



CURSO SUPERIOR DE ENGENHARIA DE COMPUTAÇÃO

Fabiano Dutra Filereno

**Controle e Automação de Vant com Visão
Computacional, para Sensoriamento
Populacional**

**Porto Alegre-RS
2020**

Fabiano Dutra Filereno

Controle e Automação de Vant com Visão Computacional, para Sensoriamento Populacional

Trabalho apresentado para o Curso de Engenharia do Computação, de Centro Universitário Uniftec como parte dos requisitos para avaliação da unidade curricular de TCC.

Centro Universitário Uniftec

Orientador: Leonardo Amaral

Porto Alegre-RS

2020

Fabiano Dutra Filereno

Controle e Automação de Vant com Visão Computacional, para Sensoriamento Populacional

Trabalho apresentado para o Curso de Engenharia do Computação, de Centro Universitário Uniftec como parte dos requisitos para avaliação da unidade curricular de TCC.

Leonardo Amaral
Orientador

Professor
Convidado 1

Professor
Convidado 2

Porto Alegre-RS
2020

Listas de ilustrações

Figura 1 – Exemplo de Redes Neurais	8
Figura 2 – Diferença entre IA, ML e DL	9
Figura 3 – Rede Neural	10
Figura 4 – Rede Neural Convolucional	11
Figura 5 – Pixel em Visão Computacional	12
Figura 6 – Exemplo Simples de Treinamento de Rede Neural com Método de Trigêmeos	15
Figura 7 – Exemplo de Comparaçāo de Vetores 128-d	16
Figura 8 – Extração de um Vetor 128-d com Reconhecimento Facial	17
Figura 9 – Exemplo de Deslocamento Utilizando Dálculos	19
Figura 10 – Exemplo de Funcionamento do SITL	23
Figura 11 – Fluxo de Operação SITL, MAVProxy, ArduPilot e MAVLink	24
Figura 12 – VANT do Tipo Quadricóptero	25
Figura 13 – Frame de um VANT Quadricóptero	26
Figura 14 – Pixhawk 1	27
Figura 15 – Diagrama Eletrônico dos Conectores da Pixhawk 1	28
Figura 16 – Motor sem Escova	29
Figura 17 – Partes do Motor	30
Figura 18 – Controlador Eletrônico de Corrente utilizado em VANTs	30
Figura 19 – Diagrama Eletrônico de um Controlador Eletrônico de Corrente	31
Figura 20 – Módulo de GPS	31
Figura 21 – Hélice	32
Figura 22 – Bateria de Polímero de Lítio	33
Figura 23 – Esquema Básico de Montagem de um VANT do tipo Quadricóptero	34
Figura 24 – Teoria de Sustentação de uma Hélice	35
Figura 25 – Diagrama de Esforços cortantes na Estrutura do Quadricóptero	37
Figura 26 – Diagrama de Momento Fletor sobre a Estrutura do Quadricóptero	38
Figura 27 – Sentido de Movimento dos Motores e Inercias	39
Figura 28 – Movimentos de um Quadricóptero	40
Figura 29 – Movimento Horizontal " <i>Roll</i> "	42
Figura 30 – Movimentos Horizontais de um Quadricóptero	43
Figura 31 – Plano Cartesiano na Superfície Esférica	44
Figura 32 – NED	46
Figura 33 – Simulação real para abstrata	48
Figura 34 – Raspberry Pi 3 B	51
Figura 35 – Diagrama de Funcionamento do Sistema	53

Figura 36 – Regiões de Interesse	54
Figura 37 – Sistema de Coordenadas do OpenCV	55
Figura 38 – Conversão do Sistema de Coordenadas OpenCV para Coordenadas Cartesianas	56
Figura 39 – Sistema de Coordenadas Cartesianas em Visão Computacional	57
Figura 40 – Projeção do Sistema de Coordenadas OpenCV para Cartesianas	58
Figura 41 – Gráfico que Representa o Gradiente de Velocidade	60
Figura 42 – Gráfico que Representa o Gradiente de Velocidade para X e Y	61
Figura 43 – Conexão MAVLink da Raspberry Pi com Pixhawk	62

Lista de tabelas

Lista de abreviaturas e siglas

VANT	veiculo Aereo não Tripulado
IA	Inteligencia Artificial
MA	Machine Learning
DL	Deep Learning
BSD	Erkeley Software Distribution
MIT	Massachusetts Institute of Technology
CUDA	Compute Unified Device Architecture
GPU	Graphics Processing Unit
GPGPU	General Purpose Graphics Processing Unit
SIMD	Single Instruction, Multiple Data
API	Application Programming Interface
NN	Neural Network
ResNET	Residual Neural Network
XML	Extensible Markup Language
GCS	Sround Sontrol Station
UAV	Unmanned Aerial Vehicle
GUI	Graphical User Interface
POSIX	Portable Operating System Interface
SITL	Software in the Loop
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
NASA	National Aeronautics and Space Administration
ROS	Robot Operating System

MEMs	Micro-Electro-Mechanical Systems
GPS	Global Positioning System
FPU	Unidade de Ponto Flutuante
MEAS	Managed Ethernet Access Service
ESD	Electrostatic Discharge
UART	Universal Asynchronous Receiver Transmitter
RSSI	Received Signal Strength Indication
PPM	Pulse Position Modulation
I2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface
CAN	Controller Area Network
ADC	Analog-to-Digital Converters
ESC	Electronic Speed Controllers
LíPo	Lítio Polimero
SSD	Unidad de Estado Solida
SATA	Serial Advanced Technology Attachment
GPIO	General Purpose Input/Output

List of symbols

α	Letra Grega Alpha
τ	Letra Grega minúscula tau
Θ	Letra Grega maiúscula Teta
λ	Letra Grega minúscula lambda
ψ	Letra Grega minúscula psi
ϕ	Letra Grega minúscula phi

Sumário

Introdução	1	
1	OBJETIVOS E JUSTIFICATIVAS	3
1.1	Objetivos Gerais	3
1.1.1	Objetivos Específicos	4
1.1.2	Resultados Esperados	4
1.2	Justificativa	5
2	FUNDAMENTAÇÃO TEÓRICA	6
2.1	Inteligência Artificial	6
2.1.1	Um pouco sobre a Historia	6
2.1.2	Machine Learnig	6
2.1.3	Deep Learnig	7
2.1.4	Visão Computacional	10
2.1.5	Redes Neurais Profundas (Deep Learning)	11
2.2	Tecnologias Open Source para Tratamento de Imagens	12
2.2.1	OpenCV	12
2.2.2	Biblioteca Dlib	13
2.2.3	Biblioteca Face Recognition	13
2.2.4	Nvidia Cuda	13
2.2.5	Reconhecimento Facial	14
2.3	Tecnologias de Linguagem de Programação para Cálculos de Álgebra e Geometria	18
2.3.1	Biblioteca NumPy	18
2.3.2	Biblioteca SciPy	18
2.4	Tecnologias Open Source de Desenvolvimento, para Veículos Aéreos não Tripulados	19
2.4.1	Firmwares para VANT	19
2.4.1.1	Ardupilot	19
2.4.1.2	PX4	20
2.4.2	Ferramentas de Desenvolvimento para VANT	20
2.4.2.1	API de Desenvolvimento Dronekit para Ardupilot	20
2.4.2.2	API de Desenvolvimento MAVSDK para PX4	20
2.4.3	Dronekit versus MAVSDK	20
2.4.4	Protocolo de Comunicação MAVLink	20
2.4.5	Proxy de Protocolo MAVLink para VANT (MAVProxy)	21

2.4.6	Simulador de VANT SITL	22
2.4.7	Simulador de Robótica Gazebo	24
2.4.8	Ferramenta para Desenvolvimento de Robótica ROS	24
2.5	Tecnologias e a Anatomia de Veículos Aéreos não Tripulados	25
2.5.1	Frame	26
2.5.2	Controladora de Voo	26
2.5.3	Motor sem Escova	29
2.5.4	Controlador Eletrônico de Corrente (ESC)	30
2.5.5	Módulo de GPS	31
2.5.6	Hélice	31
2.5.7	Bateria de Polímero de Lítio (LiPo)	32
2.5.8	Esquema de Montagem de um Quadricóptero	34
2.6	Teoria de funcionamento e dinâmica de voo de um quadricóptero	35
3	METODOLOGIA	47
3.1	Métodos	48
3.2	Componentes Físicos	49
3.2.1	Computador Utilizado nas Simulações	49
3.2.2	Computador Complementar	50
3.2.3	Dispositivo de Captura	52
3.3	Sistemas Operacionais	52
3.4	Fluxogramas dos Protótipos e Funcionamento do Sistema	53
3.4.1	Transformação do Sistema de Coordenadas do OpenCV	53
3.5	Simulações com Software	61
3.6	Simulações com Hardware	61
4	ANALISE DOS RESULTADOS E CONSIDERAÇÕES FINAIS	63
4.1	Resultados	63
4.2	Considerações Finais	63
	REFERÊNCIAS	64
	APÊNDICES	67
	APÊNDICE A – PRIMEIRO APÊNDICE	68
	APÊNDICE B – PERCEBA QUE O TEXTO DO TÍTULO DESSE SEGUNDO APÊNDICE É BEM GRANDE	69

Introdução

A segurança pública hoje é um tema muito noticiado em veículos de mídia. Segundo estudos, hoje no Brasil aproximadamente 10% dos crimes cometidos são solucionados ([BRUM, 2018](#)) e isso se dá principalmente devido ao fato das polícias terem um “baixo efetivo não dando conta de cobrirem certas áreas. Outro fator muito importante é a falta de investimento em tecnologias o que não acontece em países de primeiro mundo que possuem uma alta taxa de soluções de casos criminais pelo fato destes países terem um alto investimento tecnológico. Um bom exemplo seria a China que possui hoje o melhor sistema de vigilância existente, composto de um complexo sistema de reconhecimento facial, o que faz com que eles possuam uma taxa de crimes mais baixas que a de países da Europa e equivalente ao de países como a Suíça ([ZMOGINSKI, 2019](#)). No Brasil, os crimes são solucionados, pelo fato da polícia geralmente adquirir provas através de alguma imagem feita por uma câmera de terceiro, ou seja, equipamentos privados que foram instalados em residências ou empresas [16]. Contudo, percebemos que no Brasil existe uma grande falta de investimentos em pesquisa focada em novas tecnologias que poderiam suprir o mercado e principalmente a área de segurança. Hoje existem tecnologias de visão computacional com distribuição livre que podem ser utilizadas por exemplo no mercado Brasileiro na área de segurança. Um exemplo destas tecnologias são as ferramentas de visão computacional open source, um exemplo é o OpenCV, ferramenta muito poderosa que é utilizada hoje em vários países com foco em reconhecimento facial, identificação de padrões de comportamento, detecção de objetos, entre outras várias funcionalidades, todas focadas em visão computacional e utilizando aprendizagem de máquina (Machine Learning) [17]. Com investimento em pesquisa e desenvolvimento, a tecnologia de visão computacional focada em detecção facial e padrões de comportamento poderia ser aplicada como ferramenta para suprir a demanda de tecnologias na área de segurança pública no Brasil. Poderia ser desenvolvido um sistema integrado entre os estados utilizando um banco de dados compartilhado para detecção de possíveis criminosos, fugitivos, suspeitos com posse de armamento em meio à multidão, assim coibindo possíveis crimes. Alguns países que utilizam detecção facial já estão desenvolvendo sistemas integrados desta tecnologia com a utilização de um veículo aéreo não tripulado (Vant) ([ZMOGINSKI, 2019](#)) [2]. Com um drone é possível cobrir uma grande área utilizando o espaço aéreo e em casos de desaparecimentos, este equipamento poderá acessar áreas de difícil acesso. Um bom exemplo é o caso do rompimento da barragem de Brumadinho aonde ocorreu de vários corpos não serem localizados pelo fato da lama dificultar as buscas, e nesse caso, uma empresa estrangeira veio com uma solução utilizando drones que possibilitaram a busca em certos locais, drones esses que possuíam câmeras com tecnologias que possibilitavam

identificar um ser vivo ou um corpo a certas profundidades na lama [18]. Tecnologias de drones para visão computacional ainda estão em desenvolvimento e possuem algumas limitações como a comunicação de uma controladora de vôo com um computador que possa sustentar a tarefa de processar todo o sistema de imagem e desempenhar comandos de vôo para o equipamento de vôo. No site da Ardupilot (ARDUPILOT.ORG, 2020)[3], já existe um projeto em desenvolvimento para conectar e configurar um mini computador de baixo custo, como o Nvidia TX2, Intel Edison e a Raspberry Pi. A ideia do estudo é integrar via protocolo MAVLink os dois hardwares à Raspberry Pi com um sistema Linux embarcado e integrado à ferramenta OpenCV, e assim possibilitar o desenvolvimento de um sistema com aprendizado de máquina (Machine Learning) que envie controles de direcionamento para o equipamento de voo através da comunicação entre os dos equipamentos.

1 Objetivos e Justificativas

A ideia surgiu devido ao interesse em integrar um veículo aéreo não tripulado, no caso um drone do tipo quadricóptero, com algum dispositivo para realização de alguma tarefa, e após pesquisar várias trabalhos, protótipos experimentais, bibliografia sobre o assunto e sites especializados em tecnologias aplicadas em Vants, como exemplo a visão computacional, se descobriu que é possível conectar através de um protocolo de comunicação, uma controladora de drone ([DRONECODE, 2020](#)) [7] a um computador complementar ou como é mais abordado mini computador ([RASPBERRYPI.ORG, 2020a](#))[8], e enviar comandos para o Vant, criando um sistema de controle e automação de Vant através de visão computacional ([ARDUPILOT.ORG, 2020](#))[3]. Em seguida também se descobriu que é possível integrar a ferramenta OpenCV na Raspberry Pi fazendo a integração de visão computacional com a tecnologia de um Vant. Como o foco principal do trabalho é o controle utilizando visão computacional, primeiro se realizou uma pesquisa para descobrir se existe um computador complementar que pudesse comportar um software de visão computacional e que também tivesse capacidade computacional de compilar e rodar o algoritmo, e com processamento o suficiente para processar a tecnologia de visão computacional. O Raspberry Pi foi a melhor opção devido a alguns fatores como; custo, tamanho, tempo. A segunda pesquisa abordou como comunicar o computador complementar com a controladora do Vant e se existia a possibilidade de enviar comandos para a controladora de voo. Foi preciso buscar na internet muito material principalmente tutorias para aprender e dominar o funcionamento das ferramentas que seriam utilizadas no desenvolvimento, exemplo é o OpenCV, essa ferramenta é muito vasta possui muitas funcionalidades e módulos, então foi preciso descobrir qual seria a melhor maneira de utiliza-la no projeto, quais módulos utilizar, logo foi gasto muito tempo praticando e testando esses módulos para adequar quais seriam utilizados. Uma boa parte do trabalho foi realizando uma pesquisa bibliográfica sobre Deep Learning para o entendimento de como funcionam as tecnologias de visão computacional.

1.1 Objetivos Gerais

O objetivo geral deste trabalho é desenvolver o protótipo de um VANT controlado por um sistema embarcado com visão computacional para reconhecimento de padrões de comportamento de multidões, com reconhecimento facial, que possa ser empregado como uma ferramenta de segurança, podendo auxiliar em casos policiais, vigilância patrimonial, filmagens áreas, buscas em mata fechada, fiscalização de fronteira e aplicações civis nas quais seria inviável um ser humano trabalhar, realizando tarefas arriscadas e servindo

como verdadeira ferramenta de trabalho. Acredita-se que com o uso de imagens aéreas possa ser possível reconhecer e emitir alertas de possíveis suspeitos de crimes, e com isso buscar minimizar um pouco a falta de tecnologias que hoje são necessárias para ajudar no combate a violência. Um sistema de visão computacional integrado a um drone que através do desenvolvimento de um algoritmo e o treinamento de redes neurais, seja capaz de cumprir objetivos específicos, tendo como principal, o de identificar indivíduos através de reconhecimento facial, padrões de comportamento, e objetos específicos portados, e logo tomar a ação de rastrear o elemento que foi identificado através da movimentação aérea de um drone.

