UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
CAMPUS LUIZ MENEGHEL - CENTRO DE CIÊNCIAS TECNOLÓGICAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VINÍCIUS HENRIQUE FERREIRA DA ROSA

FERRAMENTA MÓVEL PARA DETECÇÃO DE ÁREA FOLIAR

BANDEIRANTES–PR

2017

VINÍCIUS HENRIQUE FERREIRA DA ROSA

FERRAMENTA MÓVEL PARA DETECÇÃO DE ÁREA FOLIAR

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual do Norte do Paraná para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Me. Wellington Aparecido Della Mura

BANDEIRANTES–PR

2017

VINÍCIUS HENRIQUE FERREIRA DA ROSA

FERRAMENTA MÓVEL PARA DETECÇÃO DE ÁREA FOLIAR

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual do Norte do Paraná para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA

Prof. Me. Wellington Aparecido Della Mura
Universidade Estadual do Norte do Paraná
Orientador

Prof. Me. Bruno Miguel Nogueira de Souza
Universidade Estadual no Norte do Paraná

Prof. Dr. Ederson Marcos Sgarbi
Universidade Estadual no Norte do Paraná

Bandeirantes–PR, 24 de novembro de 2017

ROSA, V. H.. **Ferramenta móvel para detecção de área foliar**. 52 p. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Estadual do Norte do Paraná, Bandeirantes–PR, 2017.

RESUMO

Com o constante avanço da computação móvel e dos algoritmos usados para o processamento de imagens digitais, cada vez mais é possível obter a solução de problemas de maneira mais fácil e prática. Este trabalho descreve e implementa um método para mensurar características foliares necessárias para pesquisadores da área de agronomia. Para isso foi realizado o desenvolvimento de uma ferramenta móvel para a plataforma *Android* capaz de identificar os elementos de interesse de uma determinada folha como área saudável e área danificada por alguma doença. As saídas da ferramenta foram comparadas com um *software* de mercado que pode ser usado para o mesmo fim, com o mesmo conjunto de imagens. Os resultados obtidos foram satisfatórios visto que mostraram uma relação de 99% para área total e 82% para área danificada entre os resultados da ferramenta proposta e do software comparado, isso mostra que a metodologia funciona e tem potencial de mercado, pois os softwares usados para este fim são de difícil aquisição e apresentam diversos problemas. No entanto, para ser lançado a ferramenta proposta precisa de ajustes para obter maior precisão na identificação de área danificada da folha.

Palavras-chave: Processamento de Imagens. Desenvolvimento Móvel. Agricultura de precisão.

ROSA, V. H.. **Mobile app to detect leaf area**. 52 p. Final Project (Bachelor of Science in Computer Science) – State University Northern of Parana , Bandeirantes–PR, 2017.

ABSTRACT

With the steady improvement of mobile computing and the processing digital images algorithims, problem solving become easier and more pratical. This work proposes and implements a method to measure foliar characteristics necessary for agronomic researchers. For this was developed a mobile app for Android able to identify interest areas of a certain leaf, as healthy area and diseased areas. The results were compared with market software that can be used for the same purpose.The results obtained was satisfactory and showed that the methodology works and has market potential, however to be released the proposed tool needs adjustments to obtain greater precision.

Keywords: Image Processing. Mobile Computing. Precision agriculture.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Com o avanço da computação móvel e o surgimento de novos dispositivos móveis, tais como *smartphones* e *tablets* com grande poder de processamento e diversos sensores, possibilita-se a realização de tarefas complexas, que a poucos anos eram inviáveis para essas plataformas. [4] A melhoria das câmeras e dos *chipsets* gráficos impulsionaram o surgimento de diversas aplicações que fazem o uso do processamento de imagens em dispositivos móveis. A utilização da plataforma móvel para o processamento de imagens é relevante tendo em vista que os dispositivos moveis possuem todo o equipamento necessário para este propósito, desde a aquisição da imagem e o seu processamento até a exibição do resultado.

O processamento digital de imagens (PDI) consiste na aplicação de algoritmos para a melhoria de imagens que possibilite a remoção ou realce de características para uma posterior interpretação. Dessa forma, pode-se observar diversas áreas de conhecimento no qual o PDI pode ser aplicado de forma a resolver ou facilitar a solução de problemas. A pesquisa agronômica é uma das áreas onde ainda se percebe muitas tarefas sendo realizadas utilizando visão humana. A automação dessas tarefas utilizando a visão computacional, seja para a sua solução ou até mesmo automatizar etapas, traz benefícios tanto para o método quanto para quem o realiza. Isso ocorre pois eventualmente torna os métodos mais exatos e com menor dependência do auxílio humano.

Dentro dessas pesquisas diversos dados são usados para avaliar novos produtos e métodos para a agricultura. A qualidade desses dados são de grande importância para a qualidade dos resultados. Dentro deles, a área da folha é usada para analises quantitativas em diversas pesquisas realizadas por agrônomos, assim como área de doenças ou danos. Por exemplo, em muitos casos é intensante saber qual a porcentagem da folha que foi atacada por certa doença para assim avaliar algum produto disposto a combate-la.

Embora o PDI seja uma área repleta de possibilidades e facilite a execução de tarefas complexas, seu custo computacional muitas vezes também é considerável. Uma imagem digital é representada por uma matriz bidimensional e os algoritmos de PDI tentem a exigir um alto processamento e grandes porções de memória [3]. Tendo em vista esta situação, aplicar o PDI na agricultura com o uso de dispositivos móveis remete a algumas dificuldades geradas principalmente pelas limitações de tais dispositivos. Entre estas dificuldades estão o gasto energético da ferramenta, a velocidade no processamento e a dificuldade de conexão à rede. Além disso, o profissional deve ser capaz de trabalhar sem perder a mobilidade e garantir o uso da aplicação durante um período relativamente longo.

Diante de tantos fatores positivos e negativos, observa-se a necessidade de buscar formas de otimizar o desenvolvimento de uma aplicação móvel que utilize técnicas de PDI visando atender todos os requisitos encontrados. Por isso, este trabalho tem como objetivo estudar técnicas que auxiliem o desenvolvimento de aplicações de processamento de imagens em dispositivos móveis que almejem a otimização de recursos disponíveis.

1.1 Objetivos

O objetivo geral desde trabalho é estudar as opções de desenvolvimento móvel disponíveis para aplicações de processamento de imagens relacionando com o desempenho e a utilização de recursos. Para que o objetivo geral seja alcançado, os seguintes objetivos específicos devem ser considerados:

- Analisar as principais aplicações do processamento de imagens na área agronômica;
- Elencar os dispositivos móveis e as plataformas mais comuns atualmente utilizados para processamento de imagem;
- Adotar um estudo de caso para uma aplicação móvel que utilize técnicas de processamento de imagens na agronomia;
- Enumerar os filtros e técnicas de processamento de imagens aplicáveis ao estudo de caso proposto;
- Analisar a melhor forma de implementação das técnicas visando o melhor aproveitamento dos recursos.

1.2 Motivação

A possibilidade de usar técnicas de processamento de imagens com mobilidade, cria novas oportunidades para o surgimento de aplicações inovadoras. Dentro desse contexto é possível elencar as áreas de realidade aumentada, reconhecimento de objetos e extração de características que estão em crescente desenvolvimento. Para o desenvolvedor desse tipo de aplicação é possível utilizar diferentes linguagens e arquiteturas diferentes, porém ainda diversas questões necessitam ser respondidas. Nos *mobiles* outras métricas de desempenho surgem além dos encontrados nas estações de trabalho fixas, neles cria-se a preocupação com gasto de bateria, com o uso de rede, armazenamento e processamento reduzido, entre outros.

Na busca por desenvolvimento em processamento de imagens móvel, ainda tem-se encontrado poucos estudos que trazem informações a respeito dessas métricas, que para usuários finais são de grande importância. Mesmo que uma aplicação tenha funcionalidades com grande valor para o cliente, dificilmente será usada se fizer o uso excessivo de

recursos. Ao gerar dados que compare diferentes métodos também se otimiza o processo de desenvolvimento para projetos futuros facilitando o crescimento da área e o surgimento de soluções inovadoras.

Com a oportunidade de aplicar os conhecimento de PDI em diferentes áreas, as agrárias se apresentam como uma boa oportunidade, especificamente o setor agrícola. Pois ainda hoje, grande parte da economia brasileira é ligada ao agronegócio, que segundo o Ministério do Desenvolvimento Agrário é constituído por 70% de pequenos e médios produtores. Devido a isso pode se perceber que muitos processos podem ser melhorados, muitas vezes feitos de forma visual, e apenas guiados pela experiência, melhorar tais processos incentivaria ainda mais esse setor. Com isso é imprescindível que haja desenvolvimento tecnológico de fácil acesso voltado aos profissionais e produtores agrícolas.

1.3 Organização do Trabalho

A organização do trabalho obedece a estrutura a seguir. No Capítulo 2 é apresentado um referencial teórico sobre Processamento Digital de Imagens e Computação Móvel. O Capítulo 3 apresenta o desenvolvimento do trabalho com uma descrição do cenário, projeto e análise da ferramenta além das técnicas utilizadas. O Capítulo 4 apresenta o processo de validação da ferramenta em comparação com uma solução disponível no mercado. Finalmente, o Capítulo 5 descreve as considerações finais do trabalho, em relação ao método proposto e aos problemas pendentes, bem como direções para trabalhos futuros.

2 FUNDAMENTAÇÃO

Este capítulo apresenta alguns conceitos significativos à elaboração deste trabalho. Inicialmente são definidas técnicas a respeito de processamento de imagens digitais e em seguida sobre computação móvel.

2.1 Processamento Digital de Imagens

As técnicas computacionais denominadas processamento de imagens buscam atender a dois principais problemas: melhorar a representação digital de uma imagem para facilitar a interpretação humana e o processamento de seus dados para o reconhecimento autônomo por uma máquina. Portanto o campo de processamento de imagens refere-se ao processamento de imagens digitais através de um computador. [3]

2.1.1 Aquisição de Representação de uma imagem

Para a manipulação de uma imagem por uma máquina é necessário inicialmente que se gere uma representação digital aproximada de imagem real de maneira finita. A aquisição de uma imagem é feita através da interação da energia eletromagnética com dispositivos capazes de transforma-la em valores que se possa mensurar, tais dispositivos são chamados de sensores. Sendo assim, sensores, dependendo da sua especificação, possuem maior poder de visão que os olhos humanos, podem responder a varias partes do espectro eletromagnético, não apenas a luz visível.

