UNIVERSIDADE ESTADUAL DE PONTA GROSSA

PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS- GRADUAÇÃO EM COMPUTAÇÃO APLICADA

MARCOS MONTEIRO JUNIOR

MÉTODO DE CÁLCULO DE TRAJETÓRIA DE MÁQUINAS AGRÍCOLAS UTILIZANDO PROCESSAMENTO DE IMAGENS EM SMARTPHONES

PONTA GROSSA
2015

MARCOS MONTEIRO JUNIOR

MÉTODO DE CÁLCULO DE TRAJETÓRIA DE MÁQUINAS AGRÍCOLAS UTILIZANDO PROCESSAMENTO DE IMAGENS EM SMARTPHONES

Dissertação apresentada ao Programa de Pós Graduação em Computação Aplicada da UEPG como requisito para a obtenção do título de Mestre em Computação Aplicada

Orientadora: Maria Salete Marcon Gomes Vaz

Co-Orientador: José Carlos Ferreira da Rocha

PONTA GROSSA

2015

Ficha Catalográfica
Elaborada pelo Setor de Tratamento da Informação BICEN/UEPG

Monteiro Junior, Marcos
M775 Método de cálculo de trajetória de máquinas agrícolas utilizando processamento de imagens em smartphones/ Marcos Monteiro Junior. Ponta Grossa, 2015.
58f.

Dissertação (Mestrado em Computação Aplicada - Área de Concentração: Agricultura), Universidade Estadual de Ponta Grossa.
Orientadora: Profª Drª Maria Salete Marcon Gomes Vaz.
Coorientadora: Profª Drª José Carlos Ferreira da Rocha.

1.Android. 2.Processamento de imagem. 3.Automação agrícola. 4.IOIO. I.Vaz, Maria Salete Marcon Gomes. II. Rocha, José Carlos Ferreira da. III. Universidade Estadual de Ponta Grossa. Mestrado em Computação Aplicada. IV. T.

CDD: 006.3

TERMO DE APROVAÇÃO

Marcos Monteiro Júnior

"MÉTODO DE CÁLCULO DE TRAJETÓRIA DE MÁQUINAS AGRÍCOLAS UTILIZANDO PROCESSAMENTO DE IMAGENS EM SMARTPHONES"

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre no Programa de Pós-Graduação em Computação Aplicada da Universidade Estadual de Ponta Grossa, pela seguinte banca examinadora:

Orientadora: *Dr. Maria Salete Marcon Gomes Vaz*
UEPG

Dr. Alceu de Souza Britto Junior
UEPG

Dr. Pedro Henrique Weirich Neto
UEPG

Ponta Grossa, 07 de agosto de 2015.

A ousadia do fazer é que abre o campo do possível. E é o fazer – com seus erros e acertos – que possibilita a construção de algo consistente.

Pedro Benjamin Garcia

Dedico aos meus pais

Marcos e Fanny

AGRADECIMENTOS

- Agradeço primeiramente a Deus.
- Aos meus familiares, em especial meus pais pelo apoio e incentivo.
- À minha amiga e companheira Tatiane Skeika, por todo apoio e por permanecer ao meu lado em todos os momentos.
- À minha Orientadora Maria Salete Marcon Gomes Vaz e ao Meu Co-Orientador José Carlos Ferreira da Rocha
- Ao professor Ariangelo Hauer Dias, por ter feito parte desse projeto como Orientador durante um grande período.
- Ao amigo Luíz Fernando Lirani Filho pela possibilidade de análise em Campo em sua propriedade.
- Aos professores e colegas da Pós Graduação em Computação Aplicada.
- À todos que direta ou indiretamente contribuíram para a realização desse Trabalho.

RESUMO

Máquinas agrícolas possuem mecanismos que as tornam autônomas, porém ainda é um recurso caro, baseado em GPS. O uso de visão computacional é uma alternativa ou um complemento para o uso do GPS. Até há pouco tempo, o uso de visão computacional, era restrito aos computadores de grande capacidade de processamento e seu uso em máquinas agrícolas era inviável devido às condições adversas do campo. Com a evolução dos processadores é possível aplicar visão computacional em celulares, cujo *hardware* é robusto, por não possuir componentes mecânicos e por ser tolerantes à poeira, e por alguns modelos até a umidade. Esta dissertação descreve a criação de um método para dispositivos móveis, do tipo smartphone, com sistema operacional *Android*, com a finalidade de calcular a trajetória de máquinas agrícolas, ou robôs, em linhas de pulverização. O método utiliza a câmera presente nos smartphones para captação da imagem da rota a ser calculada e processada pelo telefone. O sistema incorpora métodos de visão computacional, com o auxílio de um algoritmo, para suavizar os movimentos, além de colaborar para tomada de decisão de forma a não ocorrer movimentos desnecessários. O método usa *software* de código aberto, como a Biblioteca *OpenCV*, o Sistema *Android*, e as ferramentas para programação, e em *hardware* com a Plataforma IOIO. O sistema foi testado em campo, nas culturas de trigo e de soja, sobre um sistema de plantio direto. Os testes foram realizados em um Robô denominado NAVIGO, desenvolvido no Programa de Pós-Graduação de Computação Aplicada, da Universidade Estadual de Ponta Grossa. O smartphone fica acoplado no NAVIGO, com comunicação com IOIO, através de *Bluetooth*. O processamento de visão computacional foi realizado no smartphone, obtendo-se resultados satisfatórios, provando que os telefones inteligentes, são robustos, e possuem a vantagem de ter inúmeros sensores embutidos no *hardware* e são capazes de realizar tarefas que antes eram exclusivas de computadores. Além disso dispositivos de porte pequeno que utilizam a visão computacional, como o proposto no trabalho, podem ser ótimas ferramentas na agricultura em locais de difícil acesso de maquinário de grande porte.

ABSTRACT

Agricultural machinery has mechanisms which automate them; however, it is still an expensive GPS based resource. The use of computational vision is an alternative or an addition to the use of GPS. Not long ago, the use of computational vision was exclusive to computers with huge processing capabilities, and its use in agricultural machinery was impracticable due to adverse field conditions. The evolution of the processors made computational vision possible in cellphones, whose hardware is robust for not having mechanical components and for being dust and, some models, humidity proof. This work describes the creation of a method developed for smartphone based mobile devices with the Android operational system. This system has the purpose of providing the calculation of the trajectories of agricultural machinery or robots, at pulverization lines. The method uses cameras that are present on the smartphones themselves in order to capture the image of the route to be calculated and processed by the phone. The process uses methods of computational vision with the aid of an algorithm to smooth the movements and to take decisions in order to not perform unnecessary movements. The method uses open-source softwares, like the Openvc library, the Android system and its tools of the programming, and in the IOIO hardware platform. The system was field tested, in a robot named NAVIGO, developed in the graduation program of Applied Computation at the Universidade Estadual de Ponta Grossa. The smartphone is coupled to the NAVIGO and it communicates with IOIO through Bluetooth. . Computer vision processing was performed on the smartphone , obtaining satisfactory results , proving that smart phones , are robust , and have the advantage of having numerous sensors embedded in the hardware and are able to perform tasks that were previously exclusive to computers. Furthermore small devices that use computer vision , as proposed in the work can be great tools in agriculture large areas of difficult access machinery .

Sumário

1 Introdução 11

2 Revisão bibliográfica 14

2.1 Visão computacional e processamento de imagens 14

2.2 Bibliotecas *OpenCV* 15

2.3 Modelo de cores RGB 16

2.4 Modelo de cores HSV 17

2.5 Conversões 18

2.6 Pirâmide de Imagens – *pyrDown* 19

2.7 SDK - Tegra *Android* Development Pack 21

2.8 Sensores 21

2.9 Android 22

2.10 IOIO 23

2.11 Agricultura de Precisão 25

3 TRABALHOS RELACIONADOS 26

3.1 NAVIGO 32

4 MATERIAIS E MÉTODOS 34

4.1 Primeira Etapa: Delimitação do Trajeto 35

4.2 Segunda Etapa: Aplicação do Sistema no NAVIGO 43

5 Resultados e Discussões 45

6 Conclusões e perspectivas de trabalhos futuros 54

7 Bibliografia 55

ÍNDICE DE FIGURAS

Figura 1. Representação da sequência do processamento de imagens 15

Figura 2. Mistura das três cores do sistema RGB 16

Figura 3. Representação cúbica do sistema de cores RGB..... 17

Figura 4. Representação cônica do sistema de cores HSV..... 18

Figura 5. Representação de uma pirâmide de imagens 20

Figura 6. Representação da fase final do método pyrDown..... 20

Figura 7. Camadas do sistema *Android*..... 23

Figura 8. Representação da placa IOIO..... 24

Figura 9. Ciclo PWM de 50% 25

Figura 10. Reconhecimento da linha de plantas..... 27

Figura 11. Robô agrícola 28

Figura 12 Etapas para delimitar área de atuação da máquina agrícola..... 29

Figura 13. Etapas do processamento de imagens 29

Figura 14 Funcionamento do processamento de imagens do robô..... 30

Figura 15. Processamento parcial de imagens na água 31

Figura 16. Processamento de imagens para determinar a distância entre linhas de plantio 32

Figura 17. NAVIGO, visão superior. 33

Figura 18. Diagrama da interação com o sistema e o usuário..... 34

Figura 19 Exemplo da matriz *spectrum*..... 35

Figura 20. Resultado da representação das matrizes adquiridas ao longo do programa 36

Figura 21. Visão da tela do *smartphone* para o usuário 38

Figura 22. Matriz *pyrDownMat* com os trajetos e as bordas traçadas..... 38

Figura 23. Gráfico dos termos linguísticos para as regiões de CIMA e de BAIXO. 39

Figura 24 posição do servo como caixa de direção do NAVIGO 44

Figura 25. Representação das matrizes adquiridas ao longo do programa em campo ... 45

Figura 26. Saída para o usuário 46

Figura 27. Mesa coberta com papel cinza, indicando o caminho do robô, e dois panos verdes, indicando as bordas. 46

Figura 28. Suporte do celular 47

Figura 29. Saída para *HUE* 159 47

Figura 30. Matrizes *dilatedMask* para as imagens da Figura 29 48

Figura 31. Saída para *Hue* 172 48

Figura 32. Matrizes *dilatedMask* para as imagens da Figura 31 48

Figura 33 Resultados do delineamento de borda em campo 50

Figura 34. Teste em Campo NAVIGO. 51

Figura 35. Etapas para obtenção de resultados 51

Figura 36. Contagem de pixels entre bordas 52

Figura 37. Teste em campo, método do ponto médio. 52

ÍNDICE DE TABELAS

Tabela 1. Regras utilizadas no algoritmo de detecção de bordas para suavizar o movimento. 40

Tabela 2. Tabela de valores para cada termo linguístico 40

Tabela 3. Regras da borda esquerda. 41

Tabela 4. Regras da borda direita. 42

Tabela 5. Agrupamento entre CIMA e BAIXO 42

Tabela 6. Correlação da saída do algoritmo de detecção de bordas, com o ângulo da direção do NAVIGO 44

Tabela 7. Saída apresentas pelo algoritmo de Xue, Zhang e Grift (2012)para as detecções de bordas apresentadas na Figura 29 49

Tabela 8. Saída apresentas pelo algoritmo de Xue, Zhang e Grift (2012) para as detecções de bordas apresentadas na Figura 31 49

LISTA DE ABREVIATURAS

AP - Agricultura de Precisão

GPS - Sistema de Posicionamento Global

PC - Personal Computer

PDI - Processamento de imagens

PWM - (*pulse width modulation*)

RGB - *Red, Green, Blue*; (Vermelho, Verde, Azul)

VC - Visão computacional

1 INTRODUÇÃO

As pesquisas de gestões agrícolas e as novas tecnologias estão presentes desde a semeadura ou o plantio, até o armazenamento e distribuição dos produtos. Muitas pesquisas direcionam a computação para proporcionar autonomia em relação ao ser humano, tornando os processos automatizados. Uma das formas de aumentar essa autonomia acontece através da análise do ambiente agrícola utilizando sensores aliados a robótica.

Dentre as divisões da robótica, pode-se destacar o uso de visão computacional para guiar máquinas, robôs, ou até mesmo carros, chamados carros autônomos. Na agricultura o processamento de imagens, começou com o sensoriamento remoto, onde fotografias aéreas eram utilizadas para o planejamento agrícola, mapeamento de solos e previsão de safra (MERCADANTE, 2007).

Aplicações de processamento de imagens para agricultura foram desenvolvidas, como detecção de ataques por pragas ou detecção de doenças. O uso tem inúmeras vantagens sobre os métodos convencionais. Dentre as vantagens destacam-se a integração com procedimentos automáticos, a execução de medições com maior grau de acurácia e consistência que seres humanos, o monitoramento em tempo integral, além de medir a cor e a morfologia de objetos de maneira objetiva, visto que seres humanos fazem de maneira subjetiva (JAYAS, PALIWAL e VISEN, 2000).

Pesquisas relacionadas à direção autônoma de máquinas agrícolas trazem inúmeros benefícios para os agricultores, como economia de combustível, melhoria do aproveitamento da colheita, semeadura, e aplicação de insumos.

A navegação baseada em visão computacional de veículos agrícolas teve ampla cobertura na literatura, resultando no desenvolvimento de veículos autônomos (ASTRAND e BAERVELDT, 2005). A utilização de tecnologia tem como finalidade auxiliar o uso do GPS - *Global Positioning System*. Uma das alternativas que pode ser usada para vincular imagem e GPS seria a aplicação de *smartphones* como tecnologia inovadora no campo.

Os telefones inteligentes, tais como *smartphones*, são sistemas embarcados com arquitetura de *hardware* diferente de computadores, porém possui velocidade de *clock*,

quantidade de núcleos em seus processadores e placa de vídeo, capazes de processar grande quantidade de dados e informações, assim como os processadores comerciais para computadores domésticos. No início, seu uso em pesquisas foi restrito à coleta de dados para posterior processamento em computadores pessoais (MEHMOOD, TUFAIL, *et al.*, 2014).

Os *smartphones* possuem a vantagem de ter inúmeros sensores embutidos em seu *hardware*, como câmera, GPS, acelerômetro, sensores de magnetismo, etc. Há *hardware* dedicado a comunicação, como o *wifi*, *bluetooth*, conexão 2G/3G/4G (dependendo do aparelho), infravermelho etc., permitindo utilização como sistema embarcado completo e compacto (USTEV, DURMAZ e C., 2013).

Um dos sistemas operacionais presentes em *smartphone* é o *Android*. O *Android* é um sistema de código aberto, que possui maior flexibilidade na utilização do *hardware* em relação aos outros sistemas. Ainda possui suporte para aplicação de tecnologias de processamento de imagens, no caso a *OpenCV*, além de possuir ferramentas para desenvolvimento gratuitas, e sistema de simulação de *hardware*.