1.1.1 Objetivos Específicos

- Pesquisar e estudar o funcionamento da ferramenta OpenCV;
- Criar e treinar uma rede neural;
- Treinar a ferramenta inserindo fotos para reconhecimento facial;
- Treinar a ferramenta para reconhecimento de padrões de comportamento;
- Comunicar com protocolo MAVLink a Raspberry Pi e a Pixhawk;
- Realizar testes de comunicação entre a Pixhawk e a Raspberry Pi;
- Desenvolver um sistema de tolerância a falhas de comunicação;
- Pesquisar e estudar como a Raspberry Pi envia comandos de vôo;
- Desenvolver um algoritmo que interprete os dados extraídas da visão computacional e os converta em comandos de voo;
- Desenvolver um algoritmo que envie comandos de voo para o drone;

1.1.2 Resultados Esperados

- Perfeito funcionamento da ferramenta OpenCV;
- Que o algoritmo de rede neural seja compilado e executado pela Raspberry Pi;
- Reconhecimento das pessoas inseridas no algoritmo de reconhecimento facial;
- Distinção entre pessoas inseridas no algoritmo de rede neural, e as que não foram inseridas;
- Perfeita comunicação entre a Raspberry Pi e a Pixhawk;

- Perfeita tolerância de falhas do sistema;
- Desempenho satisfatório da Raspberry Pi em enviar comandos de vôo para o drone;
- Comportamento de voo do drone de maneira esperada que ele desempenhe;
- Perfeito funcionamento do algoritmo que será desenvolvido para o funcionamento do sistema que controla e corrige em que direção o drone deve seguir;
- Perfeito funcionamento dos sistemas de (software e o hardware) em seguir o objeto ou pessoa que foi inserido(a) no algoritmo de reconhecimento facial.

1.2 Justificativa

Devido ao aumento da criminalidade e dos altos índices de violência no cotidiano das pessoas, surgiu a cultura do medo e o sentimento de insegurança. Tais fatores demandaram algumas mudanças nos serviços de segurança patrimonial bem como, nas formas de monitoramento. Nesse sentido, tornou-se necessário expandir as formas de controle, seja por meio de câmeras de vigilância ou monitoramento eletrônico ([SOUZA et al., 2016](#))^[4]. No entanto, os equipamentos atuais como as câmeras utilizadas na segurança já não são mais eficientes e possuem tecnologias ultrapassadas. Na cidade do Rio de Janeiro existe um sistema de câmeras de alta tecnologia com capacidade para realizar reconhecimento facial. Este sistema foi implantado para testes através de uma parceria entre a OI e a Huawei ([SILVA, 2019](#))^[5], sendo que as câmeras e a tecnologia de reconhecimento facial embarcados são de propriedade da Chinesa Huawei e possuem alto custo monetário, e o que eu quero dizer com isso é que alguns estados Brasileiros passam por uma crise financeira, porem precisam investir em segurança ([SANTOS, 2018a](#))^[6]. Este contexto de insegurança pública e falta de investimento nacional em tecnologias avançadas de segurança é que originou a motivação para a proposta deste trabalho, ou seja, a necessidade da integração entre software e hardware de baixo custo, sendo aplicados para melhorar o desempenho do sistema de segurança no Brasil, visando a vigilância, através do sensoriamento utilizando câmeras e hardware de custo acessível e softwares livres.

2 Fundamentação Teórica

2.1 Inteligência Artificial

2.1.1 Um pouco sobre a Historia

O termo inteligência artificial foi expressa pela primeira vez por Alan Turing em 1950, ano no qual ele lançou um artigo falando sobre o seu “jogo da imitação”, hoje conhecido como “Teste de Turing”, já na época ele reconhece vários desafios a serem vencidos, inclusive com uma frase emblemática “as maquinas podem pensar” [19].

Acredito que, em cerca de 50 anos, será possível programar computadores, com uma capacidade de memória de cerca de 10^9 (TURING, 2009) tradução nossa.

O que é inteligência artificial? Este assunto tem intrigado várias áreas de estudo como a biologia, psicologia, filosofia, talvez uma maneira simples de definir é dizer que, a inteligência artificial ou “IA” é um campo de pesquisa da tecnologia que relaciona conceitos da fisiologia humana, alguns mais simples como os próprios sentidos, um exemplo é a visão e outros conceitos mais focados na capacidade do ser humano em conseguir raciocinar para tomar certas atitudes na solução de problemas complexos, por exemplo no caso da visão como conseguimos distinguir um objeto de outros, um indiviso de outro, etc. A inteligência artificial nada mais é do que a tentativa de imitar o comportamento humano em máquinas programas [20]. O conceito de aprendizado de máquina é muito importante em IA pois ele agrupa os conceitos de treinamento: aprender a interpretar entradas que vem em conjuntos finitos ou infinitos, classificá-los e assim gerar uma resposta ou como é mais conhecido uma saída de dados. Aprendizado por hábito: capacidade de um computador realizar alguma função lógica (A) e armazená-la como dado para posteriormente utilizá-la como referência para poder reagir a uma função semelhante a função (A), isso é conhecido em inteligência artificial como uma rede neural. [20].

2.1.2 Machine Learnig

Machine Learning ou aprendizado de máquina é uma subárea da inteligência artificial, ela surgiu por volta dos anos 1980 junto com a impulso de novos computadores quando eles começaram a evoluir em termos de hardware e processamento [21]. Em aprendizado de máquina, computadores são programados para aprender e evoluir com processos anteriores, para isso eles utilizam inferência que se denomina indução, eles obtêm conclusões genéricas se baseando em exemplos armazenados nos processos anteriores. Assim o algoritmo aprende a induzir uma função que será um problema a ser resolvido,

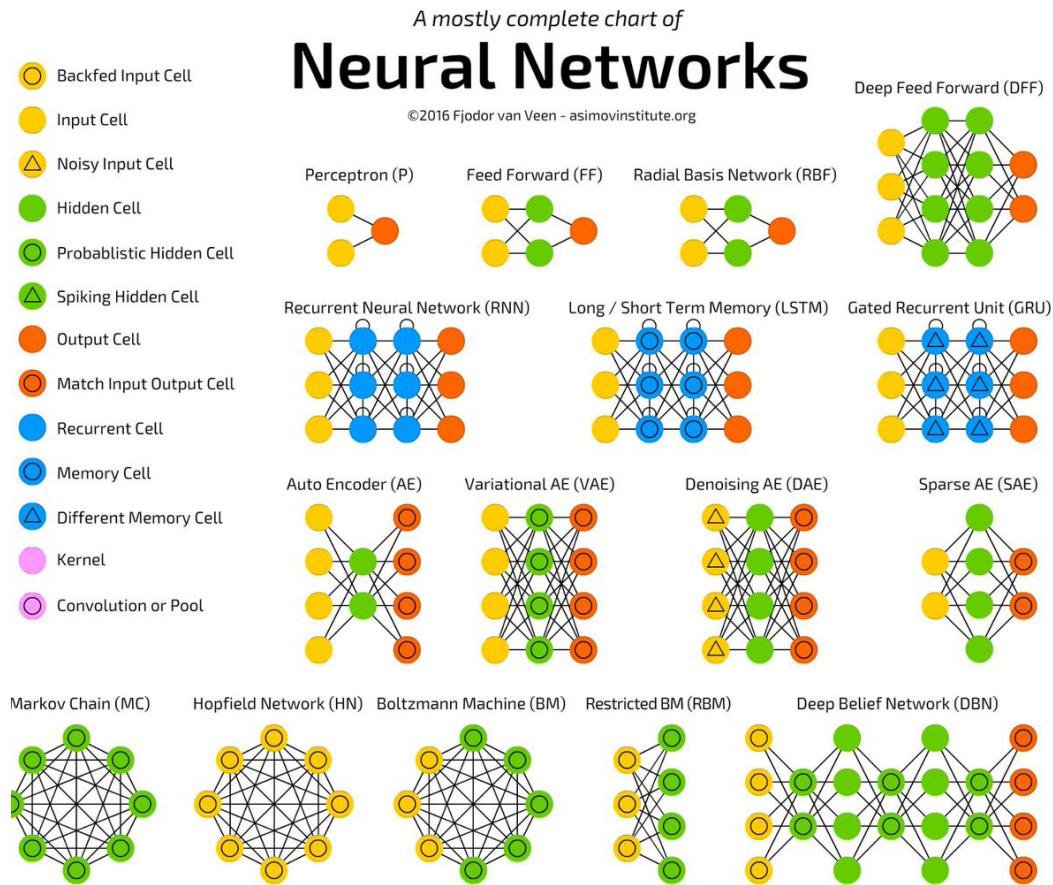
a partir de experiências anteriores que são chamados de dados [21]. Alguns exemplos bem-sucedidos:

- Reconhecimento de palavras;
- Predição de taxa de cura de pacientes;
- Detecção de fraudes em cartões de crédito;
- Veículos autônomos;
- Jogos de xadrez autônomo;
- Diagnóstico de câncer por análise de dados por expressão genética;

2.1.3 Deep Learning

Deep Learning ou aprendizado aprofundado é uma subárea do aprendizado de máquina é um conceito que utiliza redes neurais artificiais para que a máquina consiga não só resolver problemas, mas também aprender novas maneiras de resolvê-los [22]. É um conceito que estuda uma maneira de imitar o funcionamento dos neurônios humanos e contextualizá-los em modelos matemáticos, utilizando modelagem cognitiva a partir da mente humana. Esse estudo deu origem ao conceito de redes neurais artificiais e quando falamos de Deep Learning estamos nos referindo diretamente a redes neurais [22]. A figura 1 mostra os vários modelos de redes neurais desde o primeiro modelo que seria um perceptron, que é basicamente um ramo de uma rede foi o primeiro modelo, até conceitos mais profundos de redes neurais [22].

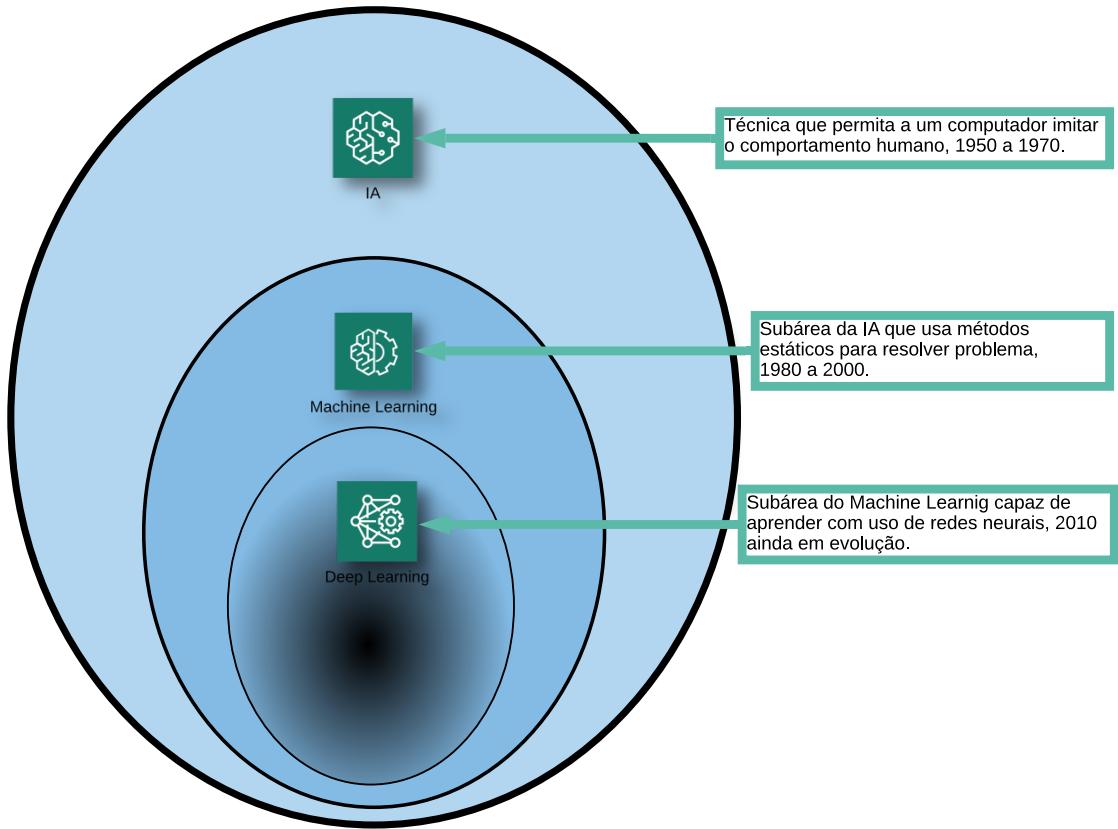
Figura 1 – Exemplo de Redes Neurais



Fonte: GitBook, ([SANTOS, 2018b](#)).

A figura 2 mostra um exemplo da diferença entre Inteligência Artificial, Machine Learning e Deep Learning, e quando aproximadamente elas começaram a ser abordadas e implementadas.

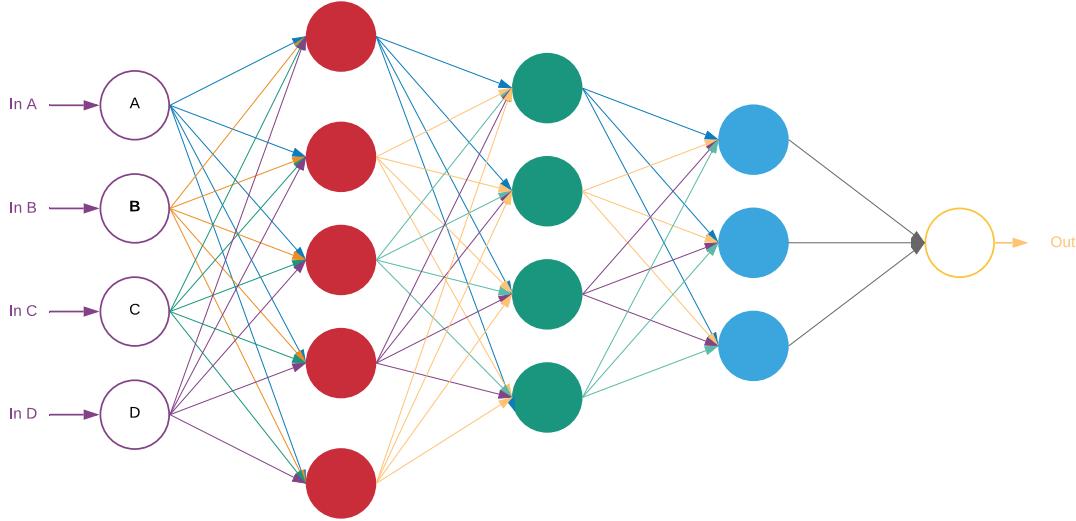
Figura 2 – Diferença entre IA, ML e DL



Fonte: o autor.

Uma rede neural nada mais é do que uma maneira de imitar o que ocorre no cérebro humana quando realizamos tarefas e assim ativamos neurônios que por sua vez ativam outros, e assim realizando uma reação em cadeia que gera o que chamamos de pensamento. A figura 3 é um simples exemplo de uma rede neural em máquinas aonde existem entradas, que ativam neurônios, que por sua vez se ligam com todos os outros neurônios e assim sucessivamente até chegarem em apenas uma saída [20].

Figura 3 – Rede Neural



Fonte: o autor com base em Oracle. ([TIERNEY ORACLE GROUNDBREAKER AMBASADOR, 2018](#)).