O espectro eletromagnético é o agrupamento das ondas eletromagnéticas de acordo com sua quantidade de energia, ondas eletromagnéticas consistem na alternância dos campos elétrico e magnético que se propagam juntos de forma perpendicular na velocidade da luz(3×10^8 metros por segundo). São elas: a luz visível, infravermelho, ultravioleta, raios x, micro-ondas, ondas de radio, e ondas gama. [5]

Além das ondas eletromagnéticas, sensores também são capazes de responder, portanto formar imagens, de energia acústica, como imagens de ultra-som ou radares que representam a distância a um objeto. Porém após a aquisição da imagem, não importa sua origem, se é luz visível, infravermelho ou ultra-som, os métodos usados para seu processamento serão os mesmos.

Para gerar uma imagem a partir dos dados coletados por um sensor, ela ganha a representação de um vetor bidimensional ou matriz de elementos chamados pixels. Cada pixel pode guardar o valor de luminosidade e cor de determinado ponto da imagem dependendo do seu tipo, entre os tipos de imagens mais comuns estão: imagens binárias, monocromáticas ou escala de cinza, coloridas e multiespectral.

Uma imagem binária é o tipo mais simples de imagem, cada pixel possui 1 bit de valores possíveis, ou seja pode assumir apenas dois valores, 0 e 1, para as cores branco e preto, normalmente obtidas a partir de outros tipos com o intuito de manter apenas informações necessárias para aplicações de visão computacional.

Imagens em escala de cinza possuem apenas uma cor, ou seja cada pixel possui um único valor representando a luminosidade daquele ponto. A quantidade de bits por pixel indicará o número de diferentes luminosidades possíveis, normalmente usado 8-bits possibilitando 256 valores, uma maior quantidade de bits pode ser usada caso necessite de grandes detalhes, como a astronomia e medicina como 12 ou 16 bits.

Imagens coloridas necessitam que cada pixel contenha também informação de uma cor, para isso diferentes modelos são usados dependendo do objetivo, como o rgb, cmy, hsi, hsl.

Imagens multiespectro normalmente são as que contém informações fora do campo de visão humano, como o infravermelho, ultravioleta, raio-x que são representadas de forma visual através do mapeamento de suas diferentes bandas para o RGB.

2.1.2 Modelos de cor

Para a representação de cor em uma imagem digital diferentes modelos foram desenvolvidos, cada qual com seu objetivo de utilização, dentre eles os mais utilizados são o RGB, o CMY e o HSI. Entre estes existem dois que são complementares: o modelo de cor aditivo (RGB) e o modelo de cor subtrativo (CMY).

No modelo RGB existe um tipo de cone que detecta e responde à cor vermelha (R de red), outro que detecta e responde à cor verde (G de green) e um terceiro correspondente à azul (B de blue). Estas cores designam-se por cores primárias aditivas, pois é possível definir qualquer cor através da especificação das quantidades de luz R, G e B que essa cor contém. Nota-se que a ausência de luz ou de cor corresponde a cor preta, enquanto a mistura dos comprimentos de onda ou das cores vermelho(R), verde(G) e azul(B) indicam a presença da luz ou a cor branca.

Este modelo é utilizado como base para a fabricação de monitores de computador e ecrãs de televisão. Nestes dispositivos, cada pixel é constituído por três pontos de três tipos diferentes de fósforo. A mistura óptica da luz emitida pelos três tipos de componentes de cada pixel faz com que aparentem ser um ponto único que possui a cor desejada. Os *scanners* detectam luz que é refletida do documento que está sendo digitalizado, por isso também trabalham com este modelo.

A quantidade de luz de cada cor é dada por um valor de 8-bits portanto a cor dentro do rgb é uma função C(r, g, b), por exemplo: C(255,0,0) - cor vermelha pura, C(0,255,0) - cor verde pura, C(0, 0, 255) - cor azul, C(255, 255, 255) corresponde à cor

branca ou seja, à mistura de proporções idênticas de luz em três cores primárias aditivas. C(0, 0, 0) ausência de cor, ou cor preta.

Já no modelo CMY que é subtrativo, ao contrário do RGB, a mistura de cores cria uma cor mais escura, porque são absorvidos mais comprimentos de onda, subtraindo-os à luz. A ausência de cor corresponde ao branco e significa que nenhum comprimento de onda é absorvido, mas sim todos refletidos.

Este modelo explica a mistura de pinturas e tintas para criarem cores que absorvem alguns comprimentos de onda da luz e refletem outros. Assim, a cor de um objeto corresponde à luz refletida por ele e que os olhos recebem. Podemos utilizá-las na pintura de impressão.

Em muitas aplicações o RGB é transformado em um espaço matemático que desacopla a informação de luz da informação de cor, esta transformação é referida como transformação de cor ou mapeamento em um espaço de cor diferente. Este novo modelo utiliza de uma dimensão para a luminosidade e duas para cor. [5]

Como por exemplo o modelo Hue/Saturation/Intensity (HSI), onde o *HUE*: É a matiz ou cor pura. O seu valor varia entre 0 (vermelho), passando pelo laranja, amarelo, verde, azul, púrpura, e novamente vermelho. *SATURATION*: Indica a quantidade de luz branca que foi misturada a cor pura. É inversamente proporcional: a cor pura tem saturação máxima, e quanto mais luz branca é adicionada a saturação vai diminuindo; *INTENSITY*: Indica a intensidade monocromática da cor (refletância), ou seja, a intensidade (em níveis de cinza) que a cor foi refletida ou absorvida. Uma vantagem do HSI é que está mais relacionado com a forma que o ser humano percebe as cores.[5]

2.1.3 Transformações de cor

Dependendo da aplicação desejada um modelo de cor pode oferecer vantagens ou desvantagens sobre outro modelo, tornando interessante a transformação de um em outro para obter melhores resultados. Abaixo pode-se observar como é realizado as principais transformações de modelos de cor.

De RGB para monocromático:

$Y = 0.3R + 0.59G + 0.11B$

A conversão do modelo RGB em CMY, pode ser feita da seguinte maneira:[6]

[C] = [255] - [R]

[M] = [255] - [G]

[Y] = [255] - [B]

Onde o R, G, B representa os valores normalizados de 0 a 255. Para conversão do

CMY para RGB:

[R] = [255] - [C]

[G] = [255] - [M]

[B] = [255] - [Y]

Para mapeamento do RGB para HSL, onde min e max são respectivamente o maior e o menor dos valores RGB, e os valores para RGB são normalizados entre 0 e 1 temos:[5]

Para Hue:

se max = min: $H = 0$

se max = r: $H = 60°x\frac{g-b}{max-min} + 360°$

se max = g: $H = 60°x\frac{b-r}{max-min} + 120°$

se max = b: $H = 60°x\frac{r-g}{max-min} + 240°$

Para Intensidade:

$L = \frac{1}{2}(max + min)$

Para Saturação:

se max = min $S = 0$

se L <= 1/2: $S = \frac{max-min}{max+min} = \frac{max-min}{2L}$

se L > 1/2: $S = \frac{max-min}{2-(max+min)} = \frac{max-min}{2-2L}$

2.1.4 Melhorias de imagens

O principal objetivo de melhorar uma imagem é que seu resultado seja mais adequado que a imagem original para uma aplicação especifica[3]. A melhoria de imagem pode ser dividida em duas categorias: domínio de espaço e domínio de frequência. As operações no domínio do espaço são caracterizadas pela manipulação direta dos pixels da imagem enquanto as operações de domínio de frequência são baseadas nas transformações de fourier.

Operações de domínio de espaço podem ser representadas pela expressão: $g(x, y) = T[f(x, y)]$ onde $f(x, y)$ é a imagem de entrada, $g(x, y)$ a imagem processada e T a operação realizada.

A operação inicia com a definição de uma vizinhança sobre um ponto, ou seja, pontos ao seu redor podendo formar um retângulo, quadrado ou aproximações de circulo, sendo chamado também de janela. A forma mais simples de operação é onde a janela tem o tamanho 1 x 1, apenas um pixel, ou pontuais, como nível de cinza, ajuste de brilho e contraste, *thresholding*.

Os valores dos pixels antes e depois da operação podem ser denotados por r e s respectivamente, como introdução as transformações em nível de cinza se tem 3 funções básicas usadas na melhoria de imagem, transformação negativa, logarítmica e *power-law*.

Transformação negativa é usada para melhorar áreas brancas ou cinzas especialmente quando áreas escuras são dominantes.

Transformações logarítmicas geralmente são da forma : $s = clog(1+r)$, usado para expandir os pixels pretos enquanto comprime o valores de alto nível.

Transformações power-law tem a forma de $s - cr^y$ onde e e y são constantes.

2.1.5 Análise de imagem

A análise de imagem digital é o fator chave para solução de qualquer problema computacional envolvendo imagem. A aquisição de imagens de amostra para a base de dados e a verificação dessas imagens para a aplicação é o primeiro passo para o desenvolvimento de uma solução.[5] Análise de imagem envolve a manipulação dos dados da imagem para determinar quais informações serão necessárias para o desenvolvimento do sistema.[5]

A análise de imagem busca também a redução da quantidade total de dados de imagem, que normalmente chegam a milhares de kilobytes, e megabytes, então é necessário isolar apenas as informações que serão relevantes para a solução do problema.

Um modelo de análise de imagem pode ser descrito em três passos principais: Pré-processamento, Redução de dados e análise de características.[5] Pré-processamento é usado para remover ruídos ou informações desnecessárias, pode incluir também transformação em nível de cinza e reajuste no tamanho da imagem. Para redução de dados são usados operações de domínio de espaço ou de domínio de frequência, e então características são extraídas para análise. Já na análise os dados gerados são examinados e avaliados para o uso na aplicação.

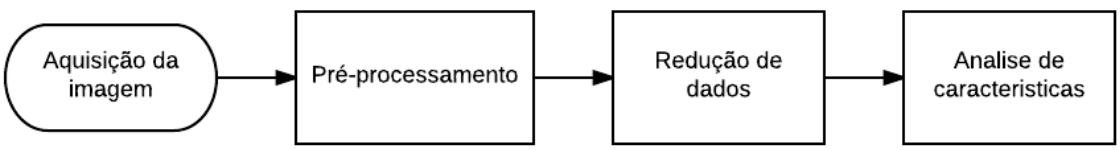

Figura 1 – Modelo analise de imagens

2.1.6 Pré-processamento

O pré-processamento utiliza dos algorítimos e técnicas para para melhorar a imagem para os passos seguintes, faz também a primeira redução de dados, pode incluir operações como: extração de regiões de interesse, operações matemáticas, melhoria de características específicas, transformações de cor e redução de dados em resolução e brilho.[5]

Isolar a região de interesse pode ser a primeira operação feita em uma imagem, para isso pode se usar funções como cortar, aumentar, estender, encolher e rotacionar. Operações matemáticas e lógicas também poder ser usadas onde imagens são combinadas de diversas maneiras como adicionando uma a outra, subtraindo, dividindo ou multiplicando, ou também aplicando operadores lógicos como AND,OR e NOT. Tais operações são aplicadas usando duas imagens onde simplesmente se realiza a operação nos pixels correspondentes, como por exemplo em uma operação de adição:

$$I1\begin{bmatrix}3&4&7\\3&4&5\\2&4&6\end{bmatrix}+I2\begin{bmatrix}6&6&6\\4&2&6\\3&5&5\end{bmatrix}=I3\begin{bmatrix}3+6&4+6&7+6\\3+4&4+2&5+6\\2+3&4+5&6+5\end{bmatrix}=\begin{bmatrix}9&10&13\\7&6&11\\5&9&11\end{bmatrix}$$

Em resumo a adição é usada para combinar informações de duas imagens, a subtração pode ser usada para detectar movimento de algum objeto em imagens sequenciais, e se o tempo e o espaço forem conhecidos identificar também a velocidade. A divisão e a multiplicação irão alterar o brilho da imagem tornando-a mais escura ou mais clara respectivamente. O operadores AND e OR podem ser usados como um jeito simples de remover as áreas de interesse e o operador NOT para criar imagens negativas ou inversas da imagem original.