Existem vários *softwares* de desenvolvimento para o sistema operacional *Android*, dentre eles, o *Tegra Android Development Pack2.0*, contendo a biblioteca de código aberto *OpenCV*, e segundo Bradsky (2008) é amplamente utilizada para processamento de imagens. De acordo com *Tegra* (2013) apesar de este pacote ter sido criado para o desenvolvimento de aplicativos em dispositivos com a tecnologia *Tegra* da *NVIDIA*, ele configura o ambiente de programação para trabalho em qualquer dispositivo *Android*.

Essa ferramenta e o uso do sistema *Android*, permitem utilizar os sensores do *smartphone*, processar informações e comunicar com *hardware* externo. Esses *hardwares* são capazes de controlar atuadores, e com circuitos auxiliares é possível controlar sistemas de diferentes potências, expandindo ainda mais a capacidade de atuação do *smartphone.*

Existem diversos *hardwares* externos com a finalidade de comunicar-se com o sistema *Android*. O IOIO é uma placa dedicada para essa finalidade. Trata-se de uma plataforma de desenvolvimento baseada no microcontrolador PIC da empresa Microchip.

Possui seu *hardware* aberto e livre para modificações, comunicação USB, e ou *Bluetooth*, além de bibliotecas para comunicação com o *Android*.

Diante disto, esta dissertação apresenta o estudo e criação um método para *smartphones* com sistema operacional *Android*, que utiliza a tecnologia de processamento de imagens e a aplicação de um algoritmo para fazer com que um robô mantenha-se em um determinado trajeto com a utilização da câmera do dispositivo para a captação de imagens. Depois dessa etapa o sistema foi implantado em um protótipo, e testado em ambiente agrícola, nas culturas de trigo e cultura da soja.

Esse trabalho utiliza a robustez dos *smartphones* como alternativa para o uso de sistemas embarcados e sistemas que fazem o uso de computadores, para controle de máquinas agrícolas. Também, mostra a execução do processamento de imagens em *smartphones Android*, para trafegar em linhas de pulverizador, hortas, linhas de plantio e até entre árvores na fruticultura. Desta maneira, é possível mostrar que a alta tecnologia presente em *smartphones*, combinada com o uso da visão computacional consegue guiar o protótipo, simulando uma máquina agrícola.

Essas aplicações permitem diversos trabalhos relacionados à pulverização localizada, poda de árvores, monitoramento de lavouras e hortas, extração de solo sobre linhas de plantio, etc. Além de auxiliar controladores existentes baseados em GPS, cujas estruturas não são capazes de identificar falhas no terreno.

O item 2 desse trabalho apresenta a revisão bibliográfica sobre o tema, onde são descritos os conceitos básicos, tecnologias empregadas, histórico da robótica na agricultura e alguns trabalhos relacionados à pesquisa apresentada.

No item 3 é apresentado o desenvolvimento do método baseado em Sistema Operacional *Android*, com aplicação de sistemas de visão computacional. A aplicação do sistema foi feita em um protótipo denominado NAVIGO. Já no item 4 são apresentados os resultados dos testes do sistema em laboratório e em campo. Finalmente, no item 5, são apresentadas as conclusões e perspectivas de trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

2.1 Visão computacional e processamento de imagens

A capacidade de interpretar imagens e vídeos possibilita inúmeras aplicações, como análise automática de uma cena, e aprimoramento de análises dos dados extraídos de uma imagem são aplicações voltadas para o Processamento Digital de Imagens.

O Processamento Digital de Imagens teve seu início com o advento dos primeiros computadores digitais de grande porte e o início do programa espacial norte-americano. As primeiras técnicas utilizadas no para aprimoramento de imagens digitais teve início no *Jet Propulsion Laboratory* (Pasadena, Califórnia - EUA) em 1964. As técnicas usadas serviam para realçar e restaurar imagens utilizadas em programas espaciais (MARQUES e N., 1999).

Processamento de imagens é uma área da computação destinada à operação de imagens e processamento, aplicando a computação de forma interpretativa, a fim de extrair dados de imagens e vídeos (FORSYTH e PONCE, 2003).

Visão computacional é transformar dados de imagens 2D/3D ou vídeos em uma nova representação para um determinado objetivo, ou seja, tirar conclusões dos dados obtidos do processamento das imagens. Esses resultados podem conter informações contextuais como “a distância entre dois objetos”, “números de células de um tecido animal”, entre outras (BRADSKY e KAEBLER, 2008).

Os problemas a serem analisados e solucionados pela visão computacional seguem uma organização, com etapas sequenciais de aplicação como segue: aquisição, pré-processamento, segmentação, estação de características e reconhecimento. Essa sequência faz parte da integração de processos e representação gráfica para o processamento de imagens (GONZAGA, 2010).

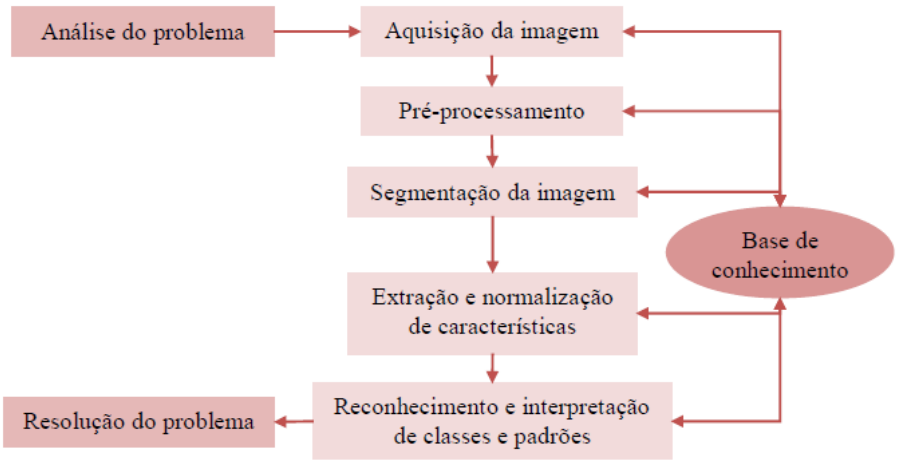


Figura 1. Representação da sequência do processamento de imagens
Fonte: (CÁSSIO, 2011).

A Figura 1 representa a sequência para manipulação de imagens e aplicação da visão computacional. Ainda, existe subdivisão desses estágios, são elas: *visão de baixo nível* (aquisição e pré-processamento); *visão em nível intermediário* (segmentação, extração e normalização de características); *visão em alto nível* (classificação, reconhecimento e correspondência) (FORSYTH e PONCE, 2003).

2.2 Bibliotecas *OpenCV*

OpenCV são códigos cobertos pela licença BSD, gratuitos para uso acadêmico e comercial. As bibliotecas são implementadas em C++, C, Python e Java, dando suporte para os ambientes Windows, Linux, Mac OS, IOS, e *Android*. *OpenCV* foi desenvolvida com o foco em aplicações de tempo real. Foi escrita, inicialmente, em C/C++ e otimizada para essas linguagens. As bibliotecas podem utilizar recursos multicores. É uma biblioteca aplicada em diferentes áreas da visão computacional, tais como inspeções de produtos, segurança, exames médicos, configurações de câmeras, robótica, entre outras áreas (OPENCV, 2014).

Entre os objetivos da biblioteca *OpenCV* estaria fornecer uma infraestrutura de visão computacional de fácil usabilidade. Isto pode permitir ao usuário criar sofisticadas aplicações de forma ágil e eficiente.

2.3 Modelo de cores RGB

Modelo de cores RGB é baseado nas três cores aditivas primárias, ou seja, no vermelho, no verde e no azul - *red, green e blue*, em inglês. A soma dessas três cores, duas a duas, resulta nas secundárias também conhecidas como as cores subtrativas. São assim denominadas pelo fato de que as cores aditivas são obtidas através destas, pela subtração da cor branca por um par de cores subtrativas. (SOUTO, 2000).

Esse modelo foi descrito por Young-Helmholtz baseado na teoria de visão colorida tricromática, Young definiu a teoria do Espaço RGB, baseada no princípio de que diversos efeitos cromáticos são obtidos pela projeção da luz branca.

O desenvolvimento tecnológico de tubos catódicos permitiu a definição do display de cores ao invés de uma fosforescência monocromática, permitindo que as cores dos pixels fossem combinadas em vários valores do vermelho, verde e azul, no Modelo RGB de cores.

As cores vermelha, verde e azul, recebem valores que variam de 0 a 255 ou de 0 a 1. No sistema RGB, o valor (0, 0, 0) equivale à cor preta e o valor (255, 255, 255) ou (1, 1, 1), equivalente à cor branca. Neste intervalo, há uma combinação possível de 16,7 milhões de valores de cores (256 x 256 x 256). Quando há interseção das três cores primárias forma-se a luz branca (Figura 2).

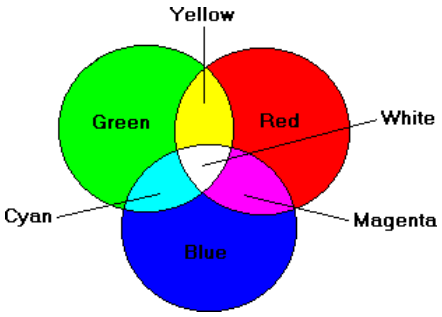


Figura 2. Mistura das três cores do sistema RGB

As cores secundárias são originadas a partir de combinações com as cores primárias, utilizando-se uma representação cúbica (Figura 3). Para obter qualquer tonalidade de cor é utilizada a combinação das cores secundárias.

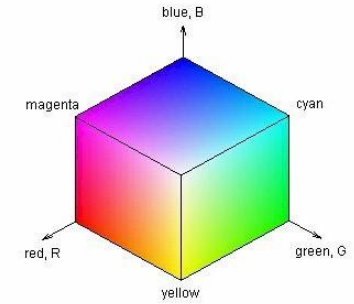


Figura 3. Representação cúbica do sistema de cores RGB

Para formar os tons de cinza, a diagonal principal contribui com as cores primárias.

2.4 Modelo de cores HSV

Munsell (1915) propôs o modelo HSV, o qual demonstra que a partir de uma mesma matiz, o ser humano discrimina ao menos 10 valores de saturação, bem como 10 valores de brilho, e dependendo do comprimento de onda, nove.

O modelo de cores HSV não se baseia nos valores computados do Modelo RGB e sim na percepção humana das cores. Elas são definidas pelos atributos de intensidade ou brilho, matiz e saturação. Simões *et al.*(2001) afirmam que nesse sistema a tonalidade é expressa em termos de um ângulo. A saturação varia ao longo do raio do ângulo e o eixo *value,* ortogonal aos demais, relaciona-se à iluminação.

O Espaço HSV (*HUE*, *Saturation* e *Value*) de representação tridimensional pode ser representado como um cone (Figura 4), onde o eixo vertical central representa a intensidade (*value*). A matiz (*HUE*) é definida por um ângulo entre 0 e 360 graus com vértices separados em intervalos de 60 graus e baseia-se se no comprimento de onda de luz refletida de um objeto ou transmitida por ele. Cada vértice possui uma cor, sendo que o ângulo 0 é a cor vermelha; o 60, amarela; 120, verde; 180, ciano; 240, azul; 300 graus, a cor magenta e novamente nos 360 graus, a cor vermelha.

A saturação ou croma (Figura 4) é a quantidade de cinza existente em relação ao matiz, medida como uma porcentagem de 0% (cinza) a 100% (totalmente saturado). A saturação é a profundidade ou pureza da cor e tem como medida a distância radial em relação ao eixo central com valores normalizados entre 0 e 1. Quanto maior o valor de saturação, mais pura será a cor.

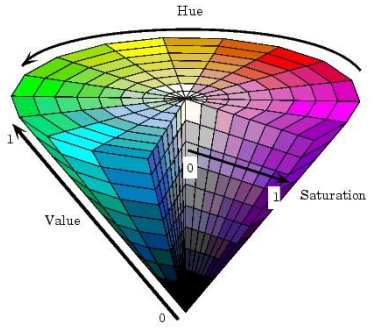


Figura 4. Representação cônica do sistema de cores HSV

O brilho (*value*) é a luminosidade relativa da cor ou a falta dela, podendo ser medida como uma porcentagem de 0% (preto) a 100% (branco). Para qualquer valor de intensidade e de matiz, se o valor da saturação variar do seu mínimo para o máximo, a alteração visualizada varia de uma cor mais escura (mais cor preta) e vai para a forma mais pura da cor representada pelo valor do matiz.

2.5 Conversões

Quando é estabelecido o Modelo RGB, deve-se levar em consideração que o método trabalha com a cor, brilho e intensidade, em conjunto. Já no Modelo HSV, ao ser selecionada uma determinada faixa de cor, utiliza-se apenas a matriz HUE, a qual verifica o tipo de cor selecionada, abrangendo todas as cores do seu espectro, independente do seu brilho e saturação. Com isso, faz-se necessário uma conversão do Modelo RGB para HSV. A Equação 1 a seguir mostra as funções necessárias para realizar a conversão entre os dois métodos.

$$H = \begin{cases} 60 * \dfrac{G - B}{MAX - MIN} + 0 \ \rightarrow if(MAX = R) e\ (G \ \geq B) \\ 60 * \dfrac{G - B}{MAX - MIN} + 360 \ \rightarrow if(MAX = R) e\ (G \ < B) \\ 60 * \dfrac{B - R}{MAX - MIN} + 120 \ \rightarrow if(MAX = G) \\ 60 * \dfrac{R - G}{MAX - MIN} + 240 \ \rightarrow if(MAX = B) \end{cases}$$

$$S = \frac{MAX - MIN}{MAX}$$

$$V = MAX$$

Equação 1. Conversão do sistema RGB para HSV

Onde, R = Red, G = Green, B = Blue, MAX = ângulo máximo, MIN = ângulo mínimo, H = HUE, S = Saturação e V = Value, MAX é o maior valor entre R, G e B, MIN é o menor valor entre R, G e B.

