UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Estudo de Algoritmos de Visão Computacional para Identificação e Rastreamento de Linhas de Transmissão de Energia Elétrica com Multirotores

Luciano do Vale Ribeiro

Itajubá, 6 de Dezembro de 2019

UNIVERSIDADE FEDERAL DE ITAJUBÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Luciano do Vale Ribeiro

Estudo de Algoritmos de Visão Computacional para Identificação e Rastreamento de Linhas de Transmissão de Energia Elétrica com Multirotores

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação como parte dos requisitos para obtenção do Título de Mestre em Ciência e Tecnologia da Computação

Área de Concentração: Matemática da Computação

Orientador: Prof. Dr. Alexandre Carlos Brandão Ramos

6 de Dezembro de 2019
Itajubá - MG

Luciano do Vale Ribeiro

Estudo de Algoritmos de Visão Computacional para Identificação e Rastreamento de Linhas de Transmissão de Energia Elétrica com Multirotores/ Luciano do Vale Ribeiro. – Itajubá, 6 de Dezembro de 2019-

51 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Alexandre Carlos Brandão Ramos

Dissertação (Mestrado)

Universidade Federal de Itajubá

Programa de Pós-graduação em Ciência e Tecnologia da Computação, 6 de Dezembro de 2019.

1. Inspeção em Linhas de Transmissão. 2. Veículo Aéreo Não Tripulado. 3. VANT. 4. Multirotor. 5. ROS. 6. Gazebo.

CDU 07:181:009.3

Agradecimentos

Agradeço aos meus familiares, em especial à minha mãe Ivone e ao meu pai Maury, por sempre terem apoiado os meus estudos e à minha namorada Eliza, por todo apoio ao longo desta jornada.

Ao meu orientador Alexandre Carlos Brandão Ramos pelos ensinamentos compartilhados e pelo apoio na realização deste trabalho.

Agradeço aos colegas do laboratório LMI e da equipe Black Bee por toda ajuda prestada na realização dos testes.

À Universidade Federal de Itajubá, por fornecer um ambiente propício para realização deste trabalho e à CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pela concessão da bolsa de mestrado.

"Se eu vi mais longe, foi por estar sobre ombros de gigantes."
(Isaac Newton)

UNIVERSIDADE FEDERAL DE ITAJUBÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Luciano do Vale Ribeiro

Estudo de Algoritmos de Visão Computacional para Identificação e Rastreamento de Linhas de Transmissão de Energia Elétrica com Multirotores.

Dissertação aprovada por banca examinadora em 06 de Dezembro de 2019, conferindo ao autor o título de **Mestre em Ciência e Tecnologia da Computação.**

Banca Examinadora:
Prof. Dr. Alexandre Carlos Brandão Ramos - UNIFEI (Orientador)
Prof. Dr. Hildebrando Ferreira de Castro Filho - ITA
Prof. Dr. Roberto Affonso da Costa Júnior - UNIFEI
Prof. Dr. Roberto Claudino da Silva - UNIFEI

Itajubá
2019

Resumo

A rede de transmissão de energia elétrica brasileira requer inspeção e manutenção preventiva regular para garantir o seu fornecimento aos consumidores. As companhias transmissoras realizam inspeções regulares para identificar anomalias na rede de transmissão, como defeitos nos cabos, trincas nos isoladores e estruturas próximos da rede, como árvores, entre outros. Estas inspeções podem ser feitas por profissionais com uso de binóculos ou através de helicópteros tripulados contendo um conjunto de sensores, tais como câmeras infravermelho para detectar pontos de alto aquecimento, entre outros sensores.

O interesse no uso de Veículos Aéreos Não Tripulados (VANTs) para realizar inspeções em linhas de transmissão têm crescido nos últimos anos, devido ao baixo custo e aumento na segurança, se comparado com as inspeções realizadas com helicópteros tripulados. Com essa tecnologia, o VANT pode ser controlado via rádio controle por um piloto ou de forma semi-automática, com uma missão pré-programada no piloto embarcado da aeronave. Uma das grandes dificuldades em se realizar vôos autônomos está na dependência do sistema GPS (Global Positioning System) para obtenção da posição do VANT. O erro na medida da posição informada pelo GPS pode afetar o desenvolvimento da missão.

Este trabalho visa desenvolver uma técnica de processamento de imagens em tempo real para realizar a identificação e rastreamento das linhas de transmissão e desta forma determinar a sua posição relativa ao VANT. Sem conhecer exatamente a posição do VANT e das linhas de transmissão, mas sabendo a posição relativa entre os dois, pode ser possível corrigir a trajetória do VANT através de comandos enviados para o piloto automático da aeronave.

O projeto foi desenvolvido sobre o framework ROS (*Robot Operating System*) e o processamento de imagens foi realizado utilizando a biblioteca OpenCV. Para avaliar o desempenho da solução, criou-se um cenário virtual no simulador de robôs Gazebo, onde foi possível processar as imagens geradas por um multirotor durante uma missão de inspeção.

Palavras-chaves: Inspeção em Linhas de Transmissão, Veículo Aéreo Não Tripulado, VANT, Multirotor, ROS, Gazebo.

Abstract

The Brazilian electricity transmission network requires regular preventive inspection and maintenance to ensure its supply to consumers. Transmission companies carry out regular inspections to identify anomalies in the transmission network, such as cable defects, insulator cracks and structures close to the network, such as trees, among others. These inspections can be performed by expert personnel using binoculars or by manned helicopters containing a set of sensors, such as infrared cameras to detect hot spots, among other sensors.

Interest in using unmanned aerial vehicles (UAVs) to conduct transmission line inspections has grown in recent years due to the lower cost and increased safety compared to manned helicopter inspections. With this technology, the UAV can be controlled via radio control by a pilot or semiautomatically, with a pre-programmed mission in the aircraft's onboard pilot. One of the major difficulties in performing autonomous flights is the dependence on GPS (Global Positioning System) to obtain the UAV position. Errors in GPS position measurements can affect mission progress.

This project aims to develop a real-time image processing technique to identify and track transmission lines and thus determine their position relative to the UAV. Without knowing exactly the position of the UAV and the transmission lines, but knowing the relative position between them, it can be possible to correct the UAV trajectory through commands sent to the aircraft autopilot.

This project was developed under the ROS (Robot Operating System) framework and the image processing was performed using the OpenCV library. To evaluate the performance of the solution, a virtual scenario was created in the Gazebo robot simulator, which made it possible to process the images generated by a multirotor during an inspection mission.

Keywords: Power Line Inspection, Unmanned Aerial Vehicle, UAV, Multirotor, ROS, Gazebo.

Sumário

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ARP	Aeronave Remotamente Pilotada	14
CEMIG	Companhia Energética de Minas Gerais	14
CHESF	Companhia Hidroelétrica do São Francisco	14
ECS	Estação de Controle em Solo	15
GNSS	*Global Navigation Satellite System*	15
GPS	*Global Positioning System*	15
ITA	Instituto Tecnológico de Aeronáutica	14
LTs	Linhas de Transmissão	13
ONS	Operador Nacional do Sistema Elétrico	13
TH	Transformada de Hough	19
VANT	Veículo Aéreo Não Tripulado	14
VTOL	*Vertical Take-Off and Landing*	14

1 Introdução

O desenvolvimento da sociedade está ligado diretamente com a qualidade da energia elétrica prestada. No Brasil, por se tratar de um país com dimensões continentais, para que seja possível atender toda a área territorial de cerca de $8,5$ milhões km^2 (IBGE, 2019), há a necessidade de transmissão da energia gerada nas usinas até os centros de distribuição. Isso se deve também pelas suas características demográficas, que faz com que as regiões que produzem energia elétrica não sejam necessariamente as que mais consomem, necessitando de um meio de transporte dessa energia para os grandes centros.

Para isso, a energia gerada caminha por milhares de quilômetros sob alta tensão a fim de minimizar as perdas, até chegar nas distribuidoras, que são responsáveis pelo abaixamento, taxação e distribuição. Segundo dados do ONS (Operador Nacional do Sistema Elétrico), em 2017, o país possuía cerca de 141 mil km de LTs (Linhas de Transmissão) na categoria $A1$, ou seja, tensão maior ou igual a 230 kV com projeção de que em 2023 este valor chegue a 185 mil km (ONS, 2019).

Para garantir a qualidade e a confiabilidade desse sistema todas as atividades de manutenção e inspeção são regulamentadas pela ONS (ANEEL-ONS, 2019) e fiscalizadas pela Aneel (ANEEL, 2019). O primeiro passo é efetuar as inspeções que serão responsáveis por detectar falhas (quando a função requerida não é mais desempenhada) ou defeitos (alteração física ou química, mas que não impede de imediato a execução da função desempenhada) nas linhas ou equipamentos. A segunda etapa é efetuar as manutenções (preventivas, preditivas ou corretivas) constatadas ou previstas pelas inspeções, de forma a evitar falhas maiores no sistema. Fica a cargo das concessionárias criar um plano de inspeções e manutenções preventivas e preditivas de acordo com as características de suas estruturas, no entanto, a Aneel estabelece um plano mínimo de ação e periodicidade (SILVA, 2013). A Figura 1 mostra alguns defeitos encontrados em inspeções.

(a) Cabo com tentos rompidos. (b) Condutores danificados. (c) Espaçador frouxo.

Figura 1 – Defeitos em LTs (Adaptado: ADAMI, 2008)

As inspeções nas LTs buscam normalmente localizar defeitos, podem ser feitas geralmente de duas formas: terrestre ou aérea. A inspeção terrestre procura identificar

espaçadores soltos, isoladores trincados, quebrados, falta de peças, cabos parcialmente rompidos etc. Esta é realizada em grande parte com uma patrulha a pé ou de carro que percorre as LTs fazendo vistorias na parte inferior dos postes e das linhas, através de binóculos especiais e principalmente se atentando à vegetação que fica sob as linhas. Com a patrulha a pé consegue-se uma inspeção mais minuciosa, porém é muito demorada.

Já a inspeção aérea é realizada como forma de complementar a inspeção terrestre. É feita no Brasil ainda em grande parte por helicópteros que pairam sobre as linhas de forma a se obter informações mais precisas. Captam imagens com câmeras especiais, como infravermelho, para realizar análises posteriormente. A vantagem de se utilizar helicópteros, é a possibilidade de cobrir grandes distâncias em um único voo. A desvantagem deste método está no grande risco envolvido pela necessidade de voar próximo da linha viva e do custo da operação com helicópteros. Além disso, apesar da inspeção apresentar bons resultados elas podem não registrar todas as anomalias existentes no ativo (falha humana) além de depender da disponibilidade de pessoal treinado para as inspeções.

Inspeções aéreas podem ainda ser realizadas com drones, que no idioma português são conhecidos como VANT (Veículo Aéreo Não Tripulado) ou ARP (Aeronave Remotamente Pilotada), sendo eles não autônomos já que a regulamentação brasileira proíbe qualquer voo sem interferência de um piloto (ANAC, 2017). Tal objeto se trata de qualquer tipo de veículo aéreo que não precisa de piloto a bordo para que seja possível guiar e/ou controlar. As ações são tomadas remotamente através de centrais que utilizam meios eletrônicos e computacionais para efetuar tal supervisionamento e controle e isso pode ser feito com ou sem intervenção humana. A Figura 2 mostra algumas formas de inspeção.

As inspeções em LTs são um fator crucial para o bom andamento da rede elétrica já que, normalmente, as manutenções só ocorrem em caso de falha ou devido à alguma incoerência detectada durante inspeções. Uma inspeção de baixa qualidade pode ocasionar manutenções mal programadas e por consequência faltas na transmissão de energia, como é o caso do apagão que aconteceu no início de 2019 e deixou 13 estados sem energia. Segundo a Aneel, a concessionária responsável teve um desempenho inadequado do sistema de proteção e tinha falha no processo de manutenção da rede (EBC, 2019).

As inspeções de LTs podem ser feitas com VANTs do tipo asa-fixa, multirotores ou um misto dos dois, os conhecidos como VTOL (*Vertical Take-Off and Landing*). VANTs de asa-fixa ou VTOL têm a habilidade de voar com velocidades maiores, atingindo assim uma grande área de cobertura. No entanto, se utilizados para inspeção de LT, podem deixar de pegar alguns detalhes das torres ou dos cabos.

O uso de VANTs em inspeções de LTs tem crescido nos últimos anos. No Brasil, o ITA (Instituto Tecnológico de Aeronáutica) desenvolveu em conjunto com a CHESF (Companhia Hidroelétrica do São Francisco) um VANT de asa fixa para inspecionar 20 mil km de LTs de energia elétrica (ITA, 2019). Além disso, em 2016, a CEMIG (Companhia

(a) Inspeção com helicóptero (NEWAIR, 2019).

(b) Inspeção Terrestre (IRISS, 2019).

(c) Helicóptero utilizado pela CEMIG (CEMIG2, 2019).

(d) Inspeção com VANT (TEXASDRONE, 2019).

Figura 2 – Meios de Inspeção de LTs

Energética de Minas Gerais) realizou testes com VANTs para monitoramento de LTs (CEMIG, 2019). Já a Enel Rio utilizou em 2018 VANTs para inspeções obtendo redução em 75% no tempo gasto se comparado com a vistoria por terra (CANALENERGIA, 2019). Os VANTs para inspeções de LTs eventualmente permitirão melhorar a gestão de ativos, aumentar a produtividade das equipes de campo, reduzir perdas técnicas, além de melhorar a gestão ambiental.

O controle do VANT ao longo de um trajeto pode ser feito através de um piloto ou por missões pré-programadas numa ECS (Estação de Controle em Solo). As missões criadas em ECS são baseadas em balizas, ou coordenadas GPS. Os VANTs possuem uma antena receptora de sinal GPS e, desta forma, se movem entre estas coordenadas para cumprir a missão.

Embora o VANT possa ser operado remotamente por um piloto, é interessante a realização de seu voo de forma autônoma e monitorada. Os sistemas de navegação atuais se baseiam principalmente em sensores inerciais acoplados na aeronave, como acelerômetros, giroscópios, bússola e de um GNSS (*Global Navigation Satellite System*). Existem

situações onde o sinal de GPS (*Global Positioning System*) pode não ser confiável, seja devido a falha de instrumentos ou interferências externas, como a Anomalia Magnética do Atlântico Sul - AMASS. Outro fator é que as próprias coordenadas das torres de linhas de transmissão informadas pelas concessionárias podem não ser exatas, impossibilitando a criação de missões automáticas seguindo coordenadas de GPS.

