máquina e visão computacional. Dado que este projeto interagirá com dados gerados pelo INFOROW e dados próprios, os critérios de sucesso poderiam ser discutidos com a própria empresa ou especialistas que possam julgar a utilidade dos resultados gerados.

2.2 Entendimento dos dados

Nesta seção será apresentada a etapa de entendimento dos dados, que consiste em fazer uma exploração dos dados adquiridos e descrevê-los detalhadamente, apontando o papel de cada tipo de dado no desenvolvimento do projeto e o relacionamento entre os diferentes tipos de dados, além de descrever a qualidade dos dados disponíveis.

Para que seja possível que o problema em questão seja inteiramente compreendido e para que se possa visualizar e apresentar possíveis abordagens de solução ao problema é necessário um entendimento aprofundado nos dados que serão utilizados ao longo do desenvolvimento deste projeto. Para isso, de uma coleta inicial do dados, obtida a partir da empresa que fornece os dados para este projeto, pode-se ter uma ideia da quantidade

de dados, sua disposição e seu papel no processo de utilização do INFOROW, isto é, na identificação de linhas de cana e identificação de falhas no plantio.

Os dados disponíveis para este trabalho são as imagens obtidas a partir da captura do VANT e as marcações dos limites das plantações para cada fazenda fotografada, que compõem os dados anteriores ao processamento, os arquivos intermediários gerados durante o processamento do INFOROW, que compõem os dados do processamento, e as marcações de linhas de cana, marcações de falhas no plantio e marcações de entre-linhas, que compõem os dados resultantes do processamento. Os dados disponíveis para este projeto estão listados a seguir.

- Imagens das plantações;

  As imagens das plantações são imagens aéreas de alta resolução de fazendas de cana compostas por um ou mais taludes, com linhas de cana se estendendo por toda sua extensão em várias direções, vegetação nativa ao local, estradas que cortam as plantações, construções, maquinário utilizado no plantio da cana e relevos no terreno. As imagens obtidas inicialmente têm um Ground Sample Distance (GSD) de 10cm/píxel, entretanto podem haver imagens com GSD de até 4cm/píxel. As imagens estão armazenadas no formato Geo Tagged Image File Format (GeoTIFF) com marcações geográficas de seus limites embutidas e podem ser visualizadas utilizando o QGIS. As imagens não tem um tamanho definido, mas do conjunto de dados inicial obtido, a menor imagem ocupa um espaço de 746,8MB e a maior imagem ocupa um espaço de mais de 2,6GB. As plantações seguem um padrão de plantio comum na agricultura de larga escala, em que as plantas são dispostas em "linhas paralelas", o que facilita o uso de maquinário como tratores e colheitadeiras, além de ser útil para a agricultura de precisão com apoio computacional, pois a padronização do plantio ajuda a facilitar a aplicar técnicas de visão computacional sobre as imagens das plantações.

  Os taludes capturados nas imagens podem conter árvores e outras plantas que não pertence à cultura de cana. Considerando que não é possível saber a partir dos dados disponíveis a altura dessas árvores e a disposição do terreno sob elas, não é possível saber também se sob elas o terreno é utilizado para plantio e se maquinário pesado consegue passar por baixo de seus galhos. Assim, toda área que não contém visivelmente plantas de cana não é considerada como parte do plantio.

- Marcações de limites;

  Cada imagem tem um ou mais arquivos de marcação dos limites de cada talude no formato KML, um formato desenvolvido e mantido pelo Google para marcações geográficas. Essas marcações são feitas manualmente com o auxílio de algum programa

Figura 2.4: Recorte de uma imagem aérea de alta resolução (esquerda) e zoom em uma região (direita)

como o QGIS. Essas marcações podem ser usadas para recortar as imagens das plantações, removendo a necessidade de processar qualquer outra parte da imagem original.

- Arquivos intermediários do INFOROW;

  Durante o processamento pelo INFOROW, diversos arquivos são gerados a cada etapa. Eles são:

  - Imagens de cada talude separado de cada fazenda no formato BIL;
  - Arquivo HDR acompanhando cada imagem BIL descrevendo seu leiaute e a formatação da imagem;
  - Para cada imagem BIL e para cada IV selecionado, uma nuvem de pontos no formato CSV e no formato EPC. Apenas a nuvem de pontos no formato CSV pode ser visualizada no QGIS. O formato EPC é um formato de arquivo binário sem cabeçalho e sem informações sobre seu conteúdo. Arquivos no formato EPC possivelmente são de uso exclusivo do INFOROW;
  - Para cada nuvem de pontos em formato EPC, um arquivo de nuvem de pontos NPC. O formato NPC é um formato de arquivo binário sem cabeçalho e sem informações sobre seu conteúdo. Arquivos no formato NPC possivelmente são de uso exclusivo do INFOROW.

- Marcações de linha de cana, de falhas no plantio e de entre-linhas.

  Ao final do processamento do INFOROW, além das logísticas de replantio apresentadas por ele, são geradas marcações de linhas de cana para cada fazenda processada,

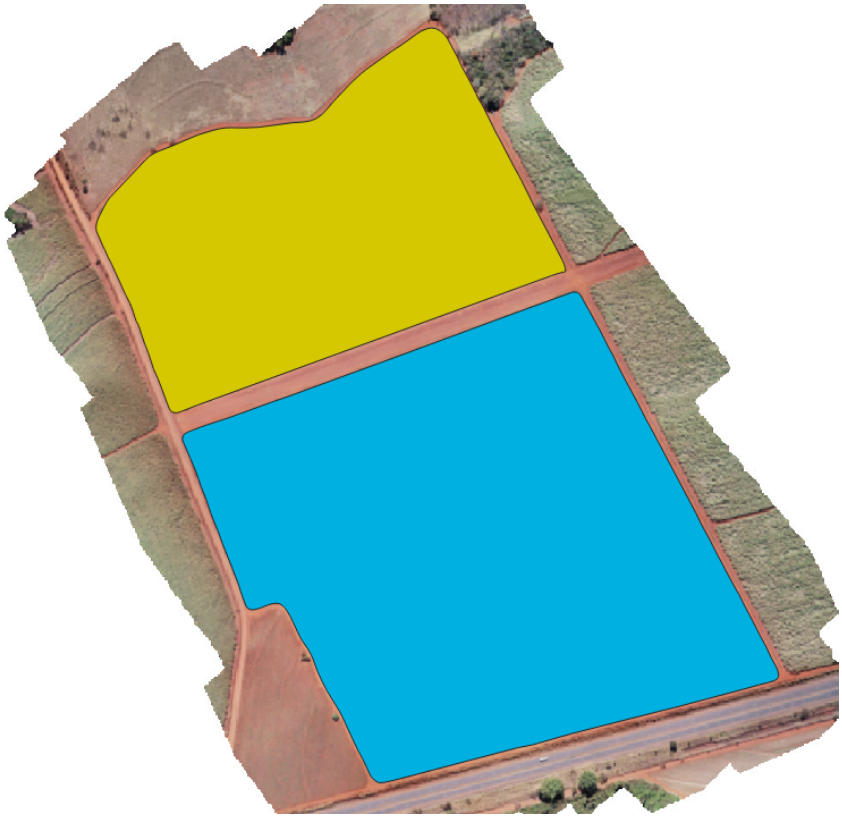

Figura 2.5: Marcação de taludes (amarelo e azul) em formato KML mostradas pelo QGIS sobre uma fazenda

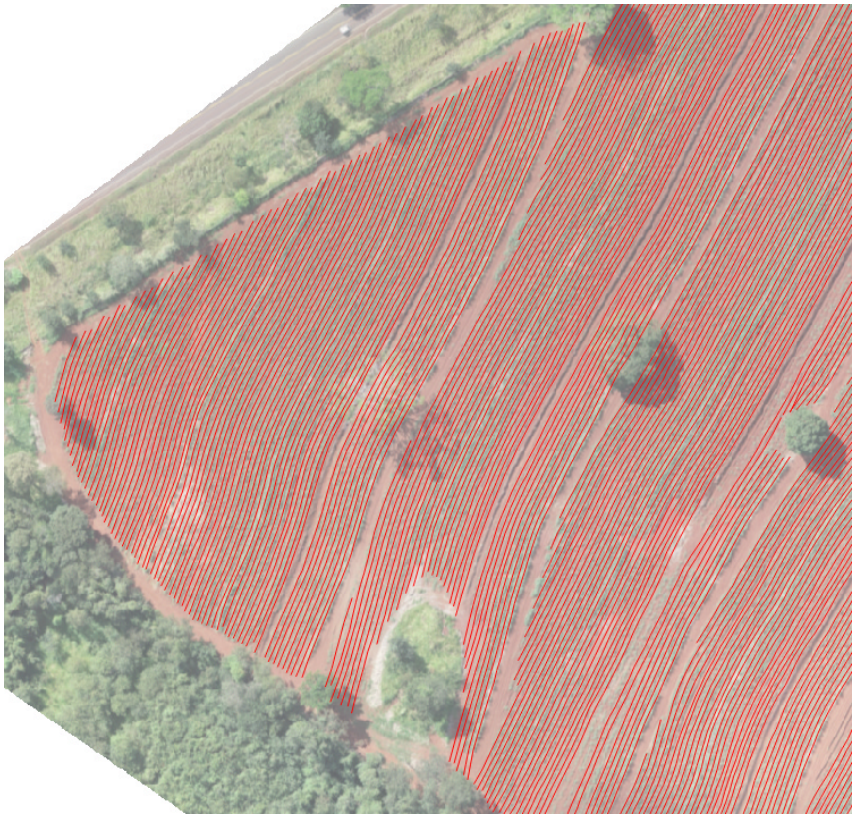

Figura 2.6: Linhas armazenadas em formato KML mostradas pelo QGIS sobre uma fazenda

marcações de falhas no plantio sobre as linhas de cana e marcações de entre-linhas. Todos esses arquivos estão no formato KML e podem ser visualizados sobre as imagens no QGIS. As marcações de linhas de cana são geradas para cada talude de uma fazenda e estão agrupadas em um único arquivo separadas internamente individualmente por linhas. As marcações de linhas não são totalmente precisas, contendo marcações de linhas de cana sobre estradas, árvores, vegetação nativa etc. As marcações de falhas no plantio são feitas sobre as marcações de linhas de cana e segmentos de linhas que podem considerar o crescimento das plantas, estendendo sua extensão por alguns centímetros. As marcações de entre-linhas são marcações das linhas que passam entre cada duas linhas de cana e trazem informações sobre a distância entre cada linha de cana em cada parte da linha para que possa ser analisado se a proximidade entre linhas está adequada, isto é, se não difere muito de 1,5m, que é o padrão adotado pela empresa que fornece os dados.

Embora para essa coleta inicial dos dados não haja informações quanto ao período em que eles foram gerados, sabe-se que a empresa gera novos dados com uma frequência de aproximadamente três meses. O acompanhamento com essa frequência tem como objetivo observar o desenvolvimento das plantas em diferentes etapas da produção. As imagens

podem conter diversos diferentes relevos, entretanto as marcações geográficas não contam com informações de altitude sobre o terreno. Além disso, apesar de se saber que todos os dados são gerados a cada processamento, nos conjunto de dados inicial há algumas imagens de fazendas que vêm desacompanhadas de suas respectivas marcações de linhas de cana e falhas ou então algumas marcações vêm desacompanhadas de suas respectivas imagens.

Para explorar o conjunto de dados inicial, foram usados os *softwares* QGIS, UnBVision, UnBMiner, linguagem de programação R com os pacotes `raster` e `rgdal` para a visualização e manipulação dos arquivos de imagem GeoTIFF e linguagem de programação Python na versão 3.x com as bibliotecas `tensorflow`, `opencv` e `gdal` para desenvolvimento de redes neurais multicamadas, processamento de imagens e manipulação de imagens georreferenciadas, respectivamente. Da exploração inicial pôde ser observado que o UnBVision não compreende o formato Geo Tagged Image File Format (GeoTIFF), sendo necessária sua conversão para um formato sem dados geográficos para a visualização e manipulação pelo UnBVision.

2.2.1 Exploração dos dados

Nesta seção serão apresentados os procedimentos feitos durante a exploração dos dados e os resultados obtidos. Diversos programas foram utilizados para manipular as imagens obtidas a fim de entender melhor a disposição dos dados e propôr uma abordagem para a solução do problema. Na Subseção 2.2.1 a linguagem R e bibliotecas foram utilizadas para visualizar e manipular os arquivos de imagens e de marcações de linhas de cana, na Subseção 2.2.1 o programa UnBVision foi utilizado para aplicar filtros nas imagens recortadas da exploração da Subseção 2.2.1 a fim de encontrar alguma manipulação das imagens que facilite a identificação de linhas de cana nelas, na Subseção 2.2.1 o programa UnBMiner foi utilizado para modelar redes neurais multicamadas para a identificação píxel-a-píxel de píxels de cana nas imagens e analisar seus desempenhos, na Subseção 2.2.1 a linguagem Python e bibliotecas foram utilizadas para criar redes neurais multicamadas para a identificação de píxels de cana nas imagens, reduzir imagens para diminuir o peso computacional do processamento de imagens, aplicar transformações sobre as imagens a fim de facilitar a identificação de linhas de cana e testar abordagens para a identificação de linhas de cana. Apesar de que nem todos os métodos testados durante essa exploração tenham se mostrado úteis para uso no desenvolvimento da solução, todos contribuíram para um melhor entendimento dos dados e serão apresentados nas subseções seguintes.

Exploração com R

Com as bibliotecas `raster` e `rgdal` para R é possível manipular arquivos de imagens georreferenciadas sem a necessidade de conversão para outros formatos além de se poder obter metadados sobre a estrutura dos dados geográficos. Com o auxílio da biblioteca `raster` é possível fazer recortes das imagens, mantendo as referências geográficas, e exportar os recortes para formatos de imagem com e sem suporte aos dados geográficos. Com o uso dessas bibliotecas é possível converter imagens das fazendas, ou recortes de imagens, para outros formatos compreendidos por uma variedade maior de programas, como o formato BMP, e manter seus dados geográficos associados salvos de outra forma, como arquivos de texto ou binários. Assim, contanto que as proporções dos recortes sejam mantidas, linhas detectadas em recortes podem ser mapeadas para suas localizações correspondentes com as coordenadas geográficas dos recortes. Além disso, também é possível manipular, com o auxílio da biblioteca `rgdal`, os arquivos de marcação de linhas de cana no formato KML, para obter os segmentos das linhas correspondentes a cada recorte da imagem. Essas linhas são armazenadas como sequências de coordenadas geográficas e podem ser visualizadas como linhas em suas posições no mapa com o QGIS. Linhas de cana que forem identificadas por este trabalho podem ser salvas no mesmo formato para que possam ser feitas comparações com as linhas geradas pelo INFOROW e determinar a qualidade dos resultados deste trabalho. Ao final desta exploração, imagens de fazendas foram recortadas em tamanhos menores e quadrados. Os recortes gerados nesta etapa foram utilizados na exploração com outras ferramentas e foram salvas no formato BMP com o tamanho de 253×253 píxels.

Exploração com o UnBVision

Com o UnBVision, é possível aplicar diversos filtros e transformações a uma imagem, como transformação de Red-Green-Blue (RGB) para tons de cinza, limiares por tons de cinza ou por RGB, operações morfológicas e outros e, dado um certo padrão, é possível contabilizar a quantidade de estruturas similares. O objetivo ao se utilizar o UnBVision neste trabalho era conferir se ele é capaz de manipular as imagens passadas das plantações de cana de modo a facilitar a identificação das linhas de cana e falhas.

O UnBVision aceita diversos formatos de imagens, mas não aceita o formato GeoTIFF. Então é necessário converter as imagens das plantações para um outro formato suportado pelo UnBVision, mas assim são perdidos os dados geográficos. Portanto é importante ter uma maneira de recuperá-los e converter novamente as imagens processadas pelo UnBVision para o formato GeoTIFF se for necessário utilizar as imagens processadas pelo UnBVision. Dentre os metadados que podem ser obtidos das imagens georreferenciadas,

as coordenadas do ponto superior esquerdo da imagem e a resolução horizontal e vertical dos píxels podem ser obtidas, isto é, qual a distância horizontal e vertical que um único píxel representa no sistema de coordenadas geográficos da imagem. Por exemplo, para uma imagem com GSD de 4cm/píxel, essas resoluções teriam valores de aproximadamente $0,04$ para um sistema de coordenadas que usa metros como unidade de distância. Assim, para recuperar as coordenadas geográficas de uma imagem pode-se usar as resoluções dos píxels e as dimensões em píxels de uma imagem para obter as outras coordenadas da imagem a partir da coordenada do ponto superior esquerdo.

Utilizando os recortes gerados na Subseção 2.2.1 com o UnBVision, foi possível aplicar filtros de realce de bordas e de limiar RGB da cor verde com uma planta de cana marcada anteriormente como exemplo para que as linhas de cana fossem identificadas na imagem como na Figura 2.7. Entretanto o UnBVision conta a quantidade de estruturas que ele identifica após a aplicação do limiar e, devido a segmentação excessiva, isto é, identificação de formas de maneira muito nítida, as linhas de cana não formam uma única estrutura reconhecida pelo UnBVision e sim centenas de pequenos pontos ao longo das linhas são identificados sendo bem diferenciáveis do solo ao redor. Assim, não é possível usar o UnBVision desta forma para identificar linhas de cana nas imagens. Operações morfológicas de abertura poderiam ser utilizadas para remover ruídos pequenos das imagens, entretanto isso pode acabar removendo linhas de cana muito finas, o que prejudica a qualidade dos dados.