O Quadro 1 mostra um pseudocódigo da transformação do Modelo RGB para o Modelo HSV, seguindo como base as fórmulas definidas na Equação 1.

```
Procedimento RGB_para_HSV (r, g, b: real; h, s, v: real)
início
   max = maximo(r,g,b)
   min = minimo(r,g,b)
   v = max
   se (max não_igual 0) então
      s = (max-min)/max
   senão s = 0                          [ocorre quando r=b=g]
se (s igual 0) então
   h = indefinido
senão
   início bloco
      delta = max -  min
       se (r igual max) então
          h = (g -  b)/delta
       senão se (g igual max) então
          h = 2 + (b -  r)/delta
       senão se (b igual max) então
          h = 4+ (r -  g)/delta
       h = h * 60
       se (h menor_que 0) então
          h = h + 360
   fim bloco
fim
```

Quadro 1: Representação em forma de pseudocódigo das equações para transformação do sistema de cores RGB para HSV

Segundo SOUTO (2000), os resultados obtidos dão a tonalidade variando de 0º a 360º, indicando o ângulo no círculo onde a tonalidade (HUE) está definida, e a saturação e o brilho variando de 0 a 1, representando o menor e o maior valor possível.

2.6 Pirâmide de Imagens – *pyrDown*

Pirâmides de imagens são utilizadas em aplicações de visões computacionais. São classificadas como um conjunto de imagens, provenientes de uma imagem inicial, onde sua resolução é sucessivamente diminuída sobre um determinado ponto de parada, até o mesmo ser alcançado. Numa pirâmide de imagens, cada camada superior à outra é menor (Figura 5), de forma que as imagens das camadas inferiores sejam maiores.

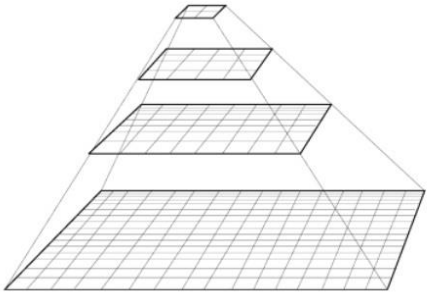


Figura 5. Representação de uma pirâmide de imagens

Para analisar uma imagem e decompô-la em partes torna-se útil para uma melhor análise e uma possível captura de detalhes para o propósito de uma seleção de coleta de bordas, de forma que não haja perda de informações de nenhuma parte. A teoria das pirâmides permite maneiras de decompor uma imagem em diversos níveis de resolução.

Existem dois tipos de métodos utilizados para alterar uma resolução de uma imagem e criar uma pirâmide de imagens: método da pirâmide Gaussiana e de Laplace. O primeiro é utilizado para diminuir a resolução sobre a imagem. Já a segunda, é utilizada para a reconstrução de uma imagem de baixa resolução na pirâmide.

O método que utiliza pirâmide Gaussiana é chamado de *pyrDown*, pela biblioteca *OpenCV*, e é descrito como um método em que borra-se a imagem e então seu tamanho é diminuído pela metade. Para borrar a imagem, a biblioteca utiliza-se de uma matriz de Gauss. Dessa forma, cada pixel da nova imagem é formado por parte dos 25 pixels presentes na imagem originária. Após a imagem ser borrada retiram-se as linhas e colunas pares dela (Figura 6), fazendo com que a altura e a largura sejam diminuídas pela metade e assim a área reduzida pela quarta parte.

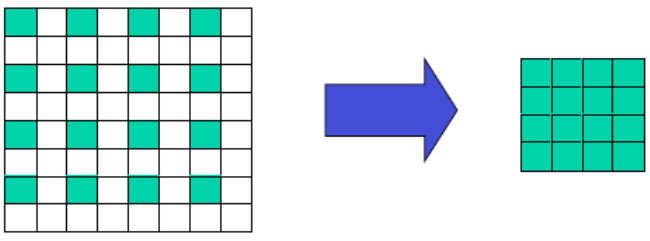


Figura 6. Representação da fase final do método pyrDown.

A Matriz de *Gauss* usada no método *pyrDown* é representada pela Equação 2.

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Equação 2. Matriz de Gauss

A aplicação desses métodos é necessária para melhorar o desempenho e condensar a imagem, assim é possível utilizar a aplicação em diferentes tipos de smartphones, já que a resolução da imagem pode variar de acordo com o aparelho.

2.7 SDK - Tegra *Android* Development Pack

O Pacote *SDK - Tedra Android Development Pack,* desenvolvido pela NVIDIA, simplifica o ambiente para os desenvolvedores de *Android* com um instalador que gerencia a complexidade.

O SDK instala todas as ferramentas de *software* necessárias para o desenvolvimento *Android,* em plataformas *Tegra,* da NVIDIA. Este conjunto de ferramentas de desenvolvimento é direcionado para dispositivos *Tegra*, mas irá configurar um ambiente de desenvolvimento, possibilitando trabalhar com praticamente qualquer dispositivo *Android*. Está disponível para as plataformas Windows, OSX e Linux Ubuntu (TEGRA, 2013).

Outra ferramenta essencial para o funcionamento correto da *OpenCV* no *Android*, é o NDK fornecido pelo Google, ele permite a implementação de aplicativos na linguagem C e C++. Além de fazer a interação do pacote JAVA para *Android* com o Kernel do *smatphone.*

2.8 Sensores

Segundo Bastos (2002), um sensor pode ser definido como um dispositivo que possui propriedades variáveis quando aplicado a alguma grandeza física, podendo fornecer sinais indicando uma grandeza direta ou indireta em relação à referida ação. Uma câmera pode ser definida como um sensor, já que capta a luminosidade através da lente e a utiliza para aquecer um chip. Assim, a imagem captada através da lente se transforma em uma imagem digital, sendo possível processá-la e analisá-la.

As câmeras de smartphone possuem características avançadas, muitas vezes de melhor configuração que câmeras pessoais. Os sensores e lentes estão em constante mudança, atingindo a casa superior a 20 megapixels de resolução máxima, com vários níveis de contrastes e abertura do obturador, e os números aumentam ano a ano.

2.9 Android

Android é um conjunto de ferramentas de *software*s para dispositivos móveis, criado pela Google e pela *Open Handset Alliance*. Tornou-se o sistema operacional móvel com maior crescimento graças à construção feita com base nas contribuições da comunidade Linux, de código aberto, e parceiros de *hardware*s, *software*s e operadoras (ANDROID, 2010).

Android é um sistema que funciona em um nível alto de abstração para os programadores. O *hardware abstraction layer* (HAL) serve como uma interface padrão que permite que o sistema *Android* interaja com a camada de *hardware,* sem que o programador se preocupe com a programação de drivers e programação de baixo nível, para a utilização dos sensores do smartphone. Caso seja necessária uma personalização de *hardware,* a licença do *Android* permite a utilização direta do Kernel do sistema. A última versão (5.1) do *Android* possui duas interfaces HAL A interface DALVIKVM, e a ART.

O sistema *Android* é baseado no Kernel do Linux, nessa camada está presente a maior parte do desenvolvimento de drivers de gerenciamento de memória, escalonamento de processos, etc. Pode-se utilizar qualquer versão de Kernel do Linux, mas é recomendada, a utilização do Kernel desenvolvido para o próprio *Android*. A Figura 7 mostra a representação das demais camadas que compõem o sistema *Android*.

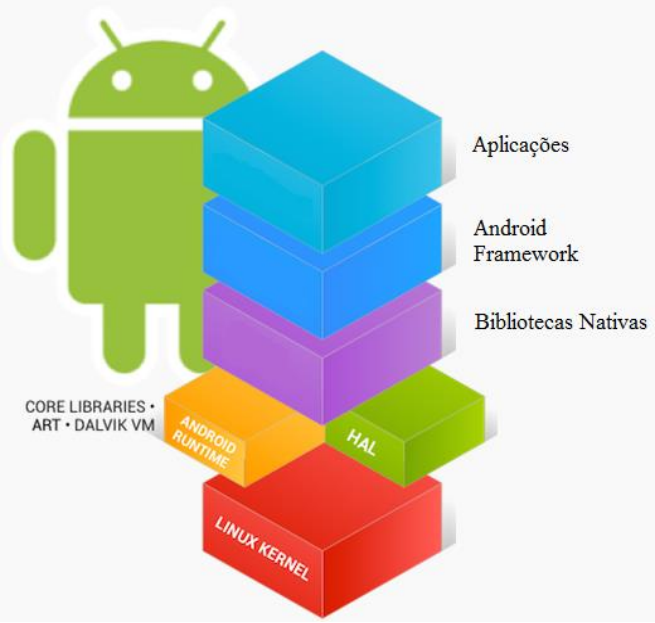


Figura 7. Camadas do sistema *Android*
Fonte: Adaptação (ANDROID, 2010).

A biblioteca *OpenCV* para *Android*, faz uso da camada aplicação e o Kernel do sistema, para isso utiliza-se a ferramenta SDK, e NDK.

2.10 IOIO

IOIO é uma plataforma de entrada e saída (I/O), baseada em telefones *Androids*, e *tablets*. Comunica-se com os telefones, através da conexão USB. É possível conectar um adaptador *bluetooth*, tornando a comunicação sem fio (MONK, 2012).

O IOIO (Figura 8) é um produto da empresa *SparkFun*, porém seu projeto de *hardware* e *software* é *open source* ou seja livre para utilização. Seu *hardware* é composto por um microcontrolador PIC reguladores de tensão osciladores, e toda a estrutura básica para o funcionamento do microcontrolador.

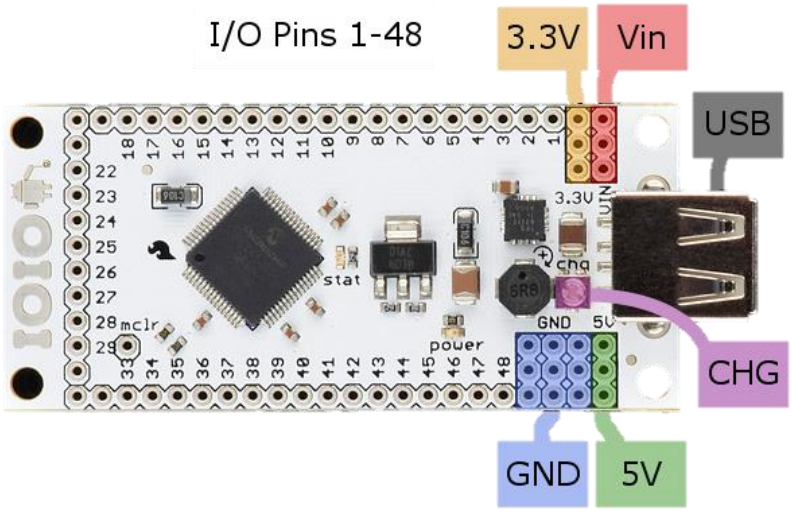


Figura 8. Representação da placa IOIO
Fonte: (IOIO, 2013). (BRAGA, 2005)

O IOIO possibilitou a comunicação do *Android* com circuitos externos. Na última versão foi acrescentado um circuito que possibilita o uso do IOIO com computadores, ampliando a sua capacidade. Como padrão, suas entradas e saídas possuem tensão de 3,3V, mas as entradas possuem uma tolerância de até 5V. O IOIO possui pinos conhecido como I/O esses pinos podem fornecer uma leitura ou escrita de sinais digitais. IOIO ainda possui pinos destinados para PWM, Pulse Input, UART, e SPI (IOIO, 2013).

A utilização de um *hardware* extra, além do uso do smartphone faz-se necessária para que o smartphone consiga acionar atuadores externos, de diferentes potências. Como exemplo, os servos motores que fazem uso do PWM (*pulse width modulation*).

PWM é um circuito destinado ao controle de potência, em diferentes níveis. É muito usado na robótica e mecatrônica e tem como característica manter o torque mesmo em baixas velocidades de motores e atuadores, fazendo uso de um funcionamento linear.

O PWM funciona variando a intensidade média de corrente no atuador, alimentando com pulsos e controlando a duração desses pulsos. Esses pulsos funcionam ligando e desligando rapidamente o circuito, de modo a produzir pulsos retangulares com a duração e o espaçamento iguais. Com isso o valor aplicado no motor será a média. Na Figura 9 está representada a duração da carga aplicada para o ciclo de 50% (BRAGA, 2005).

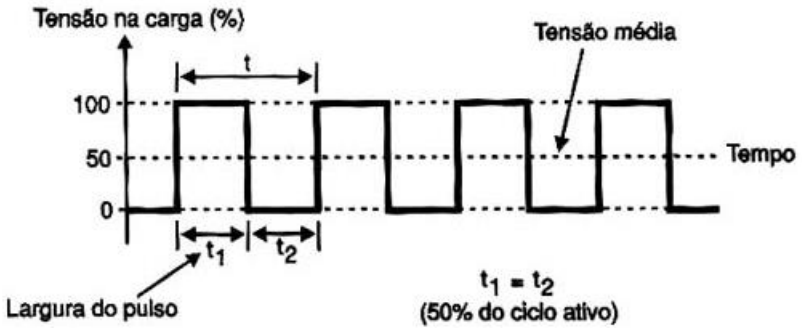


Figura 9. Ciclo PWM de 50%
Fonte: (BRAGA, 2005).

O IOIO possui circuito no seu chip, o que elimina a utilização de um circuito destinado a essa finalidade, porém existe a necessidade de amplificar esse sinal. Esse circuito permite controlar o sentido e controlar a velocidade do protótipo utilizado nesta dissertação.

2.11 Agricultura de Precisão

Na última década, a adoção da agricultura de precisão avançou, e a adoção da técnica aumentou mundialmente (ZHANG e WANG, 2002). No Brasil, a agricultura de precisão tem sido aplicada pelos produtores de soja e milho, porém novas pesquisas já mostram a expansão em novas culturas (MACHADO, BERNARDI, *et al.*, 2006).

Foram adotados sistemas embarcados, sensores, *software*s desenvolvidos voltados para agricultura, pesquisas nas áreas de medida da condutividade elétrica do solo, fotografias áreas, sensoriamento remoto, e estão sendo aprimoradas com novos instrumentos e novas técnicas. (CAMPOS, MANOEL, *et al.*, 2014). Por fim o desenvolvimento da robótica vem se destacando nos últimos anos, e está associado a agricultura de precisão.

No Brasil teve foco em máquinas agrícolas que possuíam os receptores de GPS, criando um senso comum que a agricultura de precisão está associada ao GPS e a mapas de produtividades (INAMASU, CAMPOS, *et al.*, 2009). Atualmente, sabe-se que vai muito além de mapas de produtividade, o simples fato de monitoramento de uma lavoura com o uso de imagens de satélites, ou utilização de tecnologia em campo para coleta de dados para prever doenças.

Para Menegatti e Molin (2004), a agricultura de precisão é um metodo de gestão da produção agrícola, onde a tecnologia e a lavoura devem ultrapassar a barreira lógica para que sistemas embarcados e de automação possam ser otimizados para solucionar determinados problemas.

Pesquisas para o desenvolvimento de máquinas agrícolas autônomas não deixa de ser um ramo da pesquisa de na área da agricultura de precisão, pois busca melhoramento das atividades do agricultor com a diminuição de despesas e aumento da produtividade.