Com o intuito de melhorar o sistema de navegação autônoma, uma das técnicas que tem sido estudadas e empregadas é a visão computacional. Com tal técnica é possível processar as imagens e colher informações tanto para estabilizar o VANT quanto para estimar sua localização em um ambiente desconhecido.

1.1 Objetivos

Este trabalho visa realizar a identificação e rastreamento das LTs utilizando processamento de imagens. Para isso, o algoritmo proposto irá determinar a posição relativa do VANT em relação aos cabos de transmissão de energia elétrica. Para tal, será necessária uma câmera fixa no corpo do VANT, apontada para baixo, a qual ficará constantemente capturando imagens.

As imagens coletadas passarão por uma técnica baseada na extração de bordas seguida da identificação das LTs utilizando Transformada de Hough. Além disso será utilizado a biblioteca OpenCV para aplicar os algoritmos de visão computacional. No VANT será instalado o computador embarcado Pixhawk, que é a unidade controladora de voo, onde é executado o *firmware* PX4, que contém toda a lógica para realizar a estabilização do voo e execução de missões.

Além disso é possível realizar a comunicação entre o PX4 e outro computador que estará executando os processos de visão computacional. A troca de dados entre o computador e o PX4 é feita através de uma interface serial utilizando o protocolo Mavlink. Todas essas soluções serão integradas utilizando o *framework* ROS e serão testadas em ambiente virtual com o simulador de robôs Gazebo.

1.1.1 Objetivo Geral

Desenvolver um algoritmo para rastreamento de LTs com um VANT utilizando imagens de uma câmera embarcada como entrada e realizando processamento de imagens.

1.1.2 Objetivos Específicos

- Utilizar a biblioteca de código aberto OpenCV para realizar o processamento de imagens.

- Aplicar filtros para remoção de ruído e de extração de bordas da imagem e comparar seu desempenho.
- Aplicar a Transformada de Hough para detecção de retas na imagem.
- Implementar uma solução para identificação das LTs e seu rastreamento.

1.2 Contribuições desta dissertação

As principais contribuições deste trabalho são:

- Desenvolvimento de um programa de processamento de imagens para rastreamento de LTs, utilizando OpenCV e o framework ROS, que ficará disponível para a comunidade.
- Criação de um cenário para testes no simulador de robôs Gazebo.
- Integração da solução para ser utilizada na placa controladora Pixhawk juntamente com o *firmware* PX4.

1.3 Organização do trabalho

O restante deste trabalho é organizado da seguinte maneira:

No capítulo 2 são detalhados os fundamentos básicos do processamento de imagens e é feita uma revisão da literatura relacionada à identificação e rastreamento de LTs de energia no contexto de inspeções.

No capítulo 3 são explicados os algoritmos utilizados neste projeto para a identificação e rastreamento de LTs.

No capítulo 4 é explicado como foram realizados os testes e a análise dos resultados e o Capítulo 5 conclui o trabalho.

2 Materiais e Métodos

Neste capítulo será vista uma breve descrição de processamento de imagens e os algoritmos de processamento de imagens utilizados neste trabalho. A última seção conclui com uma revisão dos trabalhos relacionados.

2.1 Processamento Digital de Imagens

Uma imagem pode ser definida como uma função bidimensional, $f(x, y)$, onde x e y são as coordenadas de um plano que define a imagem, e a amplitude de f em qualquer par de coordenadas (x, y) é chamado de intensidade ou nível de cinza da imagem naquele ponto. Quando x, y, e os valores de f são finitos e discretos, dizemos que a imagem é uma imagem digital. Cada ponto (x, y) que forma a imagem é conhecido como *pixel*.

Processamento digital de imagens é o conjunto de operações realizadas sobre os valores dos pixels a fim de melhorar características visuais (aumentar contraste, melhorar foco, reduzir ruído, eliminar distorções), extrair elementos de interesse, ou mesmo transformar a imagem, criando efeitos visuais.

A Figura 3 exibe as etapas de um sistema de processamento digital de imagens. Em cada etapa há uma série de algoritmos e técnicas disponíveis e seu uso depende em geral do problema a ser resolvido.

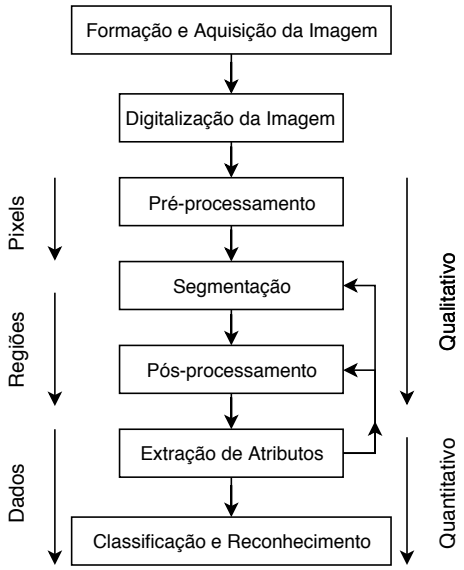


Figura 3 – Etapas realizadas num PDI (Adaptado de (IGNACIO, 2013))

Como os algoritmos utilizados em cada etapa são escolhidos de acordo com o objetivo a ser atingido, foi executado no início do projeto um estudo do que poderia ser

feito para atingir o objetivo de detecção e rastreamento de LTs. Para a detecção das retas, escolheu-se a TH (Transformada de Hough). Uma etapa anterior à TH seria a detecção de bordas. Para extração de bordas da imagem, foram estudados alguns algoritmos, e dentre estes temos os filtros Sobel, Laplaciano e o detector Canny. Uma das dificuldades encontradas foi extrair os objetos de interesse do fundo da imagem. Estudou-se algoritmos capazes de borrar a imagem, mas que, de certa forma, preservassem as bordas das mesmas.

2.1.1 Pré-processamento

As etapas existentes dentro do pré-processamento são operações que são executadas sobre uma imagem e a saída é uma imagem. O pré-processamento pode servir para realizar a conversão da imagem para tons de cinza, remoção de ruído, realce de bordas, conversão entre espaços de cores etc. Em alguns casos, sem a realização de pré-processamento, outras etapas seriam afetadas, como a extração de atributos.

2.1.1.1 Filtragem de ruído

Filtragem é uma das operações mais fundamentais do processamento de imagens, segundo (TOMASI; MANDUCHI, 1998). Em geral, a redução de ruído ou suavização da imagem para remover detalhes não interessantes tem um efeito muito importante na qualidade das outras etapas do processamento de imagem, como segmentação, extração de atributos etc.

A suavização das imagens neste projeto é feita com o objetivo de reduzir o ruído e borrar um pouco a imagem para que a próxima etapa de detecção de bordas possa ser mais eficaz. Para isto foram utilizados filtros espaciais. Cada filtro é representando por uma máscara e esta percorre a matriz da imagem. O pixel localizado logo abaixo da máscara será substituído na imagem resultante como sendo o somatório do produto dos pixels cobertos por esta máscara. A Figura 4 ilustra a operação de aplicação de uma máscara de média entre a matriz I com máscara K.

Foram utilizados três filtros neste trabalho. Filtro de média, gaussiano e bilateral. Nas próximas subseções teremos uma descrição breve de cada um.

Filtro de Média

O filtro de média consiste em substituir o pixel $f(x, y)$ pela média de seus vizinhos. Desta forma, é possível suavizar a imagem, reduzindo os seus detalhes. Uma desvantagem de seu uso está no fato de borrar as bordas da imagem, visto que uma borda é caracterizada pela mudança brusca dos tons de cinza.

Na Figura 4 foi aplicada uma máscara com tamanho 5. A expressão fora da matriz ficaria como $\frac{1}{25}$ para realizar a normalização do filtro. O filtro de média é eficaz na remoção

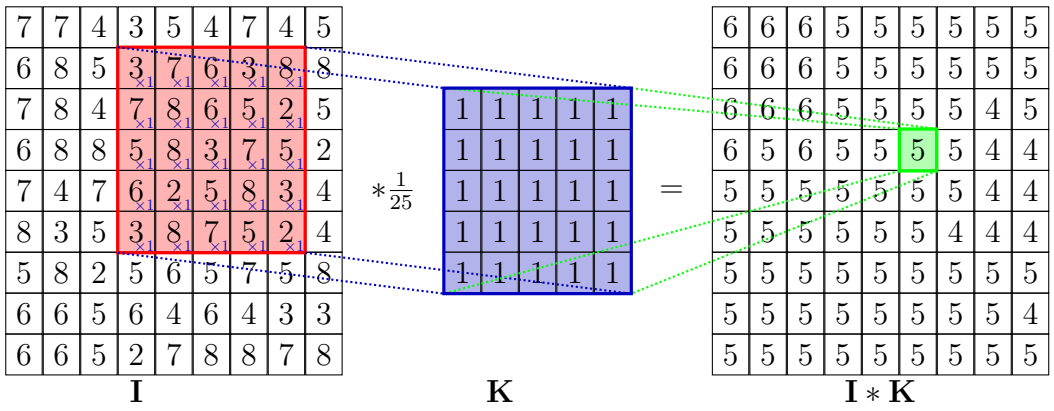


Figura 4 – Exemplo da convolução entre uma matriz I e uma máscara K (Adaptado: (PETARV, 2019))

de ruído porém nos pixels perto das bordas de um objeto na imagem, onde há uma grande diferença entre os tons de cinza, este filtro irá realizar a média desta região, borrando a borda ali presente.

Filtro Gaussiano

Este filtro, ao invés do filtro de média, utiliza uma máscara onde seus coeficientes seguem uma distribuição gaussiana. Desta forma, é dado um maior peso aos pixels vizinhos mais próximos do pixel central. A Figura 5 contêm um exemplo de uma máscara usada no filtro gaussiano.

$\frac{1}{16}$	$\frac{2}{16}$	$\frac{1}{16}$
$\frac{2}{16}$	$\frac{4}{16}$	$\frac{2}{16}$
$\frac{1}{16}$	$\frac{2}{16}$	$\frac{1}{16}$

Figura 5 – Máscara de um filtro Gaussiano (Fonte: autor)

Filtro Bilateral

Uma das desvantagens dos filtros anteriores está na característica de suavizar as bordas da imagem, indo contra o objetivo de preservá-las, já que, possuem grande importância na detecção de retas.

O filtro Bilateral (TOMASI; MANDUCHI, 1998) possui a característica de suavizar a imagem enquanto preserva as suas bordas. O seu funcionamento é baseado em duas gaussianas. Uma para filtragem espacial e outra para as cores dos pixels. Ou seja, o valor do pixel filtrado localizado no centro da máscara será determinado pelo somatório do produto de seus vizinhos pelos coeficientes da máscara, porém mesmo os pixels próximos podem ter um coeficiente baixo se a diferença de sua cor para o pixel central for grande.

Na Figura 6 é visto este comportamento. No canto esquerdo tem-se uma imagem ruidosa, bem definida. No centro, a máscara se refere a um pixel no topo da borda, próximo do centro da imagem. Pode-se ver nesta máscara que para os pixels que estão do mesmo lado da borda, a máscara tem um decaimento gaussiano nos pesos e do outro lado da borda, como o nível de cinza é muito menor que do pixel central, os coeficientes da máscara são baixos. No canto direito tem-se a imagem filtrada.

Uma desvantagem deste filtro está na sua complexidade, sendo mais lento que os outros.

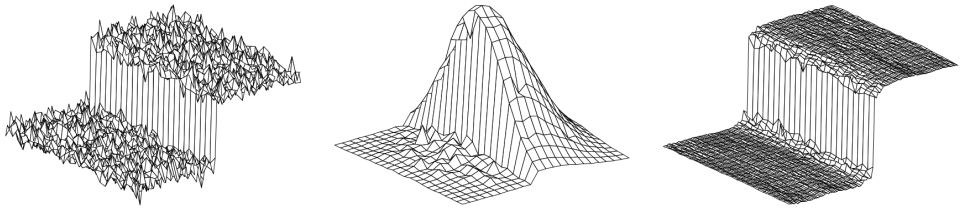


Figura 6 – Visualização do filtro Bilateral (Fonte: (TOMASI; MANDUCHI, 1998))

2.1.2 Segmentação

Na segmentação, temos como entrada uma imagem e em sua saída atributos extraídos da imagem. Segmentação subdivide a imagem em suas regiões constituintes ou objetos. Algoritmos de segmentação são baseados geralmente em duas propriedades: descontinuidade e similaridade.

Neste projeto, a imagem original foi segmentada em suas bordas, para poder facilitar na etapa futura a determinação das retas presentes.

Nas próximas subseções serão vistos os algoritmos de extração de bordas utilizados neste projeto.

Detector Sobel

O filtro Sobel faz o uso da primeira derivada para realçar bordas na imagem. A sua ideia é simples. Tomando o caso de $1D$ como ilustração, uma região contínua tem como derivada o valor zero (ausência de bordas), e a variação abrupta dos níveis de cinza são detectados pela derivada. Na prática, as bordas têm um formato de uma rampa. Ruídos na imagem (valores fora da faixa de sua vizinhança), quando derivados irão gerar um impulso. Nestes casos o ideal é antes de calcular a derivada de uma imagem aplicar um filtro de remoção de ruído.

O valor da derivada de uma imagem f no ponto (x, y) é $\nabla\mathbf{f}$, calculado por:

$$\nabla\mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{2.1}$$

A magnitude da derivada é determinado pela equação

$$\nabla f = \left(G_x^2 + G_y^2\right)^{\frac{1}{2}} \tag{2.2}$$

porém esta equação requer muita computação, sendo na prática aproximada por

$$\nabla f \approx |G_x| + |G_y| \tag{2.3}$$

O cálculo da derivada na imagem precisa ser feito de forma discreta. Desta forma, o cálculo da derivada no eixo x é feito por $G_x = f[x+1] - f[x]$ e de forma análoga no eixo y.

O filtro Sobel é implementado utilizando duas máscaras, vistas na Figura 7. O valor 2 e -2 utilizado nas máscaras tem como objetivo dar um peso maior para os pixels mais próximos do ponto onde se deseja calcular a derivada.

-1	0	$+1$
-2	0	$+2$
-1	0	$+1$

G_x

$+1$	$+2$	$+1$
0	0	0
-1	-2	-1

G_y

Figura 7 – Máscaras utilizadas pelo operador Sobel

Após a aplicação do filtro Sobel, tem-se uma imagem em tons de cinza, onde os tons mais claros representam as bordas mais fortes, enquanto que os mais escuros ausência de bordas.