2.1.4 Visão Computacional

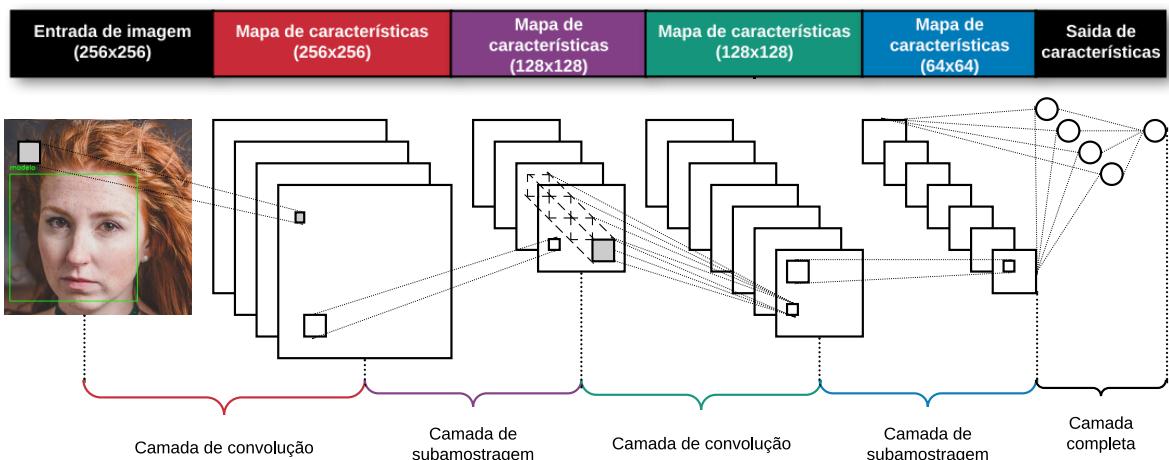
Visão computacional é um campo da inteligência artificial que treina computadores para interpretar e entender o mundo visual. Através do uso de imagens digitais, câmeras e vídeos junto a modelos de Deep Learning, as máquinas podem identificar e classificar objetos corretamente, e então, reagir ao que elas veem. [10]. Os estudos com visão computacional beiram a época do início da internet, sim isso é verdade já naquela época se estudava este campo, porém tudo era mais difícil, devido à falta de tecnologias para suportar o processamento de grande volume de dados, faltava o volume de dados hoje conhecido como (Big Data) e algoritmos de processamento paralelo [9]. Nessa década vários fatores impulsionaram o uso e estudo de visão computacional, tecnologias moveis saturaram o mundo com vídeos e fotos o (Big data), o poder computacional cresceu abruptamente e tornou-se mais barato, e novos algoritmos com redes neurais convulsionais aproveitam melhor a capacidade do hardware e do software [10]. Existem muitos tipos de visão computacional. Hoje no mercado é comum a utilização de algumas tecnologias distintas, tais como:

- Segmentação de imagem: Examina uma fração de uma imagem.
- Detecção de objetos: Detecta um ou mais objetos dentro de um campo de imagem.
- Reconhecimento facial: Detecção avançada de objetos que pode distinguir indivíduos.
- Deslocamento dinâmico de objetos: Detecta a reorganização de pixels dentro de um campo de imagem (biblioteca NumPy).

2.1.5 Redes Neurais Profundas (Deep Learning)

Aprendizagem Profunda ou Deep Learning, é uma subárea da Aprendizagem de Máquina, que emprega algoritmos para processar dados e imitar o processamento feito pelo cérebro humano [9]. Deep Learning nada mais é do que uma evolução das redes neurais, através da utilização de algoritmos de aprendizagem e camadas de aprendizagem cria-se modelos capazes de reconhecer padrões. Existem várias arquiteturas de redes neurais, uma é a rede neural convolucional profunda demonstrada na figura 4, este tipo trabalha muito bem com entrada de dados multidimensional ou espacial que é o caso do tratamento de imagens.

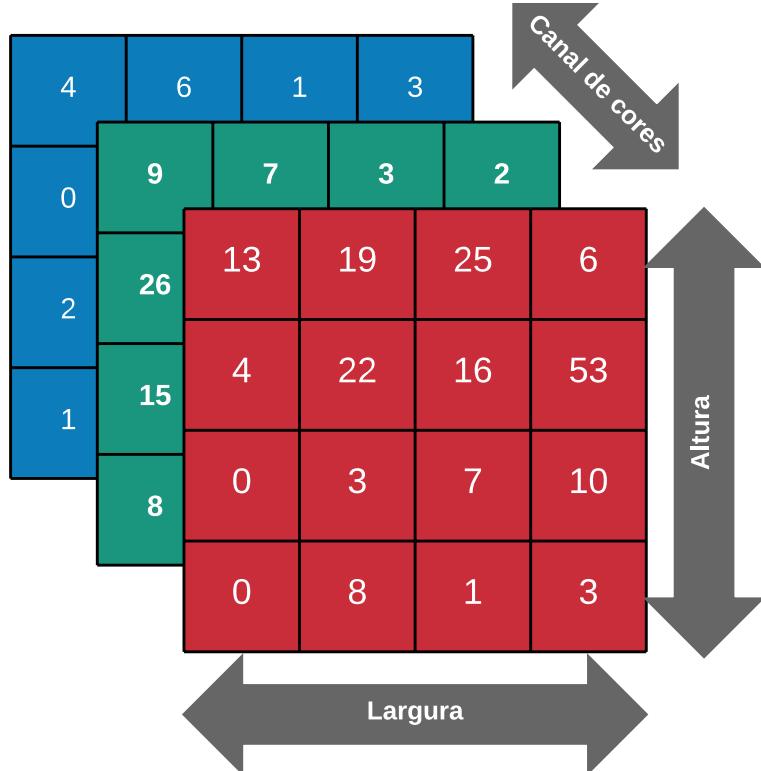
Figura 4 – Rede Neural Convolucional



Fonte: o autor com base em Lambda3, ([LONGARINI, 2019](#))

Em visão computacional as entradas são matrizes tridimensionais contendo altura e largura e profundidade, o que determina o padrão ou canal de cor do pixel, a figura 5 demonstra esse padrão.

Figura 5 – Pixel em Visão Computacional



Fonte: o autor.

A primeira coisa que precisamos para treinar um algoritmo de Deep Learning é de uma grande quantidade de imagens (Big data), e através de uma abordagem supervisionada com imagens devidamente marcadas como sendo o rosto que buscamos assim realizamos o treinamento. Então teremos um modelo que receberá imagens não marcadas e ele deverá ser capaz de classificá-las.

2.2 Tecnologias Open Source para Tratamento de Imagens

2.2.1 OpenCV

O OpenCV é uma biblioteca de código aberto licenciada por BSD que inclui várias centenas de algoritmos de visão computacional. Foi iniciada pela Intel em 1999, é uma biblioteca multiplataforma concentra seu foco no processamento de imagens em tempo real e inclui implementações sem patentes e possui todos os algoritmos mais recentes de visão computacional. Em 2008, a Willow Garage assumiu o suporte e o OpenCV 2.3.1, e ainda possui uma interface de programação para C++, Python e Android. O OpenCV possui uma estrutura modular, o que significa que o pacote inclui várias bibliotecas compartilhadas ou estáticas [12].

2.2.2 Biblioteca Dlib

O Dlib é um kit de ferramentas de linguagem de programação com licença *Boost Software License* iniciado em 2002 construído em C++ mas extensível para outras linguagens como Python, ela é utilizada largamente na indústria e academicamente para desenvolver sistemas complexos e que envolvam computação de alto desempenho. Hoje em dia ela é desenvolvida para lidar com tráfego de redes, threads, interface gráficas, estrutura de dados, e principalmente em aprendizado de máquina, processamento de imagens, mineração de dados, otimização numérica entre outros [27].

2.2.3 Biblioteca Face Recognition

É uma ferramenta de reconhecimento facial que possui simples utilização sobre a licença *MIT*, construída utilizando a biblioteca de aprendizado profundo de última geração Dlib, ela chega a uma precisão de detecções de até 99,38% segundo o site *Labeled Faces in the Wild*. É uma ferramenta muito simples de usar por ser baseada em chamadas de funções simples de identificar em sua documentação, porém é uma das melhores ou talvez a melhor ferramenta de detecção facial hoje no mundo. Com ela é possível não só detectar rostos, mas identificar partes do face como por exemplo; o nariz, boca, olhos, e com algumas outras integrações pode-se até detectar expressões faciais [28].

2.2.4 Nvidia Cuda

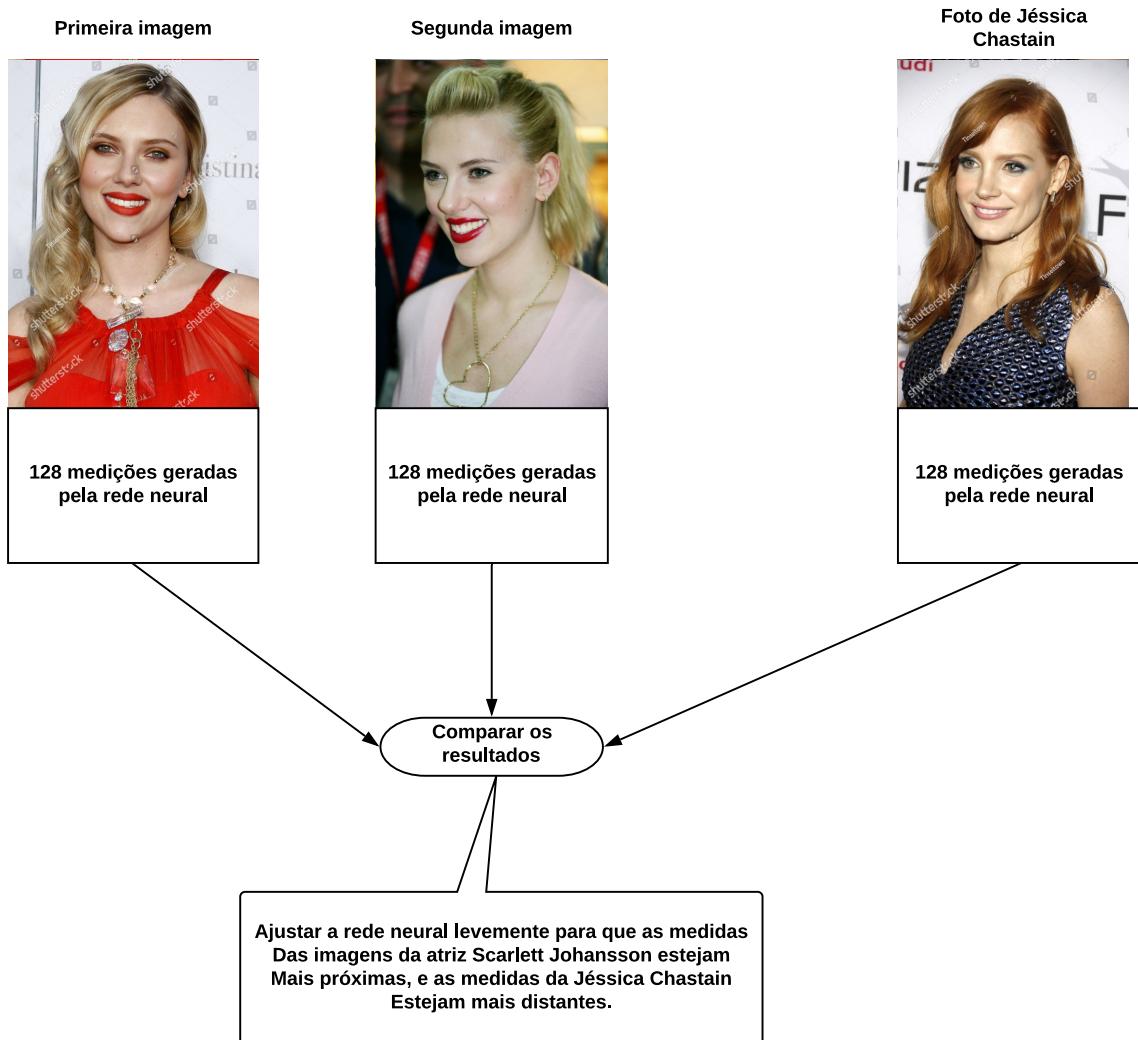
No início da computação os computadores sempre operaram com apenas um núcleo programável com códigos executados sequencialmente, porém a demanda por mais desempenho fez com que empresas empregassem esforços para criar tecnologias para melhorar o desempenho, e isso exigiu ainda mais do limite físico do silício. Para contornar isso foram criados os mecanismos de *pipeline*, *threads* e outros, uma alternativa foi adicionar processadores em paralelo e mais atualmente criaram processadores e gpus multicores com vários núcleos em um único chip. Porem foi necessário alterar as técnicas dos códigos que rodavam unicamente em paralelo para aproveitar esses novos recursos. Em 1999 a empresa americana NVIDIA ao perceber essa demanda por aumento no desempenho e um tipo de processamento específico, lançou sua primeira GPU a GeForce 256, e logo em 2000 criou as GPUs de uso geral (GPGPU). Inicialmente as GPUs tinham um processamento fixo focada em processamento de gráficos em três dimensões, mas nos últimos anos elas tem melhorado seu hardware focando no aspecto programável, e o resultado disso é uma grande capacidade de processamento focado em aritmética e não só em gráficos. Hoje as GPUs têm seu processamento focado em gráficos da classe SIMD, são desenvolvidas especificamente para cálculos de pontos flutuantes, usados em renderização de imagens, suas principais características são, massiva capacidade de processamento paralelo, total programabilidade e grande desempenho em cálculos que possua grande volume de dados. Para dominar esse

conteúdo programático era necessário um grande conhecimento do desenvolvedor de várias APIs e da arquitetura das placas gráficas além da representação através de coordenadas de vértices, texturas e shaders, aumentando drasticamente o conteúdo programado. Foi ai que foi desenvolvida a ferramenta NVIDIA CUDA em 2006, ela permite ao desenvolvedor uma facilidade ao desenvolver utilizando recursos da GPU sem a necessidade de conhecer a multiplicidade de novos componentes de programação, e utilizando todo o poder de processamento das GPUs, ela também suporta várias linguagens de programação tendo como principais o C++ e o Python. A biblioteca Dlib utiliza os recursos de processamento do CUDA para melhorar drasticamente o desempenho em processamento e tratamento de imagens, e no treinamento de redes neurais que necessitam de cálculos que possuem grande volume e dados, que seria o caso da detecção facial utilizando redes neurais profundas [29].

2.2.5 Reconhecimento Facial

Já deve ter reparado ultimamente que o Facebook possui uma grande capacidade de identificar seus amigos em suas fotos, antigamente você mesmo é quem deveria fazer essas marcações, mas agora quando você carrega uma foto, como mágica ele marca todos para você, essa técnica se chama reconhecimento facial, é uma tecnologia muito recente e incrível, o FaceBook pode chegar a um acerto de 98% de precisão ([GEITGEY, 2016](#)). Apenas reconhecer amigos é muito fácil é uma utilização muito simples para essa incrível técnica, porém no desenvolvimento desse trabalho daremos uma utilização realmente útil para essa tecnologia. As técnicas de remoção de revestimentos faciais baseados em aprendizado profundo que utilizaremos, são altamente precisas e capazes de serem utilizadas em tempo real. Será empregado o método de aprendizado métrico profundo "*deep metric learning*" que trabalha com imagens estáticas ou então fluxo de vídeo, a biblioteca de reconhecimento facial (Dlib) gera um vetor de saída conhecido como 128-d, ou seja, ele gera um vetor de tamanho 128 que quantificam a face contendo valores numéricos reais ou pesos como é mais conhecido por quem tem conhecimento na área. São chamados de pesos porque são eles que indicaram que uma imagem de entrada corresponde a uma pessoa adicionada no banco de imagens da rede neural, e ao comparar esses pesos irá gerar um resultado ou negativo (não é a mesma pessoa) ou positivo (é a mesma pessoa). Para treinar uma rede neural utilizando essas técnicas é utilizado o modelo denominado de trigêmeos, para a técnica de reconhecimento facial utilizando aprendizado métrico profundo, envolverá uma etapa de treinamento de trigêmeos. Essa técnica consiste em três imagens de entrada, onde ao menos duas das três são da mesma pessoa, o (NN) gera um vetor de 128-d para cada imagem de rosto, e logo para as duas imagens de mesmo rosto, ajustamos os pesos da rede neural para tornar o vetor o mais próximo possível via métrica de distância ([ROSEBROCK, 2018](#)),

Figura 6 – Exemplo Simples de Treinamento de Rede Neural com Método de Trigêmeos



Fonte: o autor com base em ([GEITGEY, 2016](#)).