3 TRABALHOS RELACIONADOS

A robótica é amplamente empregada em indústrias, como automobilística, indústrias de eletrônicos e eletrodomésticos. Os robôs segundo Pazos (2002) podem ser classificados como manipuladores, exploradores, máquinas ferramentas e uso geral.

Robôs manipuladores são robôs que manipulam objetos utilizando braços robóticos, pinças, etc. Já robôs exploradores são capazes de efetuar deslocamento, realizando tarefas em diferentes pontos. Já as máquinas ferramentas são máquinas destinadas a realizar uma tarefa de ferramenta e agregar valor na matéria prima como, por exemplo, robôs soldadores. Por fim, robôs de uso geral são aqueles que não se encaixam nas categorias anteriores (PAZOS, 2002).

Robôs são apresentados à agricultura graças a sua facilidade de cumprir tarefas que exigem repetições, tarefas que põem em risco a saúde do trabalhador, realizando trabalhos em menor tempo e muitas vezes com igual, ou melhor precisão.

A tecnologia na agricultura foi desenvolvida primeiramente com foco em componentes computacionais para coleta e armazenamento de dados. A robótica na agricultura é aplicada em vários problemas, mas o recorrente é na direção autônoma de máquinas agrícolas. Desde início de 1990, Sistema de Posicionamento Global (GPS) têm sido amplamente utilizados como sensores de orientação globais (BELL, 2000) (LARSEN, NIELSEN e TYLER, 1994) (YUKUMOTO, MATSUO e NOGUCHI, 2000).

As pesquisas avançaram no sentido de direção autônoma de máquinas agrícolas com a utilização do GPS, porém o custo elevado do GPS de alta precisão faz com que se pesquisem alternativas para os sistemas autômatos. A tecnologia de visão computacional

pode ser usada para guiar automaticamente uma máquina agrícola na linha de plantio. Aplicações mais comuns são guiar tratores, ou orientar colheitadeiras para operação de colheita. O único sensor de orientação é a própria câmera, já que a orientação relativa é a linha de cultivo da cultura. A tecnologia de visão computacional é muito próxima à orientação de uma máquina por um ser humano (WILSON, 2000).

Para controlar um veículo agrícola baseado em visão computacional, encontrar o referencial é crucial, a linha de colheita, linha de semeadura, ou linha do pulverizador de insumos agrícolas, são alguns dos referencias utilizados em pesquisas de visão computacional.

Nesse sentido, Astrand e Baerveldt (2005) desenvolveram um método de reconhecimento de linhas de plantas, que detecta e distingue as plantas e linhas de plantio, utilizando a transformada de Hough. Esse método adapta-se ao tamanho da planta, é capaz de fundir informações provenientes de mais de uma linha de planta. A transformada de Hough encontra um padrão de pontos (plantas) de uma imagem binária e os transforma em linhas (Figura 10).

Figura 10. Reconhecimento da linha de plantas
Fonte: (ASTRAND e BAERVELDT, 2005)

O método apresentou, quanto à posição relativa, um desvio padrão entre 0,6 e 1,2 centímetros, variando de acordo com o tamanho da planta. (ASTRAND e BAERVELDT, 2005).

A Figura 11 apresenta o robô onde foi aplicado o método, criado por Astrand e Baerveldt (2005). Para a pesquisa foi utilizado um PC (*Personal Computer*), uma câmera

COHU CCD, em escala de cinza com infravermelho (780 nm). Outra aplicação da visão computacional, para controlar veículos na agricultura, está no cultivo de citros. Máquinas que utilizam o sistema GPS sofrem frequentemente com perda de sinal devido à copa das árvores, e o uso de sistema de visão computacional colabora para correção desse problema.

Figura 11. Robô agrícola
Fonte: (ASTRAND e BAERVELDT, 2005).

Utilizando o mesmo conceito de linhas, porém as plantas de referência são as árvores. Subramanian et al. (2006) desenvolveram um método para guiar tratores sobre a cultura de laranja. O método consiste em identificar as árvores de laranja e delimitar a área onde a máquina deve trabalhar. O método utiliza um trator da marca John Deere 6410, um PC comum cuja configuração era: processador 2.4 GHz, com sistema operacional Windows 2000. A câmera usada foi Sony FCB-EX780S (SUBRAMANIAN, BURKS e ARROYO, 2006). Nesse trabalho, foram utilizados diferentes tons de valores RGB, de acordo com a intensidade da iluminação, para definir as árvores, e encontrar a linha onde a máquina deverá trafegar (Figura 12). A etapa seguinte foi a calibração da relação pixel com a distância verdadeira. Para eliminar a flutuação no rastreamento do caminho, devido à luz natural e ruído da câmera, foi utilizada a rotina de eliminação de *Outlier*, na base do tempo, e adicionou-se ao cálculo do erro (SUBRAMANIAN, BURKS e ARROYO, 2006).

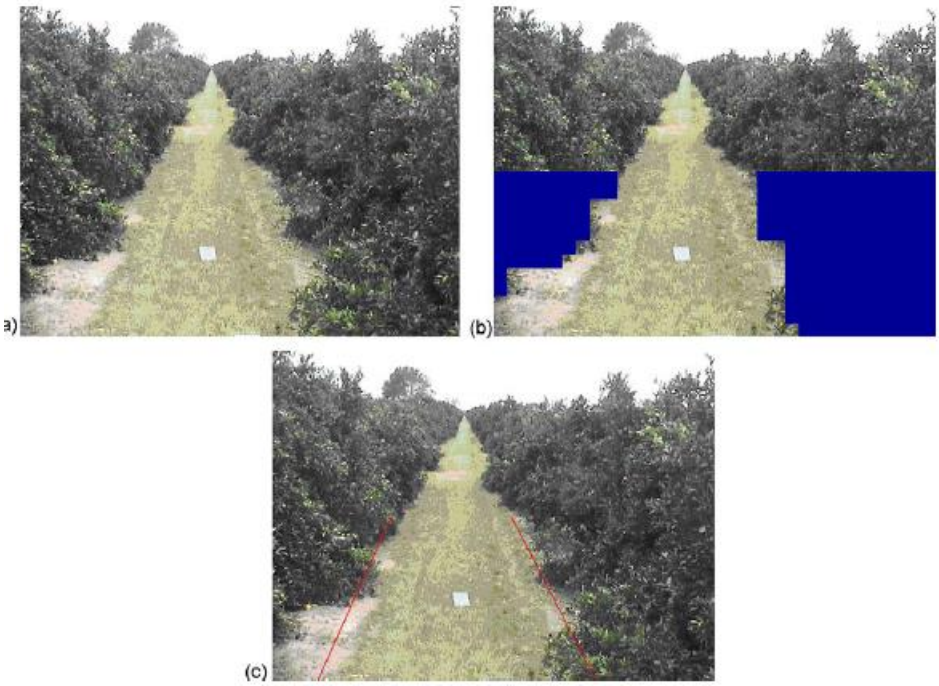


Figura 12 Etapas para delimitar área de atuação da máquina agrícola
Fonte: (SUBRAMANIAN, BURKS e ARROYO, 2006)

A navegação autônoma, utilizando uma câmera como o principal sensor, e algoritmos de processamento aplicados às imagens, foi estudada por Ortiz e Olivares (2006). O trajeto que o robô deveria seguir segundo os autores, é composto por pontos que foram extraídos das imagens capturadas pelo sensor principal, o qual foi submetido ao processamento por reconhecimento de padrões que mapeia pontos reconhecidos nas imagens.

O processamento de imagens é feito por um computador, assim como os trabalhos anteriores, o princípio do algoritmo é o mesmo, onde a imagem é segmentada e o referencial é a linha de plantio. A Figura 13 representa a as etapas do processamento de imagens estabelecido pelo sistema.

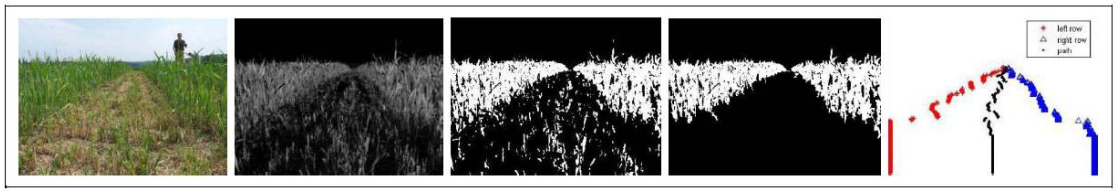


Figura 13. Etapas do processamento de imagens
Fonte: (ORTIZ e OLIVARES).

Pode ser destacado no trabalho de Xue, Zhang e Grift (2012), o desenvolvimento de um robô funcionando com auxílio de uma câmera, um computador, e um GPS. Ele trafega sobre as linhas de plantio da cultura do milho, fazendo uso da Lógica *Fuzzy*, cujas

variáveis linguísticas foram “NG, NS, ZE, PS, PG”, significando “vira muito a esquerda”, “virar a esquerda”, “mantenha seu estado”, “virar a direita”, “virar muito a direita”, respectivamente. A Figura 14 mostra como o sistema realiza o processamento de imagens, e também o sistema aplicado em campo.

Figura 14 Funcionamento do processamento de imagens do robô.
Fonte: (XUE, ZHANG e GRIFT, 2012).

O desenvolvimento de um método de processamento de imagem para determinar a diretriz de viagens para um micro robô em campos alagados de arroz, foi proposto por Chen, Tojo e Watanabe (2003). O objetivo do estudo foi realizar o controle de ervas de forma mecânica, diminuindo o uso demasiado de herbicidas. O sistema utilizou um computador pessoal com um processador *Pentium* de 400Mhz para o processamento das imagens. A captação das imagens foi feita por uma câmera TR-89C wireless. (CHEN, TOJO e WATANABE, 2003). As imagens coletadas pela câmera foram binarizadas e depois processada foi diferenciado a água da cultura de arroz. (Figura 15).

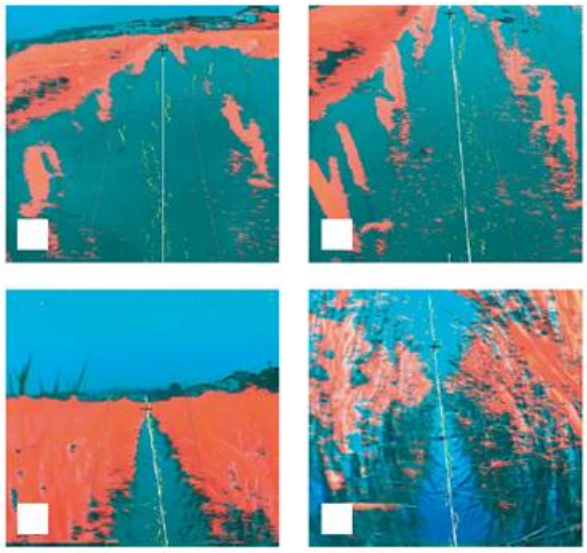


Figura 15. Processamento parcial de imagens na água

Fonte: (CHEN, TOJO e WATANABE, 2003).

Por fim o trabalho de Hana, Zhang e Nic (2004), cuja função é avaliar dinamicamente a distância entre linhas de plantio utilizando processamento de imagens. Foi utilizado um computador pessoal para o processamento de imagens, e um trator para o teste.

Esse trabalho não tinha como finalidade automatizar uma máquina agrícola, mas sim criar um algoritmo para que futuramente possa ser feita essa automatização. Nesse caso o trator foi manuseado por uma pessoa, e o algoritmo obteve um erro aproximado de 1 cm. As etapas do processamento de imagens estão na Figura 16.

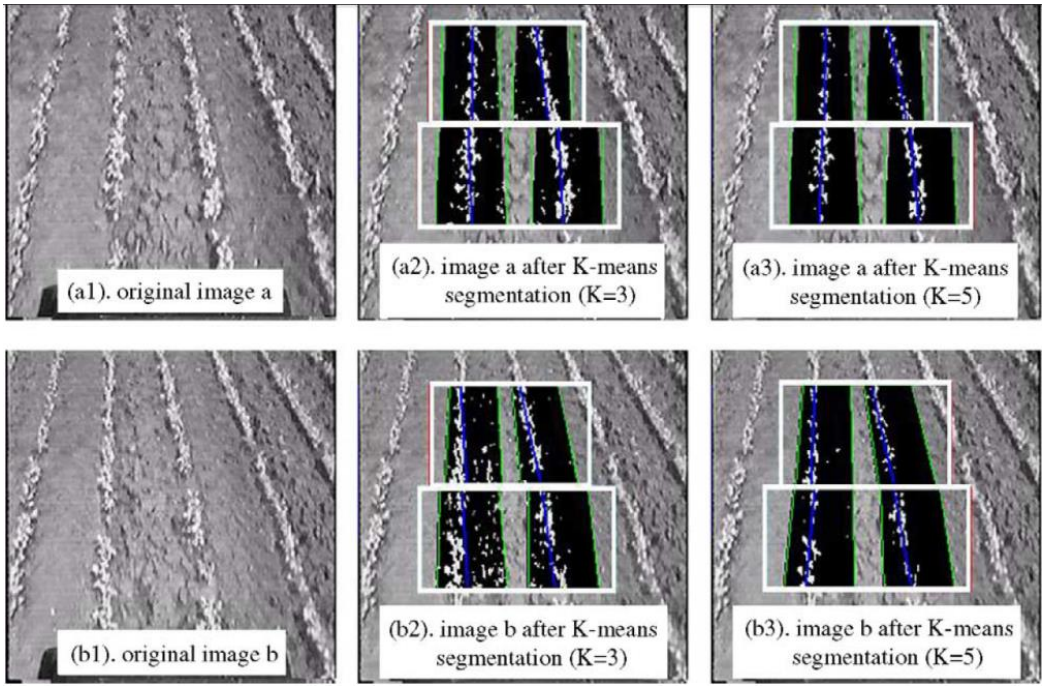


Figura 16. Processamento de imagens para determinar a distância entre linhas de plantio
Fonte: (HANA, ZHANG e NIC, 2004).

Os trabalhos serviram como base para o desenvolvimento da metodologia utilizada nesta dissertação, nota-se o uso de computadores pessoais ou notebooks para o processamento de imagens. Diferente deste trabalho que utiliza um *smartphone* para esse fim.

3.1 NAVIGO

O NAVIGO é um robô desenvolvido no Laboratório de Automação e Robótica da Universidade Estadual de Ponta Grossa dentro do projeto AGROBOT. O projeto AGROBOT, tem como objetivo desenvolver soluções na área de automação e robótica agrícola. Dentre as soluções, destaca-se a o aprimoramento de um robô agrícola – NAVIGO, o qual objetivou percorrer talhões com a função de coletar informações padronizadas, contínuas e georreferenciadas.