Detector Laplaciano

O filtro Laplaciano faz o uso da derivada de segunda ordem para realçar as bordas de uma imagem. Ele é muito sensível ao ruído, sendo necessário realizar filtragem da imagem antes de sua aplicação.

O Laplaciano de uma função $2D$ é definido pela equação abaixo:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.4}$$

onde em sua forma discreta, a derivada parcial de segunda ordem calculada no eixo x é dada por:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \tag{2.5}$$

e no eixo y:

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y) \tag{2.6}$$

.

Na prática, o filtro Laplaciano é implementado através de uma máscara e o cálculo da derivada é feita pela convolução da máscara com a imagem. As derivadas parciais de segunda ordem estão representadas na máscara da Figura 8(a). O filtro Laplaciano tem a característica de ser isotrópico, onde a máscara da Figura 8(a) calculará os mesmos resultados se a imagem for rotacionada de ângulos de 90 graus. Na Figura 8(b) temos a sua versão que também calcula a derivada nas diagonais.

0	1	0
1	−4	1
0	1	0

(a)

1	1	1
1	−8	1
1	1	1

(b)

Figura 8 – Máscaras utilizadas para implementar o filtro Laplaciano. Em (a) temos uma das máscaras para implementar a versão isotrópica para as rotações de 90 graus e em (b) a máscara isotrópica para as rotações de 45 graus

Detector Canny

Segundo o criador do detector de bordas Canny (CANNY, 1986), um bom detector deve ter as seguintes características:

- Baixa taxa de erro: não deixar de detectar bordas da imagem e também não detectar falsas bordas.
- Boa localização: A distância entre os pixels da borda detectada e da borda real devem ser a mínima possível.
- Resposta mínima: Encontrar apenas uma borda para cada borda da imagem, ou seja, evitar múltiplas respostas para uma mesma borda.

A implementação do detector Canny segue os seguintes passos:

1. Filtragem de ruído: É utilizado um filtro gaussiano para eliminação de ruído.
2. Cálculo do gradiente da imagem. Utiliza-se uma das máscaras para cálculo de gradiente, Sobel por exemplo. São calculadas duas matrizes, $G(i, j)$ que contêm a magnitude do gradiente e $\Theta(i, j)$ que contêm a direção do gradiente.

3. Aplicação do operador Sobel 3×3 nas direções x e y, como visto na Figura 7 e em seguida calcula-se o módulo do gradiente, aproximado por $G = |G_x| + |G_y|$, e a direção da borda para cada pixel como $\theta = \arctan(\frac{G_y}{G_x})$. A direção é aproximada para uma das 4 direções a seguir: 0, 45, 90 e 135.

4. Aplicação da supressão não máxima. Pixels pertencentes a uma borda são vistos como máximos locais da matriz $G(i, j)$. Para evitar a determinação de duas bordas para uma única borda real, é realizado a supressão dos valores não máximos locais. Ou seja, um pixel A será suprimido se este estiver próximo de um máximo local B e ambos tiverem a borda na mesma direção.

5. Cálculo da histerese: São utilizados dois valores de limiares, $T1$ e $T2$, para determinar se o gradiente de um pixel é ou não uma borda. Se a magnitude do gradiente de um pixel for menor que $T1$, este não é borda. Se for maior que $T2$, este pixel pertence a uma borda. Se estiver entre $T1$ e $T2$, este pixel será borda somente se ele estiver conectado a um pixel que é borda.

2.1.3 Extração de Atributos

A TH (HOUGH, 1962; DUDA; HART, 1972) é um algoritmo utilizado para encontrar curvas que possam ser parametrizadas por uma equação bem definida (retas, círculos, elipses etc.).

Como pré-requisito para este algoritmo tem-se que a imagem a ser processada deve ser binária, de tal forma que os pixels brancos representem pontos candidatos a pertencer a uma reta e os pixels pretos representem o fundo da imagem.

O objetivo da TH é extrair da imagem os parâmetros que definem as retas ali presentes. Um limiar T é utilizado e somente as retas que tiverem pelo menos T pontos colineares serão selecionados.

Uma solução trivial seria para cada par de pontos ($\approx n^2$), determinar quantos pontos são colineares a estes dois. Esta abordagem realizaria cerca de n^3 operações e, é inviável na maioria dos casos.

O algoritmo proposto em (DUDA; HART, 1972) modela cada reta através da equação:

$$\rho = x \cos\theta + y \sin\theta, \tag{2.7}$$

onde ρ representa a distância perpendicular da reta até a origem do sistema de coordenadas da imagem e θ a inclinação da mesma.

O algoritmo proposto em (DUDA; HART, 1972) é realizado da seguinte forma:

1. Os valores possíveis para ρ e θ são discretizados nos intervalos $[\rho_{min}, \rho_{max}]$ e $[\theta_{min}, \theta_{max}]$.

2. Uma matriz M é criada, onde o número de linhas e de colunas suporta a quantidade de valores distintos para ρ e para θ determinados no passo 1.

3. Para cada ponto (x_i, y_i) da imagem (pixel branco) e para cada valor discretizado de θ_j, calcula-se o ρ_j correspondente, de acordo com a equação 2.7 e a posição $M[\rho_j][\theta_j]$ é incrementada de 1.

4. Ao final deste processo, o valor acumulado numa célula da matriz $M[\rho][\theta]$ representa a quantidade de pontos colineares da imagem presentes na reta definida pelos parâmetros ρ e θ.

5. Em seguida são escolhidas as células que possuem o valor acumulado maior ou igual ao limiar T.

Temos na Figura 9, em (a) a representação de cinco retas no plano cartesiano e em (b) as suas representações como pontos no espaço de Hough. É possível notar que as três retas paralelas e igualmente espaçadas geram no espaço de Hough pontos alinhados no eixo θ e distantes de forma regular no eixo ρ.

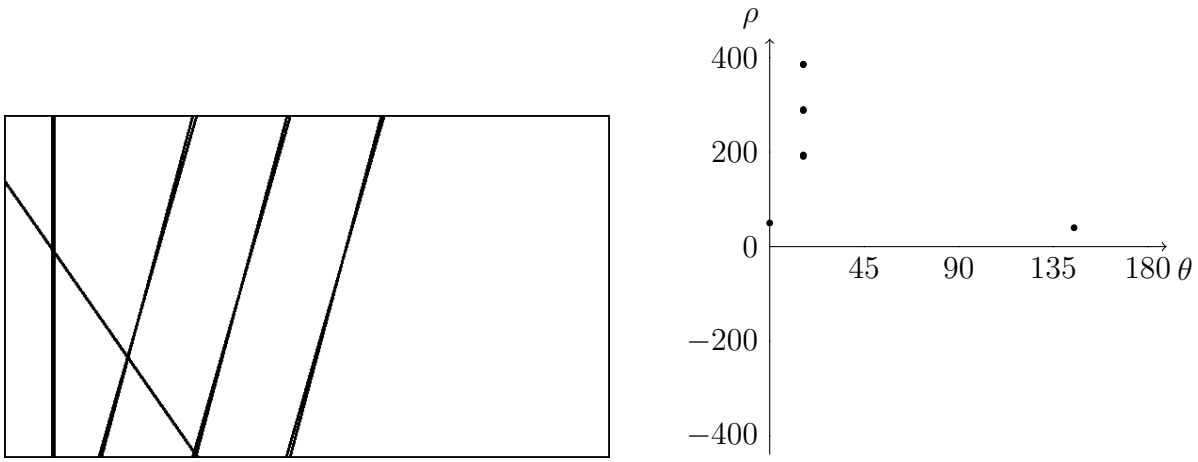


Figura 9 – Espaço de Hough

2.2 Trabalhos Relacionados

Os trabalhos sobre inspeção em LT com VANTs são divididos em dois grupos. O primeiro com foco em técnicas para identificação das LTs e outro no desenvolvimento de técnicas para controle da aeronave, dado a linha que deve ser seguida. Para a identificação e seguimento da LT, algumas considerações são feitas:

- Linhas de transmissão são objetos lineares quando vistos de cima.

- Elas possuem um brilho uniforme devido ao material que são produzidas.
- Linhas de transmissão são paralelas umas com as outras.
- Elas ocupam a maior parte da imagem.

Dadas essas características, a maioria dos algoritmos para detecção de linhas de transmissão são baseados na Transformada de Hough e em alguma outra técnica para agrupar as retas detectadas, para eliminar as que não pertencem às linhas de transmissão. Dado que a imagem é capturada com a câmera voltada para baixo, a grande dificuldade é extrair corretamente as retas que representam os cabos do fundo da imagem, que é o solo.

Zhengrong Li et al. utiliza em (LI et al., 2008; LI et al., 2010) um filtro PCNN *(Pulse-Coupled Neural Network)* com o objetivo de remover do fundo da imagem ruídos durante o processo de identificação das LTs. Então a TH foi aplicada para a identificação de retas e uma técnica de clusterização conhecida como K-means foi empregada para agrupar as retas que possuem o mesmo ângulo de inclinação. Sua implementação foi testada com dados reais capturados por uma aeronave, mas não foi utilizada para guiar um VANT.

Yang et al. também utilizou a TH para detectar retas em uma imagem binarizada (YANG et al., 2012). Fuzzy C-means foi utilizado como método de clusterização, tendo como características o tamanho e a inclinação das retas.

Mills e Aouf (MILLS; AOUF; MEJIAS, 2013) desenvolveram um sistema de controle baseado em processamento de imagem para correção da orientação de um VANT de asa fixa, para seguir uma LT na presença de vento. O algoritmo foi testado em ambiente simulado.

Zhang et al. fez um importante trabalho na área de detecção e rastreio de LT para ser utilizado em tempo real (ZHANG et al., 2012). Em adição à TH e K-means para clusterização, o filtro de Kalman foi aplicado para realizar o rastreio das retas no espaço de Hough. O tempo de processamento de cada frame ficou em 40ms, ideal para ser utilizado em sistemas em tempo real, porém o *hardware* utilizado não foi informado.

Araar e Nabil (ARAAR; AOUF, 2014) desenvolveram duas soluções, IBVS e PPBVS. Suas propostas foram testadas em simulação e também com VANT real. Não foi dado muito enfoque na detecção e rastreio de linhas de transmissão. Também não foram considerado casos com um plano de fundo da imagem complexo. No entanto, a forma como a detecção foi feita serviu de base para este projeto.

Golightly e Jones (GOLIGHTLY; JONES, 2005) propõem em seus trabalhos um VANT do tipo duto capaz de extrair energia elétrica da própria rede que está sendo inspecionada. Desta forma, o veículo aéreo estaria limitado a voos perto da linha de

transmissão, facilitando a sua aceitação pelos órgãos regulamentadores. Para manter o veículo no trajeto correto, servo visual é aplicado, utilizando a transformada de Hough para extrair características das linhas de transmissão e serem usadas para o rastreamento.

3 Desenvolvimento

Neste capítulo será visto como o problema do rastreamento de LTs foi modelado assim como a solução proposta. Também será visto a arquitetura do software e detalhes de sua implementação.

3.1 Modelagem

É proposto neste trabalho um algoritmo capaz de, dadas as imagens capturadas por uma câmera embarcada no VANT, identificar e rastrear as LTs através do processamento de imagens. Desta forma, é possível calcular a distância horizontal e a direção do VANT para as LTs. Com estes dados será possível ajustar a sua trajetória, mantendo a LT sob o campo de visão do VANT. Na Figura 10, temos uma ilustração do cenário proposto para este projeto.

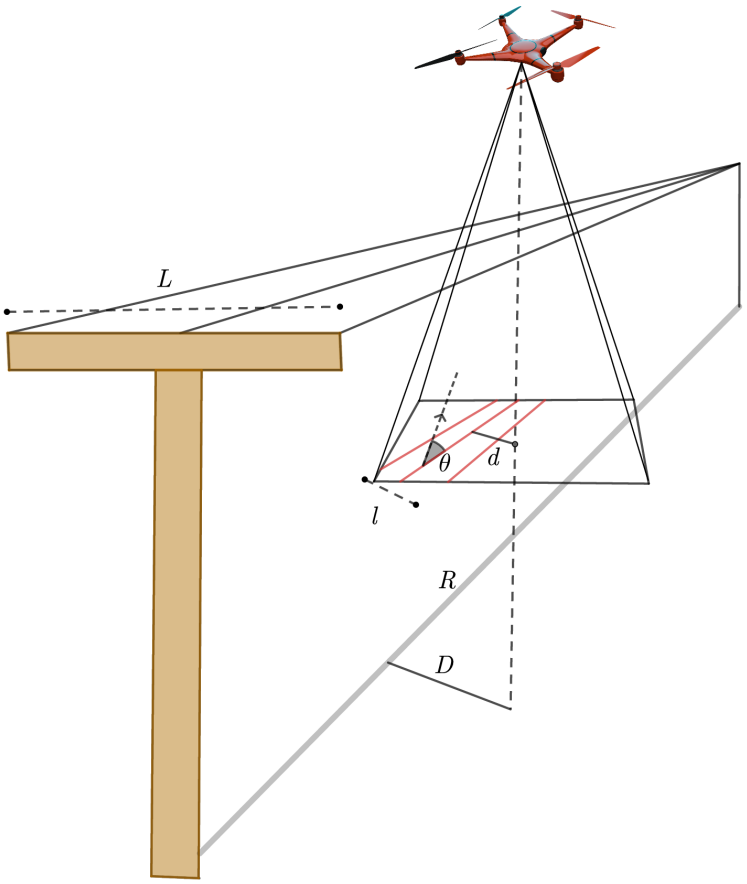


Figura 10 – Modelagem da solução proposta (Fonte: autor)

Seja a reta R a projeção da LT central no solo e D a menor distância desta reta para a projeção do VANT. Assumindo que a imagem capturada está paralela ao plano

do solo, a projeção do VANT no solo passará pelo centro da imagem. Desta forma, a distância D no plano da imagem passa a ser d, que é a distância em pixels do centro da imagem até a LT central. O valor de θ representa o erro da direção em relação as LTs.