Filtros espaciais de suavização também podem ser usados na etapa de pré-processamento, na tarefa de remoção de pequenos detalhes. A saída de um filtro espacial linear de suavização é simplesmente a média dos pixels contidos na vizinhança da máscara de filtragem. Esses filtros por vezes são chamados de filtros de média. Como mencionado na seção anterior, eles também podem ser chamados de filtros passa-baixa.[3]

Porém esse processo resulta na perca de nitidez. Como complemento se tem os filtros de estatística de ordem, cuja resposta se baseia na ordenação dos pixels contidos na área da janela e substituindo o valor do pixel central pelo valor determinado pelo resultado da classificação. O filtro mais conhecido dessa categoria é o filtro de mediana, o qual substitui o valor de um pixel pela mediana dos valores de intensidade na vizinhança desse pixel . Os filtros de mediana são bastante populares, porque para certos tipos de ruído aleatório, proporcionam excelentes resultados, com borramento consideravelmente menor do que filtros lineares de suavização.[3]

Para a redução de dados, a quantização da imagem poder ser utilizada. Onde

segundo [5] é o processo de redução dos dados da imagem, removendo algumas das informações detalhadas através do mapeamento dos grupos de pontos de dados para um único ponto. Isso pode ser feito para os próprios valores dos pixels ou para as coordenadas espaciais. A operação nos valores de pixel é referida como redução de nível de cinza, enquanto opera nas coordenadas espaciais é chamada de redução espacial.

O processo pode ser feito reduzindo o número de bits de cada pixel, por exemplo, para reduzir de 8 bit para 5 bits, uma operação *AND* é realizada entre o valor original e o binário 11111000, o equivalente a dividir por 8, o que irá diminuir de 256 valores para 32 possíveis valores. O método mais simples de redução é a limiarização onde é selecionado um valor em escala de cinza e qualquer valor acima dele é atribuído "1"e qualquer valor abaixo "0", resultando assim em uma imagem binária ou de dois níveis.

2.1.7 Segmentação

O objetivo da segmentação da imagem é encontrar regiões que representem objetos ou partes significantes de objetos. É necessário a divisão da imagem em regiões correspondentes a objetos de interesse antes de qualquer processamento, pode ser feito em um nível superior ao do pixel. A identificação de objetos reais, pseudo-objetos, sombras ou encontrar qualquer coisa de interesse dentro da imagem, requer alguma forma de segmentação.[5]

Porem a segmentação é uma das tarefas mais difíceis no processamento de imagens. A precisão da segmentação determina o sucesso ou o fracasso final dos procedimentos de análise computadorizada.Por essa razão, deve-se tomar muito cuidado para aumentar a probabilidade de se obter uma segmentação precisa. [3]

A maioria dos algoritmos de segmentação baseia-se em uma das seguintes propriedades básicas de valores de intensidade: descontinuidade e similaridade, na primeira categoria, a abordagem é dividir uma imagem com base nas mudanças bruscas de intensidade, como as bordas. As abordagens principais na segunda categoria estão baseadas na divisão de uma imagem em regiões que sejam semelhantes de acordo com um conjunto de critérios pré-definidos. A limiarização, o crescimento de região e a divisão e fusão de regiões são exemplos dos métodos desta categoria. [3]

A partir disso temos algumas técnicas que segundo [7] são bem estabelecidas para o processo de segmentação:

- Limiarização baseada em histograma.
- Crescimento de região.
- Clustering/Classificação.
- Gráfico de abordagem teórica.

- Região de *splitting* e *merging*.
- Regras baseadas em base de conhecimento.

Algum algoritmos para a segmentação também podem ser destacados como: Detecção de bordas, limiarização, segmentação baseada em região, *watersheds* morfológicas.

No algorítimo de detecção de borda, linhas ou pontos se tem o interesse em buscar os locais de mudanças abruptas de intensidade. As bordas são o conjunto desses pixels onde ocorre essa situação de forma conexa. Uma linha pode ser vista como um segmento de borda em que a intensidade do fundo de cada lado da linha ou é muito superior ou muito inferior à intensidade dos pixels da linha.

Na limiarização se tem porém maior simplicidade de implementação e velocidade computacional e devido a isso ocupa posição central nas aplicações de segmentação de imagem. [3] A limiarização utiliza técnicas para a divisão da imagem diretamente em regiões baseando principalmente nos valores de intensidade ou propriedades desses valores.

Uma maneira básica de aplicar a limiarização é buscar no histograma de intensidade da imagem o(s) valor(es) que separa os agrupamentos de valores de intensidade dominantes e aplicar o limiar nesse valor. Então qualquer ponto acima é chamado de ponto do objeto e qualquer valor abaixo de ponto de fundo.

Quando o mesmo limiar é aplicado em toda a imagem o processo e chamado de limiarização global e quando o limiar muda dependendo da região da imagem é usado o termo limiarização variável, algumas vezes também é usado o termo limiarização local quando o limiar depende das propriedades dos pixels da vizinhança

Na segmentação baseada na região o objetivo é a divisão da imagem em regiões. Como no crescimento de região onde a abordagem básica é inciar com um conjunto de pontos chamadas semente e a partir deles, fazer as regiões crescerem incluindo a cada semente aqueles pixels vizinhos que têm propriedades predefinidas semelhantes às das sementes (como os intervalos específicos de intensidade ou cor).[3]

2.2 Computação móvel

A computação móvel pode ser vista como um novo paradigma computacional no qual possibilita que seus usuários tenham acesso a informações e serviços independentemente de sua localização, podendo inclusive, estar em movimento. Tem a necessidade de envolver processamento, mobilidade e comunicação sem fio, Pois seu principio é ter acesso à informação em qualquer lugar e a qualquer momento.[8]

A computação móvel pode ser considerada a quarta revolução da computação, antecedida pelos centros de processamento de dados da década de 60, dos terminais na

década de 70 e as redes de computadores na década de 80. E o que define este novo paradigma é a mobilidade. Utilizadores podem acessar serviços independentemente de onde estejam localizados. Isso é possível graças a comunicação sem fio que elimina a necessidade de o utilizador manter-se conectado a infraestrutura física.[9]

2.2.1 Dispositivos moveis

Com o avanço da computação móvel cria-se a necessidade do desenvolvimento de dispositivos que atendam os requisitos para sua aplicação. Um dispositivo para este fim deve ter a capacidade de realizar processamento, trocar informações via rede e ser capaz de ser transportado facilmente pelo seu utilizador. Para isso, é importante que o dispositivo computacional tenha tamanho reduzido e não necessite de cabos para conectá-lo à rede de dados ou fonte de energia elétrica.[8]

Vários modelos de dispositivos são usados para os fins da computação móvel, esses dispositivos se apresentam de várias formas, possuem tamanhos e características diferentes, além de executarem softwares ou sistemas operacionais. Algum sistemas operacionais que podem ser usados nos dispositivos móveis, sendo que os mais usados são: IOS, Android, Windows Mobile, Blackberry, entre outros.

No entanto será detalhado a seguir apenas o sistema operacional Android pois será nele que o estudo de caso será implementado, e por oferecer maior informações a respeito de programação em PDI.

2.2.2 Plataforma Android

O sistema operacional Android é baseado em Java e executado no kernel do Linux. É uma plataforma para dispositivos móveis, criado para diminuir custos e melhorar a experiência dos usuários nesses dispositivos. Teve seu desenvolvimento iniciado em 2003 pela empresa Android Inc., que foi comprada pela Google e hoje lidera o desenvolvimento do Android. Desde o seu lançamento até os dias atuais o Android ganhou proporções e funcionalidades admiráveis.[10]

O seu código é aberto o que torna a principal escolha da maioria dos fabricantes de smartfones e tablets. Tem suporte a qualquer tipo de conexão sem fio e também é compatível com praticamento todos os tipos de multimídia, sendo também uma das principais escolhas para os desenvolvedores. Com a programação aberta ele é capaz de ser alterado, adaptar-se e pode manter um baixo custo.

O sistema operacional Android possui grande quantidade bibliotecas que são usadas por diversos componentes para realizar ações básicas e avançadas dos sistema. Existe vários tipos de bibliotecas, exemplos destas são: bibliotecas de mídia, gráficas, acesso e suporte ao banco de dados (SQLite), dentre outras.[1]

A arquitetura do Android é composta por várias camadas e componentes, definidos abaixo:

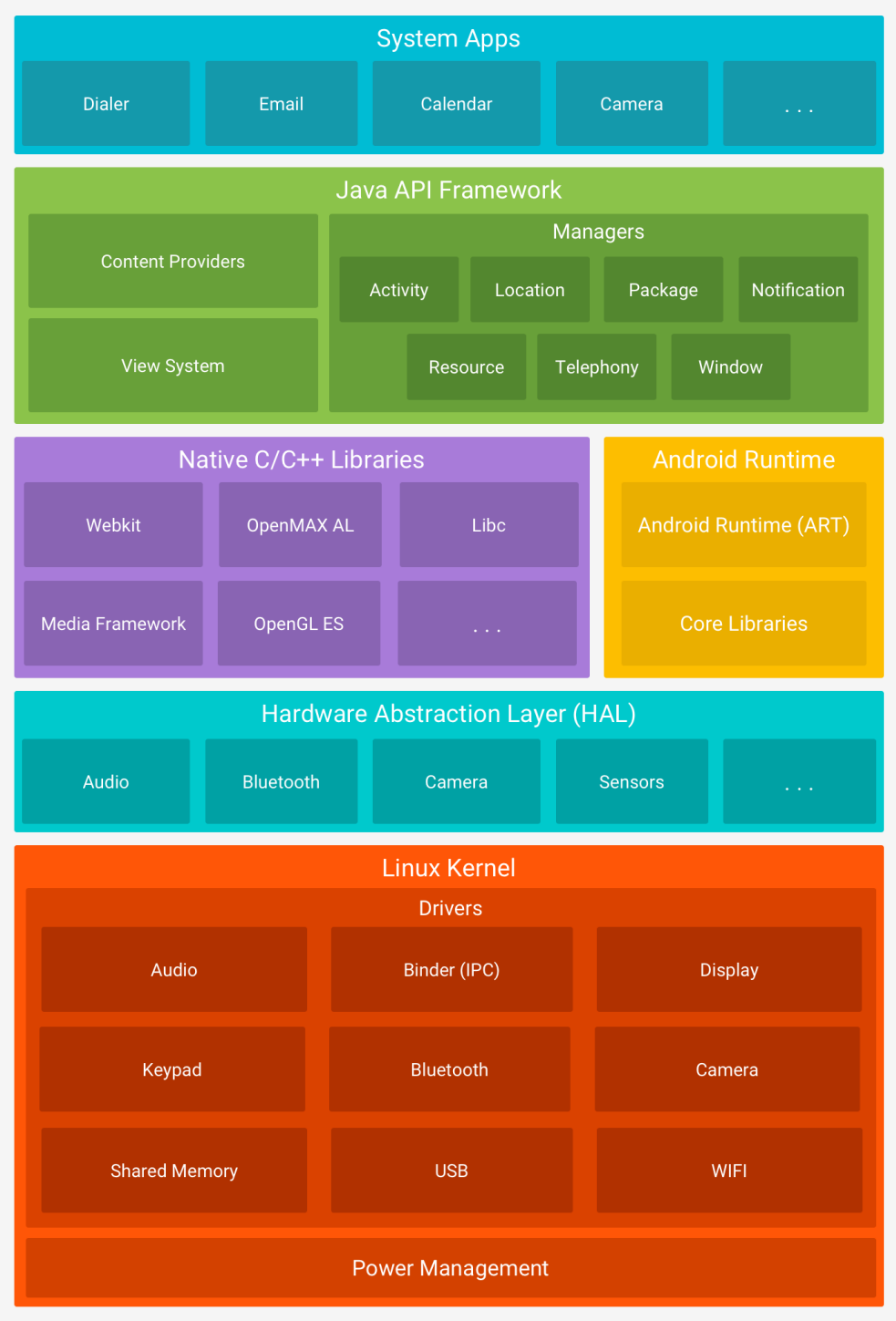

Figura 2 – Representação da arquitetura Android, adaptado de [1]

- *System Apps*: Representa as aplicações que executam sobre a plataforma. Elas podem ser tanto aplicações nativas – como o gerenciador de contatos, navegador, calendário, etc. – como aplicações criadas por terceiros.

- *API Framework*: Onde ficam as APIs do Android que são normalmente utilizadas pelas aplicações que executam sobre a plataforma.
- *Android Runtime*: Dá condições para que as aplicações baseadas na plataforma sejam executadas. Um dos componentes desta camada são as *core libraries*, que disponibilizam uma API Java utilizada para programação (grande parte das funcionalidades encontradas no Java SE estão disponíveis para o Android). Já o outro componente é a Dalvik Virtual Machine, que é uma máquina virtual para suporte à execução de aplicações.
- *Libraries*: Onde estão as bibliotecas nativas escritas em C/C++ que fazem parte da plataforma.
- *Linux Kernel*: Onde está localizado o sistema operacional da plataforma, que é baseado no Linux. Ela é responsável por serviços de mais baixo nível da plataforma, como gerenciamento de memória e processos.

3 DESENVOLVIMENTO

Neste capitulo serão apresentadas as etapas necessárias para o desenvolvimento do trabalho no qual podemos dividir em apresentação do cenário, projeto, fluxograma para o processamento e a ferramenta proposta.

3.1 Apresentação do Cenário

Por meio de observação e reuniões realizadas no Núcleo de Investigação em Tecnologia de Aplicação de Agroquímicos e Maquinas Agrícolas (NITEC) localizado na Universidade Estadual do Norte do Paraná (UENP), pode se observar que em diversas pesquisas relacionadas a agroquímicos e doenças foliares utilizam da extração de informações da folha.

Segundo os coordenadores do laboratório as informações mais usadas nestas pesquisas são:

- Área foliar, considerando a área total de determinada folha apresentada em cm^2.
- Área afetada por doença, qualquer parte da folha diferenciável de uma folha saudável pela cor por conter queimaduras ou patologias apresentada em cm^2.
- Porcentagem da folha saudável e não saudável.

Para coletar essas informações o núcleo faz o uso de um *software* proprietário, que possui algumas limitações destacadas pelos membros da equipe: a primeira delas é a aquisição do *software* que possui um preço elevado, limitando o trabalho realizado no mesmo, já que o laboratório tem apenas uma licença para o uso. Além disso requer um equipamento específico para a aquisição das imagens, contendo câmera e iluminação próprias. Outro ponto negativo do *software* é que ele requer que as plantas sejam levadas em laboratório fazendo com que a investigação seja destrutiva para a planta.

3.2 Projeto

O projeto visa desenvolver uma aplicação móvel capaz de coletar as informações descritas de forma mais prática que a utilizada pelo NITEC. Para isso pretende-se que a aquisição da imagem seja realizada pela própria câmera do *smartphone*. As configurações devem ser de maneira simplificadas por se tratar de um dispositivo móvel com tela limitada. Assim alguns processos precisam ser implícitos à aplicação não necessitando

intervenção do usuário: como a identificação de fundo e segmentação da folha. O usuário deve apenas fornecer a cor da folha saudável e a aplicação identifica o restante por proximidade ou não a essa cor. A aplicação será funcional também em campo aberto, permitindo que a planta não seja destruída e o trabalho realizado fora de laboratório.

Assim o fluxo da aplicação e os filtros utilizados pode ser descrito pela figura 3.

3.3 Fluxograma do processamento da folha

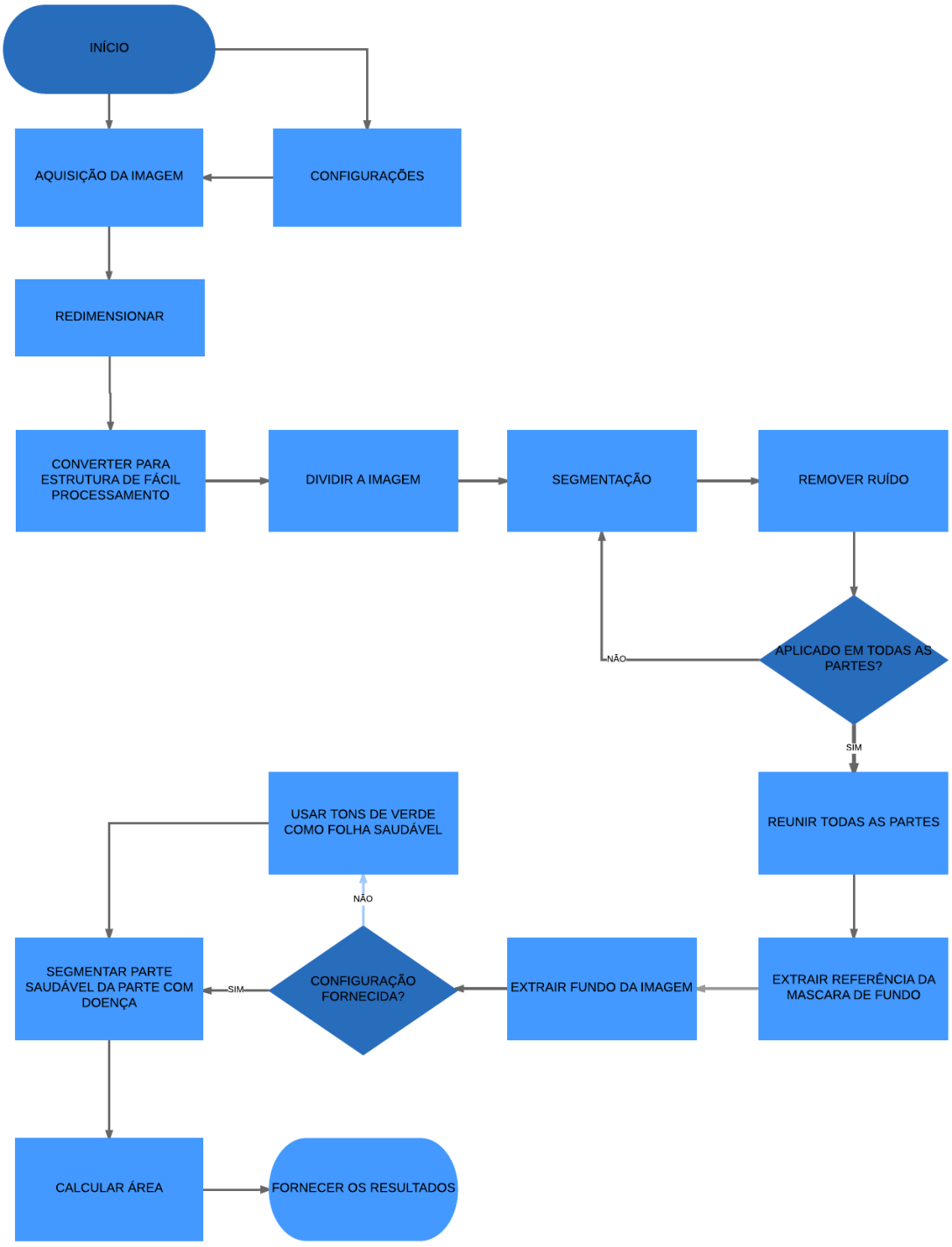

Figura 3 – Fluxograma do processamento da folha

- Inicio - A tela inicial do aplicativo fornece um menu contendo as opções de configu-

rações ou para aquisição das imagens.

- Aquisição - Imagem é obtida na através de uma câmera ou a partir de imagens capturadas anteriormente, a folha deve estar em uma superfície branca podendo ou não ter um objeto com área conhecida ao lado para auxiliar a conversão em cm^2 da área da folha posteriormente, esse objeto deve ser apenas impresso no fundo e ser menor que a folha estudada. A figura 4 ilustra o exemplo de aquisição de uma folha.