Constituído por uma arquitetura de *hardware* e *software* de código aberto o NAVIGO possibilita à comunidade seu uso, contribuições, conforme demandas. É composto por um chassi confeccionado em alumínio, Tipo *Crawler* (copia as deformações do terreno), possui rádio controlado (R/C), Modelo RCF-1, da Marca Himoto (DIAS e SILVA, 2013). Também, faz parte um braço robótico responsável pelas amostragens de solo. O controle de *hardware* e a comunicação com o smartphone é feita por uma Plataforma IOIO.

O robô NAVIGO está representado na Figura 17, destacando-se o suporte para smartphone, o braço robótico responsável pelas amostragens de solo, por fim o IOIO responsável pelo controle e comunicação com o smartphone.

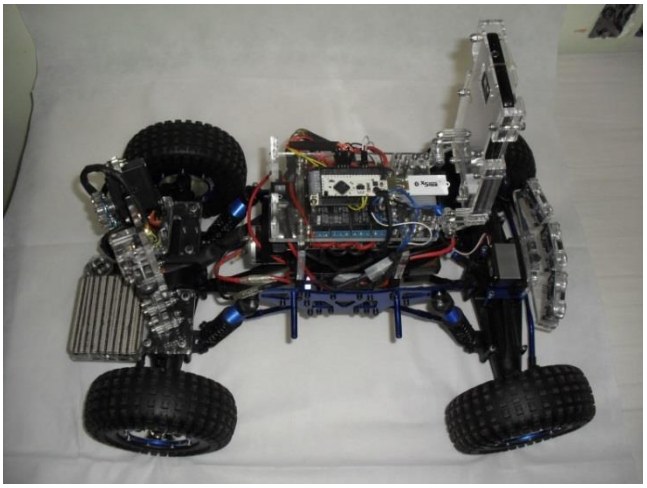


Figura 17. NAVIGO, visão superior.
Fonte: (DIAS e SILVA, 2013).

O controle da sua direção e velocidade e o braço robótico são feitos com o uso de sinais PWM. O *smartphone* realiza o processamento de imagem e envia os resultados para o IOIO, que fará o controle desses componentes e da direção do NAVIGO.

Esse protótipo fez parte dos testes em campo, não foi realizada alterações no hardware do NAVIGO. No software foram acrescentadas duas classes responsáveis pela comunicação e controle do NAVIGO. O IOIO fornece uma biblioteca em JAVA onde estão implementados alguns métodos básicos para a comunicação entre *Android* e o IOIO.

4 MATERIAIS E MÉTODOS

A metodologia foi dividida em 2 (duas) etapas. A primeira etapa consiste na separação de cores da imagem, onde se limita os pontos de trajeto do robô, a criação de bordas, o algoritmo para suavizar os movimentos do NAVIGO. Na segunda etapa foi aplicado o sistema no NAVIGO.

Para facilitar o entendimento do sistema a Figura 18, apresenta um diagrama da interação do sistema com o usuário, as etapas presentes nesse diagrama serão detalhadas nos itens seguintes.

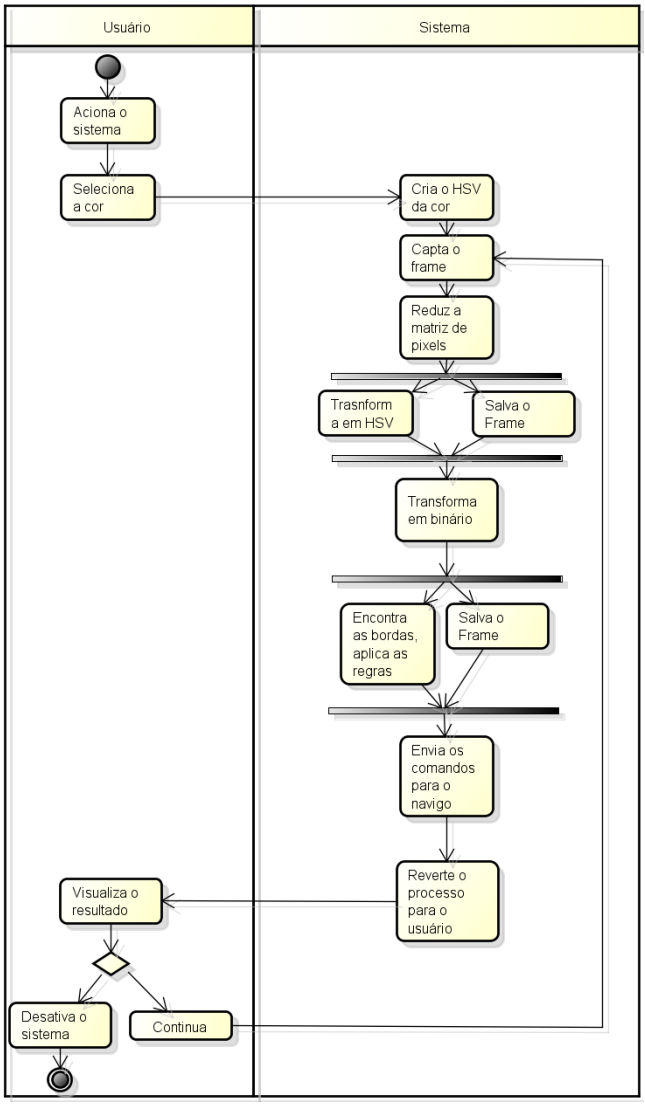


Figura 18. Diagrama da interação com o sistema e o usuário

4.1 Primeira Etapa: Delimitação do Trajeto

Nesta etapa, o sistema aguarda a escolha da cor para que seja efetuada a delimitação de área do trajeto. Depois de ser selecionada, essa cor é transformada em HSV (*HUE*, *Saturation* e *Value*), e limita-se o HUE a uma variação de aproximadamente 15 pontos, tendo como limite inferior o valor da cor 0 (zero) e o limite superior o valor 255. Também, foram limitados os outros dois valores de brilho e saturação a uma variação de 50 pontos a mais ou a menos. Isso faz com que o método desenvolvido crie, assim, uma matriz de cores abrangentes, nomeada *spectrum*, variando suas cores entre a cor de menor valor (*lowerBound*) e de maior valor (*upperBound*). A Figura 19 exemplifica a matriz *spectrum*.

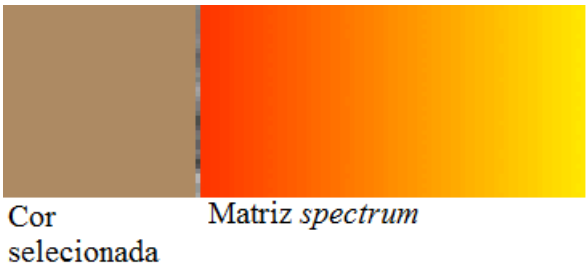


Figura 19 Exemplo da matriz *spectrum*
Fonte: O autor

Além de selecionada a cor, o processo para a verificação da imagem ocorre a medida que uma nova imagem é capturada pela câmera do smartphone e seu evento é acionado. Quando isso ocorre, recebe-se a matriz que contém toda a imagem representada, *pixel a pixel*, no formato RGB, permitindo processá-la. Contudo, essa matriz é muito extensa (aproximadamente 346.000 (346 mil pixels)).

Algumas regiões da imagem capturadas pela câmera podem ser desconsideradas como segue. A inferior, pois esta região contém parte do robô; e a superior, pois devido ao ângulo de posicionamento da câmera esta pode conter parte do céu.

Desta forma, desconsidera-se 20% da região acima da imagem e 10% da região abaixo (Figura 21). Porém, esta imagem conterá 70% da imagem original, ou seja, aproximadamente 242.000 (242 mil) *pixels*, ainda uma matriz extensa. Em virtude disso, diminui-se esta matriz, utilizando o Método *pyrDown* duas vezes, resultando em uma matriz nomeada *pyrDownMat* (Figura 20a). Após a utilização do método *pyrDown*, a resolução da imagem é convertida para aproximadamente 15.000 (15 mil) *pixels*. Ou seja, a sua área de abrangência, que pode ser caracterizada pelo número de *pixels*, sendo sua

altura e largura diminuídas em quatro vezes, fazendo com que a área seja diminuída em dezesseis vezes. Esta conversão faz com que a imagem a ser analisada necessite de um menor potencial de processamento do *smartphone*.

Na sequência, converte-se a matriz *pyrDownMat* para o sistema de cores HSV utilizando os valores *lowerBound* e *upperBound*, para definir uma nova matriz chamada *mask*, onde, se a cor do pixel posicionado na posição *(i, j)* estiver entre os valores de *lowerBound e upperBound*, será colocado o valor 255 na posição *(i, j)*, caso contrário, será colocado o valor 0. Desta forma, a matriz *mask* é uma imagem binarizada (possui apenas dois valores), onde os pixels da imagem originária, representada pela matriz *pyrDownMat*, que possuíam as cores dentro do limite estipulado, possuirão agora o valor 255 (cor branca) e os outros pixels possuirão o valor 0 (cor preta).

Com isto, tem-se agora a matriz *mask* (Figura 20b), representado o caminho a ser percorrido pelo robô (branco) e o que será considerado como obstáculos (preto). Como esta imagem possui ruídos, ou seja, pontos pretos pequenos e que podem ser ignorados por humanos e pelo robô, pois são inexpressíveis comparados as reais barreiras que impediriam o trajeto, utiliza-se a técnica de dilatação da imagem, que consiste em dilatar os pontos brancos de uma área maior do que a já alcançada e, assim, eliminando praticamente todos os ruídos desnecessários. Utiliza-se esta técnica três vezes na matriz mask produzindo assim uma nova e última matriz denominada *dilatedMask* (Figura 20c).

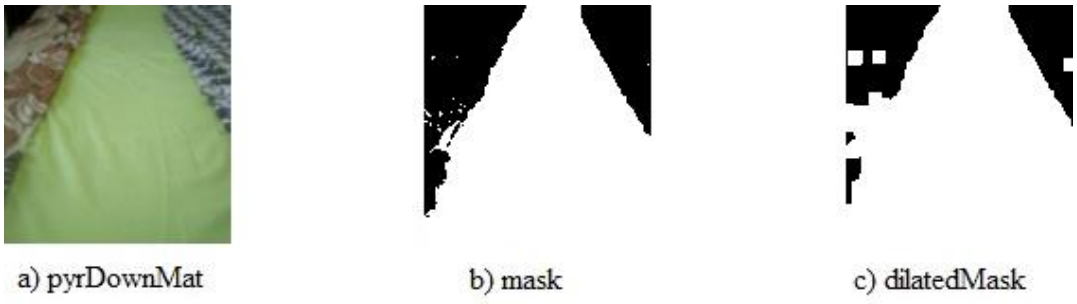


Figura 20. Resultado da representação das matrizes adquiridas ao longo do programa

Em um cenário ideal e real, contendo apenas barreiras laterais, os quais poderiam ser pés de trigo, milho, entre outros, considera-se um trapézio branco no meio da tela e trapézios menores, ou até mesmo triângulos, nas laterais da imagem. Desta forma, a maneira mais fácil de detectar o meio do percurso, seria catalogar os pontos que formam a borda da

esquerda e direita da imagem, fazendo-se um cálculo básico para encontrar o meio da imagem.

Percorre-se, então, a matriz *dilatedMask* de cima para baixo, da direita para a esquerda, em busca do primeiro pixel branco, ou não preto. Após encontrar o primeiro pixel da direita repete-se o procedimento, porém da esquerda para a direita. Com este par de pontos pode-se determinar o ponto do meio, que será então armazenado.

Diferente dos testes em laboratório, as imagens em campo apresentam linhas irregulares, dificultando o funcionamento suave da direção do protótipo, além de que em determinados trechos do percurso a direção seria acionada sem a necessidade real. Para os testes foram utilizadas as regras propostas por Xue, Zhang e Grift (2012). Essas regras serviram apenas como ferramenta para suavizar o movimento do protótipo e não como comparação.

O algoritmo para detecção de bordas foi dividido, como segue. Após serem encontrados todos os pontos da direita, esquerda, foram calculados os três tipos diferentes de bordas e pontos centrais, para fins de comparação, utilizando-se desta forma:

- **Média Aritmética:** calcula a média aritmética básica entre todos os pontos (cor verde Figura 21);
- **Primeiro / Último Pontos:** apenas utiliza-se do primeiro e o último ponto (cor azul Figura 21);
- **Ponto médio:** calcula o ponto médio entre a primeira metade de pontos como primeiro ponto e a média entre a segunda metade de pontos como segundo ponto (cor magenta Figura 21).

Os pontos centrais são calculados com base na seguinte equação:

$$ponto_{central} = \frac{|ponto_{direita} - ponto_{esquerda}|}{2} + min(ponto_{direita}, ponto_{esquerda})$$

Equação 3. Equação para a determinação do valor do ponto central

Cada um desses trajetos é mostrado na Figura 21. Nesta mesma figura nota-se na parte em tons de cinza, a cima e abaixo da região central, representando as partes ignoradas. No canto direito superior verifica-se a matriz *spectrum* contendo a variação de cores consideradas e a cor originária selecionada no início dos procedimentos, sendo mostrada logo acima da matriz *spectrum*.

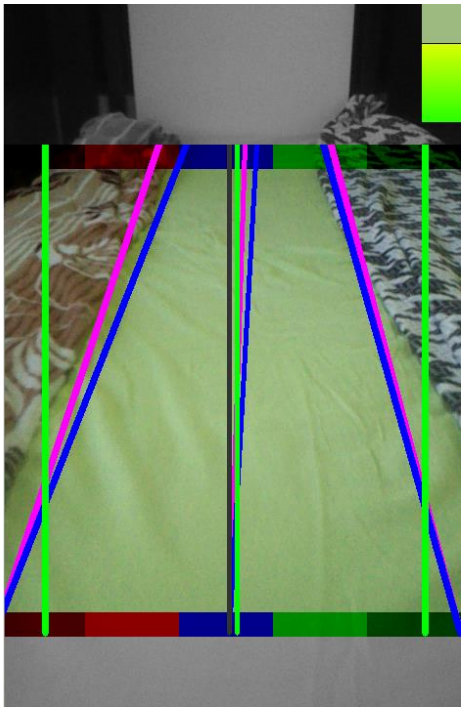


Figura 21. Visão da tela do *smartphone* para o usuário

A Figura 22 mostra a imagem representada pela matriz *pyrDownMat* com a adição das cores de borda e de centro.

Figura 22. Matriz *pyrDownMat* com os trajetos e as bordas traçadas.