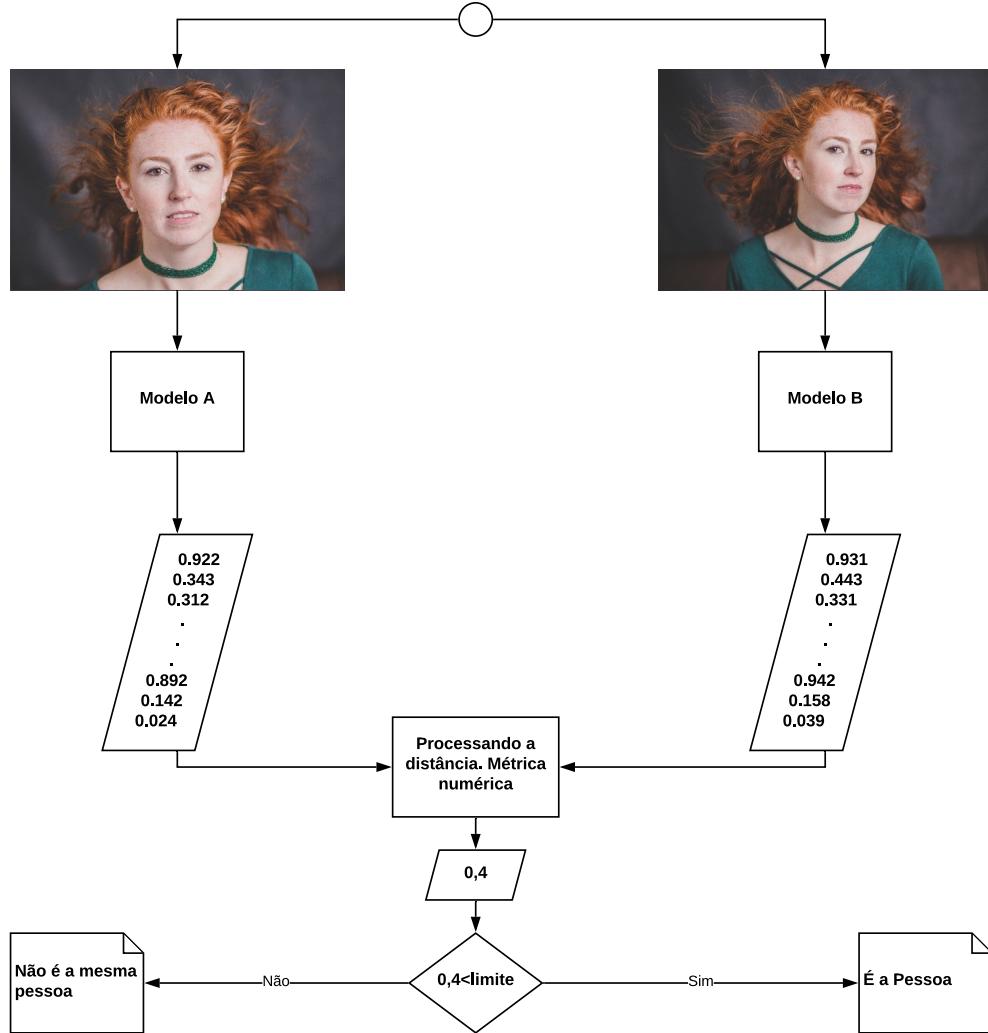
Na ilustração 6 esta uma representação que mostra uma simulação muito simples de como funciona o método com trigêmeos, duas das três faces são da atriz Scarlet Johansson que sera alvo no reconhecimento facial, e uma foto da atriz Jéssica Chastain que estará no nosso conjunto de dados. Então vamos considerar as faces, logo a ideia geral é que ajustaremos os pesos de nossa rede neural, para que as medias de 128-d das duas fotos da atriz Scarlet Johansson estejam o mais próximas o possível e mais distante da foto da Jéssica Chastain ([GEITGEY, 2016](#)).

A técnica utilizada com a biblioteca (Dlib) e (Face Recognition) é baseada no (ResNet-34) do artigo “*Deep Residual Learning for Image Recognition*” ([HE et al., 2015](#)), mas com menos camadas e a metade do numero de filtros.

A rede (Dlib) foi criada e treinada por Davis King em um conjunto de aproximadamente 3 milhões de imagens, ela pode chegar a um acerto comparado a outras técnicas de

ponta de aproximadamente 99,38% de precisão, e para concluir ela é uma rede (ResNet) com 29 camadas de convolução ([KING, 2017](#)).

Figura 7 – Exemplo de Comparaçao de Vetores 128-d



Fonte: o autor.

A figura 7 é uma representação simples de como se chega a um resultado positivo no reconhecimento facial, como podemos ver os modelos pré-treinados já com seus respectivos pesos de vetores 128-d são comparados, e se perceber os seus valores numéricos são bem próximos, perceba que os valores dentro dos parenteses da direita e esquerda são bem próximos e que o limite é 0,4, logo são comparados e como no exemplo da figura 7 se seu resultado for inferior ao limite estipulado, isso indica que são imagens de uma mesma pessoa, gerando uma saída positiva ou variável Y=1 ([GEITGEY, 2016](#)).

Na figura 8 foi realizada uma simulação utilizando o método de codificação de uma face com reconhecimento facial para extração de um vetor 128-d, e ajustamos o algoritmo para mostrar em tela os pesos que foram encontrados. Ele gera exatamente 128 valores numéricos reais.

Figura 8 – Extração de um Vetor 128-d com Reconhecimento Facial



Fonte: o autor.

2.3 Tecnologias de Linguagem de Programação para Cálculos de Álgebra e Geometria

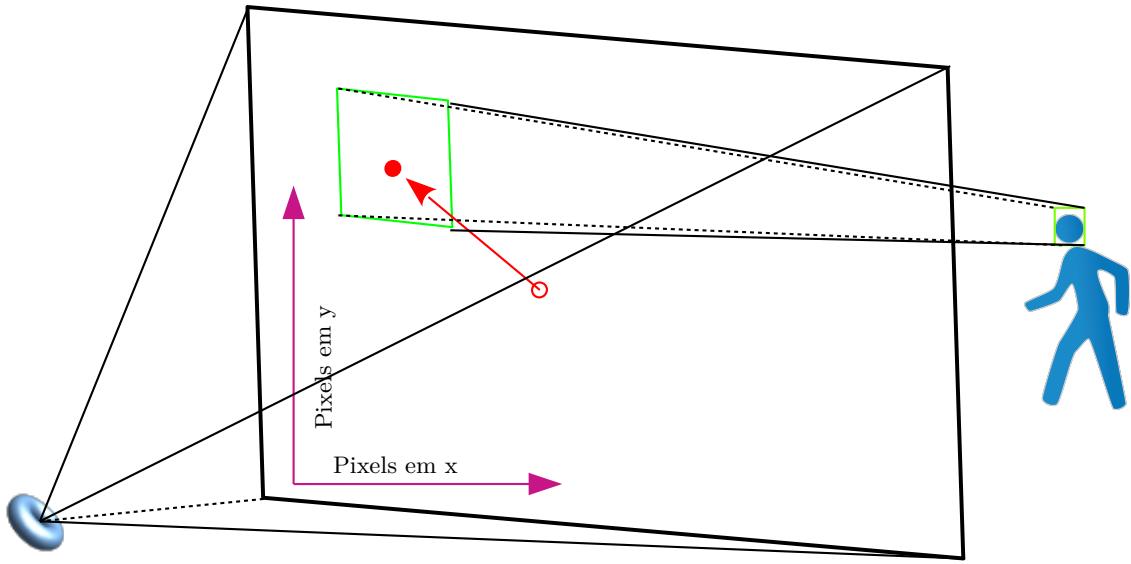
2.3.1 Biblioteca NumPy

A biblioteca NumPy é hoje uma das principais ferramentas da linguagem Python utilizadas para cálculos numéricos e estruturas de dados, entre seus recursos ela possui cálculo de matrizes muito rápido e eficiente, funções que manipulam elementos da matriz e entre matrizes, possui ferramentas para leitura e gravação de dados de matrizes em disco, operações de álgebra linear, transformada de Fourier e uma confiável geração de números aleatórios [30]. Para o trabalho de movimentação do drone, ou seguir o objeto que será focado como o ator principal do trabalho é necessário saber a posição e coordenadas x e y dos pixels, isso é possível através de cálculos principalmente de matrizes, e para isso o OpenCV suporta a biblioteca NumPy. O NumPy é uma biblioteca altamente otimizada para operações numéricas, ele fornece uma sintaxe no estilo MATLAB. Todas as estruturas de matriz do OpenCV são convertidas para matrizes NumPy de e para ela. Portanto, quaisquer que sejam as operações que você possa realizar no NumPy, você pode combiná-lo com o OpenCV, o que aumenta o número de possibilidades. Além disso, várias outras bibliotecas como SciPy, que suporta NumPy podem ser usadas [15].

2.3.2 Biblioteca SciPy

A biblioteca SciPy juntamente com a NumPy se tornam uma ferramenta extremamente madura, avançada e confiável para processamento de aplicações científicas. Há união dela com a NumPy estende muito além os recursos da NumPy, entre os recursos estão; Integração e derivações numéricas, decomposição de matrizes, solucionador de matrizes esparsas, e um grande conjunto de funções para trabalhar com cálculos que envolvem probabilidade e estatísticas. Na figura 9 um breve exemplo de como serão utilizadas essas ferramentas de cálculo no protótipo [30].

Figura 9 – Exemplo de Deslocamento Utilizando Dálculos



Fonte: o autor.

A ideia é através das ferramentas de cálculo das bibliotecas conseguir os valores absolutos em pixels da imagem total captada pela câmera, e do retângulo que é desenhado em volta da face, e através de cálculos descobrir a direção o ângulo e o valor destes vetores direcionais (seta vermelha na figura) e aproxima-los através do deslocamento do drone.

2.4 Tecnologias Open Source de Desenvolvimento, para Veículos Aéreos não Tripulados

2.4.1 Firmwares para VANT

2.4.1.1 Ardupilot

Segundo a site do desenvolvedor, o Ardupilot é o software de piloto automático de código aberto mais avançado, completo e com mais recursos disponíveis. Foi desenvolvido ao longo de mais de 5 anos por uma equipe de diversos engenheiros profissionais e cientistas da computação. É o único software de piloto automático capaz de controlar qualquer sistema de veículo imaginável, de aviões convencionais, multirotoretes e helicópteros, barcos e até submarinos. E agora sendo expandido para oferecer suporte a novos tipos de veículos emergentes, como quadriciclos e helicópteros compostos [14]. Instalado em mais de um milhão de veículos em todo o mundo e com suas ferramentas avançadas de registro de dados, análise e simulação, o Ardupilot é o software de piloto automático mais testado e comprovado. A base de código-fonte aberto significa que está evoluindo rapidamente, sempre na vanguarda do desenvolvimento da tecnologia. Com muitos fornecedores periféricos criando interfaces, os usuários se beneficiam de um amplo ecossistema de sensores,

computadores complementares e sistemas de comunicação. Por fim, como o código-fonte é aberto, ele pode ser auditado para garantir a conformidade com os requisitos de segurança e sigilo [14]. O pacote de software é instalado em aeronaves de muitas empresas como; 3DR, jDrones, PrecisionHawk, AgEagle e Kespry. Também é usado para testes e desenvolvimento por várias grandes instituições e corporações, como NASA, Intel, Boeing, além de inúmeras faculdades e universidades em todo o mundo [14].

2.4.1.2 PX4

adicionar o firmware px4

2.4.2 Ferramentas de Desenvolvimento para VANT

2.4.2.1 API de Desenvolvimento Dronekit para Ardupilot

O Dronekit é uma API feita para desenvolvedores criarem aplicativos que se comunicam com veículos (drone) por meio do MAVLink. É uma implementação em linguagem python para o controle de drones através de linguagem de programação, fornece acesso programático as informações de telemetria, estado e parâmetros a veículos conectados, permite o gerenciamento de missões e também o controle direto dos movimentos e operações do veículo. O uso e direcionado diretamente a mini computadores como Raspberry Pi, Intel Edson, Nvidia TX1 para implementar funções adicionais que a controladora do drone não suporta, entre elas podemos citar a visão computacional e a modelagem 3D. Ele suporta Linux, Mac e Windows e é disponibilizado sobre a licença de código aberto Apache 2.0.

2.4.2.2 API de Desenvolvimento MAVSDK para PX4

adicionar o mavsdk

2.4.3 Dronekit versus MAVSDK

Adicionar uma tabela de comparação entre o dronekit e o mavsdk

2.4.4 Protocolo de Comunicação MAVLink

E possível criar uma ponte de comunicação entre o computador (Raspberry Pi) e a controladora de voo (Pixhawk), e com ele pode-se capturar pacotes da controladora de voo como por exemplo; registro de telemetria, altitude, velocidade do veículo. Assim como se consegui capturar informações da controladora de voo, também é possível o contrário ou enviar comandos de voo do computador para a controladora. Todas essas informações trafegam através do protocolo de comunicação MAVLink. O MAVLink é um protocolo de

mensagens muito leve para comunicação em drones (e entre componentes de drones) [13]. O MAVLink segue um moderno padrão de design de publicação-assinatura e ponto a ponto híbrido: Os fluxos de dados são enviados/publicados como tópicos, enquanto os subprotocolo de configuração, como o protocolo de missão ou o parâmetro, são ponto a ponto com retransmissão [13]. As mensagens são definidas nos arquivos XML, cada arquivo XML define o conjunto de mensagens suportado por um sistema MAVLink específico, também conhecido como "dialeto". O conjunto de mensagens de referência implementado pela maioria das estações de controle de solo e pilotos automáticos é definido em common.xml (a maioria dos dialetos é construída sobre essa definição) [13]. A cadeia de ferramentas MAVLink usa as definições de mensagens XML para gerar bibliotecas MAVLink para cada uma das linguagens de programação suportadas. Drones, estações de controle de solo e outros sistemas MAVLink usam as bibliotecas geradas para se comunicar. Eles geralmente são licenciados pelo MIT e, portanto, podem ser usados sem limites em qualquer aplicativo de código fechado sem publicar o código-fonte do aplicativo de código fechado [13].

Características do MAVLink: Muito eficiente; o MAVLink 1 possui apenas 8 bytes de sobrecarga por pacote, incluindo sinal de início e detecção de queda de pacote. O MAVLink 2 possui apenas 14 bytes de sobrecarga (mas é um protocolo muito mais seguro e extensível). Como o MAVLink não requer nenhum enquadramento adicional, é muito adequado para aplicativos com largura de banda de comunicação muito limitada. Muito confiável; MAVLink é usado desde 2009 para se comunicar entre vários veículos, estações terrestres (e outros nós) em canais de comunicação variados e desafiadores (alta latência / ruído). Ele fornece métodos para detectar quedas de pacotes, corrupção e autenticação de pacotes. Suporta muitas linguagens de programação, executando em vários microcontroladores e sistemas operacionais (incluindo ARM7, ATMega, dsPic, STM32 e Windows, Linux, MacOS, Android e iOS). Permite até 255 conexões simultâneas na rede (veículos, estações terrestres). Permite comunicações externas e internas (por exemplo, entre um GCS e um drone, e entre o piloto automático do drone e a câmera do drone habilitada para MAVLink). A figura 6 demonstra o processo de comunicação juntamente com os processos de software, até gerar os dados de saída que são enviados para o drone por protocolo MAVLink.