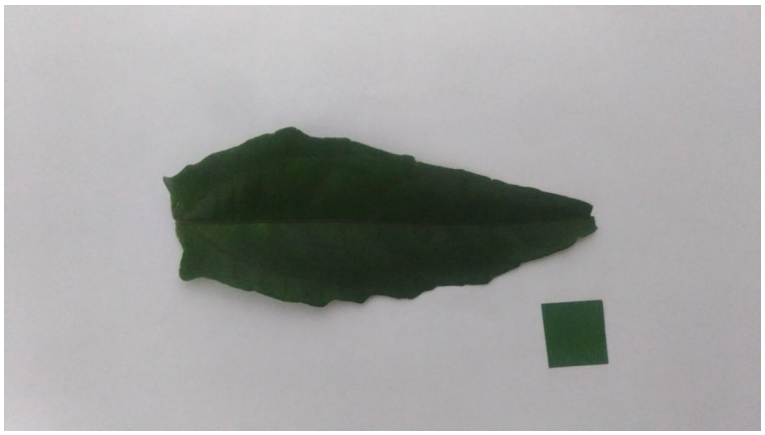

Figura 4 – Aquisição com objeto de referencia

- Configuração - O usuário devera inserir quais cores espera que sejam encontradas em determinado conjunto de folhas.
- Redimensionamento - A imagem é redimensionada para um tamanho menor do que o original, o que aumenta a performance das operações seguintes já que os smartfones atuais são capazes de capturar imagens de tamanho superior ao necessário para esta aplicação.
- Converter - Representar as Imagens em uma matriz para que as operações sejam realizadas de forma mais otimizada.
- Dividir - Com a aquisição da imagem sendo feita com o *smartphones* em ambiente não controlado, muitas vezes a iluminação pode não ser satisfatória o que pode atrapalhar no processo de segmentação da folha. O que pode ser observado na figura 5:

Figura 5 – Foto com iluminação ruim

Quando a segmentação com método de otsu[2] é aplicada nessa imagem com objetivo de separar o fundo da folha se tem como resultado:

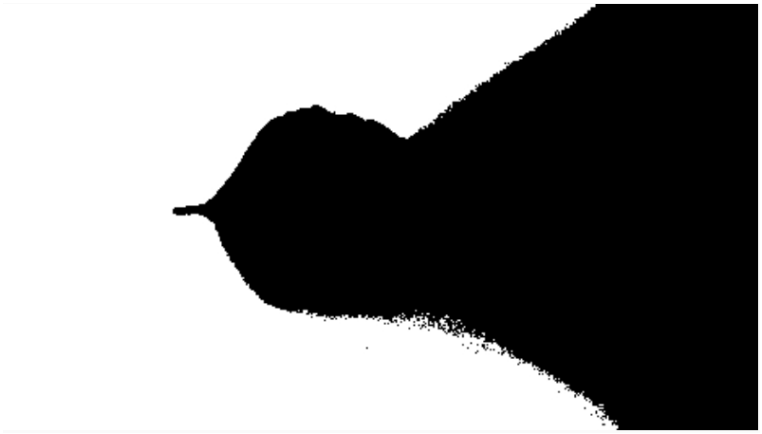

Figura 6 – Figura 3 apos algorítimo de otsu[2]

O resultado obtido é totalmente diferente do esperado, a sombra que a imagem possui é agrupada junto a folha, o que está errado, pois a intenção é apenas segmentar a folha do fundo. A maneira mais fácil encontrada para resolver esse problema e dividir em varias imagens e segmentar cada unidade, como nesse exemplo onde a imagem foi dividida em 4 processada e unida novamente:

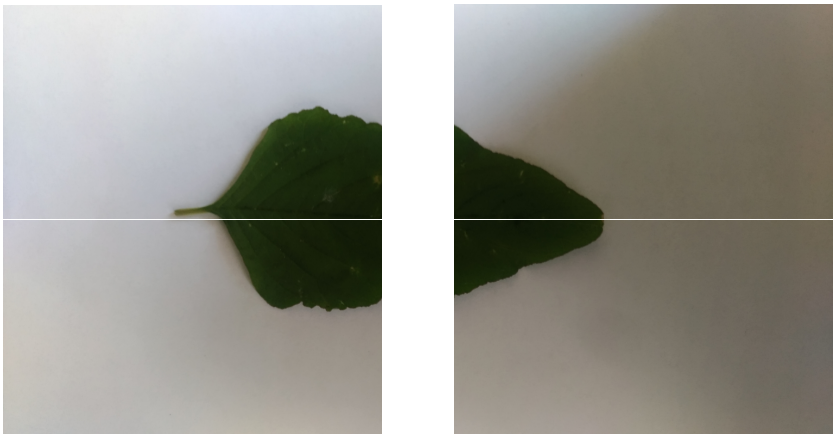

Figura 7 – Figura 5 dividida em 4 partes.

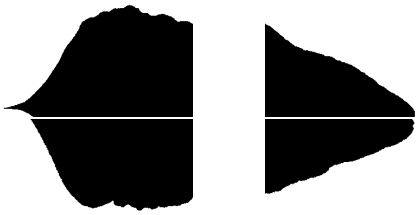

Figura 8 – Segmentação aplicada em cada uma das partes.

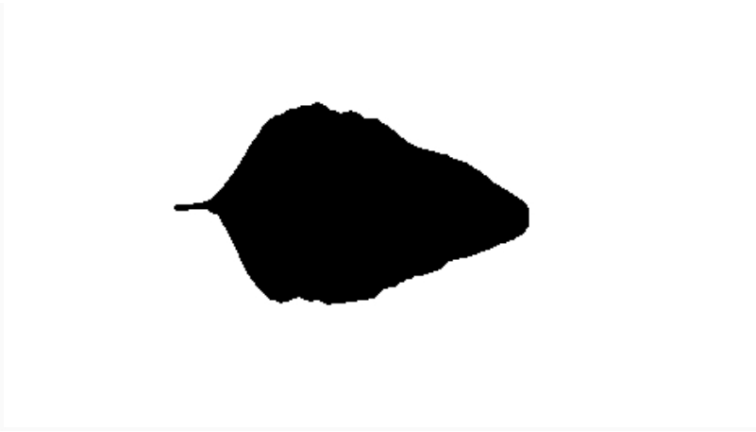

Figura 9 – Partes segmentadas concatenadas.

- Segmentação - Nesta etapa busca-se separar a imagem em duas regiões com objetivo de identificar a folha na imagem.

- Remover ruídos - Alguns ruídos podem atrapalhar a processamento posterior como os destacados na figura 8.

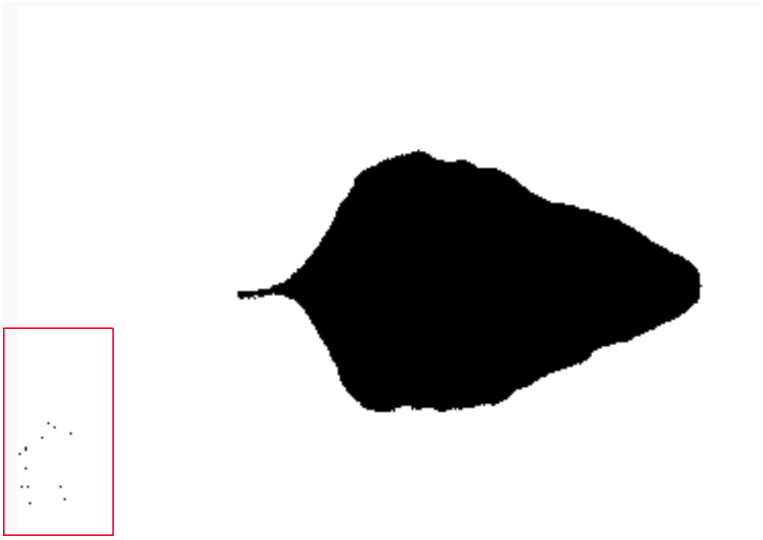

Figura 10 – Ruídos

Senso assim o filtro da mediana é um filtro de janela que calcula a mediana entre uma quantidade de pixeis pre-definida, o que elimina pixeis fora do padrão removendo ruídos e dando um efeito borrado a imagem.

- Reunir as partes - Os pedaços da imagens dividida nas etapas anteriores são concatenados novamente em uma única imagem.

- Extrair objeto de referência da mascara - Depois que a imagem é reunida se tem a segmentação da folha e do objeto de referencia do fundo:

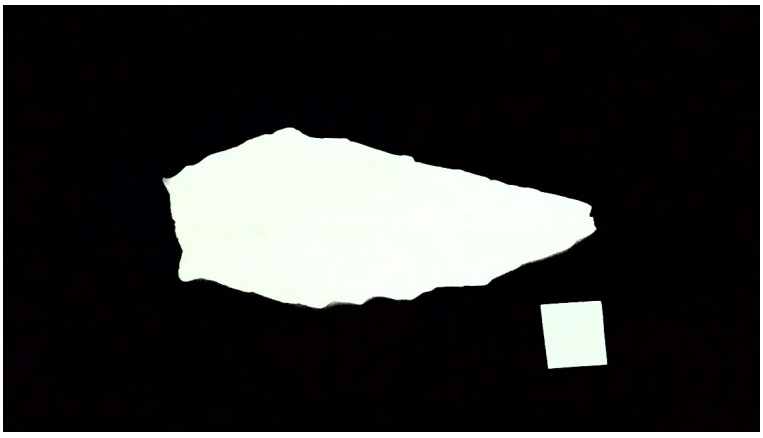

Figura 11 – Imagem segmentada com objeto de referencia

Na imagem acima pode-se ver dois objetos, a folha e o pequena quadrado que foi impresso no fundo. A área do quadrado deve ser obtida e em seguida removida da imagem, tao área sera usada como referencia para a conversão $emcm^2$, já que se sabe que o quadrado na imagem tem $4cm^2$. É necessário essa etapa devido aos requisitos

iniciais do aplicativo, em que os usuários necessitam da área em cm^2. Nas próximas etapas serão necessárias só a parte da folha da imagem.

- Remover fundo - o fundo é retirado usando a mascara obtida no passo anterior e a imagem original aplicando operação logica AND. Como pode ser visto na figura 10.

Figura 12 – Operação AND entre duas imagens, adaptado de [3]

- Segmentar parte saudável de parte doente - Com as cores fornecidas na configuração será aplicado uma segmentação por cor para encontrar essas duas partes.
- Usar tons de verde como folha saudável - Caso a cor da folha não seja fornecida na configuração utilizar todo o espectro verde para identificar a parte saudável da folha.
- Calcular área - calcular a área das duas regiões segmentadas anteriormente e converter para cm^2.

3.4 Ferramenta proposta

Para realizar o desenvolvimento da ferramenta considerou-se algumas opções de tecnologias disponíveis, necessitando então de uma análise previa.

3.4.1 Análise

Entre os dispositivos moveis modernos tem-se uma grande variedade de arquiteturas e sistemas operacionais, no entanto alguns ganham destaque como: o Android, sistema operacional baseado no kernel linux e atualmente mantido pela empresa Google, o IOS sistema operacional proprietário usado nos produtos Apple e o Windows Phone, desenvolvido e mantido pela Microsoft. Segundo pesquisas realizadas pela Gartner, em 2016 o Android estava presente na maior parte dos usuários finais, representando 86.2% do mercado com mais de 296 milhões de unidades vendidas, seguido de 12.9% do iOS e 0.6% do Windows Phone. [7] A partir disso o sistema Android será o objeto de estudo base para a esta pesquisa, fazendo necessário então, o reconhecimento e estudo das técnicas e ferramentas utilizadas para o desenvolvimento nesta plataforma.