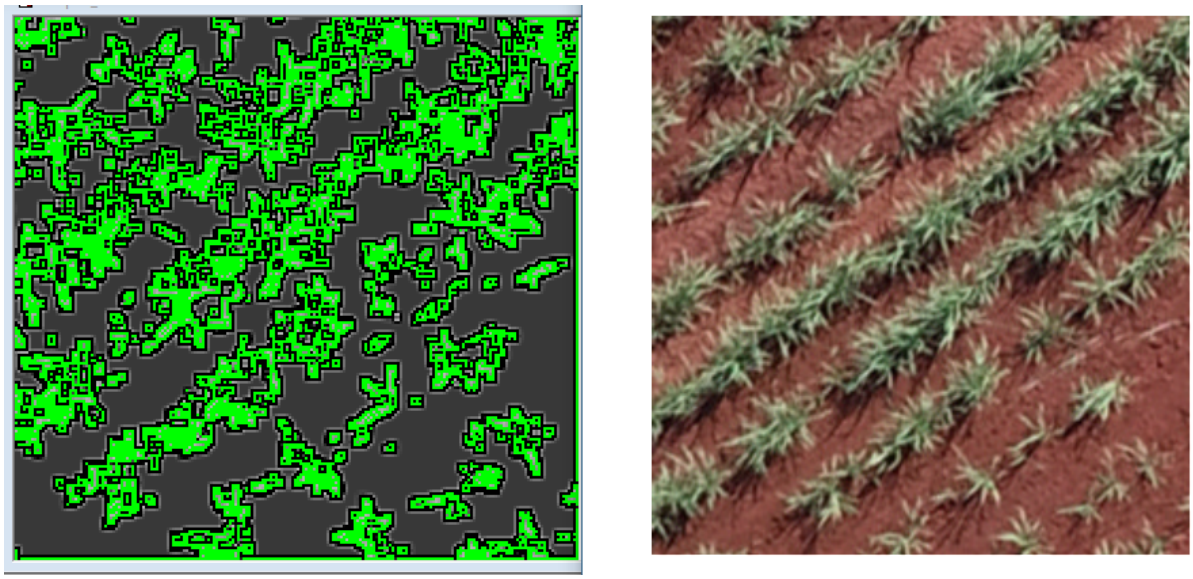

Figura 2.7: Recorte processado pelo UnBVision (esquerda) e recorte original (direita)

Se o UnBVision conseguisse contar cada linha de cana em uma imagem como uma estrutura, ainda é necessário explorar técnicas que utilizem esses resultados para efetivamente identificar e marcar as linhas de cana na imagem como dados geográficos. Resul-

tados similares poderiam ser obtidos ao processar as imagens com uma rede neural que fizesse uma marcação por píxel do tipo "verde ou não-verde" e depois com uma convolução também poderiam ser separadas as linhas de cana do solo. Essa abordagem é explorada nas Subseções 2.2.1 e 2.2.1.

Exploração com o UnBMiner

Com o UnBMiner é possível modelar rapidamente e facilmente diversos modelos probabilísticos, cada um chamado de um programa, como redes neurais multicamadas, treinar com um conjunto de dados e ver relatórios sobre o treinamento e sobre a avaliação dos dados de teste. Nesta exploração o programa de modelagem de redes neurais multicamadas foi utilizado para criar redes neurais para classificar píxel-a-píxel imagens de cana em píxels de cana e outros píxels com diferentes parâmetros de quantidade de épocas de treinamento, taxa de aprendizado, *momentum* e funções de ativação. Outros parâmetros como a quantidade de camadas ocultas e quantidade de neurônios nas camadas ocultas foram mantidos fixados em 1 camada oculta com 3 neurônios. O formato da entrada foi feito como 3 neurônios, um para cada valor RGB, normalizado para estar no intervalo $[0, 1]$, e o formato da saída foi feito como 2 neurônios, com as saídas possíveis $(0, 1)$ e $(1, 0)$, representando se um píxel é um píxel de uma planta de cana ou não. O motivo de interesse ao se utilizar esta ferramenta eram os relatórios gerados para o treinamento e avaliação dos dados, que poderiam ser utilizados para guiar a escolha dos parâmetros para a criação de uma rede neural como descrito acima com boa eficiência.

Para construir um conjunto de dados de treinamento e teste, alguns dos recortes produzidos na Subseção 2.2.1 foram selecionados à mão a fim de representar um conjunto de imagens representativo de todos os possíveis cenários que poderiam aparecer nas imagens das fazendas, como plantas de cana, árvores, estradas, maquinários etc. O programa GNU Image Manipulation Program (GIMP) foi utilizado manualmente para separar píxels de plantas de cana de outros píxels e gerar imagens para a construção de um conjunto de treino e teste para as redes neurais modeladas. As imagens de píxels separados gerados foram salvas em arquivos de texto com uma linha para cada píxel RGB, representado com três valores entre 0 e 1 e um valor em $\{0, 1\}$ representando se o píxel é um píxel de cana ou não separados por espaço. Após os arquivos de texto com píxels RGB e o de suas classes serem gerados, uma quantidade igual de valores de classe 0 e 1 foram selecionados e misturados, gerando um arquivo de conjunto de dados. Para o programa UnBMiner foi utilizada uma proporção de 90%/10% para os conjuntos de treinamento e teste, respectivamente.

Entretanto, o programa não possibilita testar lotes de diferentes configurações de parâmetros de maneira simples e, na verdade, apresenta erros quando mais de uma diferente

rede neural é treinada durante uma mesma execução do programa, como mostrado na Figura 2.8.

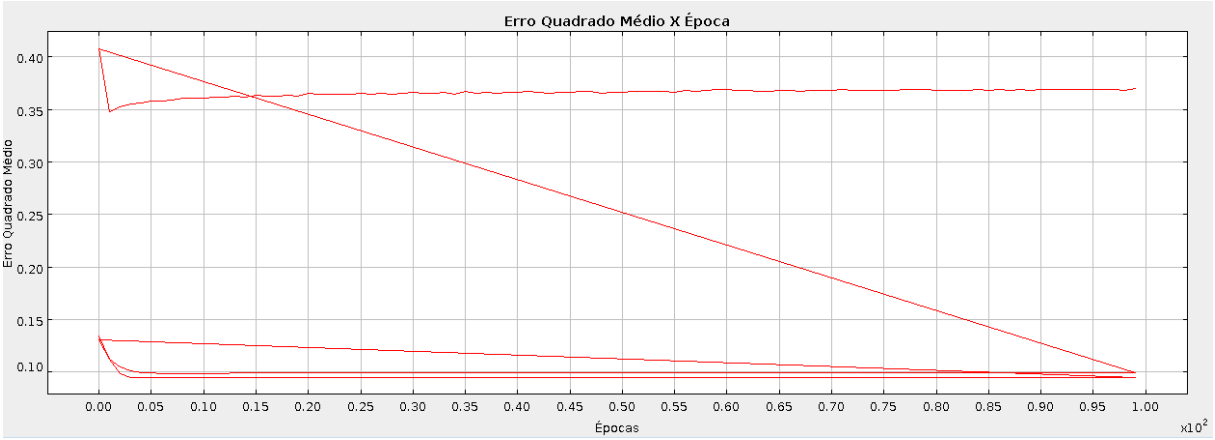

Figura 2.8: Erro no UnBMiner após múltiplos treinamentos de redes neurais

Exploração com Python

As bibliotecas `tensorflow` e `keras` são bibliotecas modernas e poderosas de código aberto para Python que possibilitam implementar diversos tipos de modelos probabilísticos. Com seu auxílio foram implementadas redes neurais multicamadas para classificar píxels de cana e de não-cana nos recortes gerados na Subseção 2.2.1 com 3 neurônios na camada de entrada para cada valor RGB, camada oculta com 10 neurônios e funções de ativação sigmoide e tangente hiperbólica e 2 neurônios na camada de saída com a função de ativação *softmax*. Para o conjunto de dados utilizado para treinar e testar as redes foram utilizados os dados gerados na Subseção 2.2.1. Diferentemente das redes neurais criadas no UnBMiner, os modelos de redes neurais criadas com o `tensorflow` podem aceitar imagens de diversos formatos se forem processadas por outras bibliotecas como `PIL` e `numpy` e as redes neurais treinadas para predizer novos valores podem receber imagens de recortes do plantio de cana e gerar como saída imagens binárias com píxels de cana e píxels de não-cana da mesma forma, como mostrado na Figura 2.9.

Com as bibliotecas `tensorflow` e `keras` também é possível passar um filtro de *max pooling* sobre as imagens geradas para reduzir seus tamanhos para diminuir o custo computacional do processamento das imagens preditas, como mostra a Figura 2.10. Com as imagens preditas e reduzidas, foram testadas três abordagens para tentar detectar as linhas de cana, a primeira, passar uma janela deslizante sobre as imagens reduzidas, tratar os píxels sob a janela como pontos no plano cartesiano e fazer a regressão linear desses pontos e ao fim juntar todas as linhas geradas pelas regressões lineares, como mostra a Figura 2.11, mas os resultados não foram satisfatórios, tendo muitas linhas geradas com

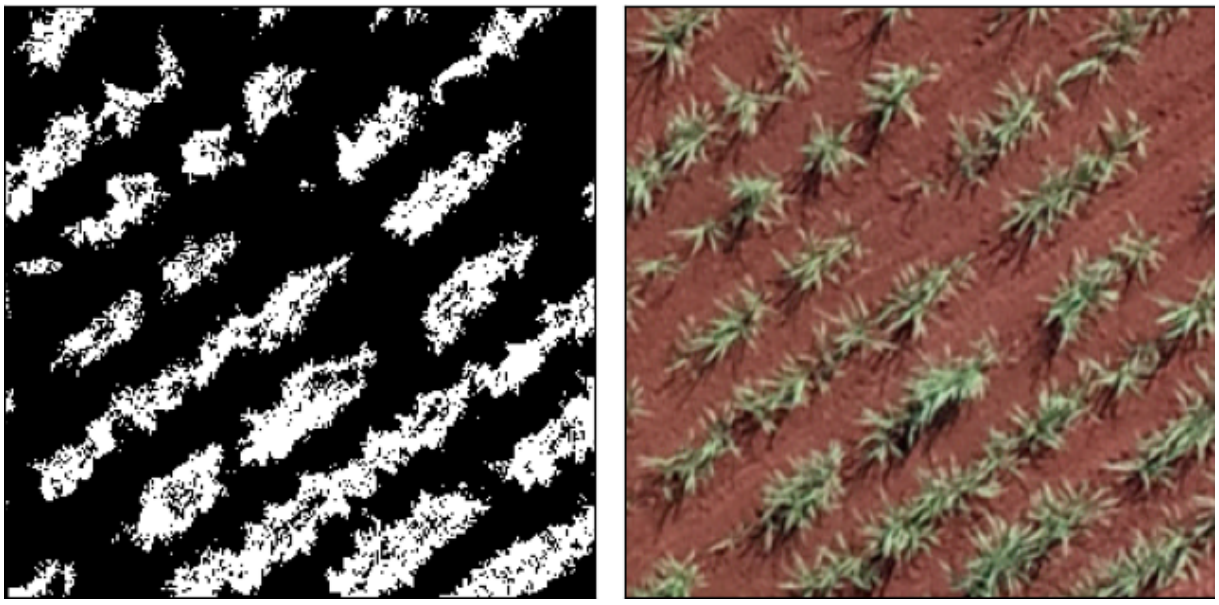

Figura 2.9: Imagem binária predita (esquerda) e imagem original (direita)

inclinações que não representavam as linhas de cana, a segunda, usar uma abordagem similar a sugerida em [3], que consiste de usar a transformada de Hough para identificar linhas em imagens binárias, que gera vários candidatos de linhas sobre uma imagem, cada uma com uma quantidade de "votos" representando a chance de serem realmente uma linha válida, e as linhas que obtiverem votos superiores a um determinado limiar são consideradas linhas válidas na imagem, e a terceira estratégia testada foi encontrar as "caixas delimitadoras" (*bounding box*) dos objetos da imagem binária e marcar a linha que atravessa essas caixas, sendo um objeto todo conjunto contíguo de píxels diferentes de 0.

Para aplicar a transformada de Hough, foi utilizada a biblioteca `opencv`, que implementa diversas funcionalidades para visão computacional. A utilização inicial da transformada de Hough foi feita usando a função `HoughTransformP` do `opencv`, que indicava utilizar o algoritmo de Canny para detecção de bordas de uma imagem, entretanto os resultados obtidos inicialmente não foram satisfatórios, como mostra a Figura 2.12, pois linhas estavam sendo detectadas nas bordas das linhas de cana e não representavam as linhas de cana realmente. Para melhorar esses resultados, era necessário fazer com que as imagens das linhas de cana parecessem mais com somente linhas, então o pré-processamento com o algoritmo de Canny foi substituído por uma estratégia de esqueletização das imagens, que consiste de uma sequência de operações de erosão e dilatação das imagens, isto é, um filtro é passado sobre as imagem e um píxel na imagem resultante só é branco caso todos os píxels sob o filtro sejam brancos, caso contrário o píxel é preto, fazendo como que as bordas da imagem sejam reduzidas, ou erodidas, e depois um filtro é passado sobre

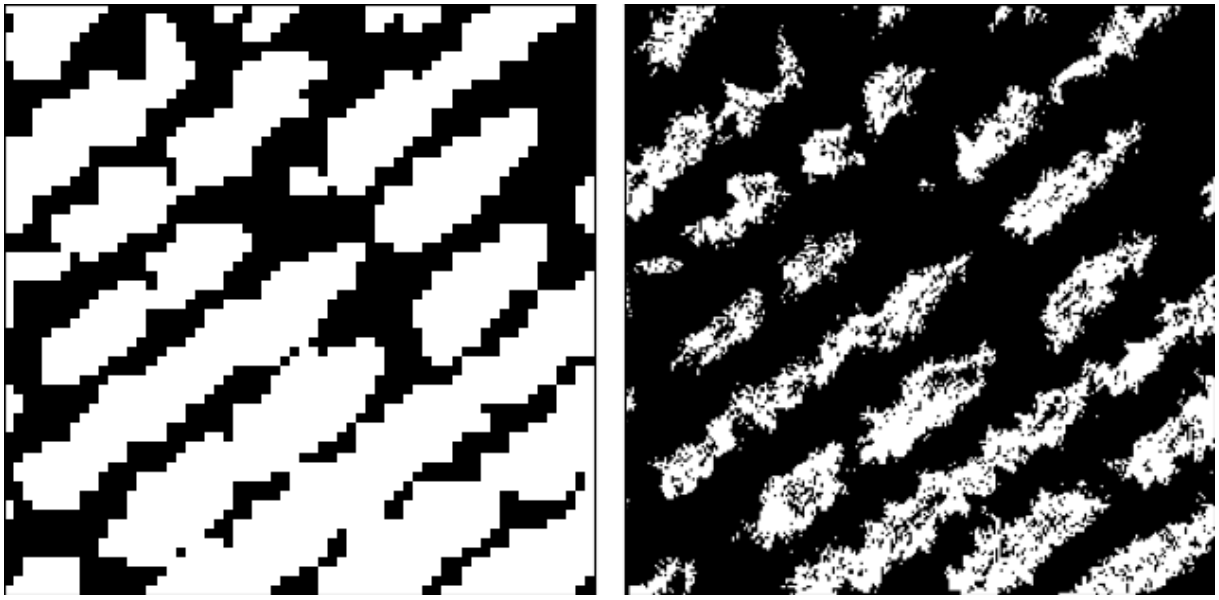

Figura 2.10: Imagem binária reduzida (esquerda) e imagem original (direita)

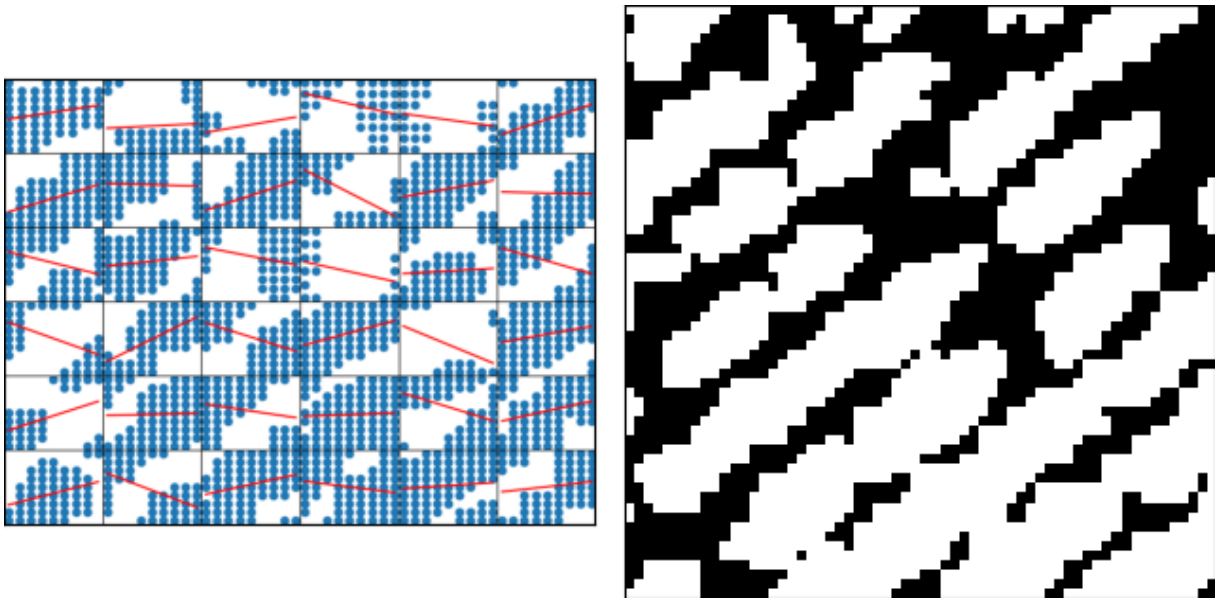

Figura 2.11: Mosaico de regressões lineares

a imagem e um píxel na imagem resultante só é branco se pelo menos um píxel sob o filtro é branco, caso contrário o píxel é preto, fazendo com que as bordas da imagem seja acentuadas, ou dilatadas. Um exemplo de uma imagem esqueletizada é mostrado na Figura 2.13. Usando as imagens esqueletizadas com a transformada de Hough, ajustando seus parâmetros, como o tamanho mínimo de uma linha para ser aceita como uma linha válida e o espaçamento máximo entre segmentos de linhas para que sejam conectados em um único segmento e a quantidade de votos que um candidato de linha deve receber para ser aceita como uma linha válida, os resultados melhoraram bastante, como pode ser visto

na Figura 2.14.