2.4.5 Proxy de Protocolo MAVLink para VANT (MAVProxy)

O mavproxy é denominado de software para estação terrestre ou (GCS) ground station software, ele é totalmente funcional para o controle de (UAV) unmanned aerial vehicle ou veículo aéreo não tripulado, mas conhecido com drone. Ele foi desenvolvido pela CanberraUAV, é minimalista, portátil e extensível para qualquer UAV que suporte o protocolo MAVlink, visto no índice 2.4.4, um exemplo é o firmware ArduPilot que será abordado mais à frente. Ele foi desenvolvido para suportar computação complementar e

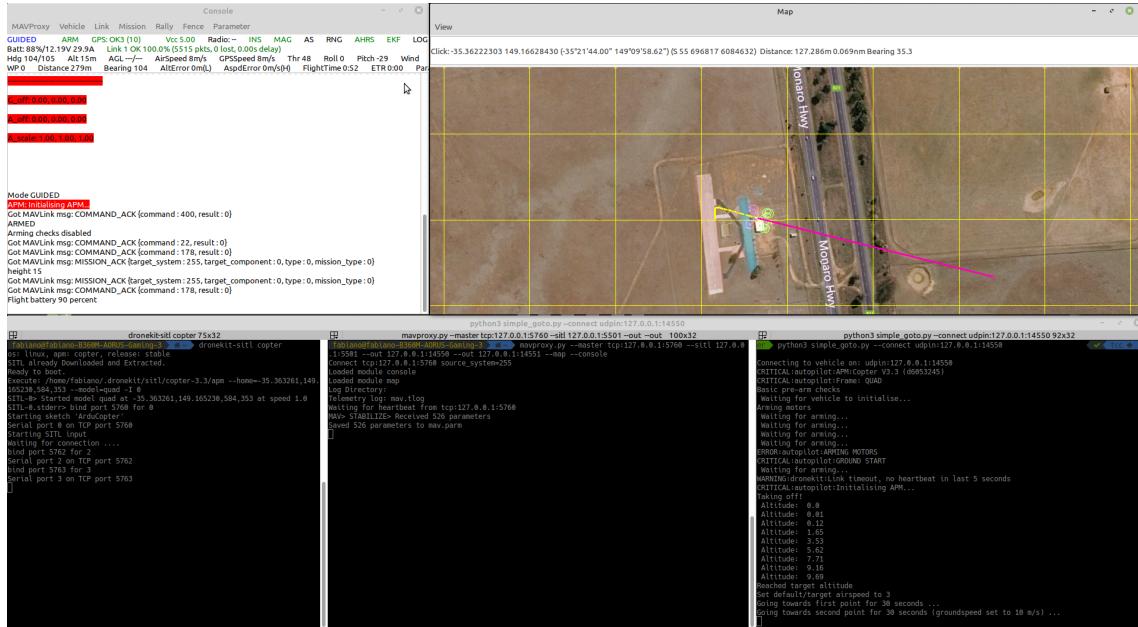
vários datalinks, é hoje a ferramenta mais versátil para o controle e desenvolvimento de software baseados em drones, principalmente aqueles direcionados ao Ardupilot. Muitos recursos existentes em outras ferramentas GCS tem sua origem no MAVProxy [25]. O MAVProxy é liberado sobre a licença GNU (general public license) v3 ou superior. Recursos:

- É baseado em linha de comando muito focada em Linux. Existem plugins que fornecem uma GUI (graphical user interface).
- Poder ser executado em localhost ou em rede em vários computadores.
- Abrange as normas POSIX (portable operating system interface), ou roda em qualquer sistema operacional que tenha chamadas na linguagem python.
- Por ser muito leve roda muito bem em computadores com processamento limitado como o Raspberry Pi.
- Tab-completion of commands.
- Possui módulos carregáveis para várias tarefas como mapas, console, joysticks.

2.4.6 Simulador de VANT SITL

O simulador SITL (software in the loop) é uma ferramenta gráfica, no caso ele possui uma GUI (graphical user interface) que implementa a simulação de funcionamento total de um drone, ele permite utilizar um plane (avião), copter (drone) ou um rover (veículo terrestre), e isso sem a necessidade de nenhum hardware, resumindo ele permite testar o comportamento do código desenvolvido com o Dronekit em um simulador de drone assim evitando possíveis imprevistos que poderiam causar algum dano material ao seu equipamento real, seu drone físico. A figura 10 mostra um exemplo do simulador rodando em um ambiente Linux, ele demonstra um drone executando um script em python que se chama simple_goto ou seja o script manda para o console do drone duas coordenadas para a qual ele deve viajar e logo em seguida retornar para a home (ponto de partida).

Figura 10 – Exemplo de Funcionamento do SITL



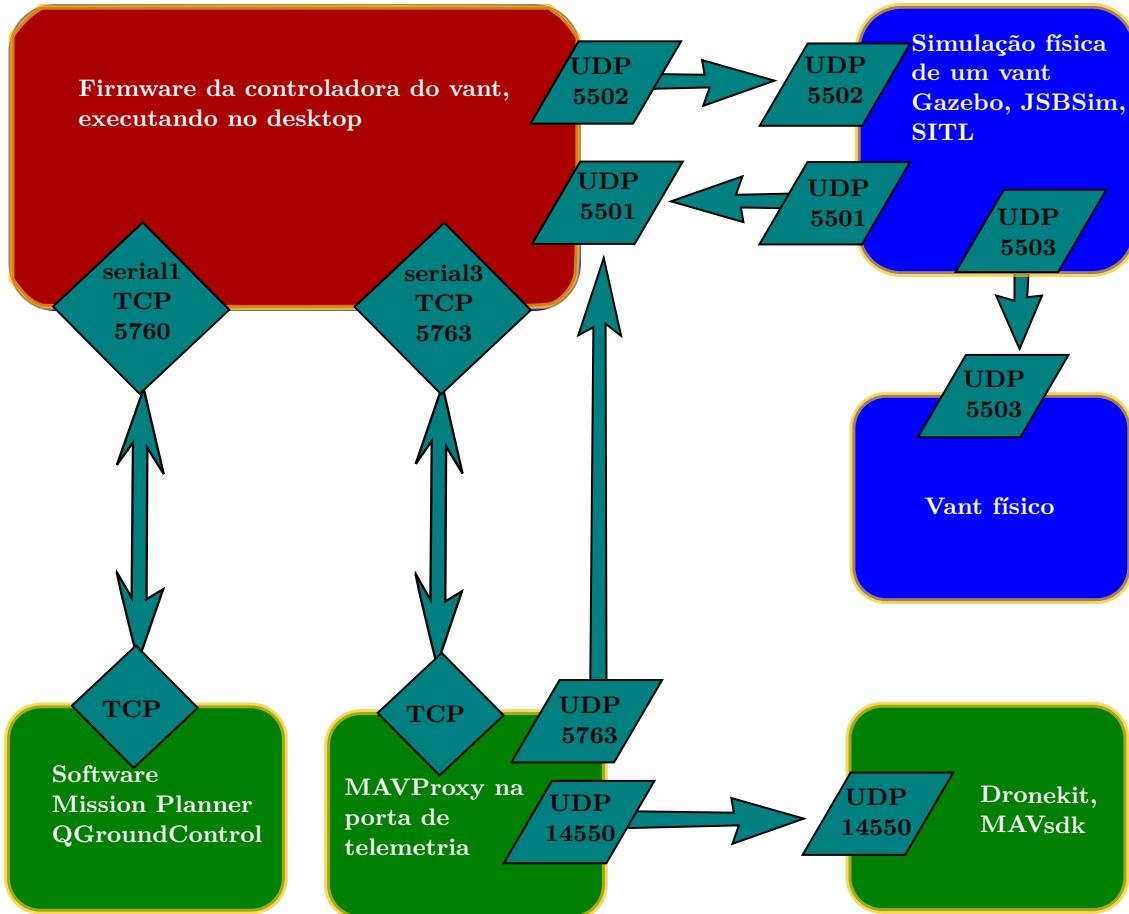
Fonte: o autor.

O SITL permite executar o firmware ArduPilot diretamente no seu computador (nootbook, ou desktop), ele suporta várias plataformas entre elas Linux, Mac e Windows. Com a junção do SITL e o ArduPilot se tem uma ampla gama de simuladores de veículos embutidos. Por exemplo:

- Drone (multi-rotor)
- Aeronave de asa fixa
- Veículo terrestre
- Veículo subaquático
- Gimbal para suporte de câmera filmadora
- É uma ampla variedade de sensores como, Lidars e sensores óticos

Com eles se torna mais fácil o desenvolvimento de aplicativos para drone, isso porque dispõe de uma gama de ferramentas como, analisadores estáticos, depuradores interativos e análise dinâmica. A simulação é uma maneira rápida, fácil e segura de testar códigos implementados para o ArduPilot antes de tentar usar (voar) no mundo real. [26]. A figura 11 mostra o fluxo de funcionamento das intercomunicações entre os módulos para criar simulações de funcionamento de um drone.

Figura 11 – Fluxo de Operação SITL, MAVProxy, Ardupilot e MAVLink



Fonte: o autor com base em Ardupilot Documents, ([ARDUPILOT, 2019](#))

2.4.7 Simulador de Robótica Gazebo

O Gazebo é um simulador de código aberto utilizado principalmente em robótica experimental, ele é um simulador muito bem projetado sendo assim tornando possível testar rapidamente algoritmos, projetar robôs, treinar sistema de IA usando cenários extremamente realísticos. O Gazebo possui um poderoso e robusto mecanismo de simulação física, ou seja, simula vários fenômenos físicos como; aceleração da gravidade, temperatura, pressão, luminosidade, e terrenos, também dispõe de gráficos de alta qualidade e uma interface gráfica e programática [31].

2.4.8 Ferramenta para Desenvolvimento de Robótica ROS

O Ros é um sistema operacional de robótica de código aberto, com ele é possível desenvolver através de linguagem de programação, comportamentos de robôs para serem testados em um simulador como por exemplo o Gazebo. Uma das principais características do ROS e o reuso de código ele disponibiliza diversas aplicações prontas para o uso, essas aplicações simulam variados tipos de robôs, sensores, atuadores, câmeras e muito mais.

Ele é implementado unicamente para o sistema operacional Linux, é baseado em versões e uma delas é a Melodic Morenia que é compatível com o Ubuntu Bionic [32].

2.5 Tecnologias e a Anatomia de Veículos Aéreos não Tripulados

Segundo a site do ArduPilot um drone ou multicoptero é um veículo aéreo mecanicamente simples, cujo movimento é controlado pela velocidade ou lentidão de várias unidades de motor/hélice para baixo. Os multicoptero são aerodinamicamente instáveis e exigem absolutamente um computador de bordo (também conhecido como controlador de voo) para um voo estável. Como resultado, eles são sistemas "Fly by Wire" e se o computador não estiver funcionando, você não estará voando. O controlador de voo combina dados de pequenos giroscópios MEMs, acelerômetros (iguais aos encontrados em smartphones) para manter uma estimativa precisa de sua orientação e posição [14].

Figura 12 – VANT do Tipo Quadricóptero



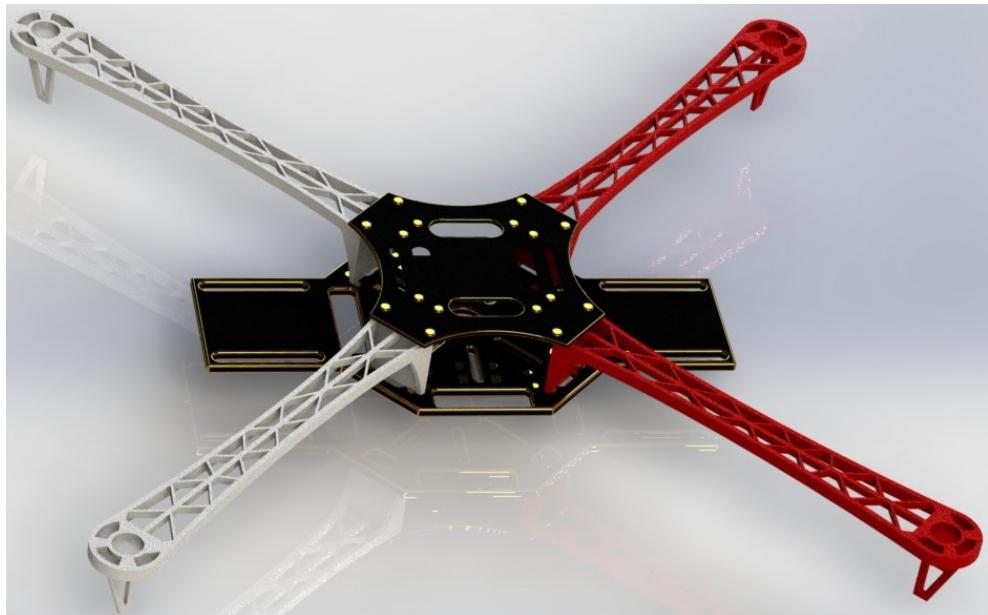
Fonte: o autor

O drone exemplo na figura 12, não será o foco do projeto porem será de importância impactante devido a sua necessidade para que o sistema projetado obtenha todas suas funcionalidades. No caso do drone ele é o meio de transporte que dará aspecto de mobilidade para que seja possível se deslocar chegando ao foco do escopo, que no caso é seguir o objeto identificado pelo sistema de visão computacional. Quando tudo estiver em funcionamento, fazendo com que o drone trabalhe de maneira autônoma ele deixara de ser um drone e se tornara um VANT (veículo aéreo não tripulado), e qual a diferença entre eles? Um drone se torna um VANT quando é capaz de vôo autônomo. Normalmente, isso significa pegar as informações de sensores como, acelerômetro e do giroscópio e combiná-las com os dados do barômetro e do GPS, para que o controlador de voo entenda não apenas sua orientação, mas também sua posição.

2.5.1 Frame

O frame é basicamente uma armação que pode ser de plástico, fibra de carbono, fibras entre outros materiais, possui alguns modelos como quadcoptero (quatro hélices), hexacoptero (seis hélices) octacoptero (oito hélices) [3]. A figura 13 mostra um frame de um quadricóptero.

Figura 13 – Frame de um VANT Quadricóptero



Fonte: o autor

2.5.2 Controladora de Voo

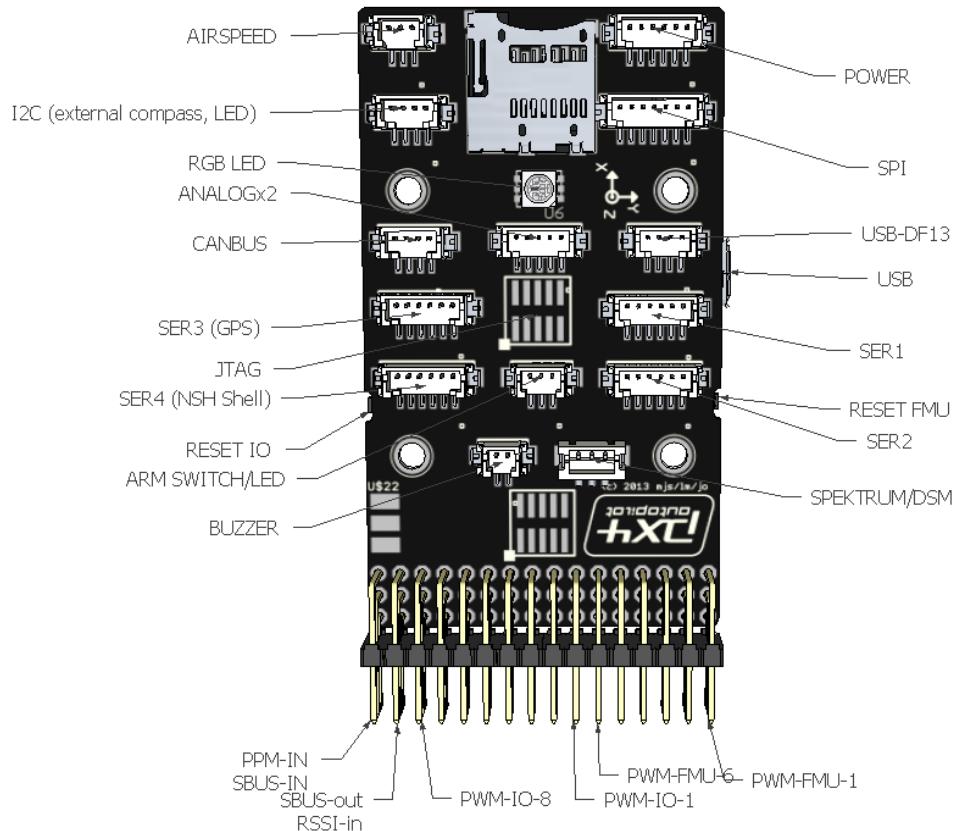
Uma controladora de voo é um hardware, é basicamente um equipamento físico composto por um circuito integrado o qual possui sensores como barômetro, gps, giroscópio, acelerômetros. Sensores que captam as atividades externas como a pressão do ar, posição geográfica, posição de equilíbrio em relação a gravidade (nívelamento), convertem essas atividades em sinais digitais e dessa forma a controladora juntamente com o firmware (Ardupilot/PX4) que possuem embarcado, conseguem interpretar esses sinais e utilizá-los para que consiga manter o equipamento em equilíbrio no ar, tornando ele um equipamento capaz de voar e manter sua estabilidade ([DRONECODE, 2020](#))[7]. As controladoras de voo ou piloto automático são utilizadas para desenvolver equipamentos de controle remoto como drones, veículos terrestres, aviões e submersíveis aquáticos. Originalmente foram fabricadas e vendido pela 3DR, hoje é possível adquirir cópias chinesas com custos bem mais em conta, devido ao projeto ser open source [3]. As figuras 14 e 15 mostram a Pixhawk 1 uma das controladoras mais utilizadas.