Após determinar o valor de d, é necessário descobrir o valor de D. No entanto, em um processo de correção de trajetória, o valor de d não poderia ser utilizado na íntegra, visto que este está relacionado com a altura do VANT relativa à LT. Em outras palavras, quanto mais alto estiver menor será o d, porém o valor de D será o mesmo. Sendo assim, tendo em vista essa situação, foi utilizada uma abordagem simples para a correlação destas medidas. Para isso é utilizado a medida de um objeto de referência que, neste caso, optou-se por utilizar a distância máxima entre os cabos da LT, a medida L dada em metros. Dessa forma, no processamento de imagens também deve ser calculado o valor de l em pixels.

A medida D, em metros, é determinada pela Equação 3.1:

$$D = \left(\frac{L}{l}\right) d \tag{3.1}$$

Para que o algoritmo funcione, dois pré-requisitos são necessários:

- A câmera embarcada deve estar em um estabilizador próprio conhecido como *gimbal*. Desta forma ela ficará estabilizada apontada para baixo, com o plano da imagem paralelo ao solo. Caso estivesse fixa no corpo do VANT, quando este se movimentasse, a imagem não ficaria mais paralela ao solo, alterando assim o valor de d, levando a um cálculo errado de D.
- Para o cálculo da distância D, assume-se que seja conhecido a distância L entre os cabos extremos das LTs.

3.2 Arquitetura

No desenvolvimento deste projeto foi utilizado o framework ROS. Com ele é possível executar os programas de forma distribuída em mais de uma máquina na rede local ou até mesmo através da Internet. Cada programa é conhecido como nó. A troca de mensagens entre os nós é feita através de tópicos e cada tópico permite o transporte de um tipo de dado específico, definido pelas bibliotecas do ROS. Podem ser dados primitivos (inteiros, reais, vetores) ou compostos.

Para que os nós possam se comunicar, é necessário que eles se localizem de alguma forma. Para isto, existe um nó chamado de Master. Sempre que um nó é iniciado, ele deve se comunicar com o Master para indicar em quais tópicos ele irá publicar e assinar. Conforme os nós iniciam, o Master vai orquestrando a descoberta deles. Desta forma, os

nós passam a se comunicar diretamente. Caso o processo do Master se encerre os demais nós serão encerrados automaticamente.

Outro motivo da escolha do ROS foi o fato dele permitir a criação de nós de forma isolada. Além disso ele possui uma gama grande de ferramentas que podem ser utilizadas para testes, como o *rosbag*, utilizado para gravação de logs de tópicos e posteriormente a reprodução destes dados, bastante utilizado nesse projeto e o *rviz*, utilizado para visualização de dados de sensores em um mundo $3D$, entre outras ferramentas.

Com o objetivo de rastrear as linhas de transmissão, foram criados 2 nós principais: *image_proc* e *line_tracking*. O primeiro é responsável pela filtragem de ruído, extração de bordas e identificação de retas. Estas etapas se enquadram em Pré-Processamento, Segmentação e Extração de Atributos, respectivamente, vistas na Figura 3, no Capítulo 2. O segundo nó criado se enquadra em Classificação e Reconhecimento, onde dadas as retas identificadas no nó *image_proc*, este irá determinar quais delas são linhas de transmissão e realizar o seu rastreio.

A Figura 11 contém o diagrama dos nós e a sequência de processamento realizada. As elipses representam os nós do ROS e os retângulos são os tópicos. As linhas direcionadas representam o fluxo das mensagens.

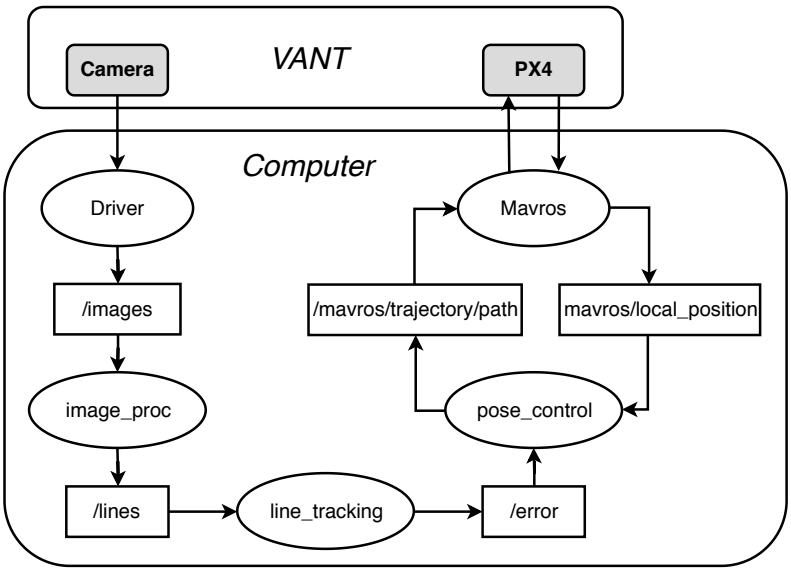


Figura 11 – Arquitetura do software desenvolvido (Fonte: autor)

A implementação se inicia pelo nó *image_proc* assinando no tópico da câmera, `/images`. Este realiza o pré-processamento, onde as retas são extraídas, e as publica no tópico `/lines`. Em seguida o nó *line_tracking* identifica quais das retas são realmente linhas de transmissão, levando em conta a posição, orientação das retas atuais e as últimas retas identificadas de imagens antigas. Após identificar as LTs são determinados os valores de d, θ e l. Em seguida, com tais valores e conhecendo L, calcula-se D usando a equação 3.1. Os valores de θ e D, erros do VANT em relação à LT, são então publicados no tópico

`/error`.

O último nó, que não foi implementado neste projeto, seria o *pose_control*. Este nó receberia como dados a posição relativa do VANT para as LTs, determinada pelos nós anteriores, sua posição estimada pelo piloto automático e a trajetória da missão e teria como saída a trajetória corrigida. O nó *Mavros*, já desenvolvido pela comunidade do ROS, seria utilizado para implementar o protocolo Mavlink e fazer a conversão das mensagens para o formato esperado pelo PX4 e pelo ROS.

3.3 Processamento de Imagens

O nó *image_proc* é responsável por receber as imagens da câmera, realizar a conversão da imagem para tons de cinza, aplicar um filtro para redução de ruído, efetuar a identificação de bordas e posteriormente identificar as retas presentes na imagem. Na Figura 12 temos o diagrama de estados do nó *image_proc*.

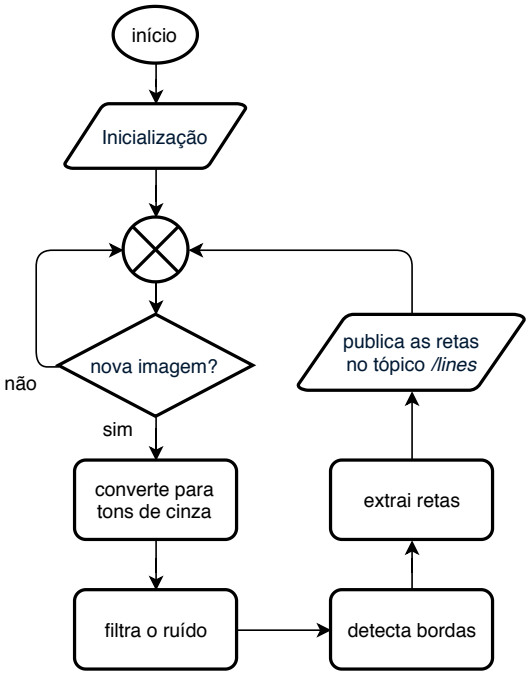


Figura 12 – Fluxograma do algoritmo de processamento de imagens do nó *image_proc* (Fonte: autor)

O processamento de imagens é realizado por *callbacks*, ou seja, o nó fica ouvindo o tópico esperando por uma nova mensagem. Quando esta chega, ele realiza o processamento e volta a esperar por outra mensagem. As mensagens recebidas no tópico */images* contêm dados de uma imagem no formato do ROS. Para que seja possível efetuar o processamento com OpenCV é necessário realizar a conversão utilizando o pacote *CV_Bridge*.

Antes de efetuar a remoção de ruído e extração de bordas, é feita a conversão da imagem para tons de cinza. Desta forma reduz-se a complexidade das etapas futuras.

Os métodos para redução de ruídos na imagem, descritos na Seção 2, são implementados pelo OpenCV usando os seguintes métodos: *cv2.blur*, *cv2.GaussianFilter* e *cv2.bilateralFilter*. Já para extração de bordas, os métodos utilizados foram: *cv2.Sobel*, *cv2.Laplacian* e *cv2.Canny*. Sendo que, para o detector Canny, utilizou-se como limiares os valores 100 e 200, escolhidos de forma empírica. No detector Sobel, calculou-se apenas a derivada no eixo X da imagem, de forma que se obtivesse apenas as componentes verticais das bordas. Já o método para cálculo do filtro Laplaciano não foram utilizados parâmetros adicionais.

Foi necessário escolher o tamanho do *kernel* dos filtros de ruído a serem utilizados. Para isto foram feitos testes comparando par a par os tipos de algoritmos para remoção de ruído e de detecção de bordas. Utilizou-se para testes os tamanhos de 3 e 5 para o *kernel*. A Figura 13 contém uma imagem da simulação utilizada neste teste. As Figuras 14 e 15 contêm os resultados do processamento. Após análise, optou-se por escolher o *kernel* de tamanho 3, pois foi o que permitiu a detecção de retas na maioria das combinações.

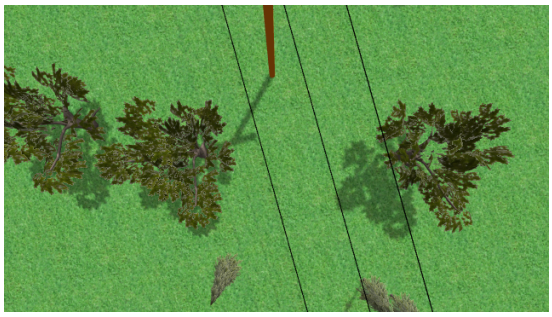
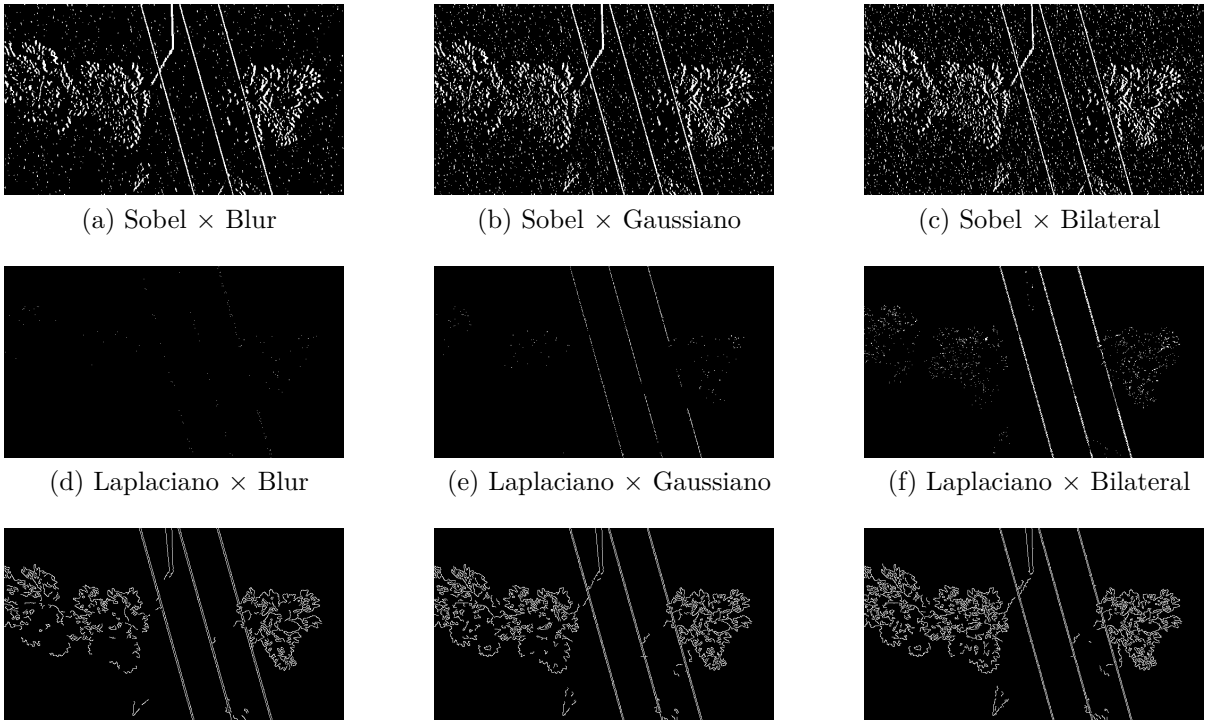


Figura 13 – Imagem capturada da simulação para testes dos algoritmos de filtragem e detecção de bordas (Fonte: autor)

A detecção de retas é feita pela Transformada de Hough. Este algoritmo, também presente na biblioteca do OpenCV, recebe como entrada uma imagem binária. Os pixels brancos da imagem são possíveis candidatos a pertencer a uma reta. Como foi visto no Capítulo 2, existe o limiar utilizado para indicar a quantidade mínima de pontos necessários para compor uma reta. Neste projeto foi utilizado o valor 150 para este limiar.

Na Figura 16, temos alguns exemplos da aplicação da TH. É possível notar nas imagens *(a)* e *(b)* que a não aplicação de filtro de remoção de ruído levará a detecção de muitas bordas do fundo da imagem, ocasionando a detecção de muitas retas após a TH.