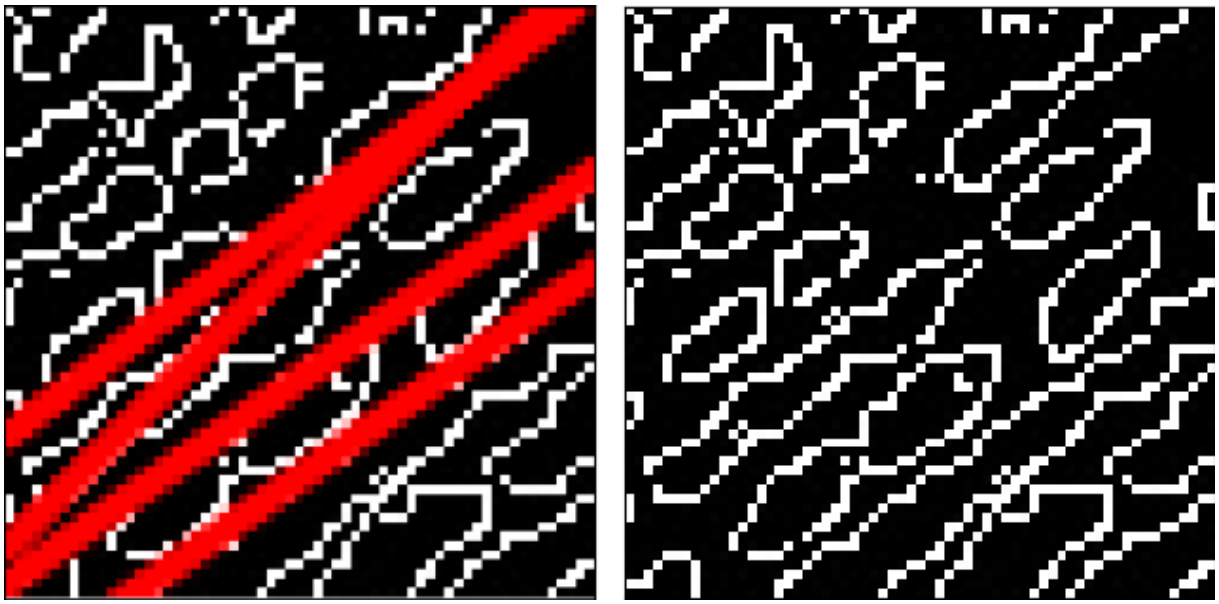

Figura 2.12: Transformada de Hough com o algoritmo de Canny

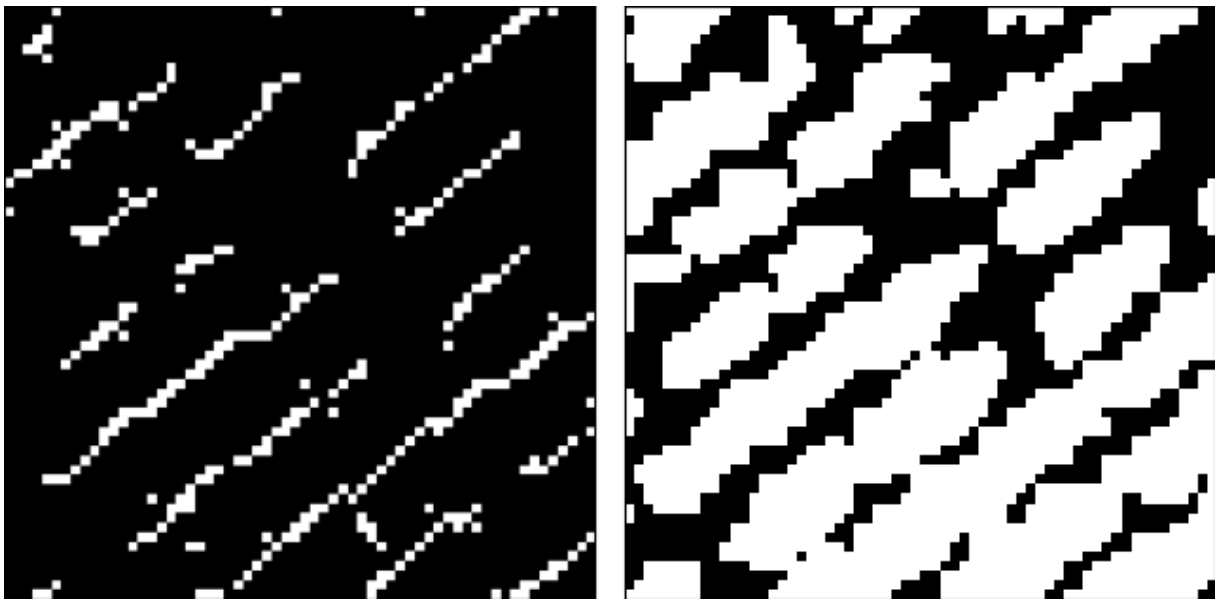

Figura 2.13: Imagem esqueletizada (esquerda) e imagem original (direita)

A abordagem que utiliza a transformada de Hough para detectar a linha é bem conhecida para a detecção de linhas, mas seu uso neste projeto não é tão simples. Para um bom funcionamento do algoritmo, é necessário que seja utilizada uma imagem com linhas bem definidas e suficientemente retas, além de um ajuste dos parâmetros adequado para detectar todos os segmentos de linhas em uma imagem. A transformada de Hough também não encontra somente um segmento para cada trecho de plantação na imagem,

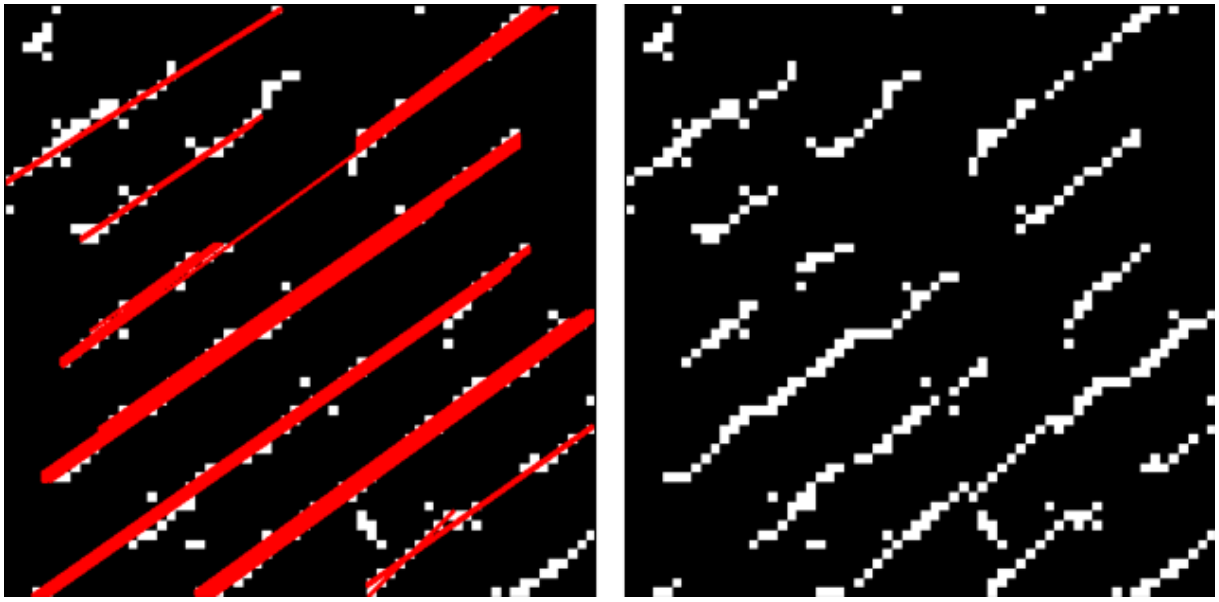

Figura 2.14: Linhas detectadas (esquerda) e esqueleto de imagem utilizado (direita)

são gerados diversos segmentos para cada pedaço de um segmento real que muitas vezes são sobrepostos. Isso pode ser visto na Figura 2.14, que tem marcado em vermelho os segmentos encontrados pelo algoritmo, na qual é possível observar diferentes grossuras para cada parte de uma linha. Portanto ao utilizar Hough para detectar as linhas de cana e falhas no plantio, outros desafios não simples aparecem: gerar um esqueleto de uma imagem binária que tenha linhas relativamente retas representativas das linhas reais a serem detectadas, ajustar os parâmetros do algoritmo para detectar todas as linhas necessárias e ignorar qualquer outra coisa, para todos os diferentes recortes de um talude, conectar os diversos segmentos que compõem uma linha de cana para que possam ser convertidos em coordenadas geográficas novamente.

Por outro lado, a abordagem de detecção pelas caixas delimitadoras é simples, não dependente da variação de algum parâmetro e detecta somente uma linha para cada objeto, o que fornece um par de coordenadas na imagem binária que podem ser traduzidas com facilidade para coordenadas geográficas, restando apena o problema de encontrar um esqueleto ou uma imagem com um conjunto de objetos de tal forma que cada objeto corresponde a um trecho de uma linha de cana. Esse último tipo de imagem é difícil de se obter, pois para uma recorte de plantação com plantas grandes pode ocorrer que duas linhas de cana diferentes acabem sendo conectadas na imagem binária como mostra a Figura 2.15, portanto é necessário que seja feito um processamento das imagens binárias antes de aplicar alguma técnica de detecção de linhas.

Com a exploração dos dados com Python foi aprofundado em um nível mais técnico o entendimento dos problemas que devem ser resolvidos a fim de alcançar o objetivo, sendo

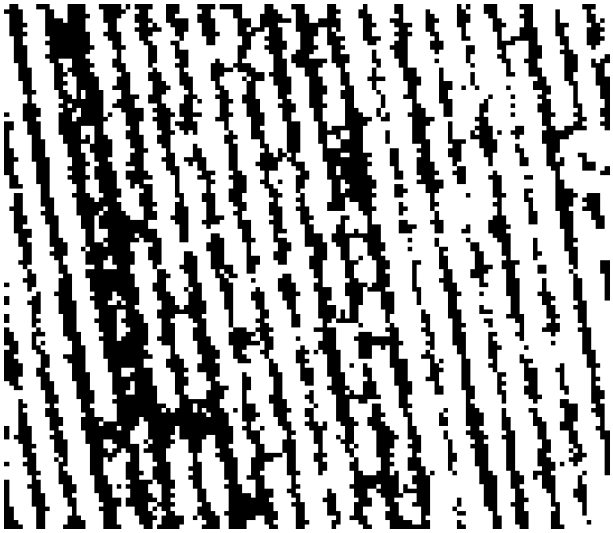

Figura 2.15: Imagem binária com linhas de cana grossas em que duas linhas de cana se conectam

um desses problemas a integração de cada parte do projeto. Dada a disponibilidade de bibliotecas Python que cobrem todas as áreas do projeto (a biblioteca `GDAL` equivalente à do R está disponível para Python, permitindo a manipulação de diversos tipos de dados georreferenciados) e para facilitar a integração entre elas, o uso de Python como linguagem de programação principal do projeto se mostra adequado. Além disso, o uso de Python para a modelagem das redes neurais foi decidido em favor do UnBMiner devido a modernidade e robustez que as bibliotecas para Python exploradas oferecem, diminuindo a chance de erros como os apresentados pelo UnBMiner, que não recebe atualizações, sem perder a possibilidade de analisar os treinamentos e desempenhos dos modelos criados com o uso da ferramenta *TensorBoard*.

2.3 Trabalhos relacionados

O problema de detecção de fileiras de plantio é um problema recorrente na agricultura de precisão, já que serve como base para muitas análises que podem ser feitas sobre plantações ao transformar as imagens originais do plantio em dados inteligíveis ao computador. As imagens das plantações podem ser capturadas do nível do solo ou serem aéreas, com o auxílio de um VANT, existindo métodos mais apropriados para detectar as linhas de plantio dependendo da forma de captura. O uso de redes neurais para resolver esse problema de detecção existe em diversos trabalhos, usando redes convolucionais profundas para classificação de píxels em uma imagem como píxels de planta, terreno, maquinário etc. Essas redes podem receber como entrada um único píxel da imagem ou uma pequena

área dela, que pode dar às redes mais contexto sobre o que deve classificar, sendo essa última capaz classificar os píxels de forma mais correta [4].

Há muitos parâmetros que devem ser levados em consideração para a modelagem de uma rede neural profunda, sendo um deles as classes que podem ser obtidas da classificação. Uma modelagem pode tentar cobrir todos os casos de diferentes objetos que podem ser encontrados em uma imagem, gerando uma classe para cada, o que faz necessário saber de antemão e ter exemplares de imagens com todas as classes de objetos, ou pode classificar binariamente os dados entre a classe desejada e uma classe de "qualquer outra coisa", o que torna a modelagem mais simples. Especificamente para o caso de classificação de imagens de vegetação obtidas por VANT, classificadores de classe única não só têm a praticidade de uma modelagem simples como também são capazes de apresentar boa qualidade dos resultados [5].

Entretanto, as redes neurais fazem somente parte do trabalho da detecção de fileiras de plantio: geram uma imagem binária a partir da imagem original limitando e simplificando o espaço de busca pelas fileiras. Com uma boas imagem gerada a partir da classificação pela rede neural, diferentes métodos de processamento de imagem podem ser utilizados para encontrar linhas. Levando em consideração o fato de que a prática de plantio em fileiras paralelas é comum, métodos que identifiquem segmentos de linhas simples podem ser utilizados, sendo o mais famoso e presente em diversos trabalhos no estado-da-arte no processamento de imagens para agricultura de precisão a transformada de Hough. Ainda assim, muitos outros métodos existem, como a ideia de passar um *scanner* de linha sobre a imagem e identificar linhas ou trechos de linha abaixo [6].

Uma outra abordagem propõe o uso de uma nuvem de pontos com o k-médias para agrupar pontos em conjuntos que representam linhas. Apesar da abordagem interessante que foge do padrão na área, o método não consegue superar a transformada de Hough na eficiência mantendo a mesma qualidade dos resultados, o que reforça o uso de Hough para esse tipo de problema [7].

Por fim, uma vez que as linhas tenham sido de fato detectadas e que possam ser usadas para análise do plantio, um dos dados mais relevantes a serem obtidos é a localização das falhas nas linhas, que são como o complemento das linhas de plantio. O desafio nesse problema não é o da localização das falhas propriamente ditas, já que se sabe onde o plantio está e sua disposição, mas sim saber o que caracteriza um trecho de desfalque no plantio como uma falha. A forma de tomar essa decisão depende do tipo da cultura do plantio, especificamente para culturas de cana-de-açúcar é comum e bem estabelecido aceitar desfalques maiores que 50cm como falhas [8].

Capítulo 3

Metodologia

Visando desenvolver uma solução para o prolema proposto que cumpra com os objetivos estabelecidos, uma aplicação com interface gráfica será desenvolvida para que os dados de entrada e saída sejam operados. Nas seções seguintes serão mostradas, detalhadamente, as transformações que serão realizadas sobre os dados de entrada, o formato dos dados de saída, a arquitetura do modelo de rede neural e da interface gráfica desenvolvida.

3.1 Preparação dos dados

Nesta seção serão discutidos os detalhes relacionados à preparação dos dados para o treinamento da rede neural. Primeiro será apresentada a seleção dos dados que serão utilizados dentre os dados disponíveis apresentados no Capítulo 2, em seguida será apresentado como esses dados serão preparados para serem processados e por fim será dado um resumo do conjunto de dados construído.

3.1.1 Seleção dos dados

A aplicação proposta neste projeto deve processar uma imagem de entrada e gerar como saída as coordenadas geográficas que definem caminhos sobre as linhas de plantio de cana e sobre as linhas de falha. Portanto é clara a necessidade de utilizar as imagens de plantações disponíveis no formato GeoTIFF.

Entretanto, como observado na Seção 2.2, essas imagens podem conter objetos indesejados à detecção de linhas de cana, como estradas, árvores, carros e outras coisas que estão fora dos limites dos taludes de cana. Assim, é interessante utilizar quando disponível as marcações de limites dos taludes, que servem como uma máscara sobre as imagens, podendo recortar o conteúdo original às áreas de interesse do processamento.

A abordagem deste projeto tenta criar as linhas de plantio e de falhas nas plantações a partir do processamento das imagens das plantações com inteligência artificial e visão computacional, então as linhas já existentes para os dados disponíveis não são utilizadas para o desenvolvimento da aplicação, mas podem ser utilizadas para a validação dos resultados.

Os outros dados intermediários gerados pelo INFOROW também não serão utilizados. Os arquivos de formato proprietário têm conteúdo desconhecido, portanto não podem ser usados. As imagens convertidas para outros formatos não têm informações a mais se comparadas com as imagens GeoTIFF originais. A nuvem de pontos gerados propaga erros que levam o INFOROW a falhar na marcação de linhas como mostrado anteriormente e portanto não são úteis para este projeto.

3.1.2 Limpeza dos dados

A limpeza e preparo dos dados selecionados são feitas de forma distinta para os dois processos aos quais os dados serão submetidos, o treinamento da rede neural e a detecção de linhas por meio da aplicação gráfica.

Para o treinamento da rede neural

Para este processo os dados devem ser preparados em forma de um conjunto de recortes das imagens das plantações com uma classificação píxel-a-píxel da imagem correspondente para serem usados como conjunto de treino e de avaliação da rede neural.

As seguintes etapas foram seguidas para preparar os dados:

1. Recortes das imagens contendo plantas de cana, terreno de plantio, outras plantas, estradas, carros, tratores e outros objetos são obtidos através do QGIS com o objetivo de ter uma amostra representativa das imagens que podem ser encontradas nos taludes de cana;

2. Cada recorte é separado em janelas não sobrepostas com a mesma topologia da camada de entrada da rede neural, por exemplo, para uma rede neural que recebe uma entrada no formato 5×5 píxels, uma janela de 5×5 píxels sobre a imagem gera uma matriz de $5 \times 5 \times 3 = 75$ valores RGB. Caso o tamanho da imagem não seja compatível com o tamanho das janelas, as bordas direita e inferior da imagem são descartadas;

3. Utilizando o GIMP, uma máscara de transparência é aplicada manualmente sobre os recortes obtidos cobrindo todos os píxels identificados como não sendo de plantas de cana;

4. Utilizando Python, os recortes mascarados são utilizados para gerar uma imagem binária correspondente de tal forma que os pixels mascarados recebem o valor 0 e os não mascarados recebem o valor 1. Os valores 0 e 1 correspondem as classes identificadas pela rede neural como “cana” e “não-cana”;

5. As imagens binárias produzidas são reduzidas por uma convolução do tipo *max pooling* para que cada píxel corresponda a classe observada dos píxels das janelas do recorte;

6. Cada janela de píxels e sua classe observada são agrupadas e colocadas em uma lista, que é embaralhada e depois tem valores removidos de forma que a mesma quantidade de pares de janela de píxels e classe observada permaneça na lista para cada classe;

7. A lista resultante é dividida em um conjunto de dados de treino e em um conjunto de dados de avaliação para a rede neural.