A partir disso, têm-se três bordas e três trajetos centrais bem definidos. Essas bordas e trajetos entram nesse contexto para fazer a correção dos desvios da rota, ou seja, para ajustar o ângulo de giro das rodas e tentar manter sempre o robô dentro do trajeto ideal.

Sabe-se que a menor distância entre dois pontos, em um plano, é uma reta. Partindo deste princípio, o método utiliza os trajetos centrais que consistem em uma reta

entre o ponto inicial, ou seja, o primeiro ponto do trajeto, considerando a imagem de baixo para cima, e o alvo que se deve alcançar, ou seja, o último ponto do trajeto, considerando a imagem de baixo para cima.

Naturalmente, presume-se que o trajeto real que o robô fará, no estado atual, é simplesmente seguir reto, ou seja, o caminho a ser seguido na imagem é definido por uma reta no centro da imagem, considerando que a câmera está posicionada no meio do robô e na parte da frente do mesmo.

A regra de inferência é criada dividindo as partes superior (CIMA) e inferior (BAIXO) da tela em cinco regiões, estas sendo denominadas: Muito à esquerda; Esquerda; Neutra; Direita; e Muito à direita. (XUE, ZHANG e GRIFT, 2012). Com essas variáveis linguísticas já determinadas montam-se os gráficos de entrada da lógica a serem usados (Figura 23).

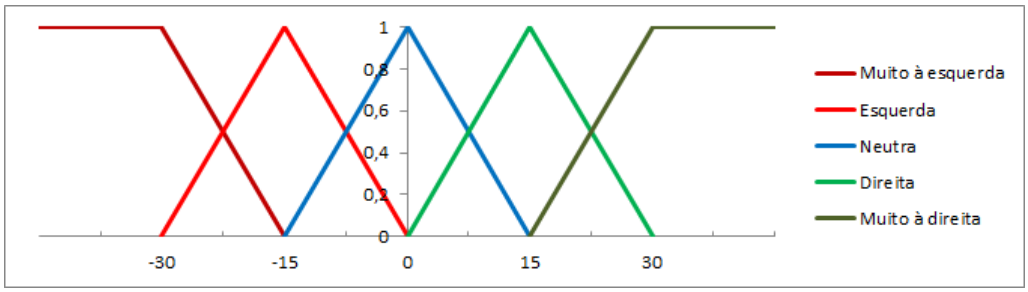


Figura 23. Gráfico dos termos linguísticos para as regiões de CIMA e de BAIXO.

As regiões definidas podem ser vistas na tela final do aplicativo (Figura 21), com as mesmas cores das apresentadas no gráfico (Figura 23). Neste gráfico, os valores de pico de cada uma das regiões são o centro da região em questão. Vale ressaltar que foram utilizados valores hipotéticos, pois, considerando que a tela do dispositivo pode mudar de tamanho, dependendo do modelo do mesmo, e que cada uma das regiões possui o mesmo tamanho, sendo todas divididas igualmente pela tela, a distância entre as dadas regiões serão sempre as mesmas, assim, pode-se afirmar que não importa os valores das distâncias entre as regiões, mas sim que as distâncias sejam as mesmas.

É possível ver as regiões de CIMA e BAIXO, nas quais possuem as mesmas cores da Figura 23 e se encontram logo abaixo da região cinza superior (CIMA) e logo acima da região cinza inferior (BAIXO). Fazendo uma correlação entre as regiões de CIMA e BAIXO é possível montar uma ação para cada possibilidade do robô, ou seja, uma ação

para cada cruzamento entre os valores em CIMA e BAIXO. Estas possibilidades foram determinadas a partir da lógica:

- Não precisa ser feito quando o robô está em um trajeto paralelo ao trajeto ideal; o trajeto ideal está indo em direção ao centro; e será feito o cruzamento do trajeto ideal num futuro hipotético.
- Virar um pouco à esquerda quando o trajeto ideal está tendendo levemente à esquerda.
- Virar bem à esquerda quando o trajeto ideal está tendendo muito à esquerda.
- Virar um pouco à direita quando o trajeto ideal está tendendo levemente à direita.
- Virar bem à direita quando o trajeto ideal está tendendo muito à direita.

Tabela 1. Regras utilizadas no algoritmo de detecção de bordas para suavizar o movimento.

		Posição de CIMA				
		Muito à Esquerda	Esquerda	Neutra	Direita	Muito à Direita
Posição de BAIXO	Muito à Esquerda	Neutra	Neutra	Neutra	Direita	Bem à direita
	Esquerda	Esquerda	Neutra	Neutra	Neutra	Bem à direita
	Neutra	Esquerda	Neutra	Neutra	Neutra	Direita
	Direita	Bem à esquerda	Neutra	Neutra	Neutra	Direita
	Muito à Direita	Bem à esquerda	Esquerda	Neutra	Neutra	Neutra

As variáveis linguísticas de saída serão: Bem à esquerda: vire bem à esquerda; Esquerda: vire um pouco para esquerda; Neutra: não faça nada; Direita: vire um pouco para direita; e Bem à direita.

Tabela 2. Tabela de valores para cada termo linguístico

Variável Linguística	Virar
Bem à Esquerda	-20
Esquerda	-10
Neutra	00
Direita	10
Bem à Direita	20

Por se tratar de um robô, percorrendo um trajeto com bordas, aplica-se a lógica para as bordas encontradas, para que o robô esteja dentro da pista, o que em muitos casos é muito mais relevante que o trajeto em si. Para isso, utilizam-se as mesmas regiões de CIMA e BAIXO definidas, porém desta vez utilizando as bordas do trajeto para a definição das lógicas. As lógicas encontradas para a borda esquerda são a borda deve sempre estar à esquerda; se a borda tender a ir para o centro, deverá virar levemente para a direita; se a borda tender a direita devera virar bem à direita; e caso contrário nada precisa ser feito.

E para a borda direita, a borda deve sempre estar à direita; se a borda tender a ir para o centro, deverá virar levemente para a esquerda; se a borda tender a esquerda deverá virar bem à esquerda; e caso contrário nada precisa ser feito. Sendo assim, podem-se gerar duas tabelas para estas lógicas, Tabela 3 para a borda esquerda e Tabela 4 para a direita.

Tabela 3. Regras da borda esquerda.

		Posição de CIMA				
		Muito à Esquerda	Esquerda	Neutra	Direita	Muito à Direita
Posição de BAIXO	Muito à Esquerda	Neutra	Neutra	Direita	Bem à direita	Bem à direita
	Esquerda	Neutra	Neutra	Direita	Bem à direita	Bem à direita
	Neutra	Neutra	Neutra	Direita	Bem à direita	Bem à direita

Posição de CIMA

		Muito à Esquerda	Esquerda	Neutra	Direita	Muito à Direita
Posição de BAIXO	Neutra	Bem à esquerda	Bem à esquerda	Esquerda	Neutra	Neutra
	Direita	Bem à esquerda	Bem à esquerda	Esquerda	Neutra	Neutra
	Muito à Direita	Bem à esquerda	Bem à esquerda	Esquerda	Neutra	Neutra

Tabela 4. Regras da borda direita.

Deste modo, a cada verificação que for feita pelo dispositivo móvel, são aplicadas as três tabelas.

Após definidas essas regras, monta-se um exemplo. Supondo a posição do trajeto ideal em CIMA seja esquerda e EMBAIXO direita, nada se precisa fazer, como nota-se na Tabela 1. Portanto, percebe-se claramente que devemos utilizar à lógica AND, para, desta forma, vincular as linhas e as colunas e conseguirmos o resultado esperado.

Porém, como se trata da lógica, o valor poderá não estar totalmente na esquerda, ou seja, ele pode possuir uma porcentagem estando na posição esquerda e outra porcentagem estando na posição muito à esquerda, como podemos ver na Figura 23. Desta forma, utiliza-se a operação AND da lógica.

Como Gomide *et. al.*(2002) mostram, a operação AND para a lógica nada mais é que o valor mínimo entre todos os valores. Ou seja, no exemplo anterior assume-se que de acordo com a Figura 23, os valores para o trajeto ideal deram:

- Para a variável CIMA, o valor é -18;
- Para a variável EMBAIXO, o valor é 18.

Para estes valores encontram-se as seguintes variáveis:

- CIMA, muito à esquerda com valor 0,2 e esquerda com valor 0,8;
- EMBAIXO: muito à direita com valor 0,2 e direita com valor 0,8.

A partir destes valores, realiza-se uma combinação entre as possíveis saídas (Tabela 5).

Tabela 5. Agrupamento entre CIMA e BAIXO

		Posição de CIMA	
		Muito à Esquerda	Esquerda
Posição de BAIXO	Direita	Bem à esquerda	Neutra
	Muito à Direita	Bem à esquerda	Esquerda

Gerando as combinações, finalmente aplicam-se os operadores lógicos AND e OR, já que existem dois casos que geram a saída “Bem à esquerda”, obtendo Equação 4.

$$\mu_{be}(\theta) = (\mu_{me}(CIMA) \wedge \mu_{md}(BAIXO))) \vee (\mu_{me}(CIMA) \wedge \mu_{d}(BAIXO)) = 0{,}2$$

$$\mu_{e}(\theta) = \mu_{e}(CIMA) \wedge \mu_{md}(BAIXO) = 0{,}2$$

$$\mu_{n}(\theta) = \mu_{e}(CIMA) \wedge \mu_{d}(BAIXO) = 0{,}8$$

Equação 4. Equações resultantes da lógica *Fuzzy*

Onde:

- ‘e’ se refere a ‘Esquerda’;
- ‘d’ se refere a ‘Direita’;
- ‘md’ se refere a ‘Muito à Direita’;
- ‘me’ se refere a ‘Muito à Esquerda’.

4.2 Segunda Etapa: Aplicação do Sistema no NAVIGO

A Segunda etapa consiste na aplicação do sistema em um protótipo, para verificar se as funcionalidades apresentadas podem ser aplicadas em sistemas reais, sem a necessidade de utilizar máquinas agrícolas, garantido a segurança e economia de recursos.

Para testar o sistema foi realizado correlação entre a saída do algoritmo proposto e o ângulo da direção do NAVIGO. A direção do protótipo é composta de um servo motor que trabalha horizontal em relação a sua caixa de direção (Figura 24).

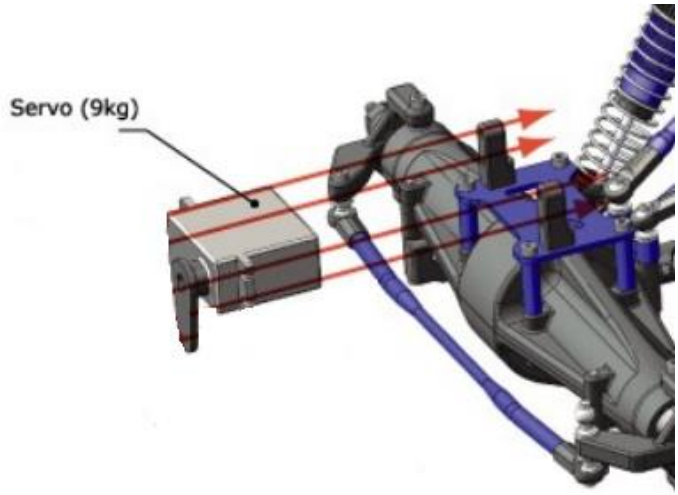


Figura 24 posição do servo como caixa de direção do NAVIGO

Apesar de, o servo trabalhar nos ângulos de 180 graus, no NAVIGO se limita entre 35 e 145 graus devido a distância do servo ao eixo da roda e o ângulo máximo de direção das rodas. Os ângulos foram divididos em cinco partes para representar os resultados da saída do algoritmo (Tabela 6).

Tabela 6. Correlação da saída do algoritmo de detecção de bordas, com o ângulo da direção do NAVIGO

	Saída	
Ângulo servo (°)	35	Muito a Direita (20)
	62,5	Direita (10)
	90	Neutra (0)
	117,5	Esquerda (-10)
	145	Muito a Esquerda (-20)

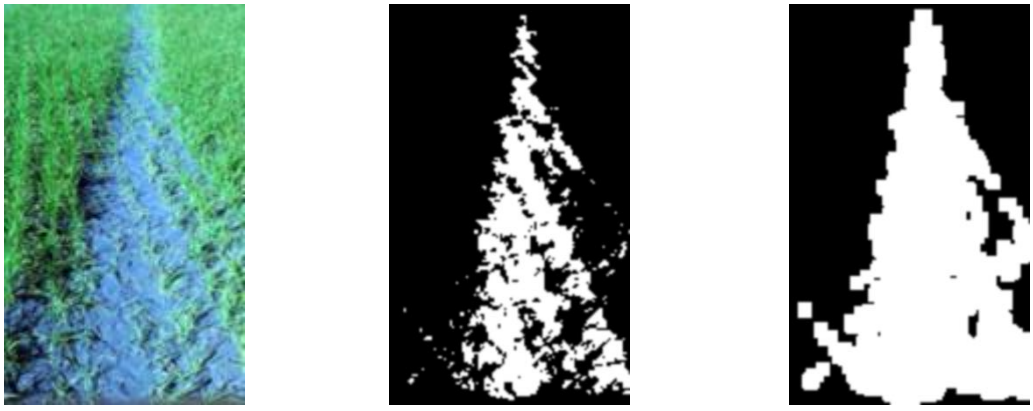
Feita a correlação, foi realizada a comunicação entre o smartphone e o IOIO. Para esse trabalho foi utilizada a comunicação *Bluetooth*, já que para a versão do IOIO utilizada a comunicação USB fornece tensão e corrente para o smartphone, diminuindo a bateria do sistema.

5 RESULTADOS E DISCUSSÕES

Os Procedimentos experimentais foram divididos em etapas para a avaliação do sistema, iniciando-se com a seleção de um local com uma cor sólida como terreno, no qual um robô se movimenta, marcando as bordas com cores diferentes, seguindo com a fixação do celular selecionando para manter a inclinação e a altura, utilizando-se em seguida o programa, que foi concebido, selecionando a cor desejada do caminho e apontando a câmera do celular na direção da pista e obtendo os resultados que serão analisados.

Primeiramente foi testado o método em campo para delimitação das bordas de acordo com a saída *dilatedMask*. Os testes foram realizados em uma fazenda localizada na região do Município de Teixeira Soares, ao Sudeste do Paraná. Nessa etapa foi realizado o delineamento e a separação da imagem por cores de acordo com os testes em cenário ideal.