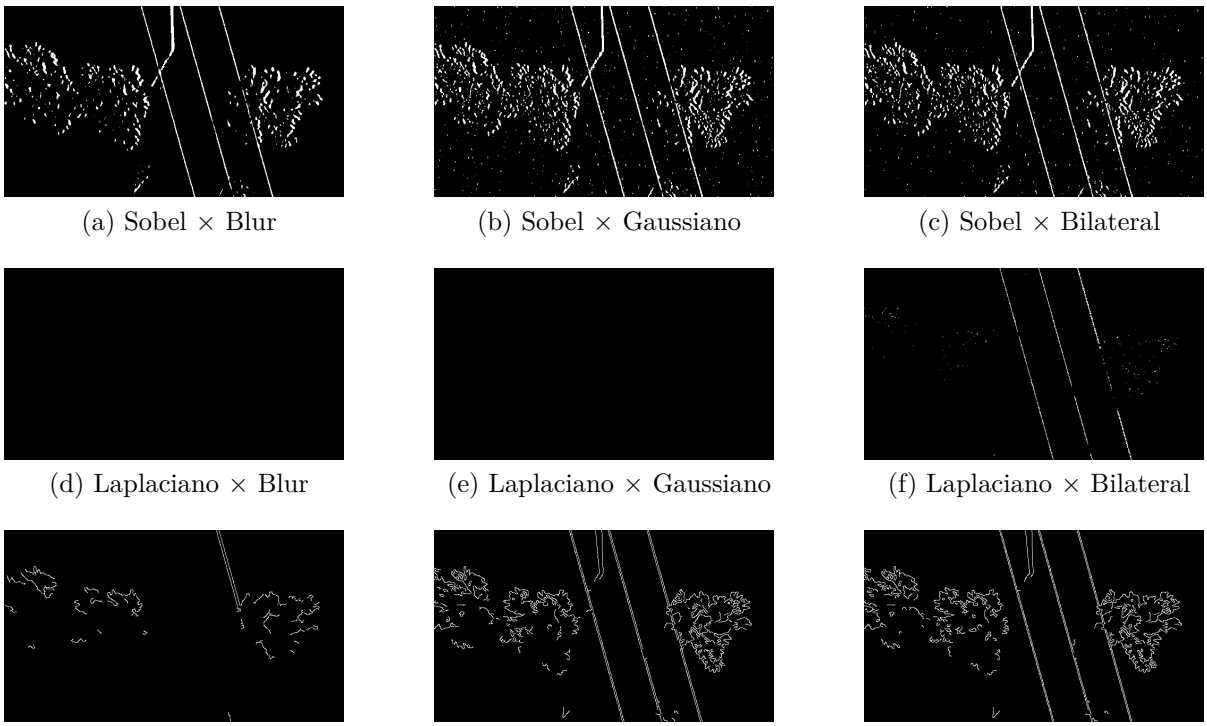


(a) Sobel × Blur (b) Sobel × Gaussiano (c) Sobel × Bilateral

(d) Laplaciano × Blur (e) Laplaciano × Gaussiano (f) Laplaciano × Bilateral

(g) Canny × Blur (h) Canny × Gaussiano (i) Canny × Bilateral

Figura 14 – Aplicação dos detectores de bordas Sobel, Laplaciano e Canny numa imagem filtrada cujo filtro filtro possui *kernel* de tamanho 3 (Fonte: autor)

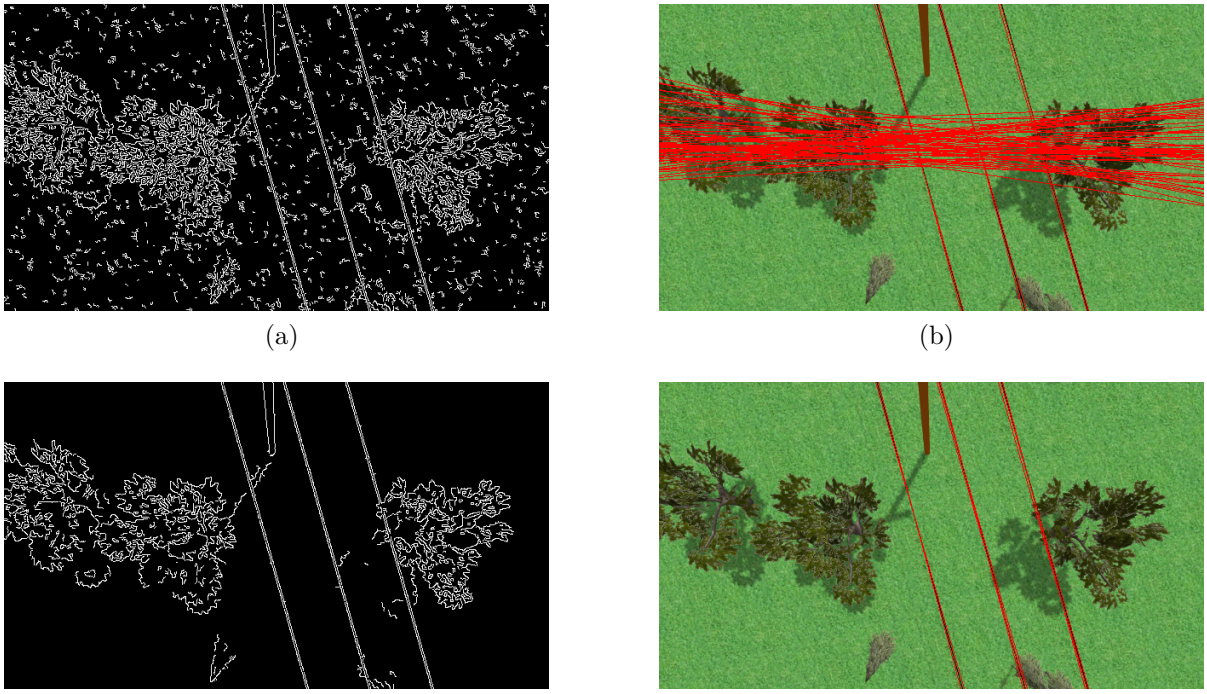


(a) Sobel × Blur (b) Sobel × Gaussiano (c) Sobel × Bilateral

(d) Laplaciano × Blur (e) Laplaciano × Gaussiano (f) Laplaciano × Bilateral

(g) Canny × Blur (h) Canny × Gaussiano (i) Canny × Bilateral

Figura 15 – Aplicação dos detectores de bordas Sobel, Laplaciano e Canny numa imagem filtrada cujo filtro filtro possui *kernel* de tamanho 5 (Fonte: autor)



(a) (b)

(c) (d)

Figura 16 – Demonstração da aplicação da Transformada de Hough. Em (a) temos uma imagem com bordas detectadas sem remoção do ruído e em (b) as retas encontradas. Em (c) temos a detecção de bordas com uma imagem filtrada usando o filtro bilateral e em (d) a detecção de retas (Fonte: autor)

3.4 Rastreamento das Linhas de Transmissão

Nesta seção é detalhado o algoritmo empregado para rastrear as linhas de transmissão dentre as retas enviadas pelo nó anterior. A Figura 17 contém passos seguidos no nó *line_tracking*. A abordagem utilizada permite que se em algum momento todas as linhas de transmissão não forem identificadas, as linhas faltantes poderão ser estimadas das outras que foram encontradas, levando em consideração a informação espacial e temporal das retas.

Ao receber uma nova mensagem, contendo uma lista com as retas vindas do nó anterior, estas retas estão parametrizadas conforme o algoritmo TH, implementada pela biblioteca OpenCV. Cada reta é definida por dois parâmetros, como visto na Seção 2.1.3, ρ que é a distância perpendicular da reta até o canto superior esquerdo da matriz que representa a imagem, e θ, o ângulo que a reta forma com o eixo x da imagem.

Para facilitar o desenvolvimento, realizou-se a transformação dos valores retornados pela TH. Seja ρ_i e θ_i os valores retornados por TH para a reta i, onde ρ_i é a distância da reta i para a origem da imagem. É feita uma mudança de coordenadas, fazendo com que este valor passe a ser referente ao centro da imagem. Desta forma, é calculado o valor d_i para cada reta, de acordo com a Equação 3.2. Os valores W e H representam a largura

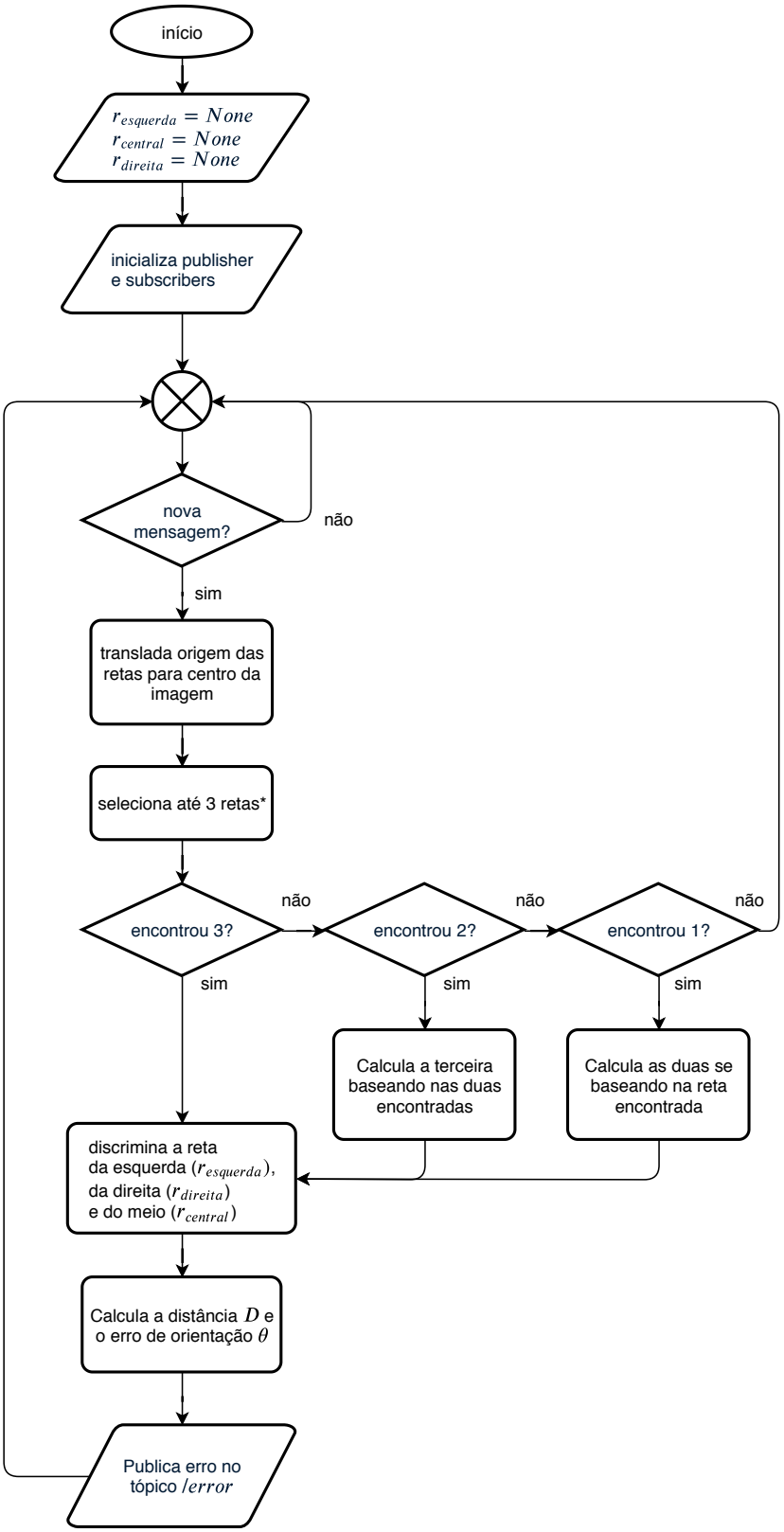


Figura 17 – Fluxograma do algoritmo de rastreamento de linhas de transmissão utilizada no nó *line_tracking* (Fonte: autor)

e altura da imagem respectivamente.

$$d_i = \rho_i - \frac{W \cos\theta_i + H \sin\theta_i}{2} \tag{3.2}$$

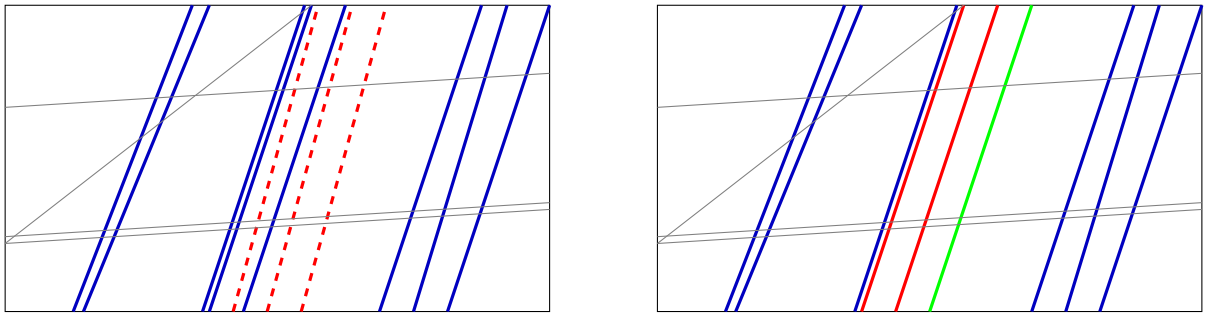
O valor de θ varia no intervalo de $[0, \pi]$. Como desejamos calcular o erro de θ em torno da direção do VANT, é necessário alterar este valor para o intervalo $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Isto é feito com a seguinte Equação 3.3.

$$se\ (\theta_i \geq \frac{\pi}{2}) \quad então: \begin{cases} \theta_i = \theta_i - \pi \\ d_i = -d_i \end{cases} \tag{3.3}$$

Após realizar a transformação de coordenadas das retas, é necessário escolher aquelas que são linhas de transmissão. Uma característica interessante da TH é que as retas retornadas por ele estão ordenadas de forma decrescente ao número de pontos pertencentes à reta. Na fase de inicialização, é escolhida a primeira reta como sendo pertencente ao conjunto de linhas de transmissão. Seja R_a a reta escolhida. Em seguida são escolhidas outras duas retas tais que $|R_{a.d} - R_{i.d}| > d_{min}$ e $|R_{a.\theta} - R_{i.\theta}| < \theta_{max}$, ou seja, outras duas retas que sejam aproximadamente paralelas a R_a mas que não sejam coincidentes, pois para uma mesma linha de transmissão, a TH pode retornar mais de uma reta. Após a determinação destas retas, elas são ordenadas com base no valor de d de cada uma. Seja R_e, R_m e R_d as retas da esquerda, meio e direita respectivamente. O valor de l é calculado como sendo a diferença entre as retas extremas, ou seja, $|R_{e.d} - R_{d.d}|$.