Para a detecção de linhas

O preparo dos dados para a detecção de linhas tem o objetivo de resolver os problemas apontados na 2.2.1 referentes a necessidade de um “esqueleto” de uma imagem binária que contenha um conjunto de objetos de forma que cada objeto represente um trecho de linha de cana na imagem original. O processo de geração desses esqueletos usa a transformada de Hough e operações morfológicas sobre a imagem binária para limpar ruídos. Considerando a Figura 3.1 como uma imagem binária de exemplo, esse processo é feito na seguinte sequência de passos:

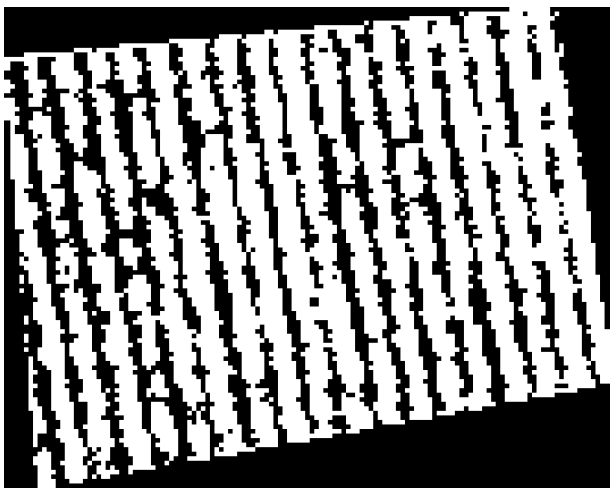

Figura 3.1: Imagem binária usada gerada pela rede neural

1. Um esqueleto inicial da imagem binária é criado para identificar a inclinação das linhas de cana na imagem;

   Levando em conta que as plantações estão dispostas em fileiras paralelas, para um recorte suficientemente pequeno, as fileiras de cana aparecem como linhas retas com inclinação similar entre si. Então, usando o esqueleto inicial gerado, que apesar de conter bastante ruído, como pode ser visto na Figura 3.2, as linhas principais ainda se destacam, e com a transformada de Hough são identificados segmentos de linhas sobre esse recorte a fim de descobrir a inclinação mediana das fileiras de cana para o recorte.

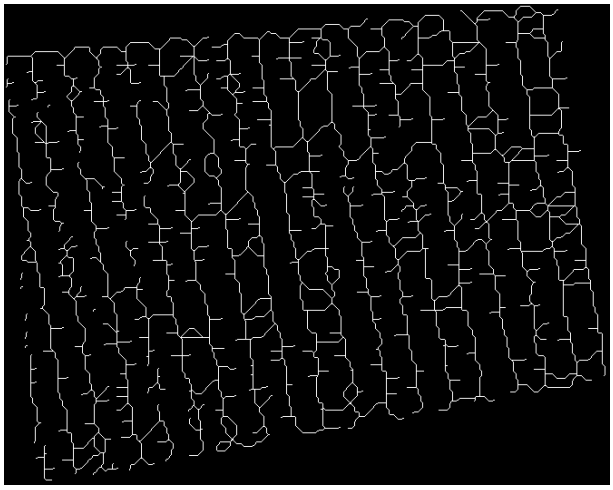

Figura 3.2: Esqueleto inicial gerado a partir da imagem binária

2. Um filtro com a mesma altura da imagem binária e com uma linha começando no canto superior esquerdo do filtro até o canto inferior direito com ângulo de inclinação mediano das linhas de cana é criado, como mostra a Figura 3.3, e passado sobre o esqueleto. Se a quantidade de pixels diferentes de 0 sob o filtro ultrapassar um determinado limiar, todos os píxels sob o filtro são transformados em 1, caso contrário, são transformados em 0.

3. A imagem filtrada é esqueletizada novamente, gerando dessa vez um esqueleto que representa mais precisamente as linhas presentes na imagem binária original.

4. Por fim, o esqueleto refinado é operado bit-a-bit com a imagem binária original com o operador lógico `E` para recuperar as falhas e interrupções nas linhas. A imagem resultante é operada com o esqueleto refinado com o operador lógico `OU EXCLUSIVO` para obter uma imagem complementar às linhas de cana, ou seja, com possíveis falhas no plantio.

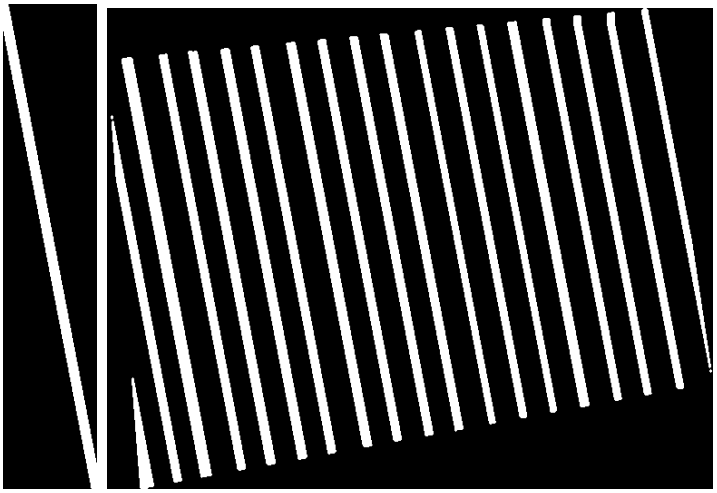

Figura 3.3: Filtro criado para o esqueleto (esquerda) e esqueleto filtrado (direita)

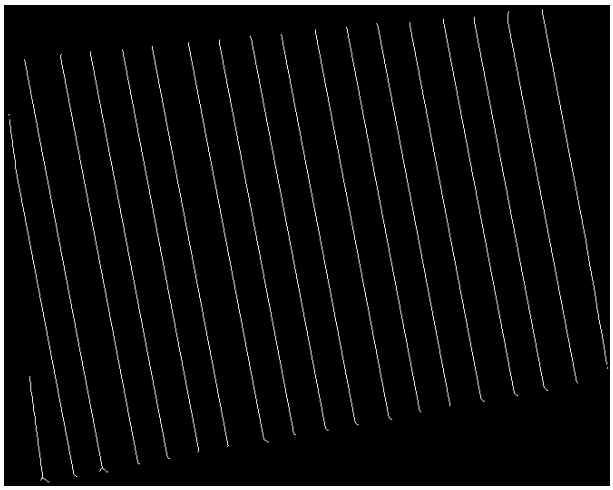

Figura 3.4: Esqueleto refinado criado a partir do esqueleto filtrado

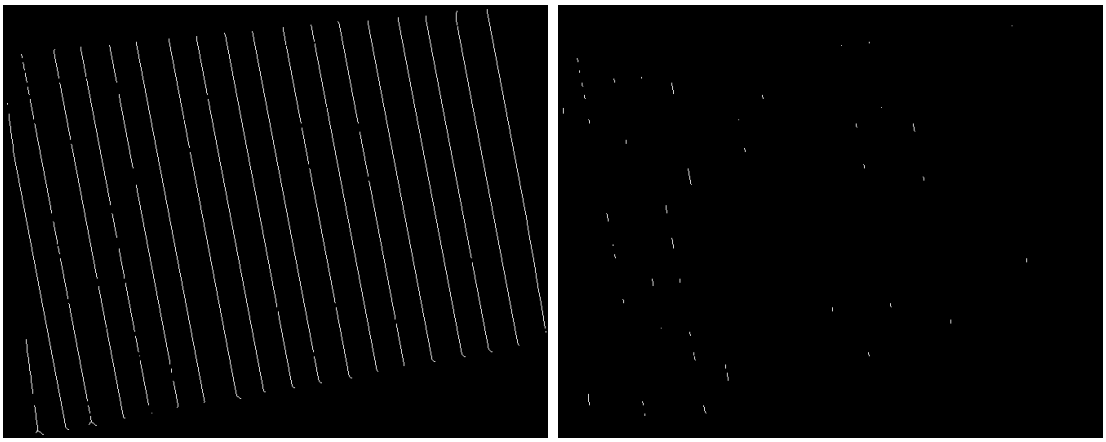

Figura 3.5: Esqueleto das linhas de cana (em cima) e das falhas nas linhas (embaixo)

Vários recortes diferentes das imagens originais foram feitos manualmente usando o QGIS para serem usados para o teste de cada etapa deste processamento. Devido ao tamanho das imagens das fazendas disponíveis, aplicar esse processo nas imagens inteiras para validação é inviável. Uma amostra de como realizar algumas operações simples no QGIS que podem ser usadas para gerar recortes e máscaras das imagens é apresentada no Anexo A.

3.2 Modelagem

Nesta seção são apresentadas as técnicas selecionadas para a classificação de píxels de uma imagem RGB e para a detecção de linhas em imagens binárias, a arquitetura dos módulos de cada um desses processo e da aplicação gráfica desenvolvida. Também é mostrado como é feita a interação do usuário com a aplicação gráfica, desde como passar uma nova imagem para ser processada até obter o arquivo de saída com as linhas e falhas nos taludes de cana.

Para o desenvolvimento de todos os módulos do modelo foi escolhida a linguagem Python, pois oferece alternativas bem conhecidas para desenvolver todos os módulos necessários, garantindo uma fácil integração entre eles, além de proporcionar uma experiência de desenvolvimento de protótipos e novas funcionalidades rápida que facilita os testes de diferentes abordagens e configurações.

Para manipular os dados geográficos, a biblioteca Geospatial Data Abstraction Library (GDAL) foi utilizada, pois é uma biblioteca completa e robusta para manipular esse tipo de dados, sendo utilizada pelo QGIS e pelo Google em produtos como Google Earth. Com essa biblioteca é possível manipular imagens georreferenciadas de tamanhos enormes e obter metadados da imagem com facilidade, bem como obter recortes de regiões específicas em matrizes no formato da biblioteca `numpy`, que se integra com as outras bibliotecas usada no projeto e permite a manipulação fácil de matrizes.

3.2.1 Classificação de píxels

Ao longo do desenvolvimento do projeto diferentes arquiteturas de rede foram testadas para classificar os píxels de uma imagem. Como o projeto seguiu o modelo do CRISP-DM, seu desenvolvimento seguiu um ciclo em que cada etapa do desenvolvimento é avaliada e, dependendo da sua avaliação, deve ser reiniciada. Dessa forma, diversos modelos foram construídos durante o desenvolvimento do projeto. A modelagem da rede neural teve várias iterações em que em cada uma foram avaliados um ou mais aspectos das redes neurais desenvolvidas e os modelos mais bem avaliados foram usados como base para as novas iterações.

Para a classificação dos píxels de uma imagem RGB foi desenvolvido um módulo Python com as funções de:

- Construir, salvar e carregar um modelo de rede neural;
- Montar conjuntos de dados para treinamento a partir de uma sequência de imagens pré-classificadas;
- Treinar um modelo de rede neural e salvar dados sobre o treinamento para análise posterior;
- Classificar os píxels de imagens usando um modelo treinado e salvar as imagens resultantes em disco.

As imagens utilizadas para o treinamento e classificação são representadas por matrizes tridimensionais com a última dimensão armazenando os valores RGB de cada píxel. As imagens que representam as classes observadas para o conjunto de treinamento e as imagens geradas pela classificação a partir do modelo são representadas como matrizes bidimensionais com os valores dos píxels sendo 0 ou 255.

Os modelos de redes neurais foram desenvolvidos utilizando a biblioteca `tensorflow`, pois oferece uma maneira simples de montar redes com diferentes topologias, otimizadores, funções de ativação e de erro, além permitir salvar e carregar modelos e os pesos de um modelo em disco, e registrar metadados dos treinamentos que podem ser visualizados de forma interativa com a ferramenta `tensorboard`.

Considerando o objetivo dessa rede neural, classificar píxels em sendo de uma planta de cana ou não, a estrutura da rede foi definida de forma simples, com uma única camada oculta. É esperado que uma rede com essa estrutura seja suficiente para resolver o problema, dependendo somente do ajuste de seus parâmetros, como taxa de aprendizado, *momentum*, função de ativação e tamanho de cada camada etc. A avaliação dos modelos treinados foi feita avaliando as métricas de acurácia e minimização da função de erro, levando também em consideração a oscilação desses valores ao longo do treinamento.

Inicialmente foram testados modelos que avaliavam a diferença entre usar píxels em tons de cinza contra píxels RGB. Então foram desenvolvidos modelos que tinham camadas de entrada com um único neurônio (píxel em tom de cinza) e com três neurônios (píxel RGB). A construção dos conjuntos de dados para o treinamento é similar ao descrito anteriormente, considerando uma janela de 1×1 píxels. Para a conversão da imagem colorida para tons de cinza foi utilizada a seguinte equação:

$$I^c(x, y) = 1.262 \times G_c(x, y) - 0.884 \times R_c(x, y) - 0.311 \times B_c(x, y) \tag{3.1}$$

na qual G_c, R_c, B_c são os canais verde, vermelho e azul da imagem RGB e I^c é a imagem em tons de cinza, com (x, y) representando um píxel específico em cada canal ou imagem [9].

Os modelos com píxels coloridos se mostraram mais precisos na classificação das imagens, indicando que a perda de informações de cor na conversão para escala de cinza impacta de forma prejudicial na classificação desejada.

Foi também comparado o uso de uma classificação píxel-a-píxel contra uma classificação baseada em objeto [4], em que um objeto é simplesmente uma janela de píxels "não-vazios" adjacentes. Inicialmente foram feitos testes com píxel-a-píxel contra uma janela de tamanho 3×3 píxels, o que mostrou uma melhor performance com a janela de píxels. Esse resultado indica que o contexto da região de um píxel traz um ganho de informação benéfico para a performance da rede, portante foram feitos testes a fim de identificar o tamanho da janela que melhor se adéqua à rede. Foram testados quatro tamanho diferentes para as janelas, 3×3, 4×4, 5×5 e 6×6 píxels. As janelas maiores obtiveram obtiveram os maiores valores para acurácia e menores para a função de erro em comparação com as menores, entretanto, quanto maior uma janela mais "quadriculada" a imagem resultante é. Isso porque a classe escolhida para cada janela é definida para todos os píxels abaixo dela, o que reduz a qualidade da imagem gerada, então, com o objetivo de maximizar a performance da rede e obter uma imagem visualmente representativa das linhas de cana, a janela de 5×5 píxels foi escolhida.

Os valores para taxa de aprendizado, *momentum* e quantidade de neurônios na camada oculta foram testados em diferentes combinações. Primeiro, foram treinadas redes com os valores para taxa da aprendizado de 0.10 até 0.19, de 0.01 em 0.01, e *momentum* de 0.1 até 0.9, variando de 0.1 em 0.1, mas mesmo as redes com os melhores valores para a acurácia por época e erro por época oscilavam muito, como mostra a Figura 3.6

O problema da oscilação foi identificado na taxa de aprendizado, que possivelmente estava grande demais para que a rede pudesse se desenvolver adequadamente. Foram treinadas redes com valores para a quantidade de neurônios na camada oculta variando de 10 até 60, de 5 em 5, para a taxa de aprendizado variando de $1 \cdot 10^{-5}$ até $1 \cdot 10^{-4}$, com passos de $1e \cdot 10^{-5}$, para o *momentum* variando de 0.5 até 0.9, com passos de 0.1. As redes com melhores resultados para acurácia por época e erro por época foram as redes com as combinações de valores $(60, 1 \cdot 10^{-4}, 0.7)$, $(40, 1 \cdot 10^{-4}, 0.5)$ e $(60, 1 \cdot 10^{-4}, 0.8)$, para tamanho da camada oculta, taxa de aprendizado e *momentum*, respectivamente.

Ao longo do desenvolvimento foram testados diferentes otimizadores para os modelos, *Nadam*, *Adam*, *RMSprop* e *SGD*, e diferentes funções objetivo, *Sparse Categorical Crossentropy* e *Mean Squared Error*, sendo o otimizador *SGD* e a função objetivo *Sparse Categorical Crossentropy* resultando nos modelos melhor avaliados.

Figura 3.6: Evolução da acurácia pelas épocas (em cima) e do erro pelas épocas (embaixo) para as combinações de taxa de aprendizado e *momentum* com melhores resultados oscilando ($(60, 1 \cdot 10^{-4}, 0.7)$ em azul escuro, $(40, 1 \cdot 10^{-4}, 0.5)$ em azul claro e $(60, 1 \cdot 10^{-4}, 0.8)$ em rosa)

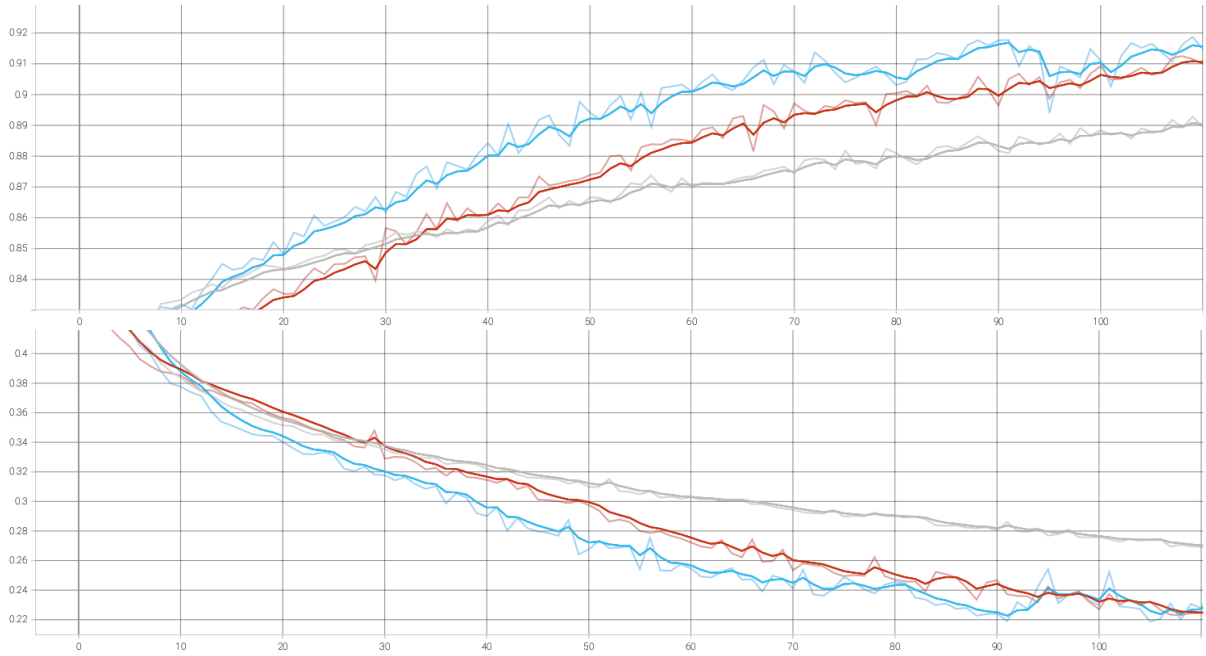

Figura 3.7: Evolução da acurácia pelas épocas (em cima) e do erro pelas épocas (embaixo) para as combinações de taxa de aprendizado e *momentum* com melhores resultados mais estáveis ($(60, 1 \cdot 10^{-4}, 0.7)$ em azul, $(40, 1 \cdot 10^{-4}, 0.5)$ em vermelho e $(60, 1 \cdot 10^{-4}, 0.8)$ em marrom)

Com essas ferramentas e com base na exploração dos dados realizada ao longo do desenvolvimento do projeto, o modelo de rede neural selecionado para ser utilizado tem a seguinte estrutura: três camadas totalmente conectadas, sendo uma de entrada com 75 neurônios, correspondente a uma janela de 5×5 píxels RGB, uma oculta com 60 neurônios e função de ativação sigmóide e uma de saída com dois neurônios com ativação *softmax*.