Figura 14 – Pixhawk 1



Fonte: Ardupilot Documents, ([PX4.IO](https://px4.io), 2020)

Figura 15 – Diagrama Eletrônico dos Conectores da Pixhawk 1



Fonte: Ardupilot Documents, ([ARDUPILOT](#), 2019)

- Processador:
 - Núcleo ARM Cortex M4 de 32 bits com FPU
 - 168 Mhz / 256 KB de RAM / 2 MB Flash
 - Co-processador à prova de falhas de 32 bits
- Sensores:
 - MPU6000 como principal acelerômetro e giroscópio
 - Giroscópio ST Micro de 16 bits
 - Acelerômetro / bússola ST Micro de 14 bits (magnetômetro)
 - Barômetro MEAS
- Alimentação:
 - Controlador de diodo ideal com failover automático
 - Todas as saídas periféricas protegidas contra sobrecorrente, todas as entradas protegidas contra ESD

- Interfaces:
 - 5x portas seriais UART, 1 com capacidade de alta potência, 2 com controle de fluxo HW
 - Entrada de Satélite Spektrum DSM / DSM2 / DSM-X
 - Entrada Futaba S.BUS (saída ainda não implementada)
 - Sinal de soma PPM
 - Entrada RSSI (PWM ou tensão)
 - I2C, SPI, 2x CAN, USB
 - Entradas ADC de 3.3V e 6.6V
- Dimensões:
 - Peso 38g
 - Largura 50 mm (2,0 pol.)
 - Altura 15,5 mm (0,6 pol.)
 - Comprimento 81,5 mm (3,2 pol.)

2.5.3 Motor sem Escova

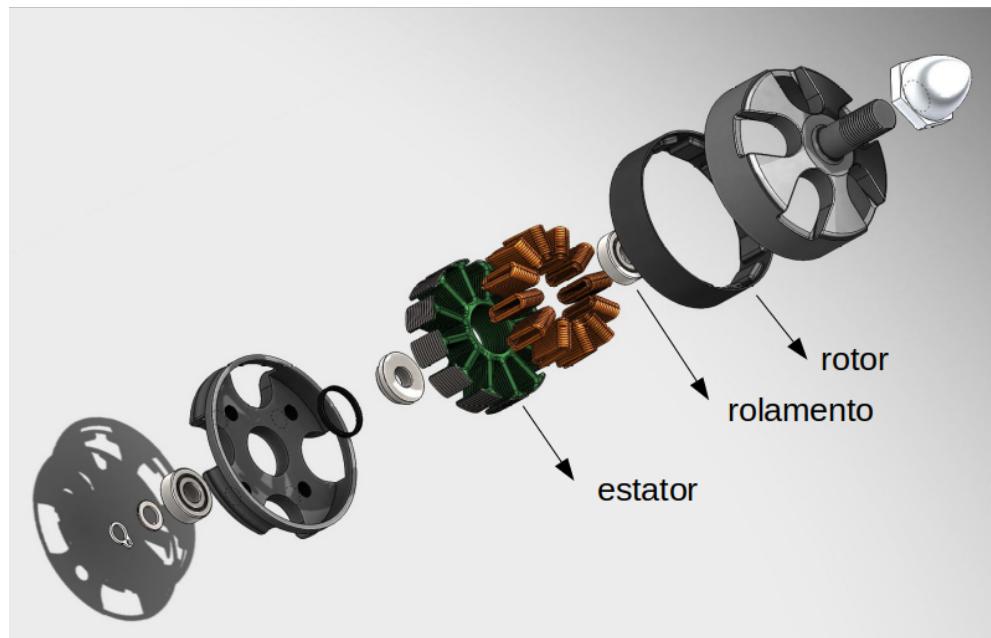
São motores de corrente elétrica continua de baixa tensão e sem escova, são síncronos e para funcionar necessitam de um (drive) ou inversor de frequência [24]. A figura 16 mostra um motor e na ilustração 17 as partes internas.

Figura 16 – Motor sem Escova



Fonte: Ardupilot Documents, ([ARDUPILOT](#), 2019)

Figura 17 – Partes do Motor



Fonte: o autor

2.5.4 Controlador Eletrônico de Corrente (ESC)

Círcuito eletrônico que controla a velocidade de rotação dos motores. A figura 18 mostra um (ESC) muito utilizado em quadricópteros comuns e na ilustração 19 o diagrama eletrônico.

Figura 18 – Controlador Eletrônico de Corrente utilizado em VANTs

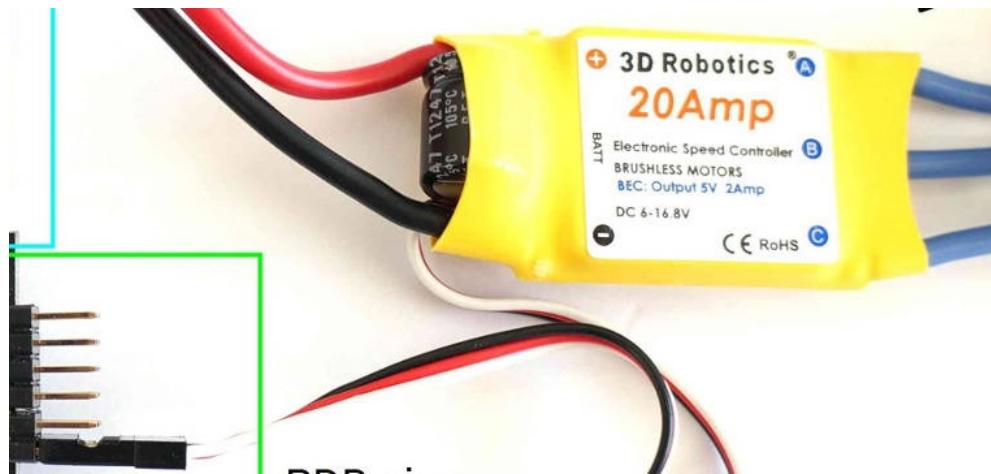
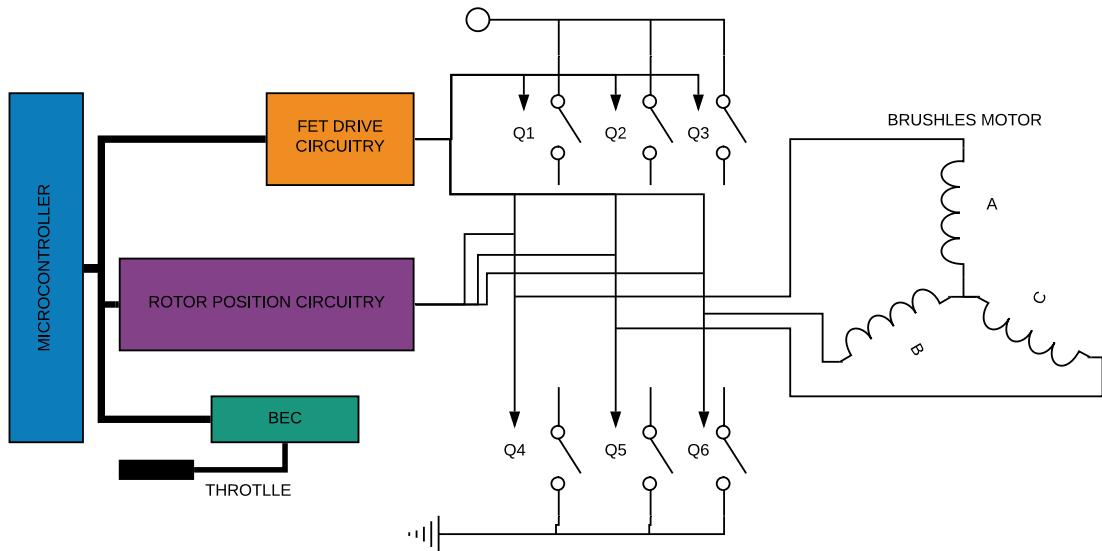
Fonte: Ardupilot Documents, ([ARDUPILOT](#), 2019)

Figura 19 – Diagrama Eletrônico de um Controlador Eletrônico de Corrente



Fonte: o autor com base em ([TEFAY et al., 2011](#))

2.5.5 Módulo de GPS

Módulo de navegação GPS + Compass, é utilizado para a orientação do drone na superfície terrestre. A figura 20 mostra um módulo de GPS.

Figura 20 – Modulo de GPS



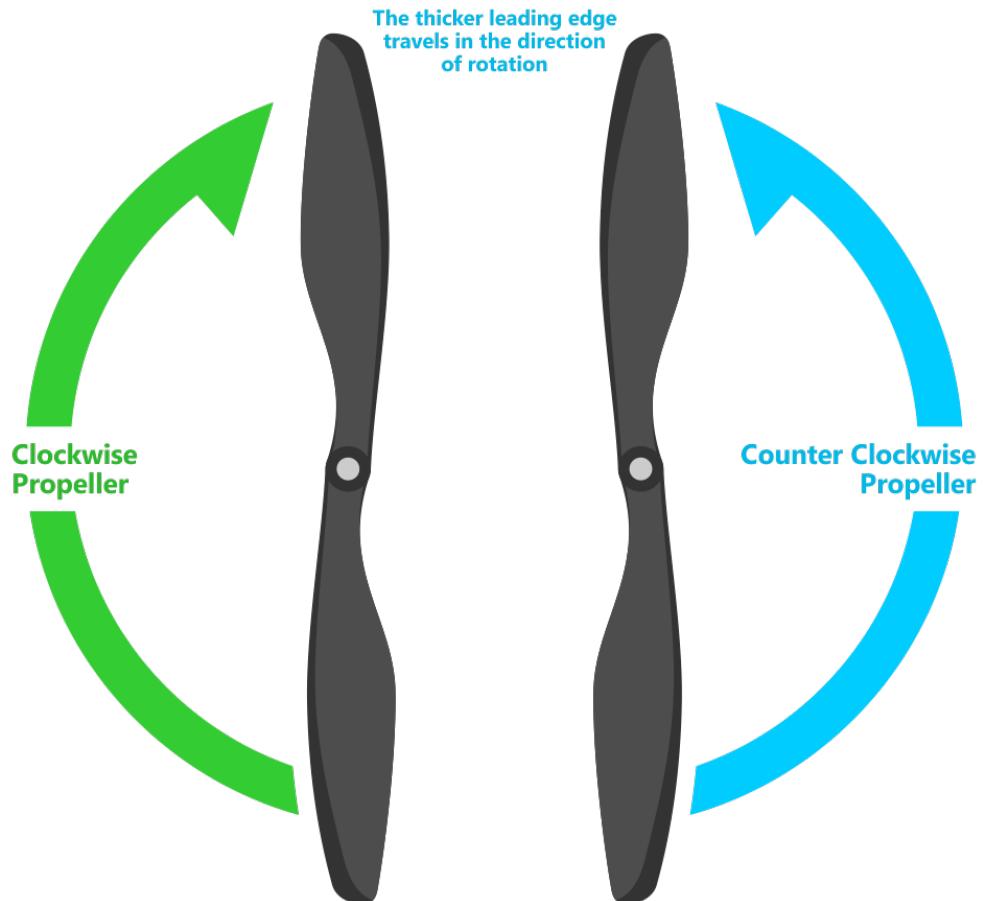
Fonte: Ardupilot Documents, ([ARDUPILOT, 2019](#))

2.5.6 Hélice

E um instrumento de propulsão ou tração, geralmente se acopla a um motor, ao girar empurra o que está na sua volta (ar ou água), assim converte energia rotacional em translacional, empurrando o objeto a que está engatada. Funcionam como asas e obedecem

aos princípios de Bernoulli e a terceira lei de Newton. Podem ser fabricadas de resinas, plásticos, fibra de carbono entre outros materiais. A figura 21 mostra um exemplo de uma hélice.

Figura 21 – Hélice



Fonte: Ardupilot Documents, ([ARDUPILOT](#), 2019)

2.5.7 Bateria de Polímero de Lítio (LíPo)

As baterias de polímero de lítio, são baterias que contêm sais de lítio num polímero sólido como o óxido de polietileno em vez de solvente, isso as torna adaptáveis ou moldáveis a diferentes formatos. Cada célula tem tensão nominal de 3,7V, sendo que utilizam elas em série chegando a mais de uma especificação de alimentação, exemplo: 2s (duas células), 3s (três células). A figura 22 mostra uma bateria do tipo LíPo 30c.

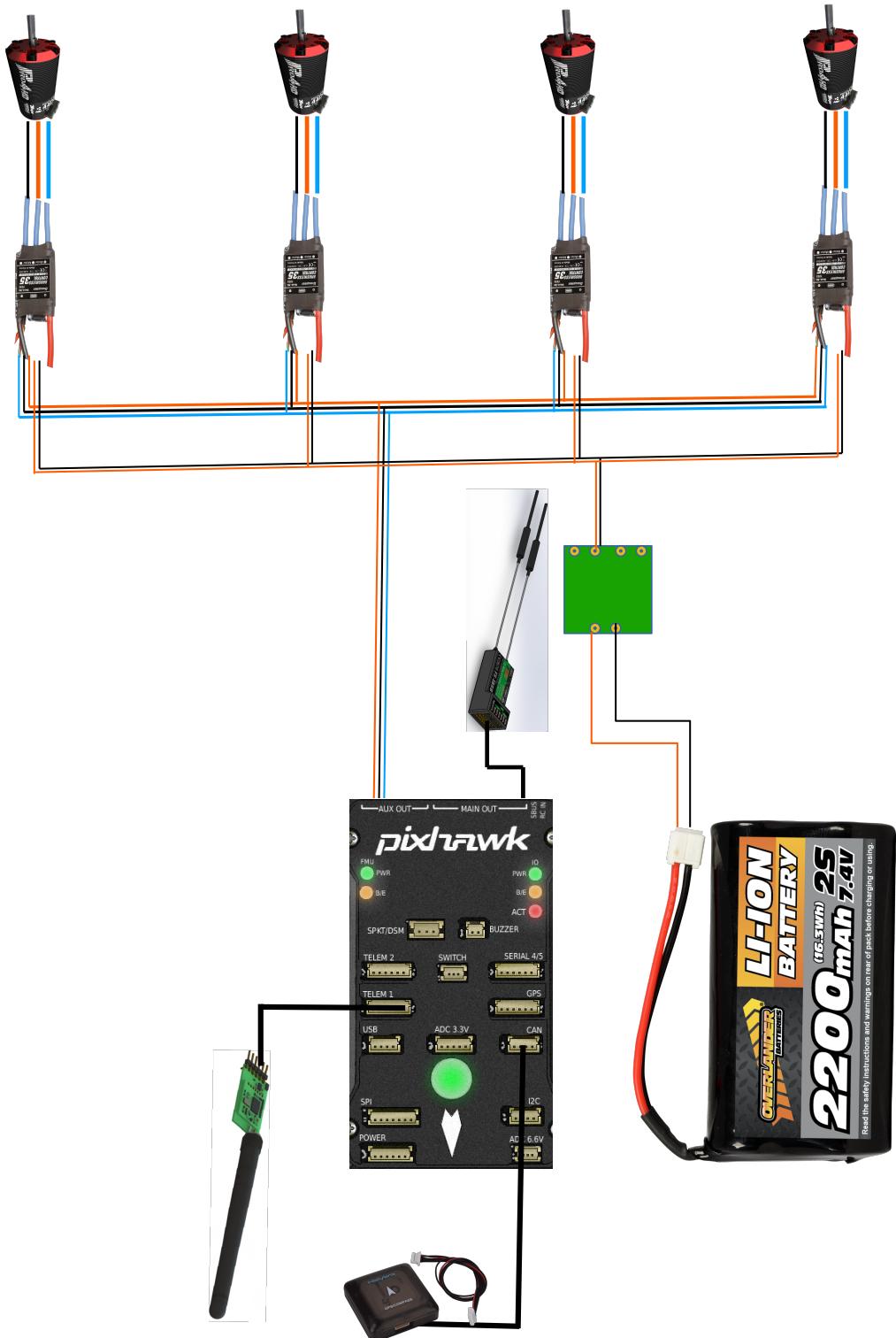
Figura 22 – Bateria de Polímero de Lítio