Em um cenário em que as linhas já tiverem sido determinadas inicialmente, eventualmente podemos cair na condição da Figura 18 *(a)*. Nesta condição, temos em tracejado vermelho as retas identificadas da imagem anterior. Com base no ângulo destas retas, descartamos as retas que estão em cinza, restando apenas as retas em azul. Levando em consideração o valor de d de cada uma destas retas, escolhe-se as retas mais próximas a elas. Novamente, toma-se o cuidado de não escolher retas muito próximas, pois podem ser referentes a uma mesma linha de transmissão. Neste caso, apenas duas retas foram identificadas, sendo necessário determinar a terceira. Após discriminar que as duas retas encontradas eram R_e e R_m, calcula-se R_d fazendo $R_{d.d} = R_{m.d} + \frac{|R_{m.d} - R_{e.d}|}{2}$ e $R_{d.\theta} = R_{m.\theta}$. A reta R_d está representada de verde na figura *(b)*.

Outros cenários, como faltar a reta do meio, da esquerda, ou faltar duas retas, são calculados de forma semelhante. Tendo encontrado todas as retas, é calculada então a distância entre as retas extremas, l. Com estes dados, é calculado o erro do VANT para a LT central utilizando a equação 3.1 e a publicação de tais valores no tópico */error*.



(a) (b)

Figura 18 – Identificação das linhas de transmissão entre as retas encontradas pela Transformada de Hough. Em (a) temos em azul as retas candidatas e em vermelho tracejado as LTs do frame anterior. Em (b) as LTs selecionadas em vermelho e em verde a LT estimada (Fonte: autor)

4 Experimentos e Análise

Para validar a solução proposta nesse trabalho foram analisados alguns experimentos com imagens gravadas de um ambiente simulado utilizando o simulador de robôs Gazebo. Inicialmente foi necessário criar o cenário de teste, com postes, linhas de transmissão e estruturas como árvores e gramados para idealizar um cenário e, por consequência, ter mais elementos para serem processados pelo algoritmo de visão computacional. Após, escolheu-se um VANT com gimbal e optou-se pelo piloto automático PX4, o qual já é amplamente utilizado e possui fácil integração com o Gazebo e ROS. As missões foram criadas na ECS QGroundControl e durante a execução utilizou-se um nó do ROS para gravar as imagens capturadas pela câmera do VANT. Desta forma, foi possível realizar testes e aprimoramentos dos algoritmos implementados, através de vídeos que foram gravados da simulação, excluindo a necessidade de ter a missão em execução simultaneamente com o algoritmo de processamento de imagens.

4.1 Preparação do ambiente para simulação

Para simular o VANT numa missão foi utilizado o simulador Gazebo (GAZEBO, 2019) na versão 7. Ele é responsável por simular a física tanto do objeto quanto do ambiente e da iteração entre ambos (iterações mecânicas, ações, forças, atrito etc.) e através de plugins, permite também a troca de mensagens com outras aplicações, como por exemplo ROS e PX4. A escolha de um simulador aliado com um cenário bem feito permite um teste rápido e eficaz de algoritmos, melhorias nos robôs, prototipação rápida com custo reduzido e testes de diversos cenários.

Como piloto automático utilizou-se o PX4 (PX4, 2019). Este é responsável por realizar o controle de baixo nível (estabilização de voo) e alto nível (execução de missões, desvio de obstáculos etc.) do VANT. Numa aplicação real, ele estaria sendo executado numa placa controladora de voo compatível, como por exemplo a *Pixhawk* (PIXHAWK, 2019). Sua escolha se deu devido ao grande suporte da comunidade de desenvolvedores em prover guias e integrações com o próprio simulador Gazebo e o ROS.

Para a simulação, utilizou-se uma versão do PX4 capaz de ser executada no computador, conhecida como *Software In The Loop (SITL)*. O PX4 SITL e o simulador Gazebo podem estar sendo executados na mesma máquina ou não. A comunicação entre eles é feita através do protocolo Mavlink. Durante a simulação, o firmware do PX4 recebe do Gazebo dados dos sensores simulados: sensores inerciais (acelerômetro e giroscópio), bússola e GPS. A origem destes dados é feita por plugins adicionados ao VANT. Ao receber estes dados, o firmware irá processar comandos necessários para controlar o mesmo, seja

para estabilização ou correção da posição para cumprir uma missão. Os comandos dos motores são então enviados para o Gazebo, que cuidará da parte física.

A aplicação que foi desenvolvida com ROS no Capítulo 3 se integra com o Gazebo através do pacote *gazebo_ros*. Este pacote contém um conjunto de plugins que permite a troca de mensagens entre ROS e Gazebo. Desta forma, a câmera que está no VANT sendo simulada por um plugin realiza a captura de imagens e estas ficam disponibilizadas para o ROS através de um determinado tópico. Com esta implementação seria possível também que a aplicação se comunicasse com o PX4, recebendo dados e enviando comandos para correção da trajetória.

Na Figura 19 é vista a integração entre os processos do PX4 em SITL e sua comunicação com o simulador através de um socket TCP na porta 4560. É possível realizar a integração também com ECS, como é o caso do QGroundControl, também utilizado neste trabalho.

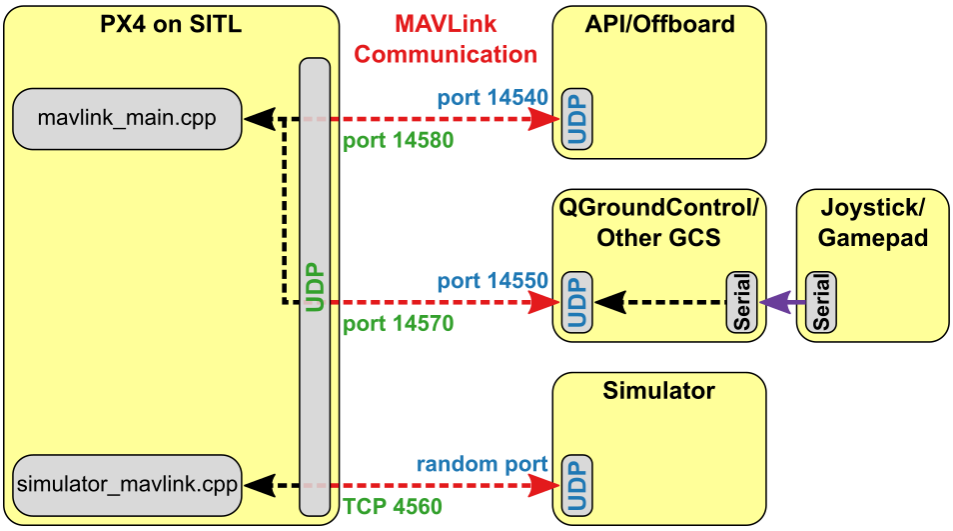


Figura 19 – Integração dos softwares utilizados para a realização dos experimentos. Temos o PX4 controlando o VANT que está sendo simulado no Gazebo e a aplicação desenvolvida recebendo dados do PX4 e do Gazebo (Adaptado: (PX4, 2019))

O último passo para a preparação do ambiente de simulação foi a criação de um cenário para o Gazebo. Foi necessário criar o modelo de um poste e dos cabos para representar as linhas de transmissão. Optou-se por desenvolver um modelo simplificado, com 10m de altura e 3 cabos de energia. Para compor o cenário e trazer mais detalhes de um mundo real, foram adicionados gramado e árvores. Desta forma, estes elementos estarão sob a visão da câmera e não deverão interferir no desempenho do algoritmo. O cenário pode ser visto na Figura 20.

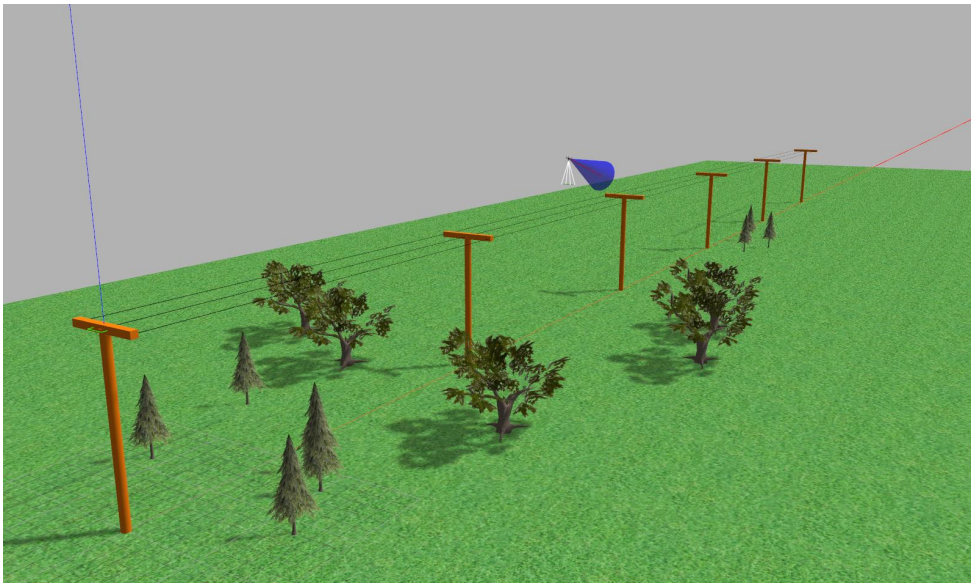


Figura 20 – Cenário criado no Gazebo para realização dos experimentos. Ele é composto de postes, cabos de energia, árvores e um gramado (Fonte: autor)

4.2 Criação das Missões

Para validar o algoritmo de identificação e rastreamento das linhas de transmissão, foram criadas e executadas 2 missões. Estas missões se diferem basicamente pela altitude do VANT em relação ao solo, visto que este parâmetro influencia na quantidade de pixels que a linha de transmissão terá na imagem. Vemos na Tabela 1 as principais características de cada missão.

Tabela 1 – Principais parâmetros das missões criadas para validação dos algoritmos

Missão	Altitude(m)	Comprimento(m)	Duração(s)	Imagens
missão 1	16	106	43.7	664
missão 2	20	106	44	664

As missões foram criadas e executadas utilizando a ECS *QGroundControl* (QGC, 2019). Desta forma é possível acompanhar os dados de voo à distância e também enviar comandos como decolar, pousar, executar uma missão etc. Na Figura 21 é possível ver uma missão em execução. As 2 missões são compostas de 6 *waypoints*. Os *waypoints* 1 e 2 são responsáveis por decolar, subir o VANT até a altura desejada e fazer com que o gimbal aponte a câmera para baixo. Os *waypoints* 3 e 4 são usados para posicionar o VANT à esquerda do primeiro poste, a uma distância próximo de 5 metros. Em seguida o VANT se move em linha reta para o *waypoint* 5, porém com uma inclinação em relação as linhas de transmissão. O objetivo deste trajeto é poder validar se é possível identificar, através do processamento de imagens, o erro da direção e distância perpendicular do VANT para as linhas de transmissão. O trajeto do *waypoint* 5 para o 6 realiza uma troca de direção e novamente um deslocamento perpendicular do VANT em relação as linhas de

transmissão. A velocidade do VANT ao longo do trajeto foi de $10,8km/h$. Esta velocidade foi escolhida porque com velocidades maiores, o voo do VANT ficou muito instável. Isto ocorria devido à um erro na modelagem do VANT e que está sem solução até o momento do desenvolvimento deste projeto.

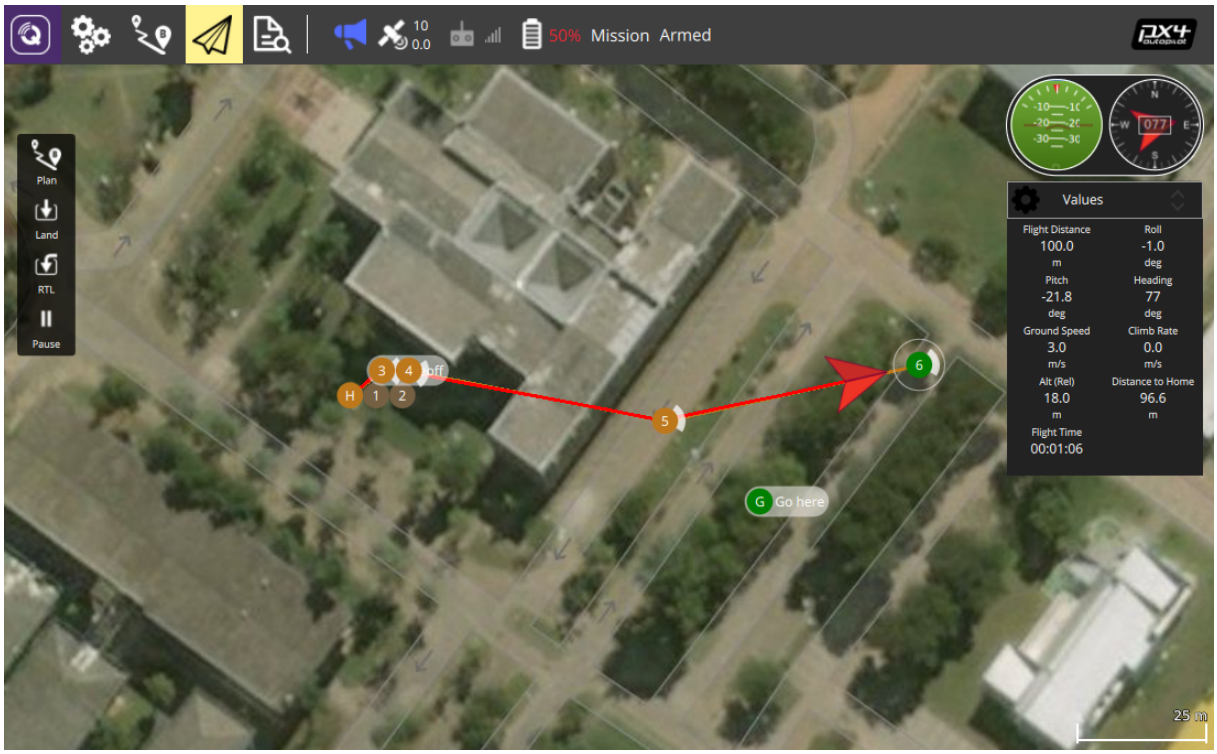


Figura 21 – Missão sendo acompanhada pelo QGroundControl (Fonte: autor)

A missão 1 e missão 2 têm como objetivo validar se alterando a altura, o algoritmo perde eficiência com relação à identificação e rastreamento das linhas de transmissão.

Após definidas as missões, estas foram enviadas ao PX4 utilizando o QGC e simuladas no Gazebo no cenário detalhado acima. Durante a execução, os dados gerados da simulação (imagens capturadas e telemetria do VANT) foram gravados em um arquivo de log utilizando uma ferramenta do ROS conhecida como *rosbag*. Para a realização dos experimentos, foram executados os nós descritos no Capítulo 3 recebendo as imagens que foram gravadas na etapa anterior pelo *rosbag*.