Os neurônios da camada de saída avaliam suas entradas e resultam em um valor entre 0 e 1, representando a probabilidade de que os píxels passados como entrada para a rede sejam de cada classe, sendo assim, seus valores são complementares e somam 1. O primeiro neurônio da camada de saída representa a classe "não-cana" e o segundo a classe "cana". Portanto, os valores de saída do primeiro neurônio são ignorados e os valores do segundo neurônio são utilizados para determinar a classe resultante de cada janela de 5×5 píxels.

Para determinar a classe resultante a partir da probabilidade obtida, o valor da probabilidade é comparado com os valores testados empiricamente para a mínima e máxima probabilidade que os píxels das classes "não-cana" e "cana" são classificados corretamente. Dependendo do resultado da comparação a probabilidade é traduzida para uma das classes citadas ou para a classe "indefinido". Com testes e avaliações visuais desta abordagem com diferentes imagens e dada a necessidade de produzir uma imagem binária a partir de uma imagem RGB, os píxels da classe "indefinida" foram classificados como "não-cana".

3.2.2 Detecção de linhas

O processo de detecção de linhas usa imagens binárias esqueletizadas como entrada. O processo para geração dessas imagens a partir de uma imagem georreferenciada fornecida é como segue:

1. A imagem RGB fornecida é separada em recortes de tamanho mínimo de 500×500 píxels, que com um GSD de 4 cm/píxel resulta em um área de aproximadamente $20m^2$. Quando um recorte não puder atender a essas dimensões ele é juntado ao recorte anterior, podendo o tamanho máximo de um recorte chegar a 999×999 píxels;

   As imagens são separadas em recortes menores para que possam ser processadas paralelamente e para que em um mesmo recorte as fileiras de cana apareçam com pouca curvatura, assemelhando-se a linhas paralelas. É importante que os segmentos de uma imagem não tenham muita curvatura pois cada dois segmentos sobrepostos detectados são simplificados em um único segmento que represente os dois, o que faria com que os segmentos detectados em um recorte grande o suficiente que contenha fileiras de cana curvadas não tivesse suas linhas de cana detectadas corretamente.

2. Cada recorte é processado pela rede neural, gerando uma imagem binária. Cada imagem binária é esqueletizada e uma detecção de linhas inicial é feita com a transformada de Hough. A mediana da inclinação dessas linhas é calculada e usada para criar um filtro com altura igual à altura do recorte e mesma inclinação das linhas detectadas;

   Apesar dos problemas apresentados anteriormente sobre o uso da transformada de Hough, ela é usada com um limiar de votos elevado e um tamanho mínimo de linha de cerca de 10% da largura de imagem de entrada para encontrar as linhas mais destacadas de um esqueleto a fim de encontrar o ângulo de inclinação mediano das linhas na imagem. O limiar de votos usado começa em 100, mas caso nenhuma linha seja encontrada, vai diminuindo de 10 em 10 até algo ser encontrado. Caso mesmo assim nenhuma linha seja detectada, o comprimento mínimo de uma linha também é diminuído sucessivamente. Por fim, se nenhum linha tiver sido encontrada um erro é lançado.

3. O filtro criado é passado sobre a imagem binária e os píxels sob o filtro são avaliados da seguinte forma: se a quantidade de píxels diferentes de 0 na imagem binária sob o filtro for maior que uma porcentagem do comprimento da linha do filtro, então os píxels sob o filtro recebem o valor 1, caso contrário, recebem o valor 0;

   Este passo tem o objetivo de suavizar as bordas das fileiras de cana na imagem binária removendo as "pontas" das plantas de cana e mantendo seu interior. Considerando que as linhas de cana em uma imagem podem aparecer de diferentes formas, finas, grossas, com falhas no meio etc., é preciso que a aplicação do filtro saiba lidar com essas situações.

   Para que o filtro identifique linhas corretamente mesmo quando existem buracos na linhas, ou seja, falhas, caso o tamanho de um buraco abaixo do filtro seja maior que uma porcentagem do comprimento do filtro, o trecho do filtro é desconsiderado no cálculo do seu comprimento. Para que também sejam identificadas linhas grossas e finas, a linha que cruza o filtro pode ter diferentes larguras, fazendo com que um filtro de largura maior facilite a identificação de linhas mais finas ou falhadas, mas pode acabar conectando duas linhas mais grossas em uma única, enquanto que um filtro com largura menor não vai conseguir detectar linhas mais finas, mas vai detectar com mais precisão linhas mais grossas, como mostra a Figura 3.8.

   Portanto, dois filtros são passados na imagem, com diferentes larguras de linha do filtro e os resultados das duas filtragens são combinados para obter uma filtragem que detecta tanto linhas finas como linhas grossas.

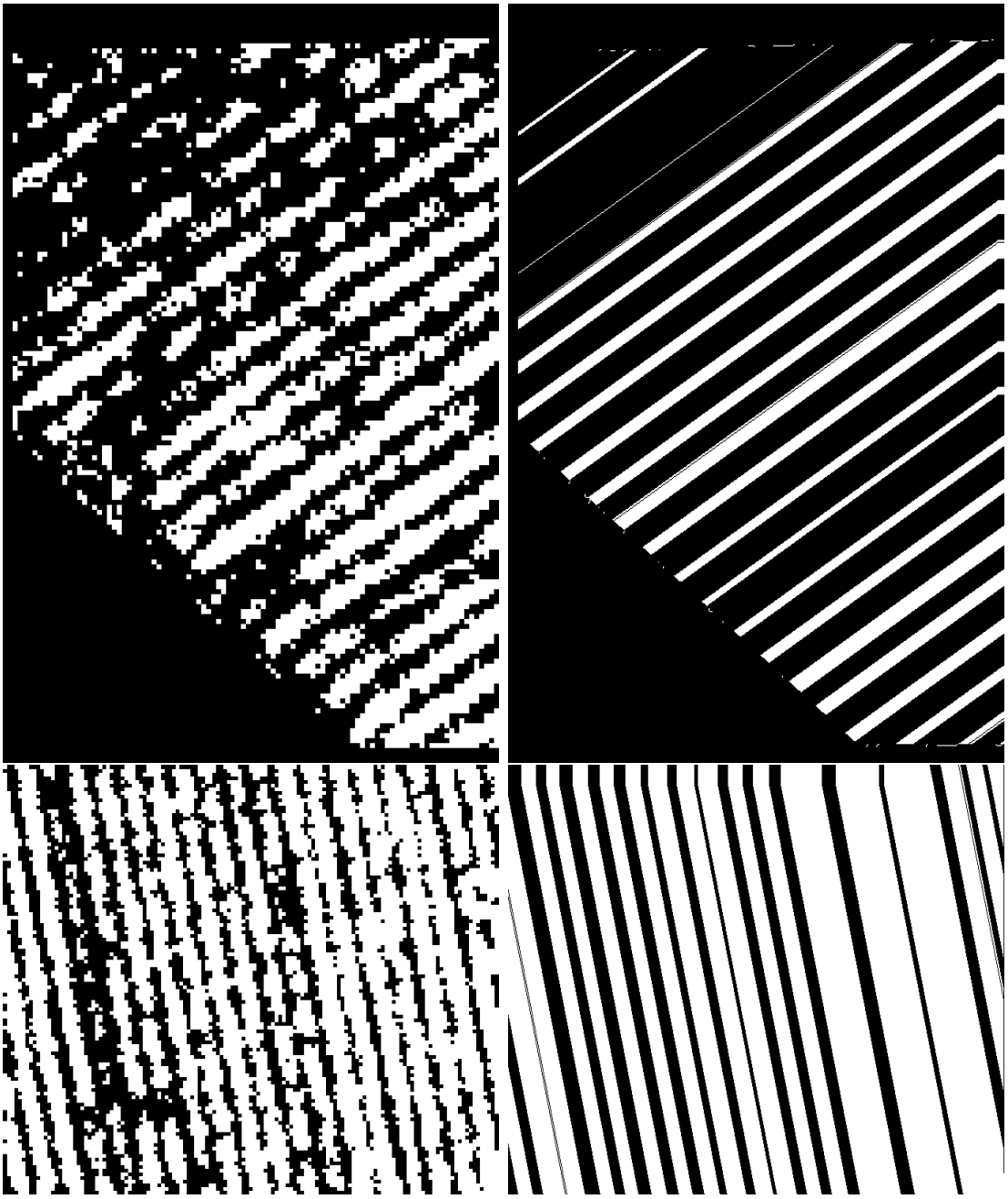

Figura 3.8: Imagem binária com linhas finas (superior esquerdo) filtrada com um filtro de largura pequena falha em detectar linhas finas (superior direito) e imagem binária com linhas grossas (inferior esquerdo) filtrada com um filtro de largura grande conecta linhas distintas em um único objeto (inferior direito)

A imagem filtrada é esqueletizada e operada bit-a-bit com a imagem original com o operador lógico `E`, para recuperar as interrupções nas fileiras de cana da imagem original. A operação morfológica de fechamento é aplicada para remover pequenos buracos no esqueleto, não grandes o suficiente para ofuscar falhas na plantação. A imagem resultante representa as linhas de cana a serem detectadas. A imagem

resultante também é operada bit-a-bit com a imagem filtrada com a operação lógica `OU EXCLUSIVO` e representa as falhas nas linhas de cana.

4. As linhas de cana e as falhas nas linhas são encontradas usando as caixas delimitadoras dos objetos nas imagens esqueletizadas, em que dois segmentos de linha são comparados, cada um vai de um lado da caixa até seu lado paralelo, e o segmento de maior comprimento é selecionado. Esse segmento é adicionado a lista de segmentos encontrados na imagem caso seja maior que um determinado comprimento mínimo e não tenha um ângulo de inclinação tão diferente do ângulo mediano das linhas na imagem.

5. Em seguida, os segmentos detectados para um determinado recorte são agrupados de forma que segmentos que pertençam a uma mesma linha de cana estejam no mesmo grupo. O agrupamento considera que a inclinação dos segmentos é igual para todos, sendo os segmentos de uma linha de cana diferenciados dos segmentos de outra linha pelo deslocamento vertical. Os segmentos são ordenados pelo seu deslocamento do eixo vertical e considerando que as linhas de cana são espaçadas de forma igual, um grupo de segmentos inicial é criado, a lista de segmentos detectados é percorrida e os elementos são inseridos no último grupo criado caso seu deslocamento não se afaste muito do deslocamento médio dos outros segmentos do grupo, caso se afaste o suficiente, um novo grupo é criado e o segmento é inserido nesse grupo. Após o agrupamento, cada grupo é percorrido e os segmentos dentro de um grupo que estiverem muito próximos um do outro são substituídos por um único segmento.

6. Após os grupos de segmentos de cada recorte da imagem serem detectados, os segmentos mais a direita e mais abaixo de um recorte são comparados com os segmentos mais a esquerda do recorte à direita e mais acima do recorte abaixo, respectivamente, e caso estejam muito próximos, um novo segmento é criado conectando-os.

3.2.3 Interface gráfica

A interface gráfica foi desenvolvida usando a biblioteca GTK, uma biblioteca que pode criar aplicações gráficas para os sistemas operacionais mais comuns, com uma Application Programming Interface (API) para Python que permite integrar as outras partes do projeto com facilidade à interface gráfica. A aplicação desenvolvida é simples e tem somente uma funcionalidade: processar uma imagem dada e gerar um arquivo com as coordenadas geográficas das linhas de plantio e das falhas nessas linhas.

Páginas da aplicação

A aplicação possui duas páginas: a página inicial serve para selecionar a imagem a ser processada e a segunda página serve para processar a imagem e salvar o arquivo resultante.

Ao abrir a aplicação, a página inicial é mostrada (Figura 3.9). O primeiro botão com nome "Selecionar imagem" abre um diálogo que permite que o usuário escolha uma imagem. Somente imagens do tipo GeoTIFF devem ser selecionadas. O Segundo botão permite que o usuário escolha um arquivo KML com uma máscara para a imagem selecionada. Caso a máscara selecionada não seja válida, um erro será apresentado. Quando uma imagem tiver sido selecionada, o botão "Avançar" ficará habilitado para que siga com o processamento. Adicionalmente, junto ao botão "Fechar", que fecha a aplicação, tem um botão que apresenta um pequeno texto de ajuda sobre o manuseio da página.

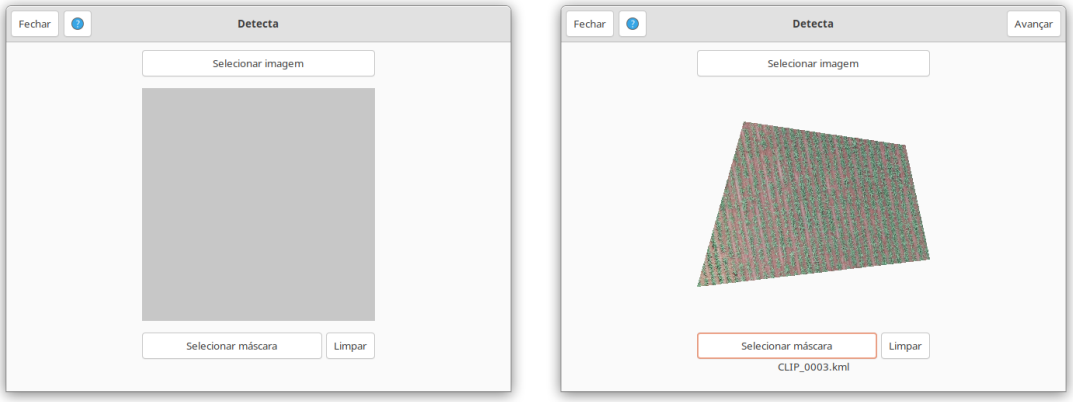

Figura 3.9: Página inicial da aplicação após ser aberta (esquerda) e após ter uma imagem e máscara selecionadas (direita)

Após o botão "Avançar" ser clicado na página inicial, a imagem selecionada é mascarada, caso uma máscara tenha sido selecionada, e a segunda página é exibida (Figura 3.10). Nesta página há apenas o botão "Executar" para ser clicado, que começa a processar a imagem e detectar linhas nela. Ao fim do processamento, um botão "Salvar" é revelado e permite que o usuário salve um arquivo KML com os resultados do último processamento.

Resultados gerados pelo processamento

As coordenadas encontradas para os segmentos de uma imagem são escritas em um único arquivo KML. Arquivos desse formato permitem que diversas camadas de diferentes dados possam ser inseridos, dessa forma, uma camada nomeada "Linhas" é criada com as linhas de plantio detectadas e uma camada nomeada "Falhas" é criada com as linhas de falhas detectadas. As coordenadas são convertidas para o mesmo sistema de coordenadas

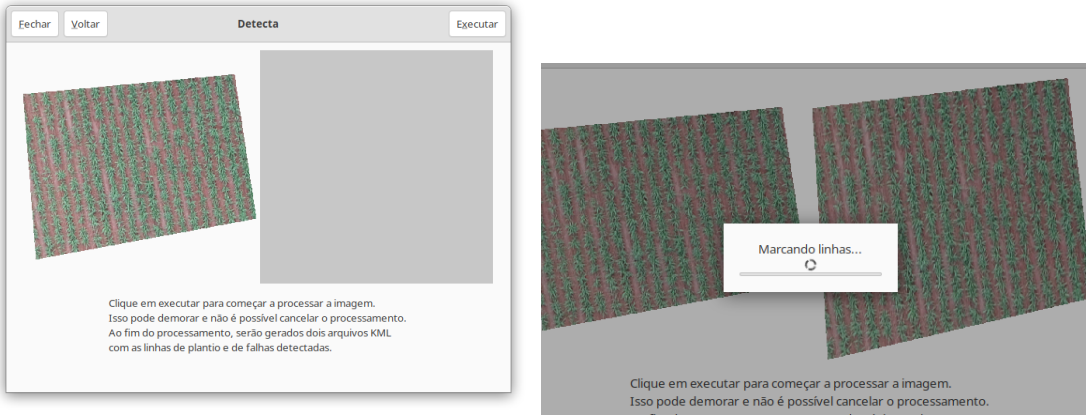

Figura 3.10: Página de detecção de linhas antes de processar a imagem selecionada (esquerda) e durante o processamento (direita)

geográficas da imagem de origem antes de serem salvas. As camadas do arquivo podem ser abertas no QGIS como camadas separadas e podem ser manipuladas independentemente.

No Capítulo 4 os resultados gerados serão avaliados e comparados com as linhas detectadas pelo INFOROW, concluindo este projeto com comentários sobre os pontos positivos e negativos, além de propôr possíveis melhorias.

Capítulo 4

Avaliação

Neste capítulo serão avaliados os resultados da classificação de píxels da rede neural e da qualidade das linhas de plantio e das linhas de falha detectadas e serão comentados possíveis trabalhos futuros, considerando os resultados analisados.

4.1 Avaliação da rede neural

A rede neural desenvolvida consegue diferenciar e classificar adequadamente píxels de planta contra píxels de terra, mesmo em diferentes densidades de plantas (Figuras 4.1 e 4.2), mas não consegue diferenciar um píxel de uma planta de cana de uma árvore no meio da plantação (Figura 4.3) e gera imagens com um pouco de ruído quando há algum tipo de objeto diferente ao redor das bordas desse objeto, como tratores e caminhões (Figura 4.4).