Fonte: Ardupilot Documents, ([ARDUPILOT](#), 2019)

2.5.8 Esquema de Montagem de um Quadricóptero

Figura 23 – Esquema Básico de Montagem de um VANT do tipo Quadricóptero



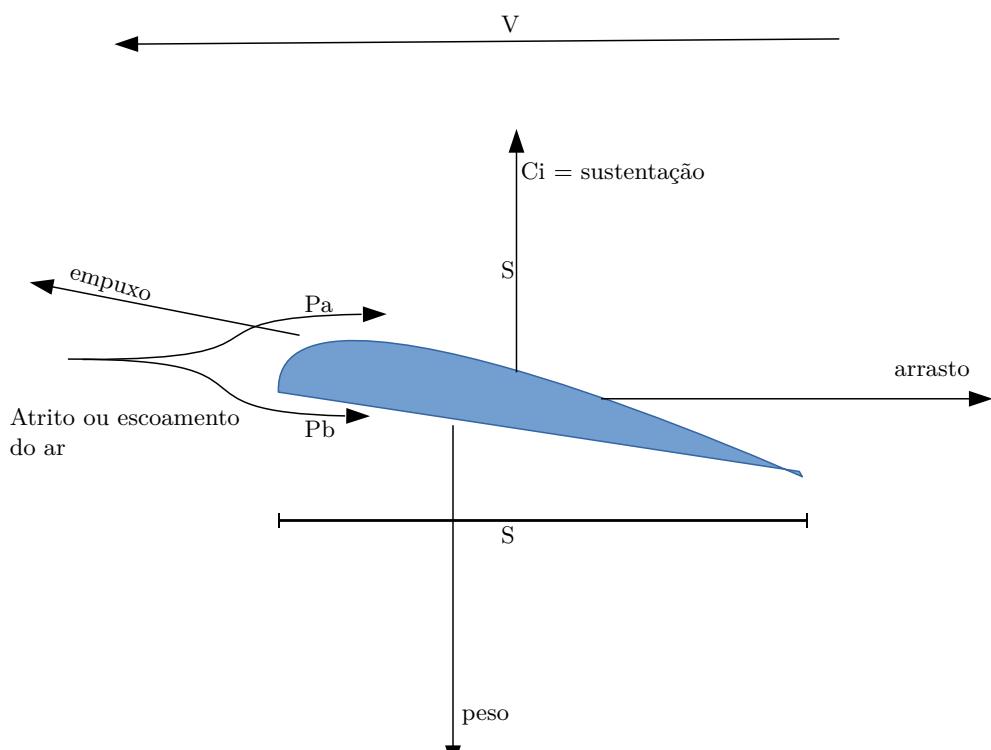
Fonte: do autor

2.6 Teoria de funcionamento e dinâmica de voo de um quadricóptero

Para iniciar a explicação deste tópico poderíamos começar falando sobre alguma das leis de Newton ou demonstrar através de cálculos que toda a ação possui uma reação de mesma intensidade e diferente direções, porem a ideia é dar uma breve introdução das reações físicas que fazem com que um VANT do tipo quadricóptero voe.

E para iniciar e preciso entender como uma estrutura do tipo hélice age para que consiga elevar e sustentar o VANT, e para isso será ilustrada a figura 24 que mostra uma representação gráfica dessas reações.

Figura 24 – Teoria de Sustentação de uma Hélice



Fonte: do autor

Como pode se ver existem quatro reações principais, o empuxo, sustentação, peso e o arrasto, na figura 24 Pb e Pa representam a pressão aplicada na hélice, na parte de baixo da hélice Pb e na parte de cima Pa, essa pressão é exercida pelo deslocamento do ar que passa pela estrutura, basicamente a sustentação é a diferença de pressão entre Pa e Pb, e o design da estrutura que faz com que a velocidade do ar embaixo da estrutura seja maior do que em cima.

- C_i = coeficiente de sustentação;

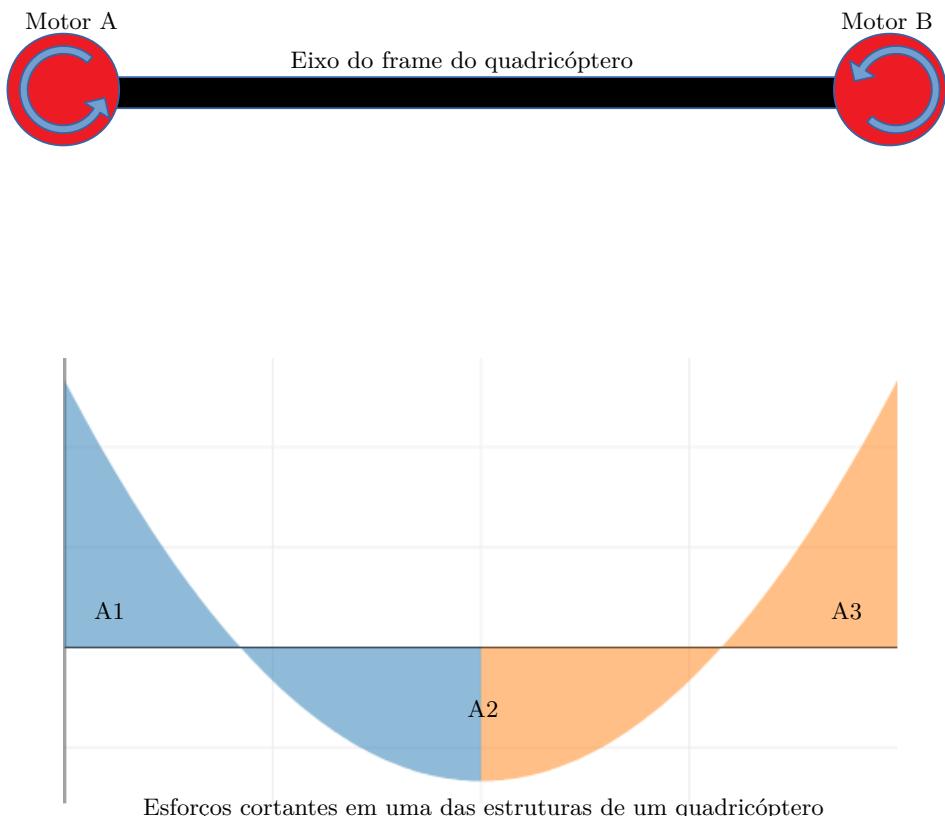
- P = densidade do ar;
- S = área da superfície da asa;
- V = velocidade do ar;
- L = força de sustentação;

A equação 2.1 prova está sustentação:

$$L = Ci \left(\frac{p}{2} \right) Sv^2 \quad (2.1)$$

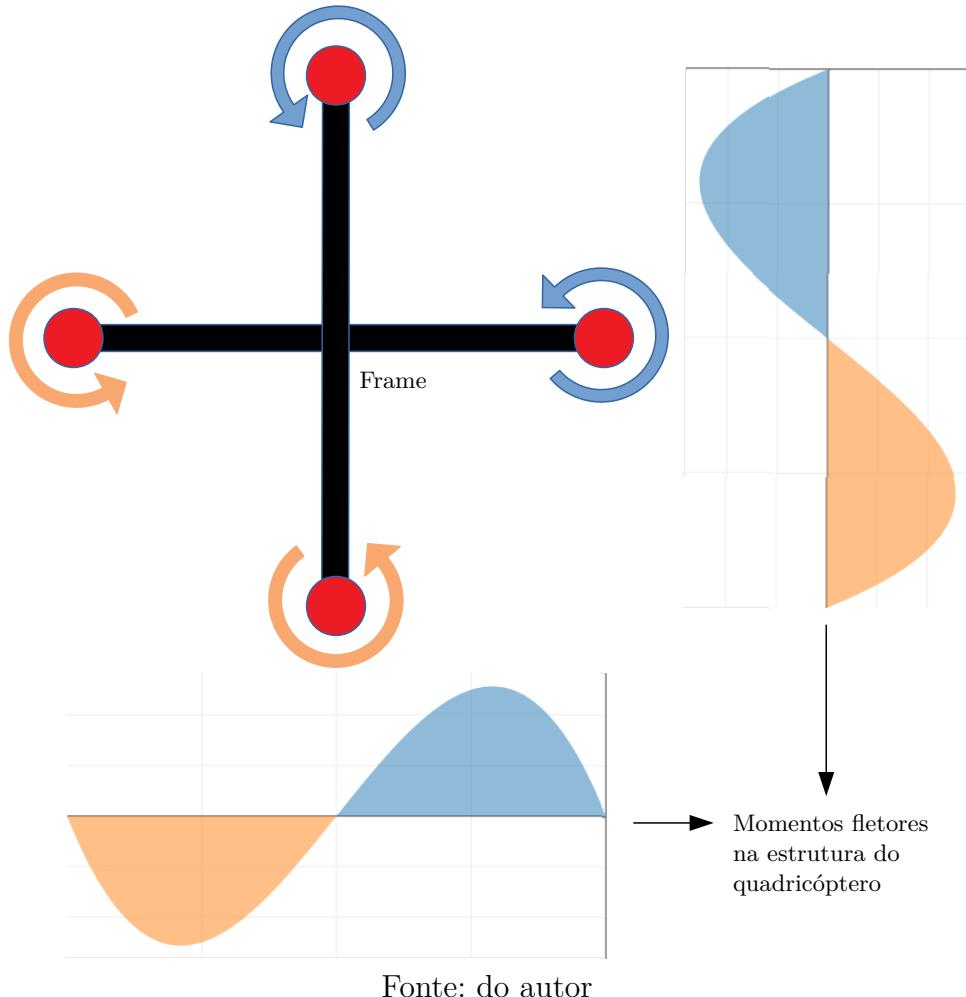
Existem outras equações matemáticas para calcular outras reações como o arrasto que seria a reação que puxa a estrutura na direção posterior da qual ela quer se deslocar ou então podemos dizer que é o rastro de ar que é deixado pela asa e tenta segura-lo, porém para este projeto é importante saber quais reações criam sustentação e fazem o veículo ganhar altitude vencendo a força da gravidade (CYPRIANO, 2014). Agora vamos dar uma olhada nas reações que são aplicadas na estrutura do quadricóptero, como elas reagem juntamente com as reações de momento que a rotação do motor aplicam sobre a estrutura. E para expressar as reações a figura 25 mostra elas graficamente.

Figura 25 – Diagrama de Esforços cortantes na Estrutura do Quadricóptero



Fonte: do autor

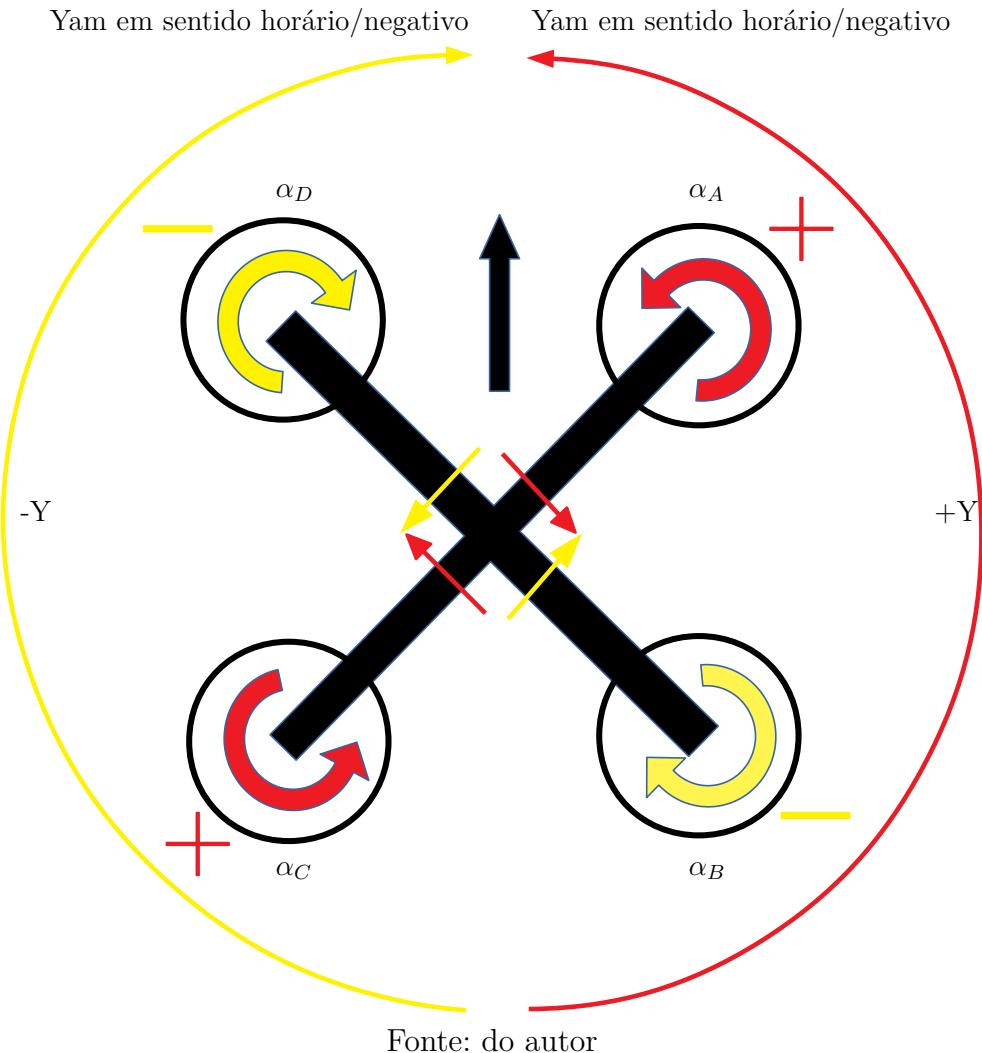
Figura 26 – Diagrama de Momento Fletor sobre a Estrutura do Quadricóptero



Na figura 25 as reações que a rotação do motor geram na estrutura no frame dois pontos de esforço cortante, logo é possível observar que elas se cancelam, no caso a área de $A_1=A_2$ e $A_3=A_2$ o que faz com que elas se cancelam e com isso o equipamento não rotacione continuamente em um sentido. Já na figura 26 os gráficos de momento fletor demonstram que todas as forças no frame do quadricóptero acabam sempre se anulando. No ponto central do frame do quadricóptero o momento angular é zero dessa forma ele não rotaciona ([BURGGRÄF ALEJANDRO RICARDO PÉREZ MARTÍNEZ, 2019](#)).

No plano vertical um quadricóptero pode subir, descer e pairar, com isso é possível deduzir que para subir serão aplicadas forças nos motores de maneira que crie uma força ascendente que supere seu peso, e logo para pairar essas forças precisam se igualar ao máximo possível, mas tendo em mente que se igualarem é fisicamente impossível devido as varias reações da física que são exercidas no quadricóptero, logo para descer ele deve diminuir a velocidade de rotação dos motores de maneira que ascenda lentamente devido ao seu peso exercer empuxo gravitacional.

Figura 27 – Sentido de Movimento dos Motores e Inercias



Na prática isso é simples os rotores giram jogando ar para baixo o ar empurra o equipamento para cima, e quem faz com que ele fique nivelado é a controladora de voo que simultaneamente com os sensores iniciais controlam a inclinação aplicando mais potência para cada motor independente de maneira que ele fique nivelado com a gravidade da terra, sim isso acontece muito rápido e freneticamente não é perceptível a olho nu (MENG et al., 2018).