4.3 Análise das Missões

Nesta seção será feita uma análise da detecção de retas variando-se os algoritmos para filtragem de ruído e de detecção de bordas. Na Figura 22 temos todas as combinações de algoritmos para detecção de bordas e filtragem de imagens usados por cada missão.

Para cada missão, analisou-se 4 principais fatores: maior erro da medida da distância do VANT para a LT, maior erro da medida direção do VANT em relação à LT, total de

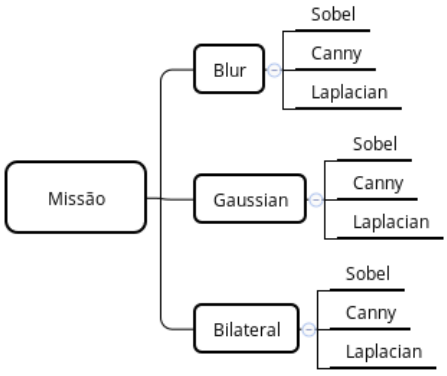


Figura 22 – Combinação de métodos de filtragem de ruído e extração de bordas utilizadas nos experimentos (Fonte: autor)

retas encontradas e total de LTs encontradas. Os erros das medidas foram determinados utilizando-se a posição e direção do VANT estimados pelo PX4 com relação aos valores estimados pelo algoritmo desenvolvido no Capítulo 3. O parâmetro *Total Lines* representa o número de retas encontradas pelo algoritmo Transformada de Hough, etapa final do nó *ImageProc*. O ideal é que este parâmetro fique próximo de 3, uma reta para cada LT, ou até 6, que é o caso de encontrar duas retas para cada borda da LT. O parâmetro *Cables Detected* representa, dentre todas as retas, quantas delas foram identificadas como LT. São escolhidas no máximo 3, conforme Capítulo 3. Em casos onde o número de linhas for menor que 3, as posições das linhas faltantes são estimadas com base nas linhas encontradas e nas linhas de imagens anteriores. Quando isto é feito, percebe-se um aumento no erro máximo da distância do VANT para a LT.

4.3.1 Influência dos filtros de ruído na missão 1

Utilizando o detector de bordas Sobel, os filtros Blur e Gaussiano tiveram resultados parecidos. Ambos encontraram de 3 a 12 retas, e o erro máximo da medida da posição ficou em $25cm$. Com relação às linhas de transmissão, todos os filtros encontraram os 3 fios ao longo do trajeto. O que teve melhor desempenho foi o filtro Bilateral, com erro máximo de $21cm$. Vale notar também que o erro máximo na direção ficou em 6 graus, para todas as combinações.

O detector de bordas Canny também obteve o melhor desempenho com o filtro Bilateral, com erro máximo de $23cm$ na medida da posição. Já o Blur teve o pior desempenho, com erro máximo de $39cm$. Isto se deve ao fato do filtro borrar demais a imagem, visto que o próprio Canny já realiza uma filtragem. Desta forma, com o Blur, em alguns momentos apenas uma reta foi detectada, fazendo com que a posição das linhas de transmissão fossem estimadas, ocasionando aumento no erro.

Já com o detector de bordas Laplaciano, novamente o pior resultado foi o filtro

Blur e o melhor com o filtro Bilateral. Nesta simulação, com o filtro Blur não foi possível identificar retas, visto que não houve pontos colineares suficientes para que a Transformada de Hough as identificassem. Com o filtro Bilateral, o erro máximo foi de 21cm identificando os 3 cabos ao longo de todo o trajeto.

As Figuras 23 e 24 contêm os gráficos da melhor combinação, Sobel com filtro Bilateral, e pior combinação, Laplaciano com Gaussiano (não foi possível mostrar os gráficos do Blur pois nenhuma reta foi identificada).

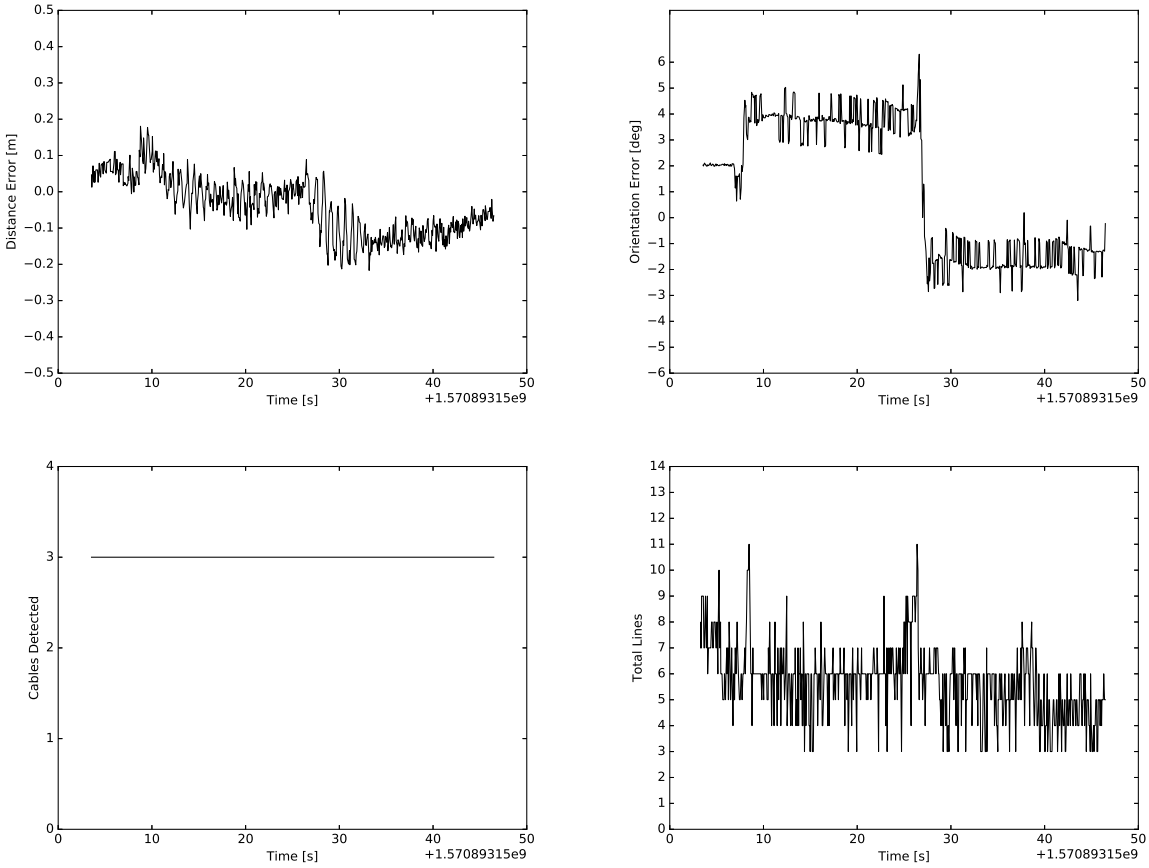


Figura 23 – Resultados utilizando filtro Bilateral e filtro de borda Sobel para a primeira Missão (Fonte: autor)

4.3.2 Influência da altura na missão 2

O teste feito na missão 2 possui exatamente o mesmo trajeto da missão 1, porém com a altura de 20 metros. Logo espera-se obter uma distância parecida do VANT em relação à linha nesta missão. Outro ponto que se deseja verificar é se o aumento da altura influencia na identificação das linhas de transmissão.

Após o processo de remoção de ruído, e detecção de bordas com o detector Sobel, foi possível identificar as 3 linhas de transmissão em todas imagens. Para o detector de bordas Sobel, todos os filtros de remoção de ruído tiveram desempenho semelhante. Erro

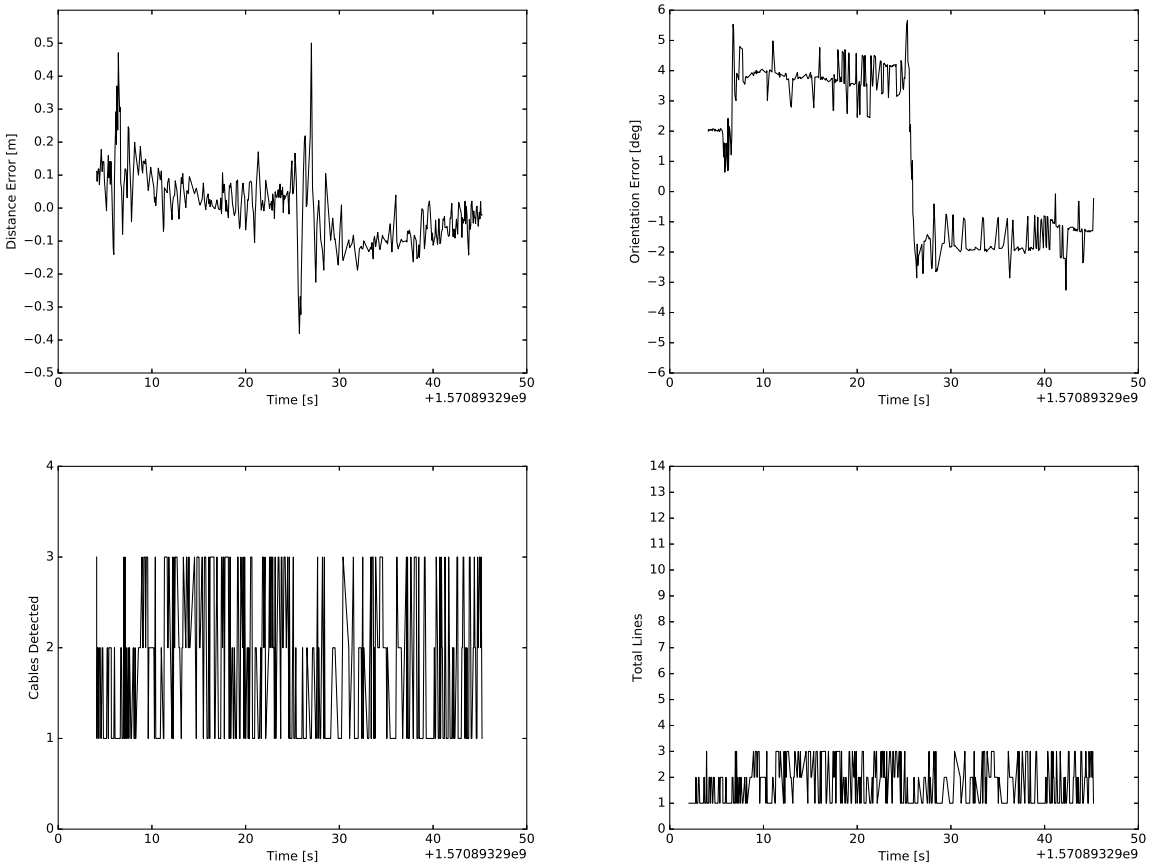


Figura 24 – Resultados utilizando filtro Gaussiano e filtro de borda Laplaciano para a primeira Missão (Fonte: autor)

máximo da medida da distância próximo de 40cm e total de retas encontradas ficou em torno de 6.

Já com o filtro Canny, a sua utilização conjunta com o filtro Blur não foi eficaz. Com esta combinação, só foi possível identificar as linhas de transmissão até o *waypoint* 5. Os filtros Gaussiano e Bilateral tiveram desempenho semelhantes, com erro na medida da distância em torno de 40cm e encontrando até 9 retas. Em alguns pontos, encontrou-se apenas 1 ou 2 linhas de transmissão, sendo necessário estimar a posição das faltantes.

Com o filtro Laplaciano, o único que foi possível rastrear as linhas de transmissão foi o filtro Bilateral, com erro de até 30 cm e encontrando até 3 retas. A quantidade de linhas de transmissão detectadas também variou, encontrando de 1 até 3 linhas. Com o uso dos filtros Blur e Gaussiano, não foi possível identificar nenhuma reta.

Pode-se concluir que o aumento da altura, algumas combinações de filtros e detectores de bordas ficaram mais sensíveis, como Laplaciano com Blur, Laplaciano com Gaussiano e Canny com Blur. Como a espessura da reta diminuiu, estes filtros borraram a imagem a ponto de não ser possível identificar retas. O filtro Bilateral tem a característica de preservar bordas, permitindo assim a identificação de retas. Outro ponto que

pode ser verificado é que mesmo variando a altura, o erro do cálculo da distância ficou parecido.

As Figuras 25 e 26 contêm os gráficos da melhor combinação, Sobel com filtro Bilateral, e a pior combinação que conseguiu pelo menos identificar as retas foi Canny com Blur.