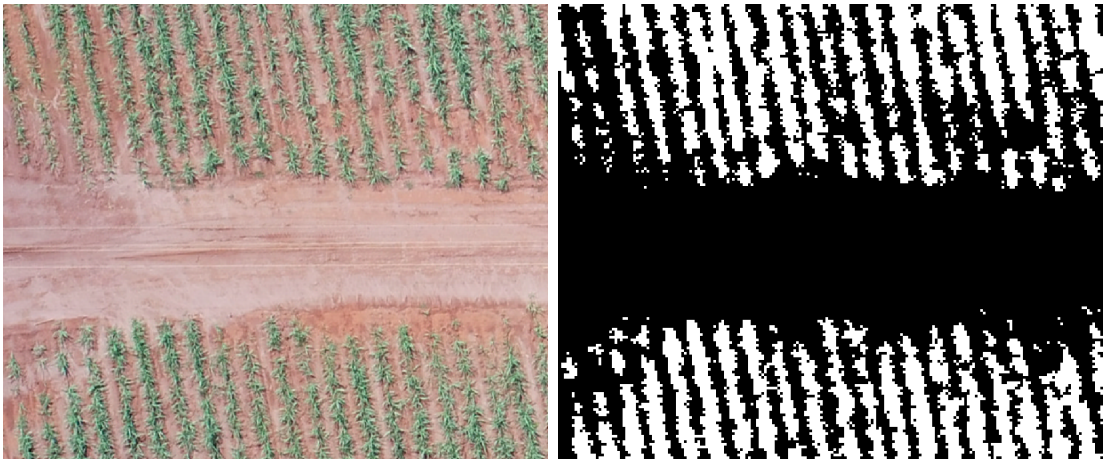

Figura 4.1: Imagem do encontro de uma plantação com uma estrada (esquerda) e a classificação gerada (direita)

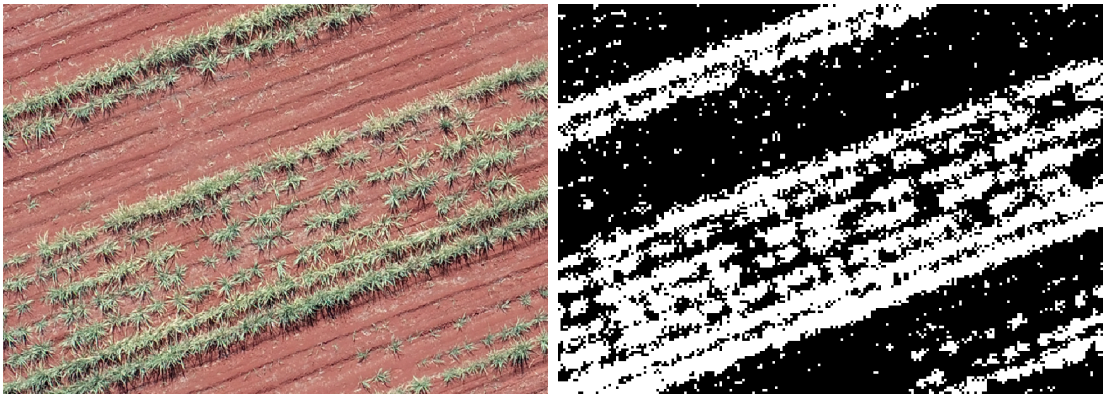

Figura 4.2: Imagem de um plantação com plantas bem escassas (esquerda) e a classificação gerada (direita)

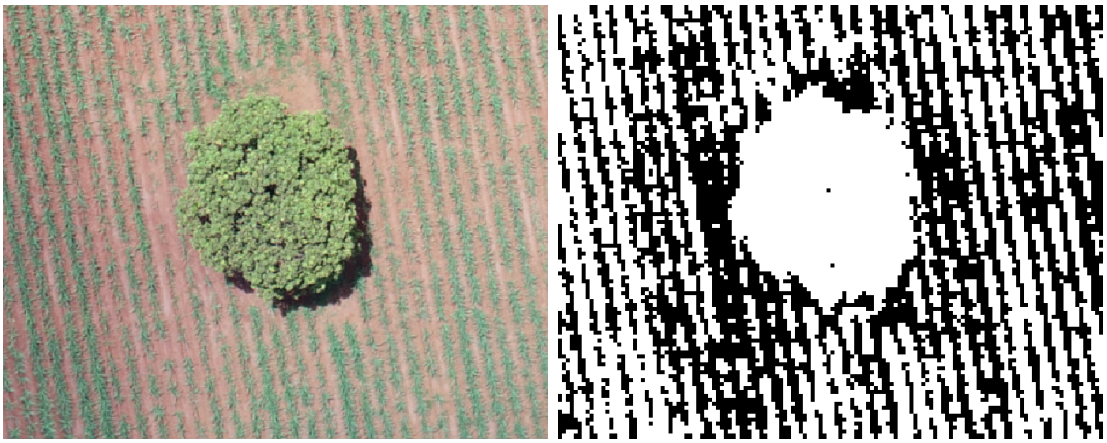

Figura 4.3: Imagem de uma árvore no meio de uma plantação (esquerda) e a classificação gerada (direita)

Figura 4.4: Imagem de máquinas próximas a uma plantação (esquerda) e a classificação gerada (direita)

Esses erros de classificação e artefatos gerados prejudicam a detecção correta de linhas nessas imagens, mas, como pode ser observado na Figura 4.5 e em todos os dados disponíveis, os limites de cada talude são bem definidos, com estradas de terra fazendo uma separação clara de qualquer outra vegetação local, e a ocorrência de árvores no meio das plantações não é muito frequente. Isso permite que a criação de uma máscara sobre as plantações que remova árvores e recorte os limites das plantações de forma que máquinas que estejam ao lado sejam mascaradas resolva o problema de forma simples e sem demandar muito esforço, como apresentado na Figura 4.6. Ainda que haja espaço para melhora na classificação gerada pela rede neural, para este projeto, esses problemas não se mostraram impeditivos ao avanço e à obtenção de bons resultados.

Figura 4.5: Imagem completa de uma fazenda, com diferentes taludes delimitados em azul

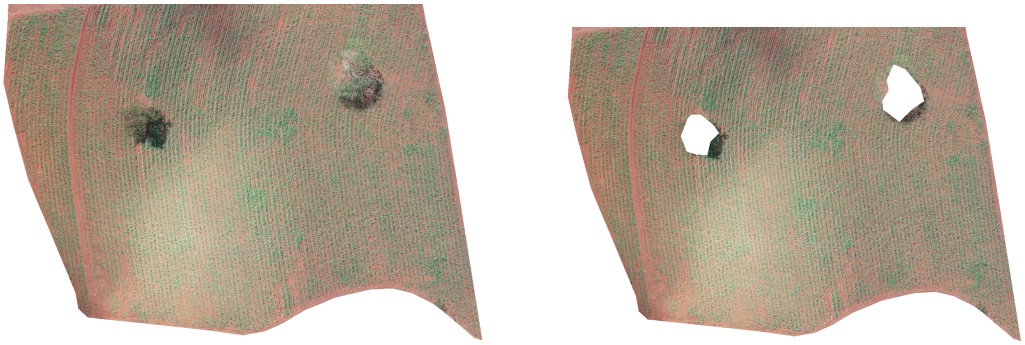

Figura 4.6: Recorte de uma fazenda mascarada pelos seus limites (esquerda) e mascarada com uma máscara modificada para cobrir árvores na plantação (direita)

4.2 Avaliação da detecção de linhas

4.2.1 Linhas piloto

A avaliação da detecção de linhas deve ser feita comparando as linhas geradas pela aplicação com as linhas já detectadas pelo INFOROW mencionadas na Seção 2.2. A primeira diferença notável entre as linhas geradas e as linhas obtidas do INFOROW é a forma como pequenos espaços entre as plantas de uma linha de cana são tratados. As linhas obtidas neste projeto são quebradas nos espaços entre plantas, devido a operação lógica `E` feita no esqueleto filtrado com a imagem binária mostrada na Seção 3.2.2, e as linhas no INFOROW são contíguas, como pode ser comparado na Figura 4.7.

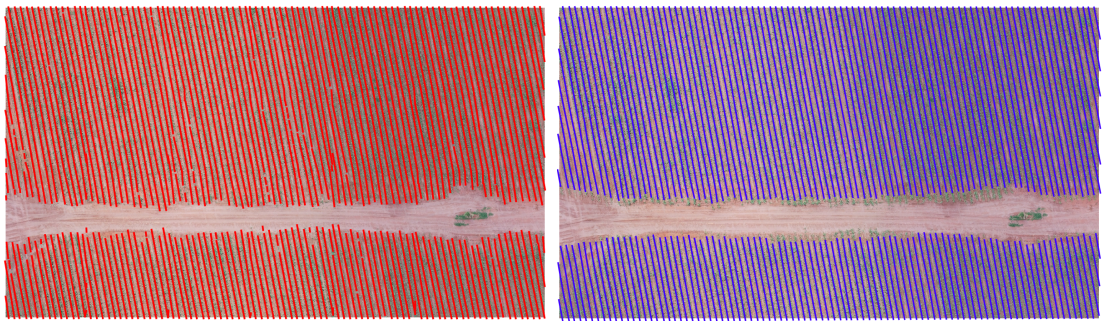

Figura 4.7: Linhas geradas (esquerda) e linhas detectadas pelo INFOROW (direita)

A implementação escolhida neste projeto faz sentido pois serve para encontrar as falhas, mas caso fosse necessário fazer com que as linhas detectadas fossem contíguas como as do INFOROW, a aplicação pode ser alterada para fazê-lo sem maiores problemas, uma vez que um esqueleto das linhas indo de ponta a ponta da imagem já é obtido durante o processamento como mostrado na Subseção 3.1.2. Por mais que essa escolha não esteja disponível na aplicação para o usuário, para fins de comparação, a Figura 4.8 mostra as linhas detectadas dessa forma.

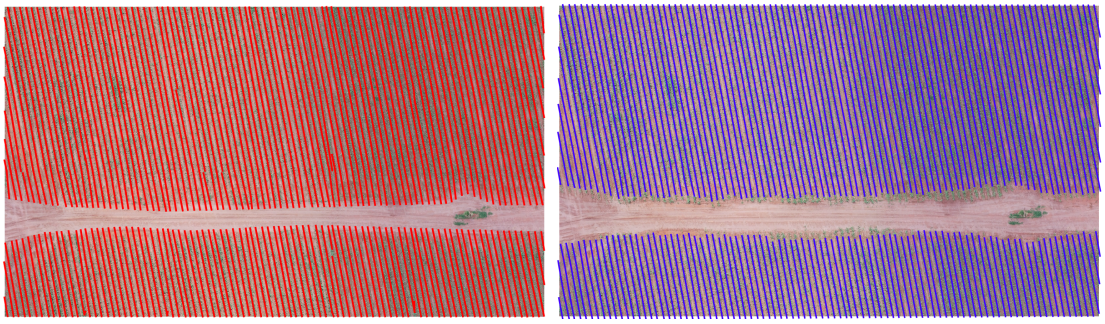

Figura 4.8: Linhas contíguas geradas (esquerda) e linhas detectadas pelo INFOROW (direita)

Nas situações em que árvores aparecem no meio da imagem, como na Figura 4.9, o INFOROW não detecta as linhas de plantio muito precisamente, enquanto que o método proposto com a utilização de uma máscara sobre a árvore mostra-se capaz de superar algumas das dificuldades do INFOROW, que não consegue continuar identificando uma linha que passe muito perto das árvores ou parar de identificar uma linha que não mais existe perto das árvores.

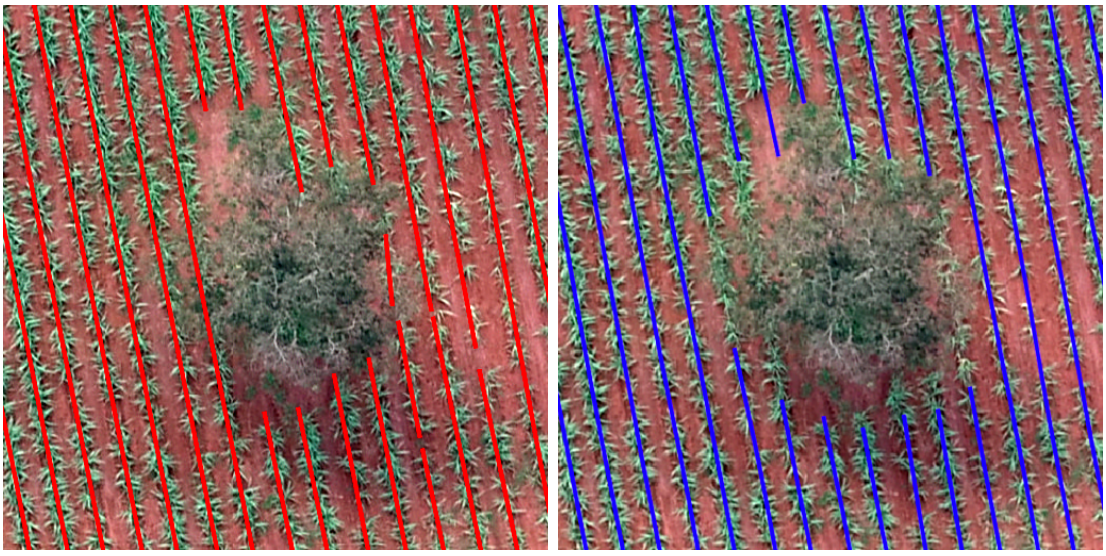

Figura 4.9: Linhas geradas (esquerda) e linhas detectadas pelo INFOROW (direita) em um recorte com uma árvore

Em outros casos em que as plantações têm plantas escassas, muito pequenas ou até mesmo sem plantas, o INFOROW continua identificando linhas, resultando em linhas completamente erradas. Entretanto, nesses casos, a solução proposta também apresenta dificuldades em encontrar as linhas de plantas nas imagens binárias, como mostram os resultados nas Figura 4.10 e 4.11. Apesar do INFOROW marcar linhas onde sequer existem plantas, as marcações sobre as plantas continuam corretas e a exclusão manual dessas linhas erradas é muito mais simples do que corrigir as linhas detectadas com este projeto.

As imagens intermediárias da Figura 4.11 indicam que o ruído está fazendo com que a imagem fique mais difícil de ser filtrada para encontrar linhas. Sendo esse o caso, no processamento de um recorte maior o ruído deve impactar menos a aplicação do filtro. Na Figura 4.12 é mostrado o processamento de uma área com o mesmo tipo de vegetação e escassez mostrados na Figura 4.10 mas com tamanho de cerca 70×50 metros, ou 1836×1311 píxels com GSD de 4cm/píxel, que obteve resultados melhores que os obtidos usando recortes de 20×20 metros. Entretanto, como discutido nos capítulo anteriores, aumentar o tamanho da área processada pode fazer com que a detecção de linhas falhe caso as linhas de cana apresentem uma curvatura. Mas como forma de solução para esse

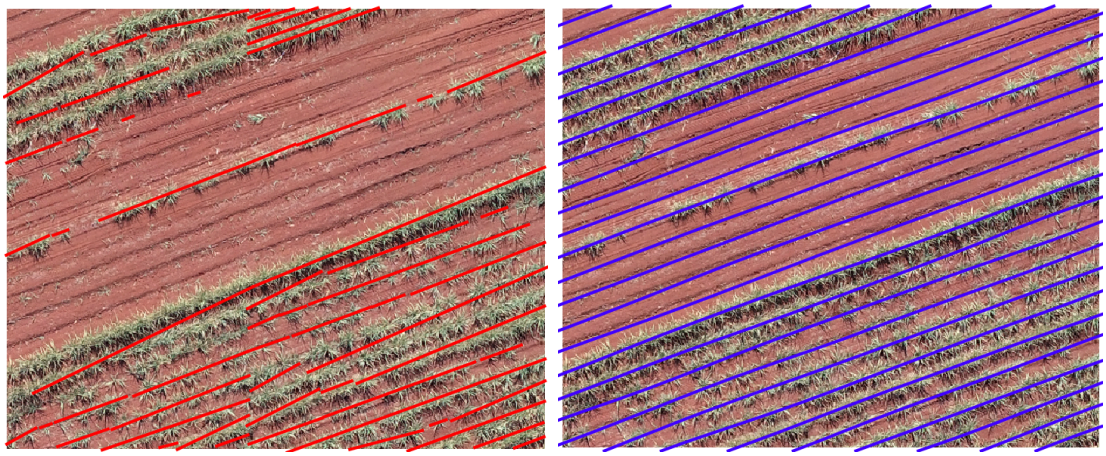

Figura 4.10: Linhas geradas (esquerda) e linhas detectadas pelo INFOROW (direita) em um recorte com plantas escassas

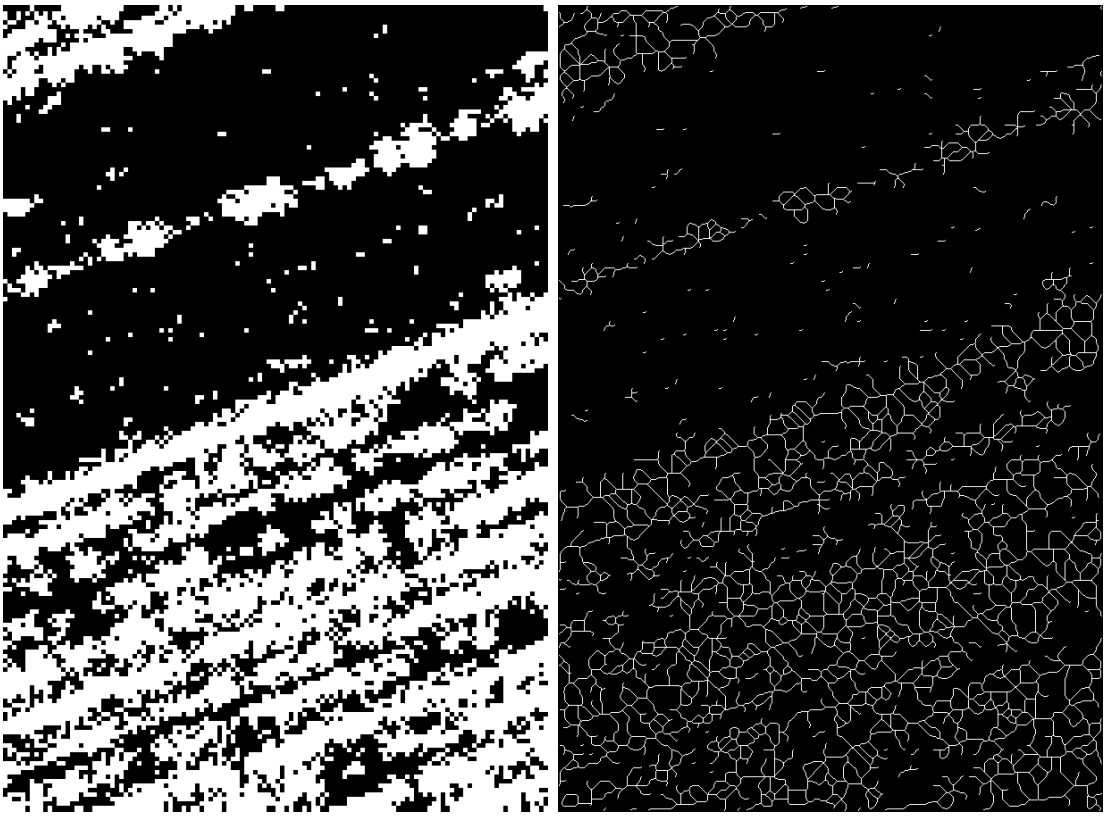

Figura 4.11: A imagem binária predita (esquerda) resulta em um esqueleto inicial com muito ruído (direita)

problema, o tamanho da área processada poderia ser configurável a partir da aplicação gráfica, permitindo que caso um usuário se depare com uma situação como essa possa avaliar se o aumento da área de processamento seria prejudicial ou não aos resultados.