A cultura utilizada para o teste foi o trigo (*T. aestivum*), cultivado no período entre Junho e Julho/2014. Foram salvas algumas imagens do funcionamento do sistema e na Figura 25a é mostrado o comportamento do *pyrDownMat*. Na Figura 25b é mostrado o comportamento mask e na Figura 25c o comportamento *dilatedMask* é apresentado.



a) pyrDown b) mask c) dilatedMask

Figura 25. Representação das matrizes adquiridas ao longo do programa em campo

A Figura 26 representa a imagem já processada na tela do celular, onde essa imagem corresponde a saída para o usuário. A imagem apresenta as limitações onde o robô deve traçar seu caminho, linhas azuis, e a linha onde se caracteriza a média da soma de pontos de um lado com o outro, linha vermelha.

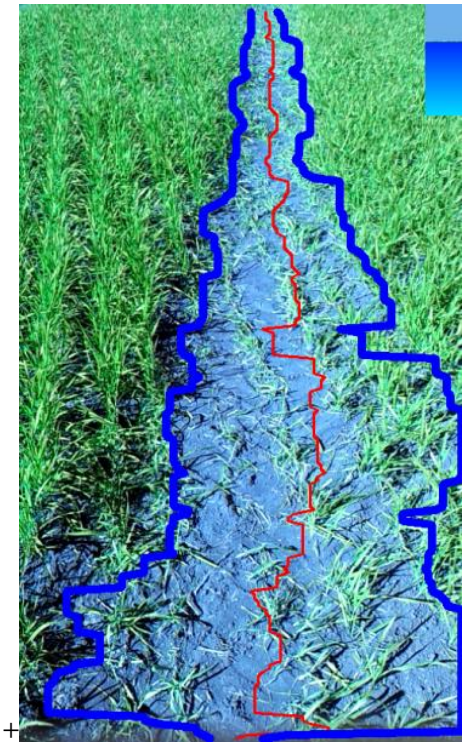


Figura 26. Saída para o usuário

Após a verificação da funcionalidade do delineamento de bordas em campo, foi tratar as bordas nas 3 (três) formas propostas. Foi selecionada uma tábua de mármore para os testes, porém, por essa possuir muitas variações de cores cobriu-se a mesma com um papel cinza e utilizou-se de panos verdes para caracterizar as bordas (Figura 27).

Figura 27. Mesa coberta com papel cinza, indicando o caminho do robô, e dois panos verdes, indicando as bordas.

Para a segunda etapa foi utilizado um suporte de celular (Figura 28), deixando desta forma a base inferior do celular a aproximadamente 19 centímetros de altura com uma inclinação

para frente de 40º, considerando como posição inicial do celular na vertical. O celular utilizado é o Sony Xperia U (Figura 28).

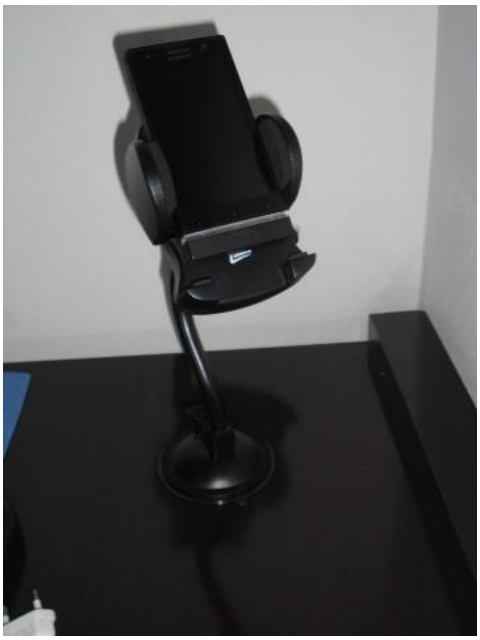


Figura 28. Suporte do celular

Para a seleção da cor foram selecionados os valores de *HUE*, como 159 e 172. As imagens encontradas, com ambas as cores, são mostradas nas Figura 29 e Figura 31, respectivamente. As matrizes *dilatedMask* das saídas são mostradas nas Figura 30 e Figura 32, respectivamente.

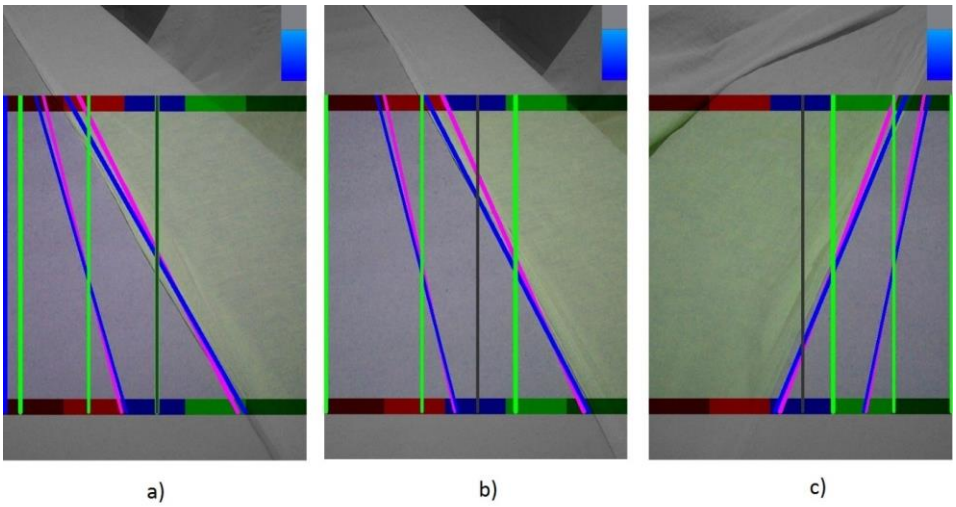


Figura 29. Saída para *HUE* 159

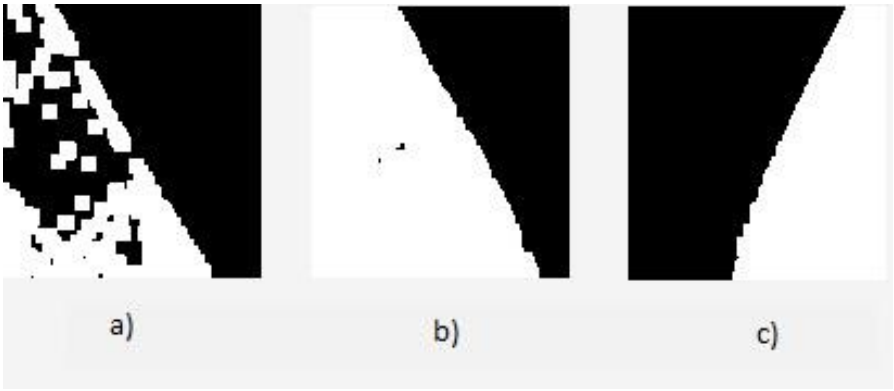


Figura 30. Matrizes *dilatedMask* para as imagens da Figura 29

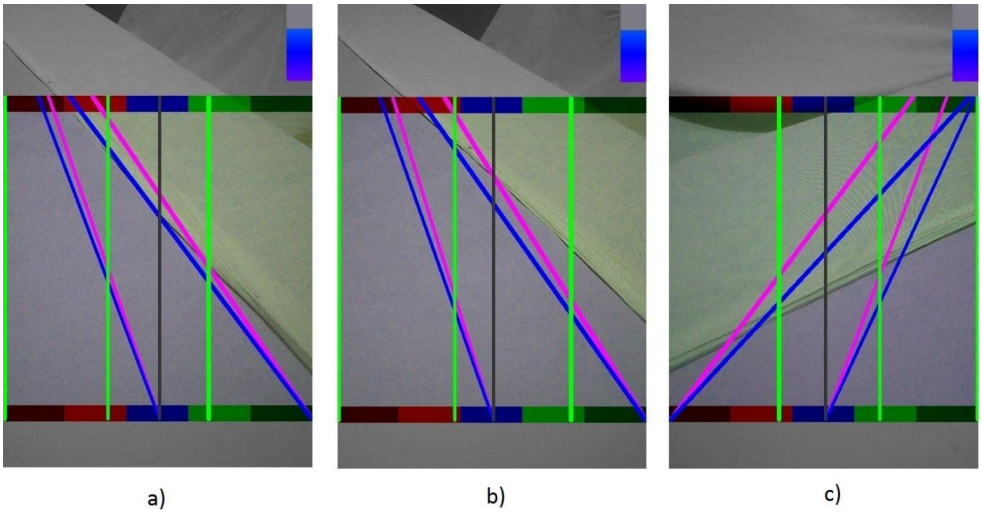


Figura 31. Saída para *Hue* 172

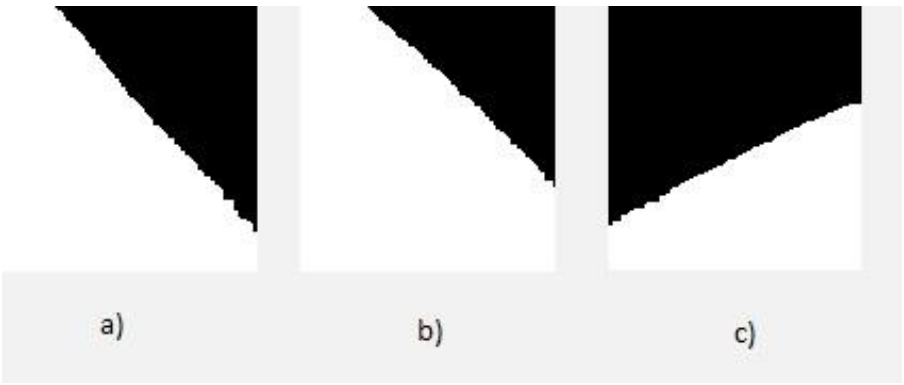


Figura 32. Matrizes *dilatedMask* para as imagens da Figura 31

Pelas Figura 30 e Figura 32, percebe-se que a seleção do valor de *HUE*, no primeiro caso sendo 159 e 172 e no segundo, influencia drasticamente na seleção da pista pelos métodos descritos. Já que a pista apresenta certas variações de cores devido a diferentes graus de iluminação, em diversos pontos, podendo ser relevante em várias outras aplicações práticas, tais como identificação de objetos em ambiente robóticos

(SIMÕES e COSTA, 2001). As saídas da lógica para as Figura 29 e Figura 31 podem ser vistas, nas Tabela 7 e Tabela 8.

Tabela 7. Saída apresentas pelo algoritmo de Xue, Zhang e Grift (2012)para as detecções de bordas apresentadas na Figura 29

		Método		
		Média Aritimética	Primeiro / Último Pontos	ponto médio
Figura	A	0	-20	-20
	B	0	-10	-10
	C	0	20	20

Dentre os outros dois métodos, “Primeiro / Último Pontos” e “ponto médio”, observa-se pelas Tabela 7 e Tabela 8, e suas respectivas imagens de saída (Figuras 28 e 30), que o método do “ponto médio” parece ser mais conservador, apresentando resultados iguais ou menores, já que muitas vezes as bordas laterais deste tendem a se encontrarem mais externamente. Estes resultados conservadores se diferenciam do caso anterior para o método da “Média”, pois são situações que pode ser visto claramente que o robô não bate se virar pouco. O método para média aritmética foi descartado das etapas seguintes, devido a pouca confiabilidade do método.

Tabela 8. Saída apresentas pelo algoritmo de Xue, Zhang e Grift (2012) para as detecções de bordas apresentadas na Figura 31

		Método		
		Média Aritimética	Primeiro / Último Pontos	Ponto médio
Figura	A	0	-20	-20
	B	0	-20	-10
	C	10	20	20

Realizados os testes de bancada, foi aplicado o teste em campo, no NAVIGO, sob as linhas do pulverizador. A fazenda utilizada foi a mesma do primeiro teste, porém a cultura que estava sendo cultivada no período de teste foi a da soja (*Glycine max*). Como o

sistema foi projetado para funcionar em diferentes culturas, a mudança de cultura não deve afetar os resultados. O período de testes foi no mês de Dezembro/2014.

As imagens apresentadas são as saídas para o usuário, onde a imagem já está processada. Na Figura 33a, o sistema apresenta a detecção de borda mais comum, ou seja, onde não há curvas ou tomadas de decisões. Pode-se notar uma pequena tendência das bordas azuis e magentas para a direita. Na Figura 33b, mesmo com caminho bifurcado, devido ao cruzamento de linhas de plantio com a linha do pulverizador, o sistema detectou e realizou o delineamento de bordas correto. Já na Figura 33c, o sistema apresentou ponto crítico, onde deverá ser tomada uma decisão como, por exemplo, virar à esquerda.

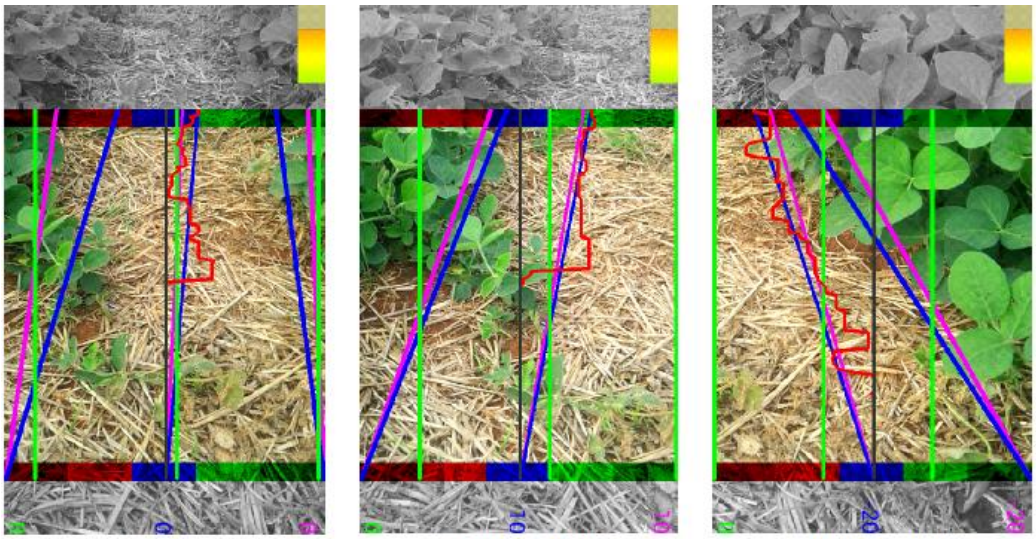


Figura 33 Resultados do delineamento de borda em campo

Na Figura 34 é apresentado um talhão da fazenda onde foi deixado o NAVIGO sobre a linha do pulverizador. O sistema coletou imagens processadas pelo *smartphone*, durante o processo de apresentação para o usuário, ou seja, imagens em fase de processamento.

Figura 34. Teste em Campo NAVIGO.