Para o desenvolvimento Android diversas opções são encontradas e devido a sua popularidade novas maneiras surgem a cada dia, sua linguagem nativa para programação é a linguagem Java, porem podemos citar alternativas que se destacam, como:

- Mono para Android, plataforma de desenvolvimento *Open source* baseada no *framework* .NET permitindo a utilização de diversas linguagem (C#, VB 8, Java, Python, Ruby, Eiffel, F#, Oxygene).[11]
- Android NDK, conjunto de ferramentas que permite implementar partes do aplicativo usando linguagens de código nativo, como C e C ++. [12]
- Basic4Android, ferramenta para desenvolvimento Android nativo utilizando Visual Basic.[13]
- Apache Cordova, *framework* que permite o desenvolvimento para Android, Ios e WP, utilizando as linguagens HTML, CSS e JavaScript.[14]
- Appcelerator Titanium, *framework* de código aberto que permite a criação de aplicativos móveis nativos para as plataformas iOS, Android e Windows UWP a partir de um único código JavaScript.[15]
- Python for Android, possibilita desenvolvedores desta migrarem aplicativos em Python para a plataforma Android.

Entretanto mesmo muitas dessas alternativas tem como objetivo facilitar e aumentar a produtividade, o que pode simplificar suas funcionalidades impossibilitando ou dificultando o acesso a recursos nativos. Tendo em vista possíveis dificuldades a programação do aplicativo de estudo será feita de forma nativa utilizando a linguagem Java em conjunto com Android SDK. O que torna necessário para o avanço da pesquisa o estudo desta linguagem e dos componentes Android.

Dentro do processamento de imagens encontra-se também diferentes abordagens que devem ser compreendidas em conjunto como o estudo da biblioteca OpenCV. OpenCV (Open Source Computer Vision Library) é uma biblioteca de visão computacional e aprendizado de maquina de código aberto. OpenCV foi construído para fornecer uma infra-estrutura comum para aplicações de visão computacional e para acelerar o uso da percepção da máquina nos produtos comerciais. [16]

3.4.2 Desenvolvimento da ferramenta

A primeira etapa para o desenvolvimento do aplicativo Android, foi a integração da biblioteca opencv[16] na IDE Android Studio[17], todos os arquivos foram obtidos do

endereço web opencv.org[16] assim como o tutorial de instalação. A tela inicial da aplicação consiste em fornecer ao usuário opções para a aquisição da imagem a ser processada, contendo a opção de utilizar a câmera do dispositivo ou selecionar imagens previamente obtidas em sua memória interna ou externa. Além de um botão para configurar a cor da folha.

Para a tela de configuração deve-se abrir uma imagem a partir da galeria do dispositivo contendo o tipo de folha desejado. Nessa tela o usuário tem a possibilidade de aumentar, diminuir ou arrastar a imagem através de movimentos de toque para facilitar a visão dos detalhes. A seleção da a cor desejada é feita com um toque único na parte da imagem onde a cor se encontra. Apos a escolha da cores (parte saudável e com doença) se tem a opção de salvar ou não e prosseguir para aquisição das imagens

Figura 13 – Protótipo da tela de configuração

Com a aquisição da imagem uma classe do tipo *Leaf* é instanciada, ela recebe em seu construtor uma imagem do tipo *Bitmap*. Faz a conversão para Matriz que é usada pela biblioteca opencv[16] para suas operações e armazena em um atributo. Essa classe tem como função implementar os métodos responsáveis pela chamada dos métodos do opencv[16] pelo processamento da imagem propriamente dito.

O primeiro método chamado é o *resize*, que redimensiona a imagem para 10% de seu tamanho original. Com objetivo de otimizar o processamento já que diminuir o tamanho da imagem nesse nível não ocasiona perca de detalhes necessários para os passos seguintes. Com isso a imagem é convertida para escala de cinza utilizando o filtro próprio do opencv[16] chamado *cvtColor*.

Em seguida o método *splitIn4* é chamado e retorna um *ArrayList* java com quatro matrizes cada qual representando uma parte da imagem. Essa etapa é realizada para contornar problemas de iluminação citado anteriormente. O processo é feito instanciando 4 objetos do tipo Rect com as dimensões desejadas, assim Uma Matriz é criada passando a imagens original e o Retângulo onde a imagem será cortada.

O limiar então já pode ser aplicado, assim o método *threshold* do opencv é chamado e como parâmetro é especificado para usar o algorítimo de otsu nas 4 imagens. O resultado disso então é passado para a função *concatSplitedLeaf* que uni as partes utilizando as função do opencv[16] *hconcat* e *vconcat* que concatenam listas de matrizes de maneira horizontal e vertical.

Após a concatenação um filtro de mediana é aplicado, chamado no opencv[16] de *medianBlur*. E o resultado do processo até aqui representa a segmentação de fundo e da folha, essa imagem é salva em um atributo da classe.

Com essa imagem é identificado os objetos presentes, se tiver 2 objetos o menor é considerado o objeto de referência e o maior a folha, se essa configuração ocorrer já são calculados as áreas dos 2. O objeto de referência foi definido que deve ter $4cm^2$ com esses dados é atualizado a função que converte a área para centímetros quadrados. Após isso o objeto de referencia é removido da imagem deixando apenas a folha e passa para próxima etapa. Caso seja identificado apenas um objeto, ele é considerado a folha e passa para o próximo passo.

Com base na imagem de fundo (imagem 1) obtida é chamado então o método *removeBackground* que utiliza a imagem original colorida da folha (imagem 2) e a imagem 1 para remover o fundo da imagem 2. O procedimento é realizado percorrendo a imagem 1 e atribuindo zero na imagem 2 em todos os pixeis correspondentes ao fundo na imagem 1 resultando em uma imagem colorida contendo apenas a folha.

Figura 14 – Imagem colorida sem fundo

O próximo passo então é separar a folha saudável da folha doente nessa imagem resultante. A cor obtida no processo de configuração é usado aqui, com o método *segmentLeafAndDisease* que percorre a imagem colorida sem fundo verificando a proximidade do

pixel com a cor que está salva na configuração, se a cor for próxima é considerada folha se for distante é considera doença. Nessa função também são contabilizados a área de cada parte.

A área obtida então passa por uma função de conversão para centímetros quadrados caso a opção de objeto de referência foi fornecida e então é salva em um atributo da classe. Caso não tenha sido fornecido o objeto de referencia, a área é salva em seu valor bruto. As áreas são então acessadas na classe *ResultActivity*, que configura a tela que exibe os resultados, por seus métodos *get* específicos.

Para o armazenamento temporário das informações da ferramenta uma classe herdeira da Classe *Application* do Android nomeada de *ApplicationData* é criada. A classe *Application* é usada pelo Android para manter estados globais da aplicação, e pode ser acessada através do método estático *getApplicationContext*. Dados como as imagens obtidas da etapa de aquisição, cor e resultados são armazenados por ora nessa classe para simplificar a implementação e facilitar a comunicação entre as telas.

As classe *CameraActivity* e *CameraPreview* foram adaptados dos exemplos dados pelo android[18] e implementam as funções para o uso da câmera do dispositivo e a exibição da mesma, assim como a requisição de permissões e captura e armazenamento da foto.

A implementação da ferramenta em seu desenvolvimento atual pode ser melhor compreendida através de um diagrama de classes apresentado abaixo:

Nos métodos onde se faz necessário a manipulação direta nos pixels, usando o opencv pode ser feito através dos métodos get e put, como por exemplo:

```
double[] pixel = img.get(linha,coluna);

pixel = [0,0,0];

img.put(linha,coluna, pixel);
```

No entanto ao usar esse método para percorrer uma imagem, a performance pode ser prejudicada. Uma forma melhor de manipular a imagem é representar-la em um array de bytes para alterar seus valores, para isso deve se criar um array do tamanho da quantidade de pixels multiplicado pela quantidade de canais que a imagem possui:

```
byte[] buff_img = new byte[(int)(img.total() * img.channels())];
```

Em seguida usar o método get passando o array para pagar todos os valores da imagem começando do ponto (0,0):

```
img.get(0, 0, buff_img);
```

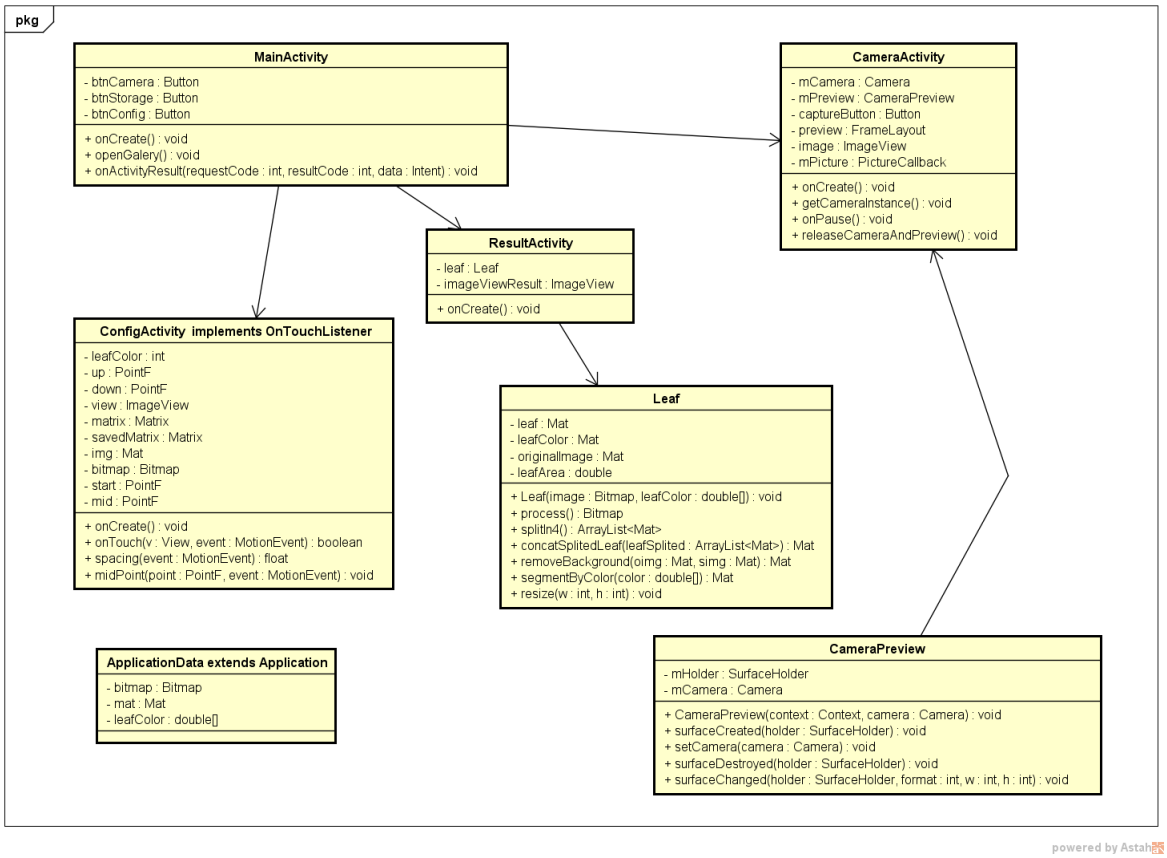

Figura 15 – Diagrama de classe da ferramenta

Para percorrer esse array deve se atentar que cada pixel ocupa uma posição para cada canal, se uma imagem tem 3 canais e 4 pixels seu array tera 12 posições, sendo: [p1r, p1g, p1b, p2r, p2g, p2b, p3r, p3g, p3b, p4r, p4g, p4b], onde o primeiro pixel ocupa as 3 primeiras posições o segundo as posições 4, 5, 6 e assim por diante.