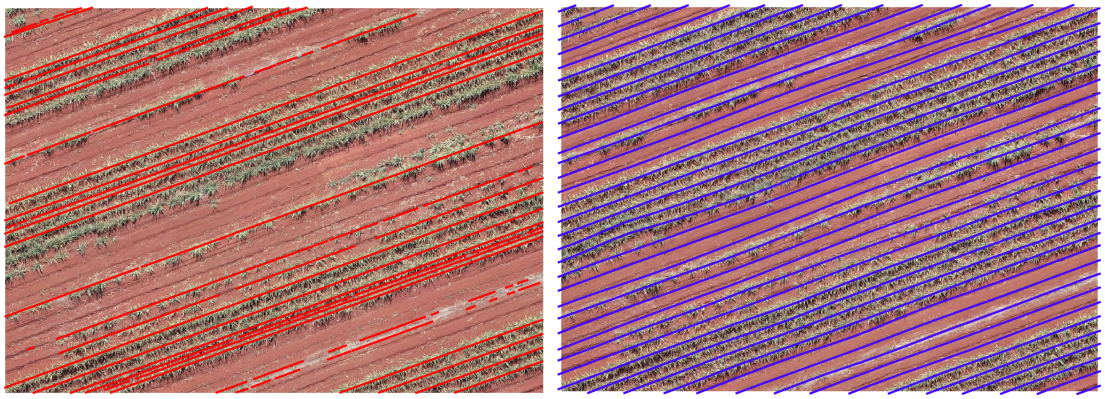

Figura 4.12: Linhas geradas (esquerda) e linhas detectadas pelo INFOROW (direita) em um recorte de área de cerca $70 \times 50m$

4.2.2 Linhas de falha

A comparação das falhas não é tão simples, pois apesar de ser estabelecido que uma interrupção nas linhas de cana de mais de meio metro pode ser considerada uma falha [8], sem a opinião de um profissional não é tão simples determinar quando uma interrupção começa e quando ela termina. A Figura 4.13 mostra algumas falhas detectadas pelo INFOROW e pela aplicação desenvolvida, na qual é fácil ver a dificuldade de julgar quais linhas estão mais certas ou mais erradas.

Ao comparar as falhas detectadas pelo INFOROW e pela aplicação em outros locais, alguns padrões podem ser observados. O INFOROW detecta consistentemente mais falhas, mas também tende a marcar mais possíveis falso-positivos, isto é, falhas visualmente ambíguas (Figura 4.14). A solução desenvolvida neste projeto julga que "fins de linhas" como falhas, algo que o INFOROW raramente faz (Figura 4.15). O INFOROW também julga linhas de cana inexistentes como falhas, enquanto que a aplicação desenvolvida sequer reconhece essas "linhas" e, portanto, não encontra falhas nelas (Figura 4.16). O INFOROW sendo mais agressivo na detecção de falhas também detecta mais falhas reais, que passam despercebidas na detecção do programa desenvolvido, como na Figura 4.17, que mostra diversas falhas detectadas nas linhas pelo INFOROW mas a aplicação desenvolvida encontra somente duas falhas.

Como o INFOROW foi utilizado como base para este trabalho, é normal que se espere que as falhas detectadas pela aplicação se assemelhem às falhas detectadas pelo INFOROW, entretanto é difícil dizer o que é que faz com o que o INFOROW detecte uma falha em uma imagem para poder traduzir a mesma definição de uma falha para a solução proposta neste trabalho. Isso torna fazer uma avaliação objetiva e quantitativa dos resultados complicada, restando apenas uma avaliação qualitativa dos dados obtidos.

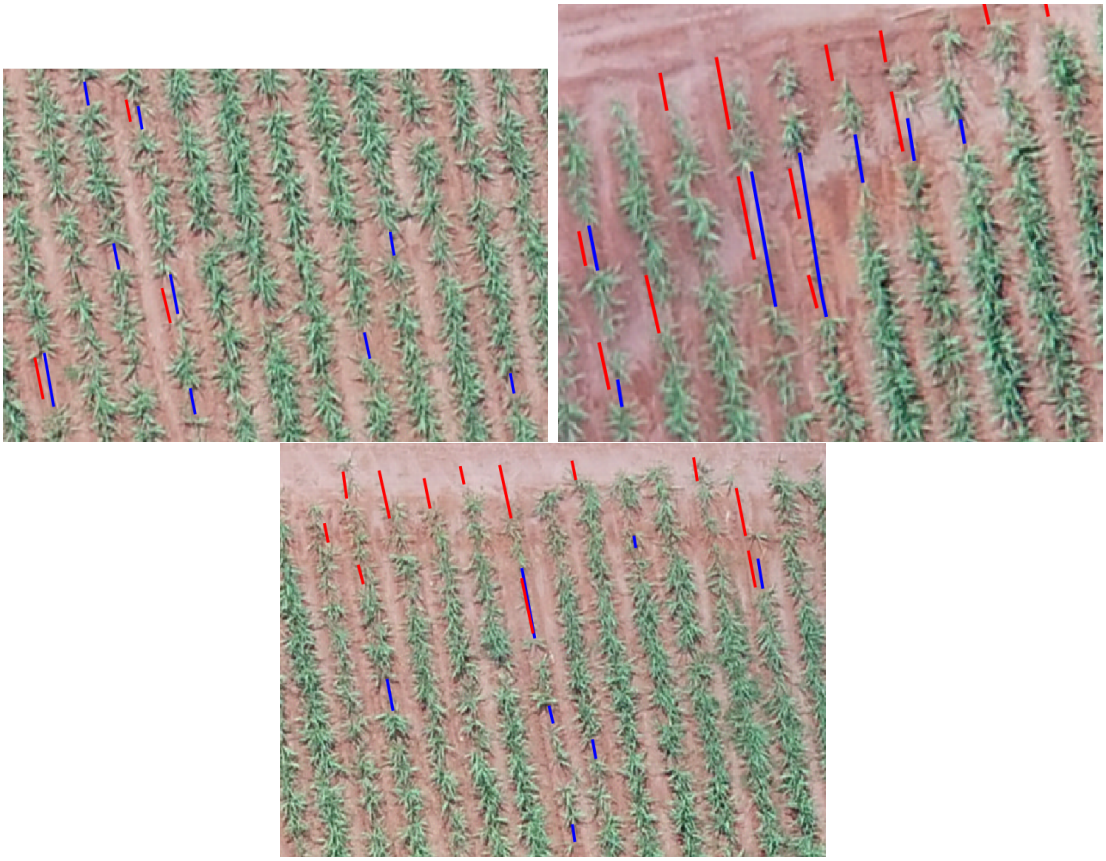

Figura 4.13: Falhas detectadas pelo INFOROW (azul) e pela aplicação (vermelho)

Avaliando visualmente os resultados obtidos neste trabalho com os resultados do INFOROW, nota-se uma clara diferença no entendimento do que é uma falha para as duas aplicações, entretanto pode-se observar que ambas tem falhas e acertos não presentes na outra, como foi apresentado anteriormente.

4.3 Conclusão

O objetivo estabelecido para este projeto foi o de usar parte dos dados gerados pelo INFOROW e substituir parte do seu processamento dos dados para gerar resultados melhores para a detecção de linhas e de falhas nos taludes de cana, tanto na qualidade dos dados, eliminando marcações de linhas erradas, como na facilidade de aplicação do processo de detecção de linhas, diminuindo o peso do trabalho do usuário ao automatizar completamente a detecção de linhas.

A abordagem selecionada para solucionar o problema resultou em uma aplicação que não exige nem permite muito ajuste pelo usuário, esse podendo apenas criar máscaras para recortar a área de processamento de uma imagem. Isso se mostrou benéfico ao proporcionar um processo completamente automático de processamento das imagens ge-

Figura 4.14: Falhas visualmente ambíguas detectadas pelo INFOROW

rando arquivos em formatos compatíveis com os gerados pelo INFOROW, possibilitando que as outras etapas do trabalho possam continuar as mesmas sem qualquer necessidade de adaptação. Entretanto, também teve suas consequências, pois a falta de ajustes pelo usuário impossibilita-o de adaptar o processamento das imagens em uma situação complicada como no caso apresentado na Subseção 4.2.1 sobre o processamento de uma região com poucas plantas, em que a aplicação desenvolvida não conseguiu gerar linhas precisas com o tamanho padrão do recorte utilizado.

Mas na maiorias dos casos, a solução desenvolvida foi capaz de detectar linhas tão precisas quanto as detectadas pelo INFOROW ou até mesmo identificar algumas linhas que não foram encontradas pelo INFOROW perto de árvores no meio dos taludes de cana.

Quanto a detecção de falhas, não é possível dizer que os resultados deste trabalho podem substituir os resultados do INFOROW sem alguma consequência, pois as falhas

Figura 4.15: Falhas detectadas pelo nos fins das linhas pela aplicação (vermelho) e falhas detectadas pelo INFOROW (azul)

detectadas diferem em vários aspectos, sendo necessário um melhor entendimento da interpretação de uma falha pelo INFOROW para propor uma melhora e adaptar a maneira como falhas são detectadas na abordagem utilizada. Um dos aspectos que dificulta saber qual o impacto da diferença das falhas detectadas no resultado final da análise da situação dos taludes é a falta de acesso ao próprio INFOROW. Uma vez que as falhas detectadas são válidas e compatíveis com o INFOROW, finalizar o processamento das falhas detectadas neste projeto pelo INFOROW permitiria ver o impacto dessas diferenças nas políticas de replantio apresentadas por ele.

Considerando todos os pontos abordados, este trabalho no estado que foi finalizado apresenta resultados interessantes, equiparáveis e até melhores que os dados originais em alguns pontos, mas que devido a natureza dos dados e o contexto em que este trabalho foi desenvolvido, são de difícil comparação.

4.4 Trabalhos Futuros

A fim de cumprir com o objetivo proposto neste trabalho, uma aplicação com um objetivo bem específico foi desenvolvida, o de detectar linhas de cana e linhas de falha de forma automatizada, o que não oferece muitas opções de ajuste do processamento dos dados ao usuário, mesmo que em caráter opcional. Assim, a fim de melhorar a aplicação para que

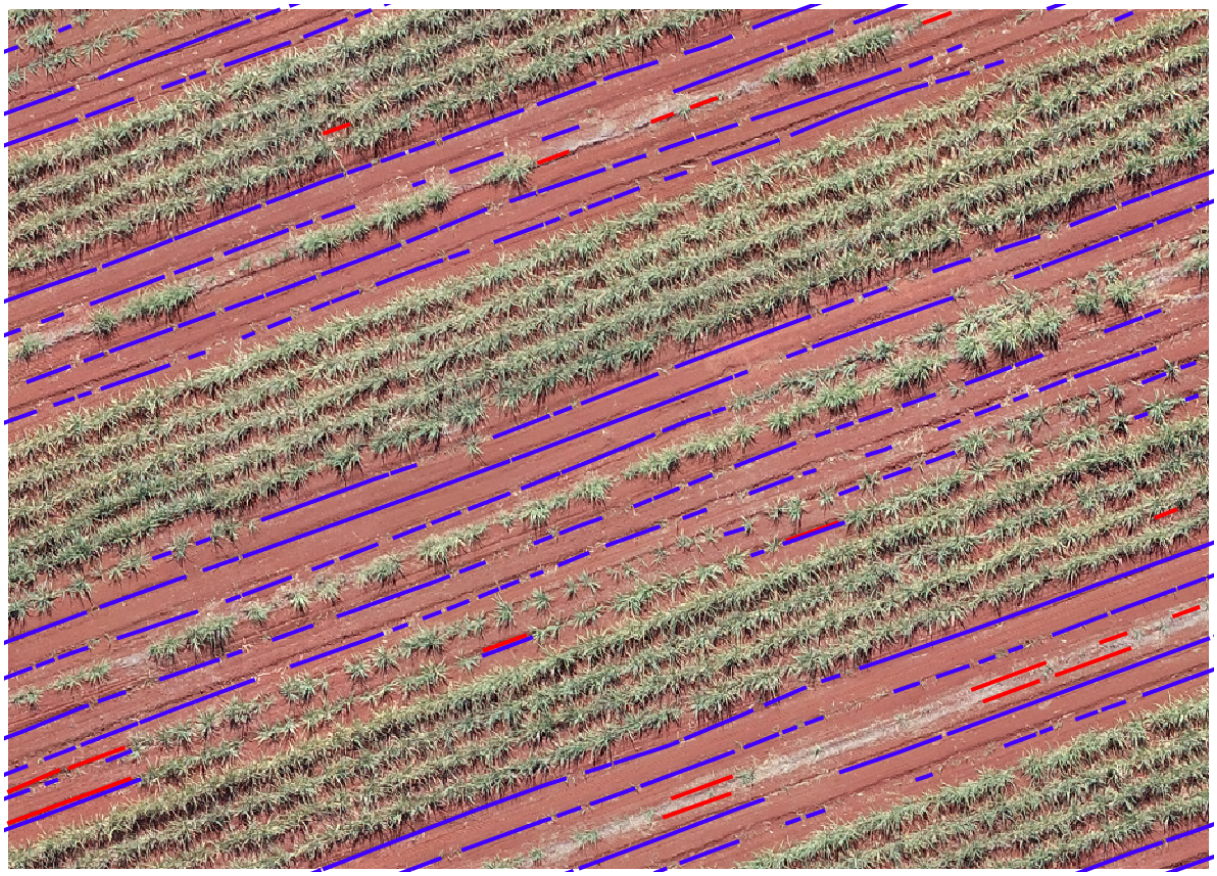

Figura 4.16: Falhas detectadas pelo INFOROW (azul) em linhas de cana completamente falhas e falhas detectadas pela aplicação (vermelho)

possa se adaptar aos diferentes cenários aos quais pode ser submetida, um estudo sobre como a variação de alguns valores estabelecidos como constantes nesse projeto, como o tamanho da área dos recortes processados ou o tamanho mínimo de um segmento de linha de cana, podem afetar seu desempenho poderia ser realizado.

Outro ponto com espaço para melhora é o modelo de rede neural utilizado, que neste trabalho não consegue diferenciar com clareza plantas de cana de outros tipo de planta e árvores ou que gera ruídos perto das bordas de objetos diversos, como carros e tratores. O entendimento do que uma falha é de acordo com o INFOROW também pode ser estudado mais a fundo, o que pode ser usado para melhorar a forma como são detectadas falhas para que se assemelhem mais com as falhas detectadas pelo INFOROW mas eliminando marcações errôneas.

Não ter acesso ao INFOROW também se mostrou um problema durante a avaliação dos resultados. A definição de métricas para teste que não dependam do INFOROW, como a IOU dos resultados gerados com resultados marcados manualmente, poderia permitir uma avaliação mais objetiva dos resultados. A técnica IOU também poderia ser usada com os resultados do INFOROW diretamente contra os resultados gerados ou contra mar-

Figura 4.17: Falhas detectadas pelo INFOROW (azul) ignoradas pela detecção da aplicação desenvolvida (vermelho)

cações feitas manualmente para que possa ser estimado mais precisamente a semelhança e qualidade dos resultados.

Por fim, com o objetivo de integrar as soluções desenvolvidas com ferramentas populares no manuseio de imagens georreferenciadas, a aplicação gráfica poderia ser desenvolvida no formato de uma extensão para o QGIS, que integra facilmente novas funcionalidades desenvolvidas em Python, que facilitaria o uso e adoção do modelo construído.

Referências

[1] Mendonça, Fernando Nicolau: *Detecção de linhas de plantio da cana de açúcar por meio de veículo aéreo não tripulado*. Tese de Mestrado, Universidade Estadual Paulista (UNESP), 2019. 1

[2] Chapman, Pete, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer e Rudiger Wirth: *Crisp-dm 1.0 step-by-step data mining guide*. Relatório Técnico, The CRISP-DM consortium, August 2000. `https://maestria-datamining-2010.googlecode.com/svn-history/r282/trunk/dmct-teorica/tp1/CRISPWP-0800.pdf`. 2

[3] Basso, Maik e Edison Pignaton de Freitas: *A uav guidance system using crop row detection and line follower algorithms*. Journal of Intelligent & Robotic Systems, 97:17, março 2019. 18

[4] Sibaruddin, H I, H Z M Shafri, B Pradhan e N A Haron: *Comparison of pixel-based and object-based image classification techniques in extracting information from UAV imagery data*. 169:012098, jul 2018. `https://doi.org/10.1088/1755-1315/169/1/012098`. 23, 31

[5] Rapinel, Sébastien e Laurence Hubert-Moy: *One-class classification of natural vegetation using remote sensing: A review*. Remote Sensing, 13(10), 2021, ISSN 2072-4292. `https://www.mdpi.com/2072-4292/13/10/1892`. 23

[6] Hassanein, M., M. Khedr e N. El-Sheimy: *Crop row detection procedure using low-cost uav imagery system*. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-2/W13:349–356, 2019. `https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2-W13/349/2019/`. 23

[7] Sabo, Kristian, Danijel Grahovac e Rudolf Scitovski: *Incremental method for multiple line detection problem — iterative reweighted approach*. Mathematics and Computers in Simulation, 178:588–602, 2020, ISSN 0378-4754. `https://www.sciencedirect.com/science/article/pii/S0378475420302421`. 23

[8] STOLF, Rubismar: *Metodologia de avaliação de falhas nas linhas de cana-de-açúcar*. Stab, Piracicaba, 4(6):22–36, 1986. 23, 45

[9] Rabab, Saba, Pieter Badenhorst, Yi Ping Chen e Hans D. Daetwyler: *A template-free machine vision-based crop row detection algorithm*. Precision Agriculture, 22, fevereiro 2021. 31

Anexo I

Operação do INFOROW

Serão apresentados nas seções seguintes: IR_Prepare, na Subseção .1, VegID, na Subseção .2, SOMO Validador, na Subseção .3, Row Finder, na Subseção .4, e IR_Deliver, na Subseção .5.

.1 IR_Prepare

O IR_Prepare é o primeiro software a ser usado após o recebimento das imagens capturadas pelo VANT já montadas em formato de fazendas completas. O objetivo do IR_Prepare é preparar a imagem para o processamento, isto é, redimensioná-la e remover áreas que não são de interesse. Ao final do seu uso, o mosaico da fazenda estará redimensionado de acordo com o tamanho do pixel desejado assim como deverá conter somente os polígonos das áreas de interesse, isto é, apenas as áreas correspondentes aos talhões da fazenda permanecerão na imagem.