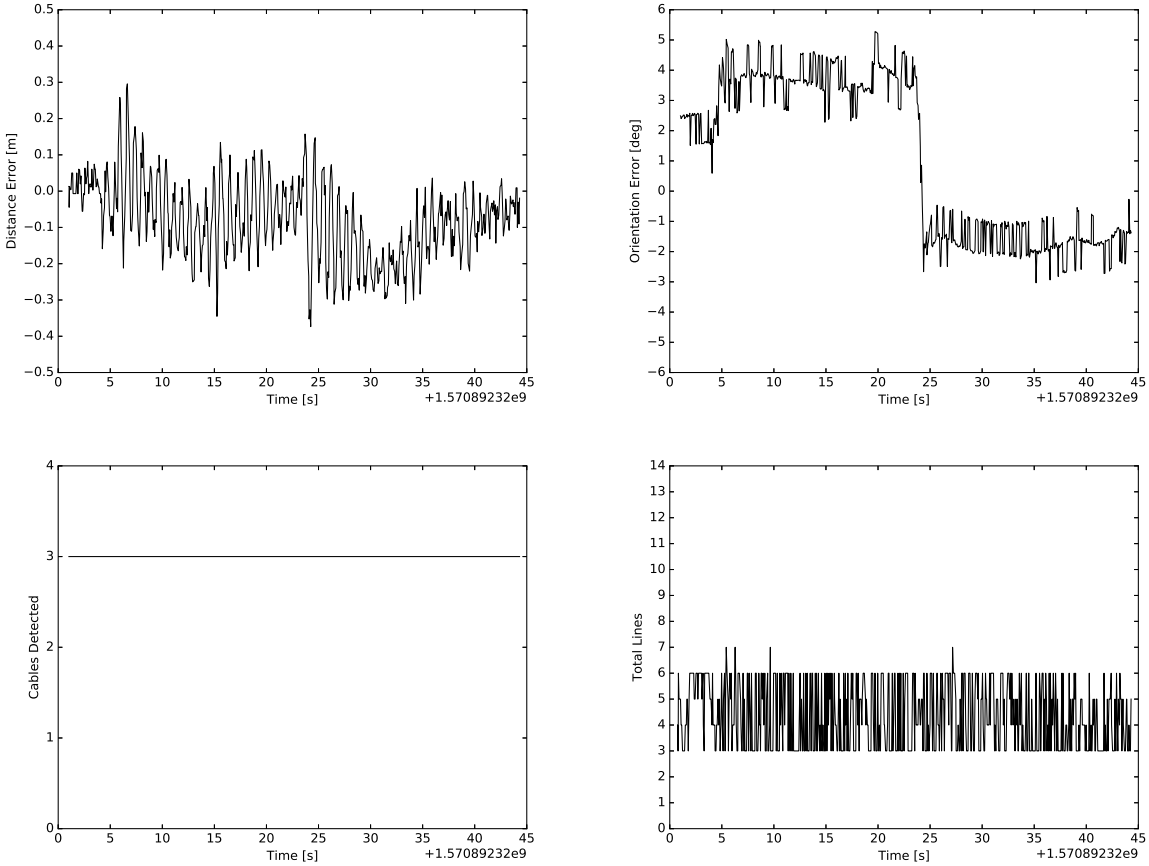


Figura 25 – Resultados utilizando filtro Bilateral e filtro de borda Sobel para a segunda Missão (Fonte: autor)

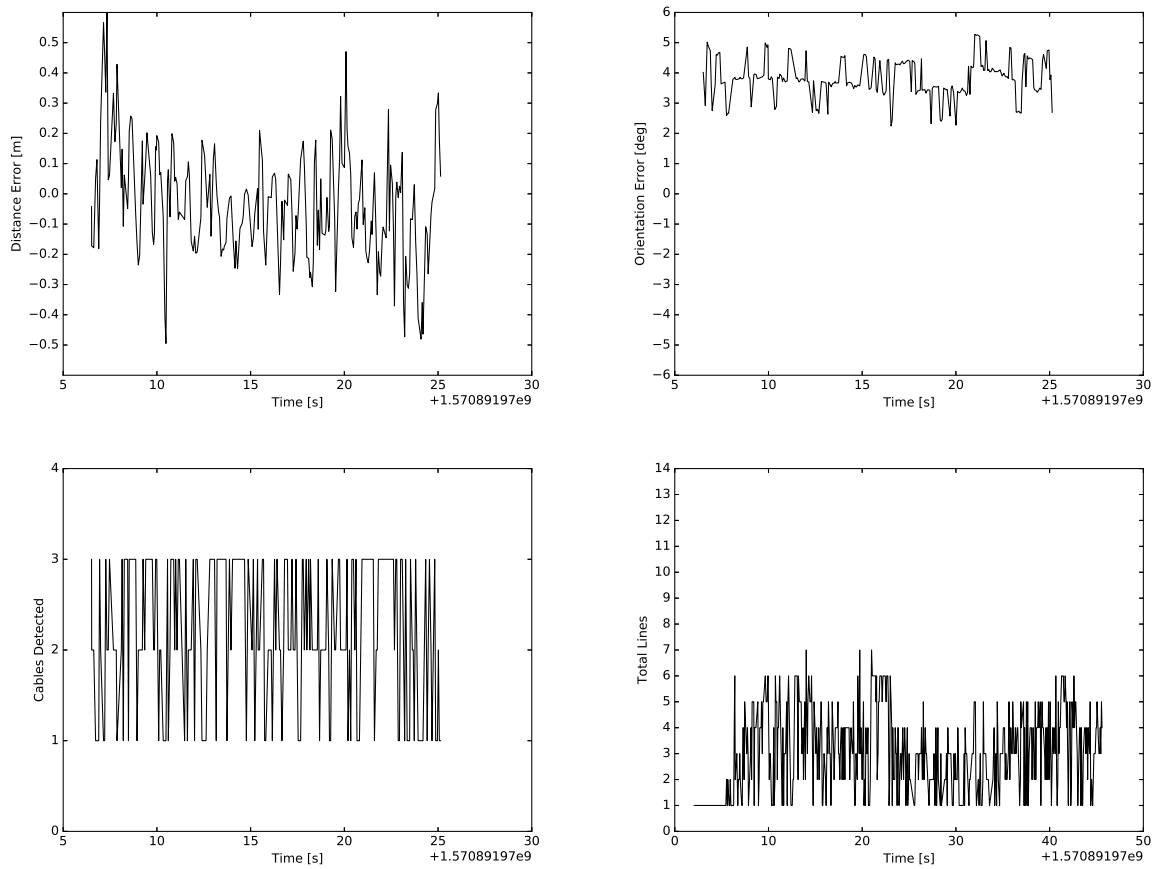


Figura 26 – Resultados utilizando filtro Blur e detector de borda Canny para a segunda Missão (Fonte: autor)

5 Conclusão

Neste trabalho foi desenvolvido uma técnica de processamento de imagens capaz de permitir voos autônomos de VANTs num cenário de inspeção de linhas de transmissão. Durante o trajeto de inspeção entre duas torres, uma câmera embarcada ao VANT e direcionada para baixo irá captar imagens que contêm as linhas de transmissão no seu campo de visão. Através do processamento de imagens, foi possível identificar as linhas de transmissão, realizar o seu rastreio ao longo do trajeto e calcular a posição relativa do VANT para elas. O algoritmo foi implementado em Python utilizando o *framework* ROS sendo o processamento de imagens feito pela biblioteca OpenCV.

A parte crítica da solução foram os algoritmos para o processamento de imagens. São necessários vários passos até extrair as linhas de transmissão, dentre eles o pré-processamento das imagens para remoção de ruído, segmentação da imagem identificando as bordas, identificação das retas e por último, dentre as retas, quais delas são os cabos de energia. Foram comparados alguns algoritmos para redução de ruído e extração das bordas.

Para redução do ruído foram utilizados os filtros Blur, Gaussiano e Bilateral. Para extração de bordas foram utilizados os operadores Sobel, Laplaciano e o detector de bordas Canny. Para identificação de retas foi escolhido a Transformada de Hough.

O algoritmo foi testado em ambiente simulado, utilizando o simulador de robôs Gazebo. Foram construídos cenários contendo torres e linhas de transmissão permitindo assim avaliar o desempenho do algoritmo de identificação e rastreamento das linhas de transmissão. Foi possível realizar a integração do Gazebo com o firmware PX4 SITL, este que originalmente é executado na placa controladora de voo *Pixhawk*.

Foi possível identificar nos testes que a escolha do filtro de remoção de ruído tem grande impacto na hora de detectar as retas, principalmente quando o VANT está mais alto em relação às LTs e na imagem, a LT possui apenas 1 pixel de largura. O filtro Bilateral se mostrou ser mais eficaz, visto sua capacidade de remover detalhes desnecessários da imagem e ainda preservar as suas bordas. A sua combinação com os detectores de bordas Sobel e Canny foram os que tiveram os melhores resultados. Um ponto negativo de sua escolha está no fato de possuir a maior complexidade temporal (mais lento em tempo de execução), podendo ser um empecilho em aplicações de tempo real num hardware embarcado no VANT.

Este trabalho permitiu, após a criação do cenário e integração entre ROS, Gazebo e PX4 a prototipação e validação rápida de novos algoritmos e novas implementações para controle de VANTs e processamento de imagens.

Como melhorias e trabalhos futuros, sugere-se:

- Validar em ambiente real e controlado.
- Estudar o filtro de Kalman para efetuar o rastreamento e estimação das posições das LTs.
- Estudar outras maneiras de calcular a distância D sem precisar saber a medida L.
- Implementar uma técnica de controle para correção da trajetória do VANT.

Referências

ANAC. *Orientações para usuários de drones*. 1ª edição. ed. [S.l.], 2017. 14

ANEEL. *Aneel*. 2019. Disponível em: <https://www.aneel.gov.br/aneel-essencial/-/asset_publisher/c4M6OIoMkLad/content/o-que-a-aneel-faz-?inheritRedirect=false>. Acesso em: 14 out. 2019. 13

ANEEL-ONS. *Operador Nacional do Sistema Elétrico - ONS*. 2019. Disponível em: <https://www.aneel.gov.br/home?p_p_id=101&p_p_lifecycle=0&p_p_state=maximized&p_p_mode=view&_101_struts_action=/asset_publisher/view_content&_101_returnToFullPageURL=/&_101_assetEntryId=15817751&_101_type=content&_101_groupId=656835&_101_urlTitle=operador-nacional-do-sistema-eletrico-ons&inheritRedirect=true>. Acesso em: 14 out. 2019. 13

ARAAR, O.; AOUF, N. Visual servoing of a quadrotor uav for autonomous power lines inspection. *22nd Mediterranean Conference on Control and Automation*, p. 1418–1424, 2014. Disponível em: <http://ieeexplore.ieee.org/document/6961575/>. 26

CANALENERGIA. *Canal Energia*. 2019. Disponível em: <https://www.canalenergia.com.br/noticias/53059239/enel-rio-ira-utilizar-drones-em-inspecoes-na-rede-eletrica>. Acesso em: 11 out. 2019. 15

CANNY, J. A computational approach to edge detection. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, v. 8, n. 6, p. 679–698, 1986. 23

CEMIG. *Cemig*. 2019. Disponível em: <http://www.cemig.com.br/sites/Imprensa/pt-br/Paginas/cemig-nave-nao-tripulada-testes.aspx>. Acesso em: 11 out. 2019. 15

CEMIG2. *Cemig Inovação*. 2019. Disponível em: <http://www.cemig.com.br/pt-br/A_Cemig_e_o_Futuro/inovacao/Documents/Inovacao/PeD%202016%202017%20C07%2001%20-%20Drones_Vants.pdf>. Acesso em: 11 out. 2019. 15

DUDA, R. O.; HART, P. E. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, ACM, New York, NY, USA, v. 15, n. 1, p. 11–15, jan. 1972. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/361237.361242>. 24

EBC. *Aneel multa ONS e operadora por apagão de março do ano passado*. 2019. Disponível em: <http://agenciabrasil.ebc.com.br/economia/noticia/2019-02/aneel-multa-ons-e-operadora-por-apagao-de-marco-do-ano-passado>. Acesso em: 27 out. 2019. 14

GAZEBO. *Gazebo homepage*. 2019. Disponível em: <http://gazebosim.org>. Acesso em: 06 out. 2019. 38

GOLIGHTLY, I.; JONES, D. Visual control of an unmanned aerial vehicle for power line inspection. In: IEEE. *ICAR'05. Proceedings., 12th International Conference on Advanced Robotics, 2005.* [S.l.], 2005. p. 288–295. 26

HOUGH, P. Method and means for recognizing complex patterns. 12 1962. 24

IBGE. *IBGE*. 2019. Disponível em: <https://www.ibge.gov.br/geociencias/organizacao-do-territorio/estrutura-territorial/15761-areas-dos-municipios.html?=&t=o-que-e>. Acesso em: 14 out. 2019. 13

IGNACIO, J. da S. *Processamento e Análise Digital de Imagens em Estudo da Cinética de Recristalização de Ligas de Al-Mg-X*. Dissertação (Mestrado) — Instituto de Pesquisas Energéticas e Nucleares, 2013. 18

IRISS. 2019. Disponível em: <https://www.iriss.com/white-papers/the-critical-angle-of-ultrasonic-electrical-inspection-of-transmission-lines-and-enclosed-system/attachment/wp10_banner_image/>. Acesso em: 20 out. 2019. 15

ITA. *ITA Notícias*. 2019. Disponível em: <http://www.ita.br/noticias/vantsinspecionarolinhasdachesf>. Acesso em: 11 out. 2019. 14

LI, Z. et al. Knowledge-based power line detection for UAV surveillance and inspection systems. In: *2008 23rd International Conference Image and Vision Computing New Zealand, IVCNZ*. [S.l.: s.n.], 2008. ISBN 9781424425822. ISSN 2151-2191. 26

LI, Z. et al. Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved Hough transform. *Machine Vision and Applications*, 2010. 26

MILLS, S.; AOUF, N.; MEJIAS, L. Image based visual servo control for fixed wing UAVs tracking linear infrastructure in wind. In: *Proceedings - IEEE International Conference on Robotics and Automation*. [S.l.: s.n.], 2013. ISBN 9781467356411. ISSN 10504729. 26

NEWAIR. 2019. Disponível em: <http://servicoaereo.com.br/inspecao-de-linhas-de-transmissao/>. Acesso em: 20 out. 2019. 15

ONS. *Capacidade instalada no SIN - 2018/2023*. 2019. Disponível em: <http://www.ons.org.br/paginas/sobre-o-sin/o-sistema-em-numeros>. Acesso em: 27 out. 2019. 13

PETARV. *PetarV*. 2019. Disponível em: <https://github.com/PetarV-/TikZ/tree/master/2D%20Convolution/>. Acesso em: 11 out. 2019. 19

PIXHAWK. *Pixhawk homepage*. 2019. Disponível em: <https://pixhawk.org>. Acesso em: 27 ago. 2019. 38

PX4. *PX4 homepage*. 2019. Disponível em: <https://px4.io>. Acesso em: 7 out. 2019. 38, 39

QGC. *QGC*. 2019. Disponível em: <http://qgroundcontrol.com/>. Acesso em: 11 out. 2019. 40

SILVA, R. L. da. *Introdução à manutenção de linhas de transmissão e subestações*. [S.l.], 2013. 13

TEXASDRONE. 2019. Disponível em: <http://texasdroneprofessionals.com/sample-drone-data/>. Acesso em: 20 out. 2019. 15

TOMASI, C.; MANDUCHI, R. Bilateral filtering for gray and color images. In: *ICCV*. [S.l.: s.n.], 1998. v. 98, n. 1, p. 2. 19, 20, 21

YANG, T. W. et al. Overhead power line detection from UAV video images. *International Conference on Mechatronics and Machine Vision in Practice*, p. 28–30, 2012. 26

ZHANG, J. et al. High speed automatic power line detection and tracking for a UAV-based inspection. *Proceedings of the 2012 International Conference on Industrial Control and Electronics Engineering, ICICEE 2012*, p. 266–269, 2012. 26