Este processo otimiza o processamento em até 10 vezes, sendo essencial em aplicações moveis.

4 VALIDAÇÃO

Para a validação da ferramenta foi utilizado um banco de imagens cedido por uma pesquisa realizada pelo NITEC. A pesquisa em questão teve como objetivo analisar os sintomas de fitotoxidez de óleo mineral na cultura da soja. A Fitotoxidez é a concentração excessiva do ingrediente ativo na superfície da folha, causando queima ou destruição das células.

A determinação da área foliar de cada folha foi feita a partir do sistema análise de imagem WinDIAS 3® (Delta – T Devices Ltd) que permite através de escalas de cores criadas manualmente e posteriormente detectadas na planta, medir a área foliar de um folíolo e também o seu percentual comprometido por danos.

As mesmas imagem desta pesquisa, foram processadas com a ferramenta proposta e foram gerados para cada folha os mesmos dados já obtidos, com a diferença que as áreas obtidas com a ferramenta não estarem em cm^2. Isso se deve ao fato dessas imagens não possuírem o objeto de referencia necessário para ferramenta converter em centímetros. No entanto uma correlação entre os dois dados pode ser criada de qualquer forma.

Figura 16 – Exemplo folha usada na validação

Para aquisição as folhas foram retiradas das plantas e as imagens coletadas em 2 dias, no mesmo dia da retirada e no dia seguinte, totalizando 64 imagem. O resultado para área total da folha quando comparado com o software Windias 3, tanto para o primeiro quanto para o segundo dia apresentaram forte correlação com um R=0.9998 para os dois dias.

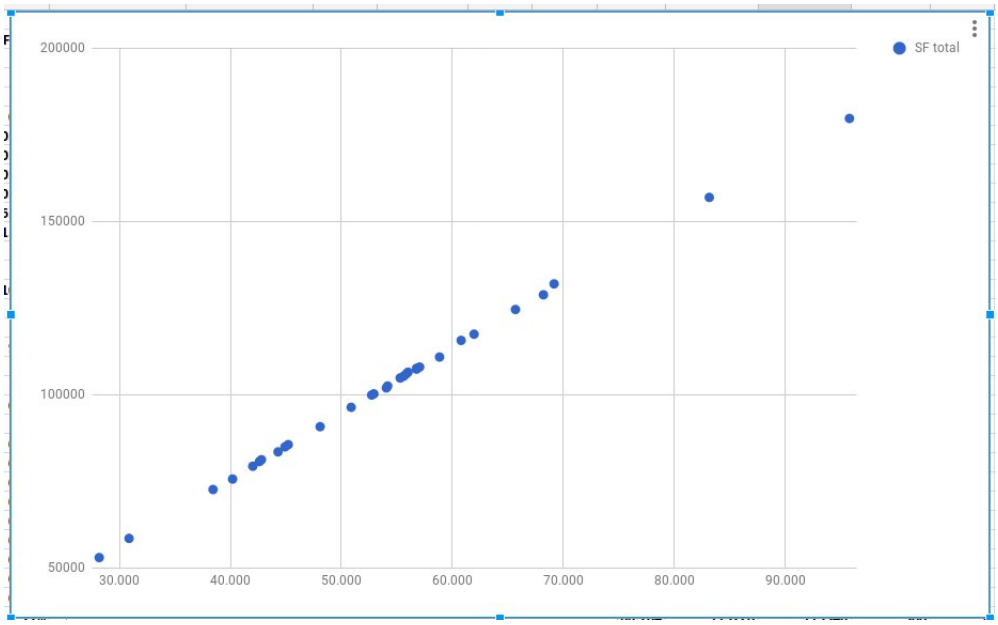

Figura 17 – Gráfico de dispersão entre resultado da área total ferramenta e software windias 3 - dia 1

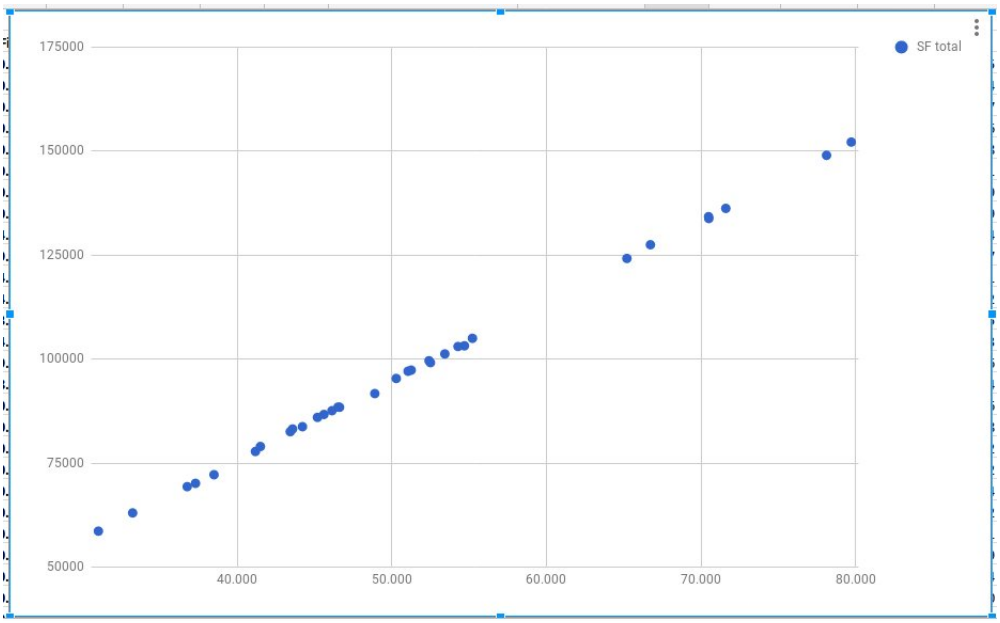

Figura 18 – Gráfico de dispersão entre resultado da área total ferramenta e software windias 3 - dia2

Já para a área danificada, as ferramentas apresentam resultados mais distintos com um R=0.9177 no primeiro dia e R=0.9182 no segundo dia. Mesmo ainda tendo uma forte relação entre os resultados, a ferramenta proposta teve a tendencia de calcular no geral um percentual danificado menor que o apresentado pelo software Windias 3.

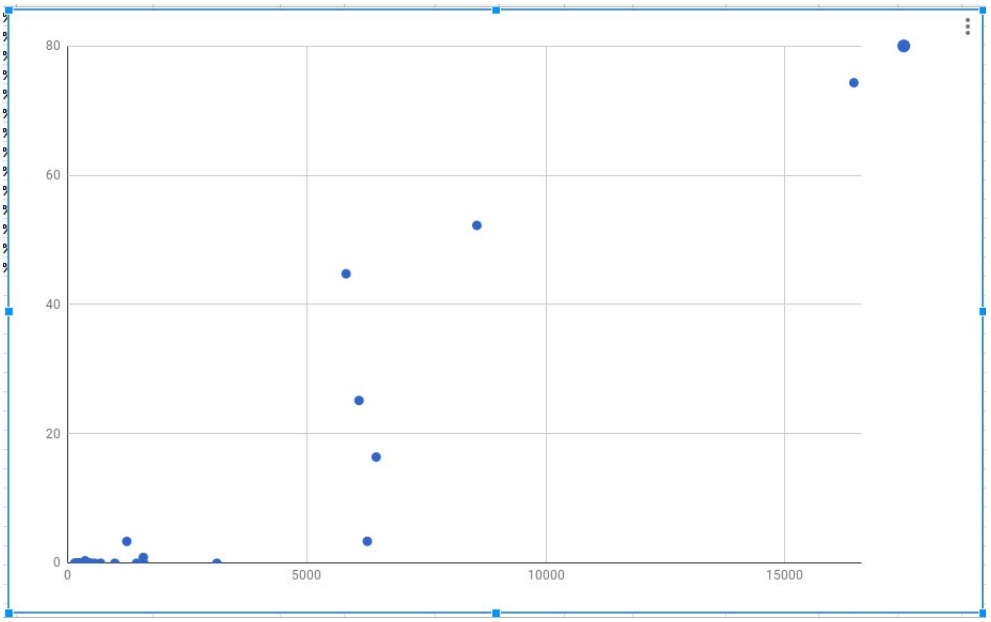

Figura 19 – Gráfico de dispersão entre resultado da área danificada ferramenta e software Windias 3 - dia 1

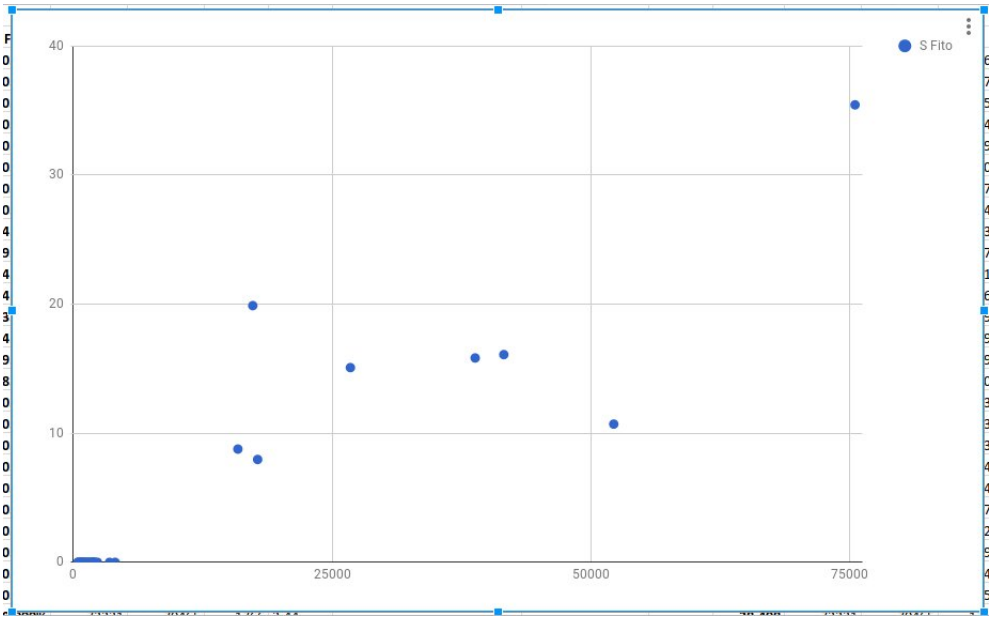

Figura 20 – Gráfico de dispersão entre resultado da área danificada ferramenta e software Windias 3 - dia2

Algumas justificativas podem ser levantada para entender esse resultado, como o baixo numero de amostras, já que apenas 8 das 32 folhas coletadas apresentaram danos visíveis. E o fato do dano estudado na pesquisa utilizada muitas vezes apresentar a mesma tonalidade da folha saudável, o que não foi considerado para a ferramenta proposta.