O seu uso é descrito brevemente no manual do usuário do INFOROW e é composto das seguintes etapas:

1. Adicionar imagens em formato Tagged Image File (TIF) ou Enhanced Compression Wavelet (ECW) e marcações dos limites de recorte (*boundary* de recorte) em formato Keyhole Markup Language (KML) ou formato ESRI Shapefile (SHP) (que é convertido para KML pelo IR_Prepare);

2. Para cada imagem, configurar o tamanho do pixel (indicado 10cm/px) e selecionar o tipo de amostragem mais adequado para o mosaico entre as opções disponíveis;

    - Máximo RGB: para imagens nítidas e que se pode fazer distinção entre linha e entre linha;
    - Máximo de Vegetação: para imagens com plantas muito pequenas;

- Mínimo de Vegetação: para imagens com plantas muito altas ou se não pode distinguir entre linha e entre linha.

3. Processar as imagens.

Como resultado, um arquivo Binary Interleaved (BIL), um formato para imagens *raster* usado para representar superfícies e elevações em terrenos, e um arquivo High Dynamic Range (HDR), um formato que descreve o leiaute e a formatação da imagem *raster* e normalmente acompanha um arquivo BIL, são gerados para serem usados no VegID.

.2 VegID

O VegID é usado sobre os arquivos de saída do IR_Prepare (BIL e HDR). Seu objetivo é gerar uma nuvem de pontos sobre a vegetação da imagem passada. A nuvem de pontos pode ser visualizada com o programa de código aberto QGIS mesmo durante a geração da nuvem de pontos.

O seu uso é descrito no manual do usuário do INFOROW e é composto das seguintes etapas:

1. Adicionar um arquivo HDR;
2. Indicar a zona Universal Transversa de Mercator (UTM) da imagem *raster* (visível no QGIS);
3. Selecionar Índice de Vegetação (IV) (um ou mais);
4. Executar.

Como resultado, um arquivo Comma Separeted Values (CSV) e um arquivo EPC para cada IV selecionado contendo a nuvem de pontos gerados sobre a plantas na imagem.

.3 SOMO Validador

O SOMO Validador é utilizado para converter a nuvem de pontos no formato EPC para uma nuvem de pontos no formato NPC. Apesar de ser parte do INFOROW, é distribuído como um software hospedado no serviço de computação em nuvem *Azure*, precisando de usuário e senha para ser acessado.

O seu uso é descrito no manual do usuário do INFOROW e é composto das seguintes etapas:

1. Acessar o site *frontend* para o SOMO Validador;

2. Enviar arquivos EPC;
3. Converter arquivos EPC enviados;
4. Baixar arquivos NPC gerados.

Como resultado, um arquivo NPC é gerado com uma nuvem de pontos.

.4 Row Finder

O Row Finder tem como objetivo identificar e marcar as linhas de cana de uma plantação. O Row Finder utiliza um arquivo com uma nuvem de pontos NPC e um arquivo com os limites da área de plantio KML. A identificação das linhas de cana é feita sobre a nuvem de pontos, ligando pontos de um limite da imagem a outro, formando linhas. O Row Finder procura por uma linha por vez e tenta identificar segmentos da linha a partir de segmentos já marcados. O Row Finder tenta identificar outras linhas paralelas a linhas já marcadas.

O seu uso é descrito no manual do usuário do INFOROW e é composto das seguintes etapas:

1. Carregue a nuvem de pontos e os limites da área de plantio;
2. Clique em “Procurar” para começar a procurar por segmentos de linha;
3. Ligue pontos da nuvem de pontos que formam um segmento da linha clicando neles quando o Row Finder não conseguir identificar mais segmentos na linha;
4. Quando uma linha estiver completamente marcada clique em "Procurar"novamente para que outra linha seja procurada (rastreios ao longo da linha atual indicam a direção em que as próximas linhas vão ser procuradas);
5. Ao final, salve um arquivo KML com as linhas encontradas.

O Row Finder ainda conta com algumas funcionalidades que podem ajudar durante a marcação das linhas, como desenhar linhas manualmente, reverter segmentos encontrados, inverter a direção de busca de novas linhas, interromper uma linha antes de chegar aos limites da área etc. Para a identificação de segmentos, algumas configurações devem ser ajustadas, como a distância entre linhas da plantação, comprimento padrão para os segmentos encontrados, variação do ângulo de procura por novos segmentos etc. Como resultado, um arquivo KML com as linhas que foram marcadas é gerado.

.5 IR_Deliver

O IR_Deliver utiliza os arquivos de nuvem de pontos NPC, da imagem processada BIL e as marcações de linhas geradas pelo Row Finder KML. Seu objetivo é extrair informações sobre a plantação. O IR_Deliver apresenta os locais de falha no plantio, a porcentagem de falhas, a distância entre linhas em cada local, mostrando entre linhas grandes ou pequenas demais, e logísticas de plantio baseado nos dados obtidos.

O seu uso é descrito no manual do usuário do INFOROW e é composto das seguintes etapas:

1. Carregue os arquivos NPC, BIL e KML necessários;
2. Ajuste a distância entre linhas (o IR_Deliver tenta identificar automaticamente);
3. Clique em "Extrair dados das linhas" para gerar arquivos KML de linhas suavizadas, de pontos de paralelismo e de linhas de transbordo;
4. Clique em "Extrair dados da imagem" para iniciar uma nova aba para procurar por falhas na plantação;
5. Na nova aba, selecione o IV adequado à plantação, dentre os cinco disponíveis;
6. Escolha entre identificar falhas usando um critério local ou global (usado em condições onde a área é homogênea);
    - Para o critério local: ajuste a porcentagem máxima de biomassa local para a classificação de falhas;
    - Para o critério global: clique em "Sugerir IV" para que o IR_Deliver sugira um IV e configure o valor máximo do IV para a classificação de falhas.
7. Clique na aba "Definição";
8. Configure o comprimento mínimo de uma falha, a distância das extremidades da linha para que não sejam identificadas falhas e escolha entre estender falhas sob a folhagem ou não;
9. Clique em "Aplicar" e é possível visualizar as falhas encontradas e a porcentagem de falhas antes de salvar os resultados;
10. Salve os resultados e uma aba de logística de replantio aparecerá;
11. Configure informações sobre o replantio desejado, como a distância entre as plantas existentes e mudas novas, a distância entre mudas novas, a porcentagem mínima de falhas na linha para o replantio, etc;

12. Clique em “Identificar” para ver informações sobre o impacto na produtividade pelo replantio e o tempo de replantio;

13. Arquivos geográficos de pontos e linhas de replantio são gerados.

Como resultado o IR_Deliver gera arquivos KML de linhas suavizadas, de pontos de paralelismo, de linhas de transbordo, de marcação de falhas, de pontos de replantio e de linhas de replantio. Os arquivos podem ser visualizados sobre a imagem usando o QGIS.

Anexo A

Operação do QGIS

Para o uso da aplicação gráfica desenvolvida, uma imagem georreferenciada deve ser fornecida opcionalmente com uma máscara sobre ela. A criação e edição dessas máscaras foge do escopo deste projeto, mas como elas são utilizadas na aplicação desenvolvida, vale apresentar uma maneira de criá-las e editá-las. Uma maneira conveniente de fazer isso é com o uso do QGIS, que oferece uma interface de simples operação. Assim, seguem exemplos simples de como manipular esse tipo de arquivo no QGIS:

A primeira etapa para usar o QGIS é abrir imagens para iniciar um novo projeto:

- No navegador de arquivos à esquerda, navegue até o arquivo desejado e clique duas vezes nele para abri-lo como uma camada no novo projeto. Caso o arquivo selecionado seja uma imagem válida, ela aparecerá na janela principal do programa, caso seja um arquivo de vetores, como um KML, os vetores serão desenhados na janela principal do programa como pontos, linhas ou polígonos, dependendo do tipo dos dados do arquivo.

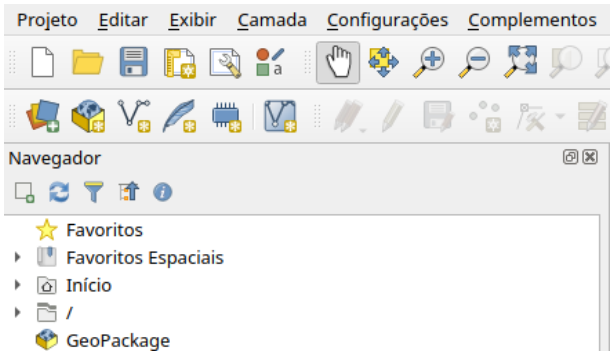

Figura A.1: Navegador de arquivos do QGIS

- No gerenciador de camadas, cada camada pode ser arrastada para cima e para baixo de outras camadas, sendo importante pôr as camadas de imagens abaixo das

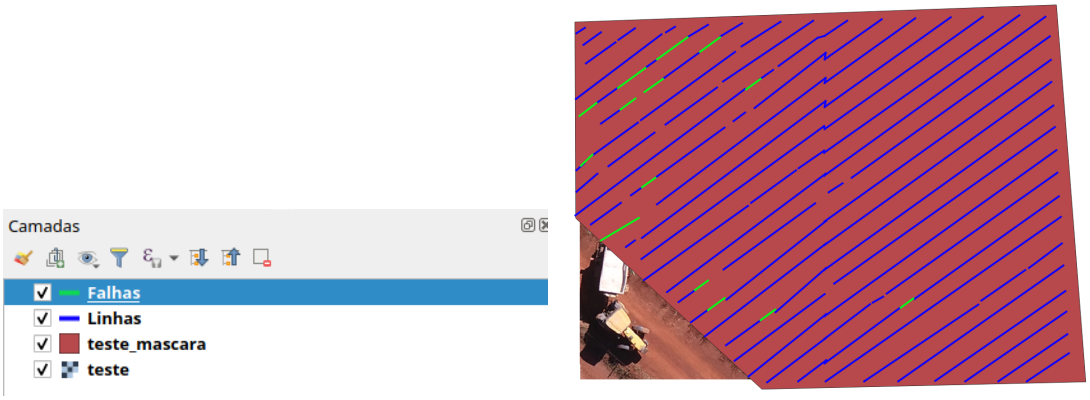

Figura A.2: Gerenciador de camadas (esquerda) e janela principal do QGIS (direita) com uma imagem aberta, um polígono e dois conjuntos de linhas desenhados

camadas de máscara sobre elas para que as máscaras possam ser visualizadas na janela principal do programa.

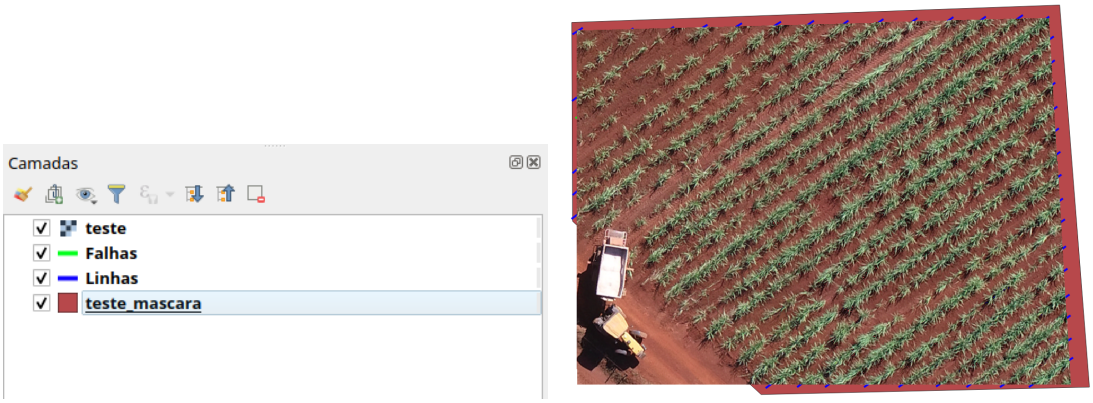

Figura A.3: Linhas e polígonos sendo ocultados pela imagem aberta por causa da ordem errada de camadas

- Clicando com o botão direito na camada de imagem ou de vetores, um menu de opções aparece mostrando algumas operações e propriedades da camada.

Com uma imagem aberta em um novo projeto no QGIS, segue como criar e manipular uma nova camada de forma no projeto e como exportá-la para o formato KML:

- Criação de máscaras sobre uma imagem (polígonos)
  - Clique em “Nova camada Shapefile” na barra de ferramentas (Figura A.5);
  - Clique em “Adicionar Polígono” na barra de ferramentas (Figura A.6);
  - Clique com o botão esquerdo na imagem carregada para adicionar vértices ao polígono (Figura A.7);

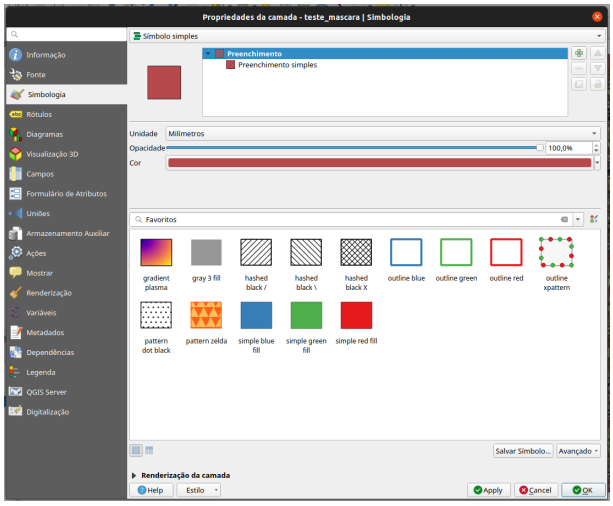

Figura A.4: Janela de propriedade de uma camada

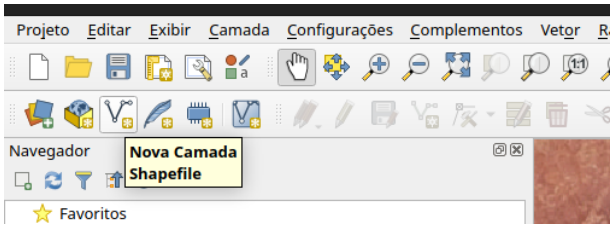

Figura A.5: Criar nova camada

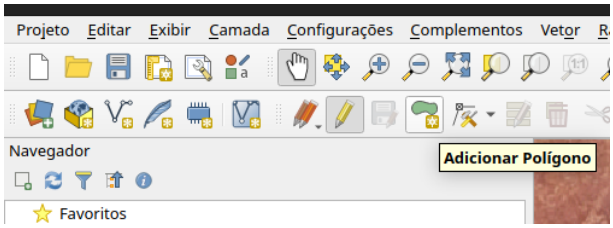

Figura A.6: Adicionar polígono

Quando marcar todos os vértices necessários, clique com o botão direito para salvar o polígono e em "Alternar edição" na barra de ferramentas para finalizar a edição dessa camada.

- Edição da máscara criada

  Clique em "Alternar edição" na barra de ferramentas para entrar no modo de edição da camada selecionada e depois em "Ferramenta Vértice" para manipular os vértices do polígono criado. Para editar um vértice, clique duas vezes em cima de um vértice e, sem soltar o botão do mouse, arraste o vértice para a nova posição (Figura A.8). Para sair do modo de edição clique em "Alternar edição" novamente.

- Aplicação da máscara sobre uma camada de imagem

Figura A.7: Adicionar vértices

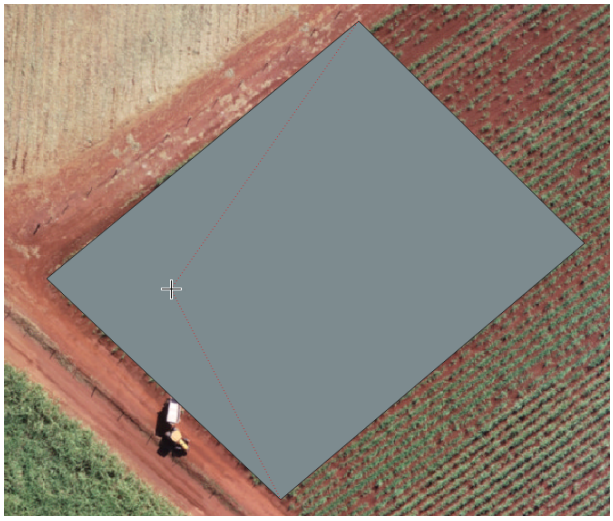

Figura A.8: Edição de vértices de uma camada de polígono

No menu da aplicação, navegue pelo caminho “Raster” → “Extrair” → “Recortar raster pela camada de máscara...” e abrirá um diálogo para configurar os parâmetros do recorte (Figura A.9). Selecione a camada de imagem que deve ser recortada e a camada de máscara que descreve o corte. Os outros parâmetros são opcionais e devem ser configurados dependendo da necessidade de cada caso. Por exemplo, a configuração “Criar uma banda alfa de saída” pode ser marcada para que as partes da imagem que forem mascaradas sejam transparentes na imagem resultante.

- Exportação da camada de máscara e de imagem mascarada

  Para salvar uma camada, clique com o botão direito na camada criada e depois em “Exportar” → “Salvar Como...” ou “Salvar Feições Como...” e aparecerá um diálogo para salvar a camada (Figura A.10).

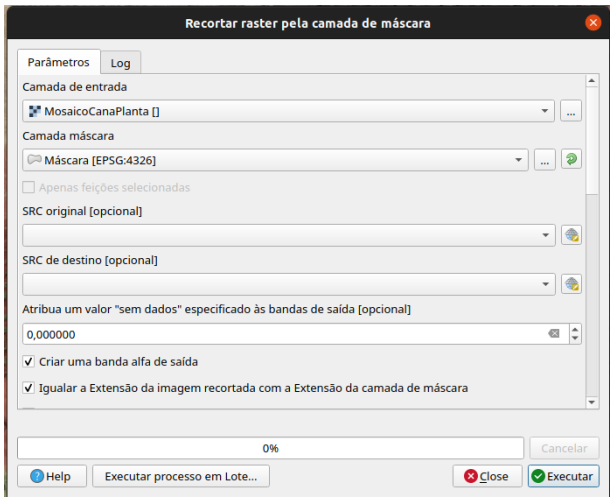

Figura A.9: Diálogo de configuração de parâmetros para recorte de uma imagem por uma máscara

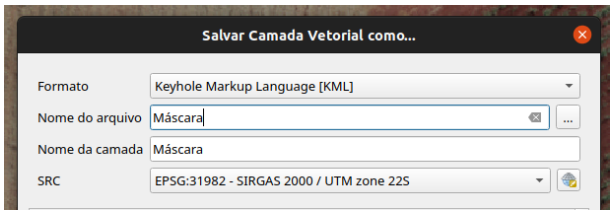

Figura A.10: Diálogo de exportação de uma camada