Para comparação entre os métodos “primeiro e último” e “ponto médio”, foi contada a quantidade de pixels pretos, das imagens binárias, entre as bordas, ou seja, o erro. Dessa maneira pode-se concluir que o método que apresentar menos pixel entre as bordas, apresenta o melhor resultado para o delineamento das bordas.

A Figura 35 apresenta a coleta e o processamento de um único *frame*. A Figura 35a a apresenta a saída para usuário. A Figura 35b está na forma binária, dilatada com as bordas, e por fim a Figura 35c possui os pixels externos da borda azul retirados.

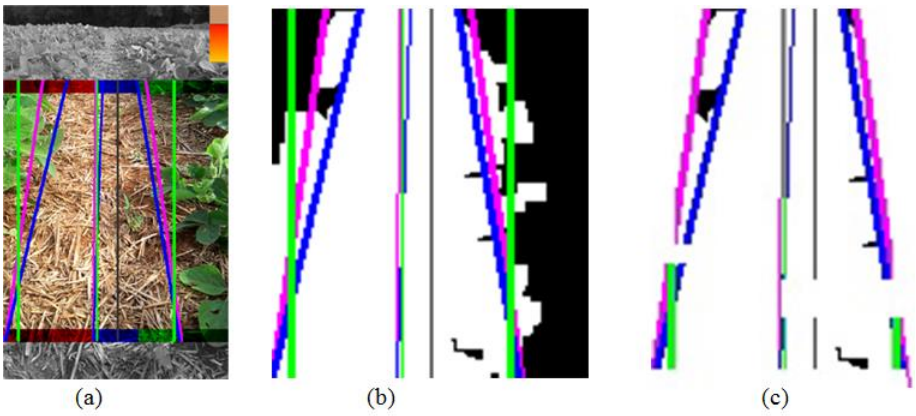


Figura 35. Etapas para obtenção de resultados

Foram utilizadas 62 imagens, a correlação entre as duas bordas (magenta e azul), utilizando o teste *t*. Obteve-se o resultado de 0,5914, ou seja ambas amostras não diferem entre si. Na Figura 36 apresenta o gráfico com a contagem de pixels pretos entre as duas bordas, onde o método “primeiro e último” de cor azul, apresentou melhores resultados em números de pixels.

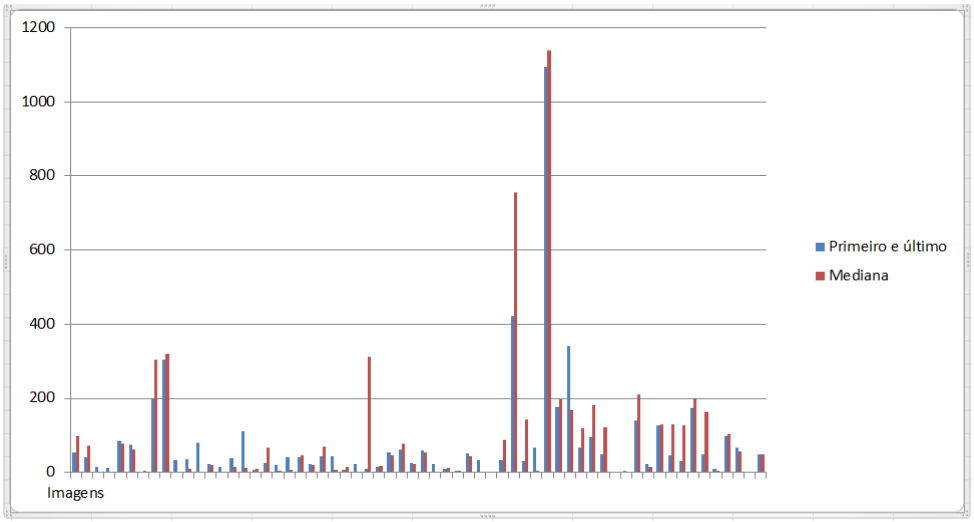


Figura 36. Contagem de pixels entre bordas

Foram feitos 10 (dez) testes durante o período da tarde, e nesse tempo o NAVIGO percorreu uma faixa de 50 metros por teste, em condições de céu aberto e claro. Os métodos do “ponto médio” e “primeiro e último” apresentaram resultados muito semelhantes em campo, assim como nos testes de bancada, porém em algumas situações o método do “primeiro e último”, apresentou um resultado mais plausível de correção de rota. Isso foi notado na prática (Figura 37), onde algumas vezes o método do ponto médio, levou o NAVIGO a sair da rota, invadindo a plantação, para depois retornar à linha do pulverizador.

Figura 37. Teste em campo, método do ponto médio.

Trabalhos como de Astrand e Baerveldt (2005), Subramanian, Burks e Arroyo (2006) entre outros, também usam como padrão linhas de semeadura, ou linhas do pulverizador, para usar como referência no uso de visão computacional. As variáveis linguísticas usadas por XUE *et al* (2012), foram aplicadas da mesma maneira nesta dissertação.

Por fim, em relação aos trabalhos correlatos, percebe-se que soluções propostas, como Pradhan et al.(2008), onde realizaram um estudo entre diversas técnicas de navegação de robôs móveis, comprovaram que o uso da lógica e associações de Gauss para a navegação autônoma de robôs em ambientes altamente desordenados e com obstáculos obtiveram melhores resultados. CHEN et. al. (2003), afirmam que processamento de imagens é um método eficaz para orientar um robô em um campo de plantio de arroz, identificando os limites entre as plantações. Simões et. al. (2001), já citado anteriormente, entre outros, serviram de base para a realização deste estudo, pois o mesmo utiliza da combinação das técnicas.

6 CONCLUSÕES E PERSPECTIVAS DE TRABALHOS FUTUROS

Os objetivos propostos neste trabalho de dissertação puderam ser atingidos, pois foi possível explorar o potencial de *smartphones*, para realizar tarefas que exijam maior grau de processamento.

Os resultados demonstraram a capacidade de implementação de um algoritmo capaz de captar a imagem traçar diferentes tipos de bordas, calcular o melhor caminho, armazenar, e transmitir comandos para uma segunda plataforma, tudo em velocidade suficiente para que o protótipo consiga tomar a decisão correta.

Dentre os diferentes tipos de bordas apresentados nesse trabalho, conclui-se que as bordas que utilizam apenas o primeiro e último ponto apresentam um resultado prático um pouco melhor para a finalidade. Porém estatisticamente o método que utiliza o "ponto médio" e o "primeiro e último" ponto apresentam resultados com correlação de 91,5%.

Sendo assim dependendo da finalidade, ambos os métodos são eficazes. Já o método que utiliza a média dos pontos apresentou os piores resultados visíveis e práticos, sendo ele descartado para as etapas finais.

O método de Xue, Zhang e Grift (2012), foi adaptado para o sistema e se mostrou eficiente e rápido, para a tomada de decisões. Com base na experiência acumulada durante a elaboração desta dissertação, e das observações feitas sobre todo o processo de trabalho, foi elaboradas algumas sugestões para trabalhos futuros, com o intuito de aprimorar a técnica utilizada e visando uma maior utilização da mesma entre a comunidade acadêmica.

- Validação do método através da *Lógica Fuzzy*, já que o sistema apresenta todas as matrizes de inferência. Podendo ser acrescento mais variáveis linguísticas.
- Alteração da comunicação *bluetooth* por comunicações mais eficazes, rápidas e de menor consumo de energia.
- Ampliação do sistema elétrico, para um sistema de maior potência, permitindo assim, a aplicação do sistema em máquinas de maior porte.

7 BIBLIOGRAFIA

ANDROID. The world`s most popular mobile platform. **Android Developer**, 2010. Disponivel em: <http://developer.android.com/about/index.html>. Acesso em: 22 Novembro 2014.

ASTRAND, B.; BAERVELDT, A. J. A vision based row-following system for agricultural field machinery. **Mechatronics**, 2005. 125-136.

BASTOS, A. **Instrumentação, Eletrônica Analógica e digital para Telecomunicações**. Rio de Janeiro: ANTENNA, 2002.

BELL, T. Automatic tractor guidance using carrier-phase differential GPS. **Computers and Electronics in Agriculture**, v. 25, p. 53 - 66, 2000.

BRADSKY, G.; KAEBLER, A. **Learning OpenCV:** Computer Vision with the OpenCV Library. 1. ed. [S.l.]: O`REILLY, 2008.

BRAGA, N. C. **Eletrônica básica para mecatrônica.** São Paulo: Saber, 2005.

CAMPOS, A. C. B. et al. **Agricultura de precisão no Brasil:** Adoção, resultados e perspectivas. 8 Curso de agricultura de precisión. [S.l.]: [s.n.]. 2014.

CÁSSIO, L. L. Técnicas de visão computacional aplicas ao reconhecimento de cenas naturais e locomoção autônoma em robôs agrícolas móveis. **Dissertação de Mestrado à Escola de Engenharia de São Carlos de São Paulo**, São Carlos, 2011.

CHEN, B.; TOJO, S.; WATANABE, K. Machine Vision for a Micro WeedingRobot in a Paddy Field. **Biosystems Engineering**, n. 85, p. 393 - 404, 2003.

DIAS, A. H.; SILVA, D. M. NAVIGO Robô autônomo para navegação agrícola. **Dissertação de mestrado do Programa de Pós-Graduação em Computação Aplicada, UEPG**, Ponta Grossa, 2013.

FORSYTH, D. A.; PONCE, J. **Computer Vision:** Modern Approach. [S.l.]: Prentice Hall, 2003.

GOMIDE, F. A. C.; GUDWIN, R. R.; TANSCHEIT, R. **Conceitos fundamentais da teoria de conjuntos , lógica e aplicações**. Cempinas: [s.n.], 2002.

GONZAGA, A. Notas de aula em visão computacional (disciplina Pós-Graduação). **Univercidade de São Paulo, Escola de Engenharia de São Carlos, Departamento de Engenharia Elétrica - Laboratório de Visão Computacional (USP/EESC/SEL/LAVI)**, São Carlos, 2010.

HANA, S.; ZHANG, Q.; NIC, B. R. J. A guidance directrix approach to vision-based vehicle guidance systems. **Computers and Electronics in Agriculture**, v. 43, p. 179 - 195, 2004.

INAMASU, R. Y. et al. Agricultura de precisão para sustentabilidade de sistemas produtivos do agronegócio brasileiro. **Agricultura de precisão: Um novo olhar. EMBRAPA**, 2009.

IOIO. Software, firmware and hardware of the IOIO - I/O for Android. **GitHub Ytai Ben-Tsvi**, 2013. Disponivel em: <https://github.com/ytai/ioio>. Acesso em: 2014.

JAYAS, D. S.; PALIWAL, J.; VISEN, N. S. Multi-layer neural networks for image analysis of agricultural products. **Journal of Agricultural Engineering Research**, Londres, 22, 2000. 119-128.

LACOMBE, F. J. M. **Dicinário de Administração**. [S.l.]: Saraiva, 2004.

LARSEN, W. E.; NIELSEN, G. A.; TYLER, D. A. Precision navigation with GPS. **Computers and Electronics in Agriculture**, v. 11, p. 85 - 95, 1994.

MACHADO, P. L. O. A. et al. Mapeamento da condutividade eletrica e rela~ao com a argila de Latossolo sob plantio. **Pesquisa Agropecuaria Brasileira**, Brasília, 2006. 1023-1031.

MARQUES, O. F.; N., V. H. **Processamento digital de imagens**. Rio de Janeiro: Brasport, 1999.

MEHMOOD, A. K. et al. Activity Recognition on Smartphones via Sensor-Fusion and KDA-Based SVMs. **International Journal of Distributed Sensor Networks**, 2014, 2014.

MENEGATTI, L. A.; MOLIN, J. P. A cana de e a Agricultura de Precisão. **Idea News**, v. 4, 2004.

MERCADANTE, E. Dinâmica espectral da cultura da soja ao longo do ciclo vegetativo e sua relação com a produtividade na região centro oste do Paraná. **Tese de Doutorado**, Campinas, 2007.

MONK, S. **Making Android Accesories with IOIO**. [S.l.]: O`Reilly, 2012.

MUNSELL, A. H. **Atlas of the Munsell color system**. [S.l.]: Howland & Company, 1915.

OPENCV. OpenCV. **OpenCV**, 2014. Disponivel em: <opencv.org>.

ORTIZ, J. M.; OLIVARES, M. A Vision Based Navigation System for an Agricultural Field Robot. **Department of Electronics, Technical University Federico Santa María**, Valparaíso.

PAZOS, F. **Automação de Sistemas e Robótica.** Rio de Janeiro: Axcel Books do Brasil, 2002.

PRADHAN, S. K.; PARHI, D. R.; PANDA, A. K. Logic techniques for navigation of several mobile robots. **Applied Soft Computing**, 2008. 290-304.

SIMÕES, A. S.; COSTA, A. H. R. **Classificação de cores por redes neurais artificiais:** um estudo do uso de diferentes sistemas de representação de cores no futebol de robôs móveis autônomos. Encontro Nacional de Intêligencia Artificial - ENIA`2001 - Anais do XXI Congresso da Sociedade Brasileirade Computão. Fortaleza: [s.n.]. 2001. p. 182.

SOUTO, R. P. Segmentação de imagem multiespectral utilizando-se o atributo matiz. **Dissertação de mestrado** , São José dos Campos , 2000.

SUBRAMANIAN, V.; BURKS, T. F.; ARROYO, A. A. Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. **Computers and Electronics in Agriculture**, v. 53, p. 130 - 143, 2006.

TEGRA. Android Development Pack Overview. **NVIDIA DEVELOPER ZONE**, 2013. Disponivel em: <https://developer.nvidia.com/tegra-android-development-pack>. Acesso em: Julho 2014.

USTEV, Y. E.; DURMAZ, O. I.; C., E. User, device and orientation independent human activity recognition on mobile phones: challenges and a proposal. **Proceedings of**

the ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication, New York, 2013. 1427–1436.

WILSON, J. N. Guidance of agricultural vehicles—a historical perspective. **Computers and Electronics in Agriculture**, v. 25, p. 3 - 9, 2000.

XUE, J.; ZHANG, L.; GRIFT, T. E. Variable field-of-view machine vision based row guidance of an agricultural robot. **Computers and Electronics in Agriculture**, v. 84, p. 85 - 91, 2012.

YUKUMOTO, O.; MATSUO, Y.; NOGUCHI, N. Robotization of agricultural vehicles (Part 2)—description of the tilling robot 34, 107–114. **Journal of Agricultural Engineering Research**, v. 34, p. 107 - 114, 2000.

ZHANG, N. . W. M.; WANG, N. Precision agriculture. **Computers & Electronics in Agriculture**, 2002. 113-132.