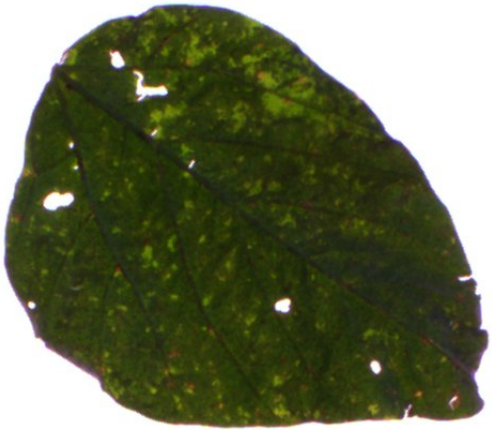

Figura 21 – Folha área danificada possui mesma tonalidade da folha

No entanto foi verificado também, a partir das imagens usadas que o software Windias 3 não apresenta resultados 100% condizentes com a realidade, mesmo podendo ser configurados com varias escalas de cores que representam cada elemento, ainda pode se encontrar entre as imagens usadas algumas processadas de forma incorreta.

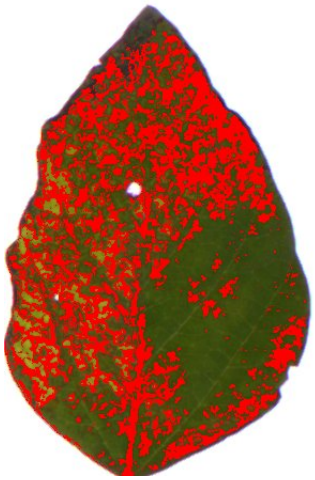

Figura 22 – Exemplo 1 de folha processada incorretamente pelo software Windias 3 (Em vermelho o que o software considerou como dano)

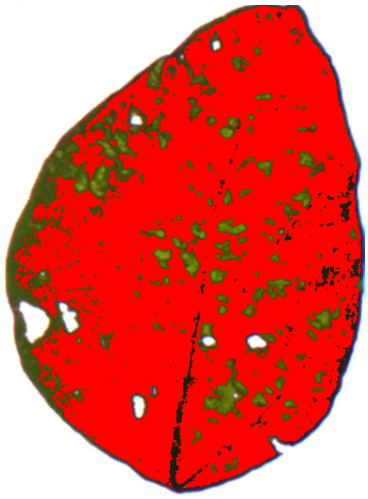

Figura 23 – Exemplo 2 de folha processada incorretamente pelo software Windias 3 (Em vermelho o que o software considerou como dano)

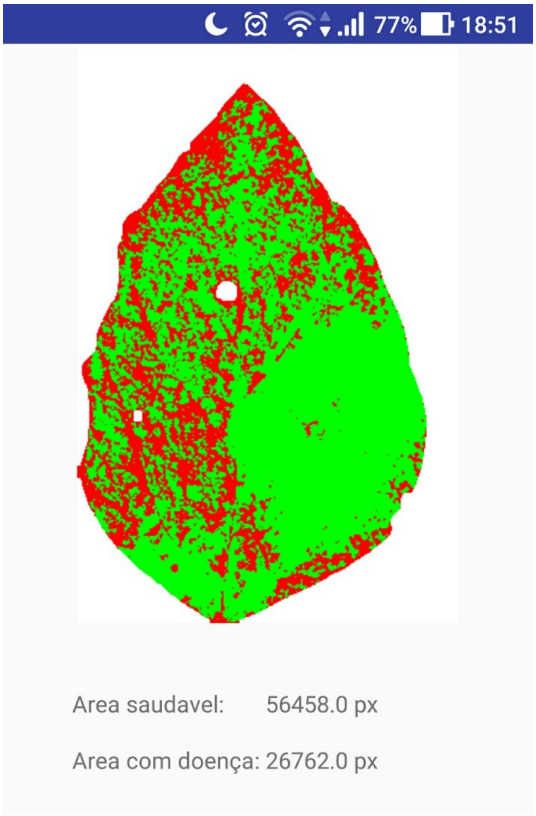

Figura 24 – Exemplo 1 de folha processada pela ferramenta.

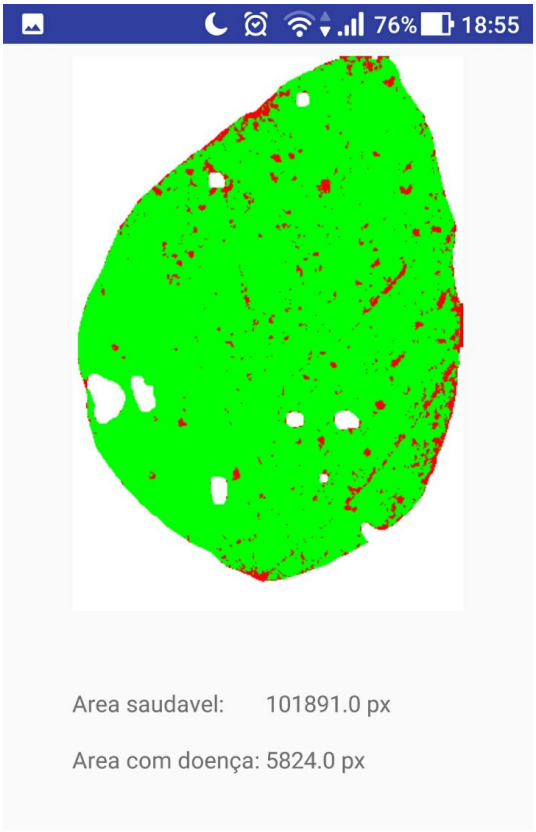

Figura 25 – Exemplo 2 de folha processada pela ferramenta.

5 CONCLUSÃO

Este trabalho desenvolveu um método capaz de extrair informações de uma determinada folha e implementou uma ferramenta móvel em plataforma *Android* para aplicar esse método, as informações extraídas foram a área total da folha, área danificada e porcentagem danificada da folha.

Os resultados mostraram que a ferramenta consegue calcular a área total com dados quase 100% correlatos com o *software* que é considerado o padrão para este fim, e que realiza funções similares. Isso mostra que a ferramenta tem grande potencial de mercado, Por tornar o processo mais simples, mais pratico e com um custo consideravelmente reduzido. No entanto a ferramenta apresentou um pouco de deficiência para segmentar corretamente a parte danificada da folha em alguns casos, evidenciando que está etapa necessita de ajustes para que a ferramenta seja lançada no mercado.

Para trabalhos futuros espera-se que novas abordagens sejam testadas na etapa de identificação da área danificada, com intuito de aumentar as diferentes doenças que a ferramenta consegue identificar. Com isso se propõe que diferentes folhas com diferentes tipo de danos sejam utilizadas nos testes e na validação. Também a comparação dos dados deve ser realizada com imagens que foram segmentadas manualmente, para representar os dados reais observados, tendo em vista que o software utilizado também pode apresentar erros em seus resultados.

REFERÊNCIAS

[1] GOOGLE. *about android*. 2017. Disponível em: <https://developer.android.com/about/android.html>.

[2] OTSU, N. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, IEEE, v. 9, n. 1, p. 62–66, 1979.

[3] GONZALEZ, R.; WOODS, R. *Processamento Digital de Imagens*. [S.l.: s.n.], 2009. ISBN 9788576054016.

[4] THABET, R.; MAHMOUDI, R.; BEDOUI, M. H. Image processing on mobile devices: An overview. In: IEEE. *Image Processing, Applications and Systems Conference (IPAS), 2014 First International*. [S.l.], 2014. p. 1–8.

[5] UMBAUGH, S. E. *DIGITAL IMAGE PROCESSING AND ANALYSIS*. Taylor Francis Group LLC, 2010. ISBN 0. Disponível em: <https://books.google.com.br/books?id=r5f0RgAACAAJ>.

[6] ACHARYA, T.; RAY, A. K. *Image Processing - Principles and Applications*. [S.l.]: Wiley-Interscience, 2005. ISBN 0471719986.

[7] GARTNER, I. *Gartner Says Five of Top 10 Worldwide Mobile Phone Vendors Increased Sales in Second Quarter of 2016*. 2016. Disponível em: <http://www.gartner.com/newsroom/id/3415117>.

[8] FIGUEIREDO, C. M.; NAKAMURA, E. Computação móvel: Novas oportunidades e novos desafios. *T&C Amazônia*, v. 1, n. 2, p. 21, 2003.

[9] MATEUS, G. R.; LOUREIRO, A. A. F. *Introdução à computação móvel*. [S.l.]: DCC/IM, COPPE/UFRJ, 1998.

[10] MONTEIRO, J. B. *Google Android: Crie aplicações para celulares e tablets*. [S.l.]: Editora Casa do Código, 2014.

[11] MONO. *mono*. 2017. Disponível em: <http://www.monodevelop.com/>.

[12] GOOGLE. *Android ndk*. 2017. Disponível em: <https://developer.android.com/ndk/index.html?hl=pt-br>.

[13] B4X. *about*. 2017. Disponível em: <https://www.b4x.com/>.

[14] CORDOVA. *about*. 2017. Disponível em: <https://cordova.apache.org/docs/en/latest/guide/platforms/android/>.

[15] APPCELERATOR. *about*. 2017. Disponível em: <http://www.appcelerator.com/>.

[16] OPENCV.ORG. *About OpenCV*. 2017. Disponível em: <http://opencv.org/about.html>.

[17] ANDROID. *about android studio*. 2017. Disponível em: <https://developer.android.com/studio/index.html>.

[18] ANDROID. *about*. 2017. Disponível em: <https://developer.android.com/develop/index.html>.