Até aqui já sabemos como o quadricóptero sobe, desce, e paira, também sabemos o porque dele não girar em apenas um sentido, agora será abordada a dinâmica do quadricóptero, ação que faz do quadricóptero um equipamento incrível, tendo capacidade de se movimentar rapidamente e em direções variadas, para quem não sabe é possível realizar uma manobra de "loop" ou "flip", é a capacidade de dar um giro muito rápido virando de ponta cabeça e voltando rapidamente a posição inicial.

A figura 28 mostra todos os movimentos e seus respectivos nomes assim como são

conhecidos, o "yaw" é o giro que o quadricóptero da em cima do seu próprio eixo pode ser em sentido horário e antiaéreo, "roll" é uma inclinação lateral em relação a frente fazendo com que ele vá para a direita ou esquerda e logo o "pitch" é a inclinação que faz com que ele se desloque na direção frontal ou traseira.

Figura 28 – Movimentos de um Quadricóptero



Fonte: do autor

O VANT de quatro hélices possui basicamente dois movimentos que estimulam deslocamento, um deles é a guinada que faz com que ele se desloque horizontalmente dentro do plano cartesiano, já o movimento de giro ele rotaciona mudando a sua posição frontal e traseira respectivamente, muda o sentido de orientação da frente do equipamento como por exemplo de norte para leste e o sensor que rege esse movimento é a bussola do modulo de GPS. O movimento de giro acontece alterando a rotação dos motores, dois ao mesmo tempo no caso, um exemplo é como rotacionar ele no sentido horário. Perceba que na figura 27 α_D e α_B são negativos e α_A e α_C positivos, logo se atribuirmos a α_D e α_A um valor numérico 2 e a α_C e α_B o valor 2 igualmente chegaremos a um resultado zero o que indica que o VANT esta pairando sem girar.

A soma dos torques dos motores gera a equação 2.2 que rege o movimento de Yam.

$$(k) (\alpha_D + \alpha_A - \alpha_C - \alpha_B) = Yd^2 \left(\frac{\theta Y}{dt^2} \right) \quad (2.2)$$

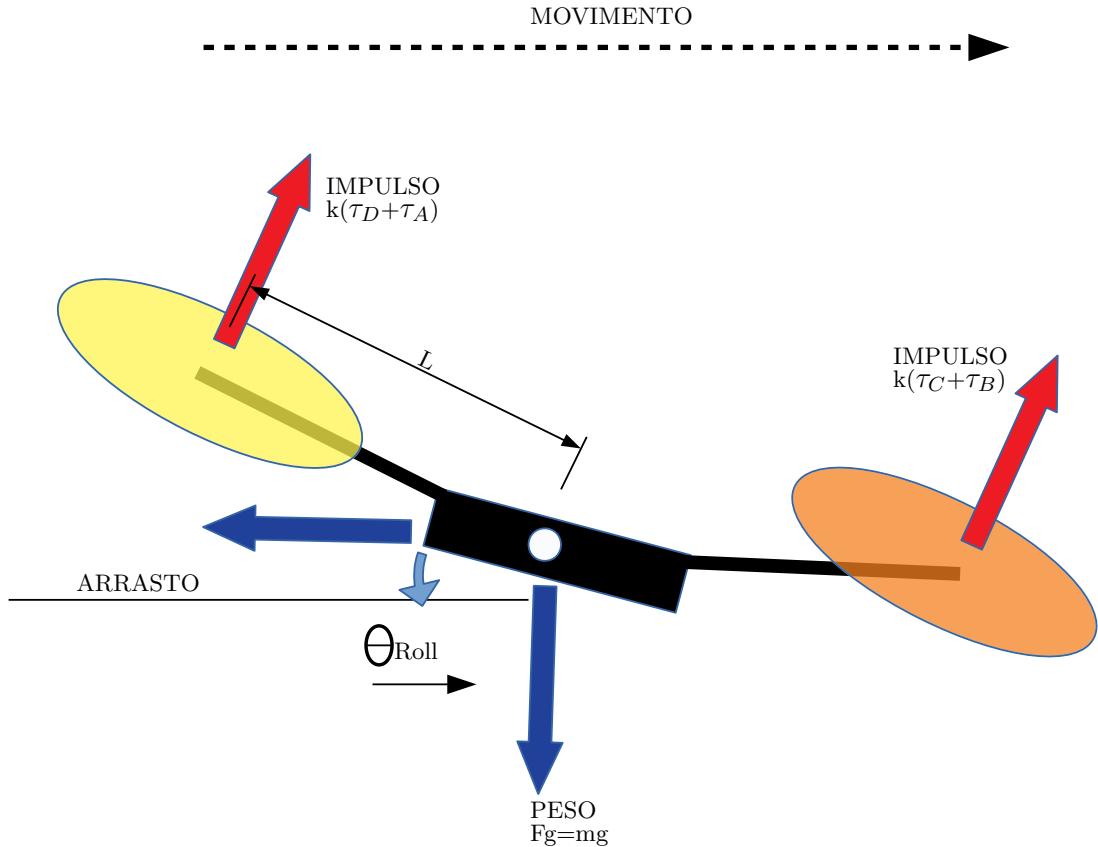
Vamos realizar outro exemplo de movimento, vamos supor que por exemplo queira girar para esquerda, segundo a figura 27 e a equação 2.2 atribuiremos para α_C e α_A o valor 3 e a α_D e α_B o valor 1, então chegaremos novamente a zero o que indica que o VANT esta estabilizado, porem a maior propulsão nos motores α_B e α_D faz com que ele gire no sentido anti-horário ou seu Y sera negativo como a ilustração 27 mostra ([BURGGRÄF ALEJANDRO RICARDO PéREZ MARTíNEZ, 2019](#)).

A mesma equação apenas com uma pequena alteração pode ser aplicada para criar o movimento horizontal conhecido com "roll".

A soma dos momentos no centro de massa gera a equação que rege "roll".

$$Lk (k) (\alpha_D + \alpha_A - \alpha_C - \alpha_B) = Rd^2 \left(\frac{\theta R}{dt^2} \right) \quad (2.3)$$

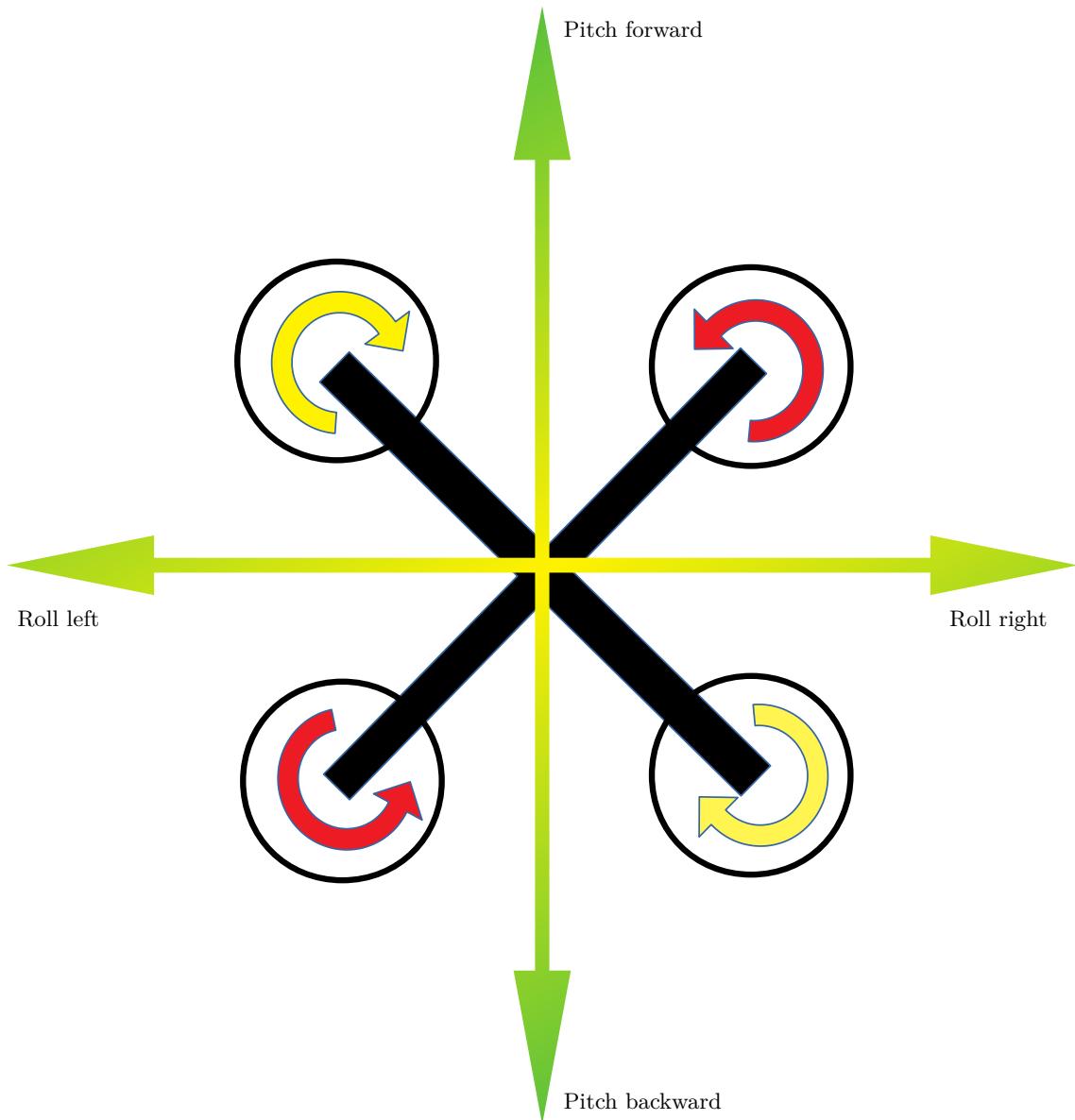
Figura 29 – Movimento Horizontal "Roll"



Fonte: do autor

A figura 29 ilustra uma vista lateral do movimento horizontal "roll" e forças que o regem, perceba que o VANT esta inclinado mas como se consegue coloca-lo nesta posição? O aumento do impulso nos motores α_C e α_B e diminuição nos motores α_D e α_A . O impulso total continua igual zero ou seu momento angular resulta em zero, porem a inclinação faz com que ele vá na direção em que esta inclinado, e a equação 2.3 e quem define esse movimento horizontal. E para concluir a ilustração 30 denomina todos os movimentos horizontais de um quadricóptero (MSARDONINI, 2015).

Figura 30 – Movimentos Horizontais de um Huadricóptero

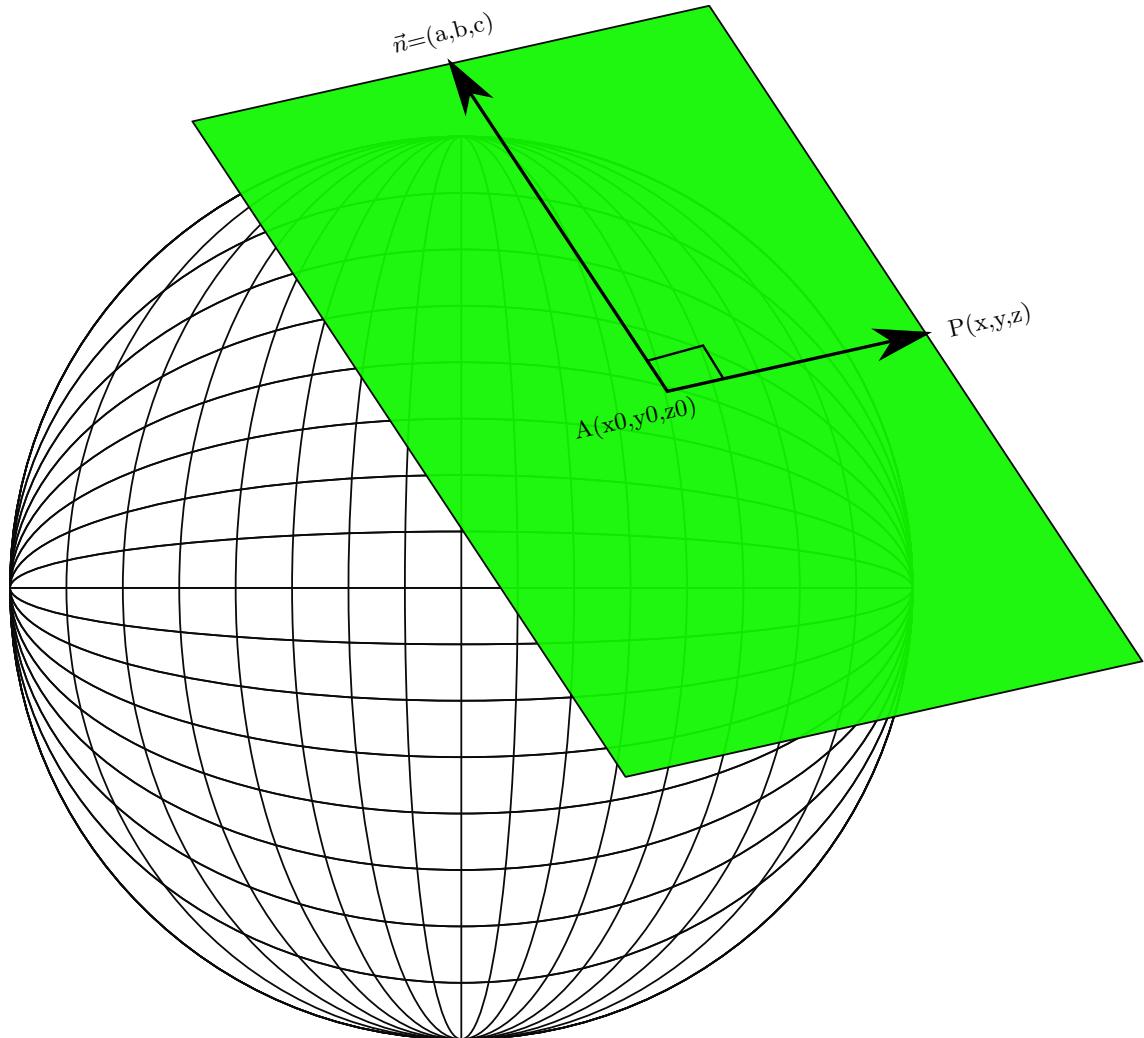


Fonte: do autor

Para orientar-se e se deslocar na superfície terrestre o VANT se orienta pelo sistema NED (North, East ,Down) que se baseia nos "cálculos de coordenadas do plano tangencial", o qual equaciona como um plano se desloca por uma superfície esférica, sendo regido e se parecendo com a equação 2.4. Isso é um conteúdo de geometria analítica e não é de difícil entendimento sendo encontrado em qualquer material de álgebra básica e para melhor entendimento a ilustração 31 foi adicionada.

$$PT = a(x - x_0) + b(y - y_0) + c(z - z_0) \quad (2.4)$$

Figura 31 – Plano Cartesiano na Superfície Esférica



Fonte: do autor

Como foi abordado o VANT se orientar usando o sistema de coordenadas NED aonde N é o eixo Norte e Leste D é um eixo que aponta para baixo do VANT chegando próximo ao centroide da terra, são vetores de orientação usados na aviação aérea e cibernética marinha. Ele é usado comumente em sistemas GPS para movimentar-se ao redor do globo e são tangentes as linhas das coordenadas geográficas. A ilustração 32 mostra graficamente todas as referencias entre o frame de um VANT com o sistema de coordenadas:

- A tangente Leste-Oeste em paralelo com os paralelos;
- A tangente Norte-Sul em paralelo com os meridianos;

Também é possível observar na ilustração 32 como funciona o sistema de orientação do Dronekit e do firmware:

- O movimento pitch esta relacionado com o angulo de inclinação ϕ aonde uma rotação ante-horário o movimenta para frente tendo X negativo e uma rotação horário movimenta para traz tendo X positivo;
- O movimento roll esta relacionado com o angulo de inclinação λ aonde uma rotação ante-horário o movimenta para esquerda tendo Y negativo e uma rotação horário movimenta para direita tendo Y positivo;
- O movimento de rotação yaw esta relacionado com ψ , também pode ser chamado de azimute porque o giro do VANT em yaw ocorre ao longo do azimute da terra;

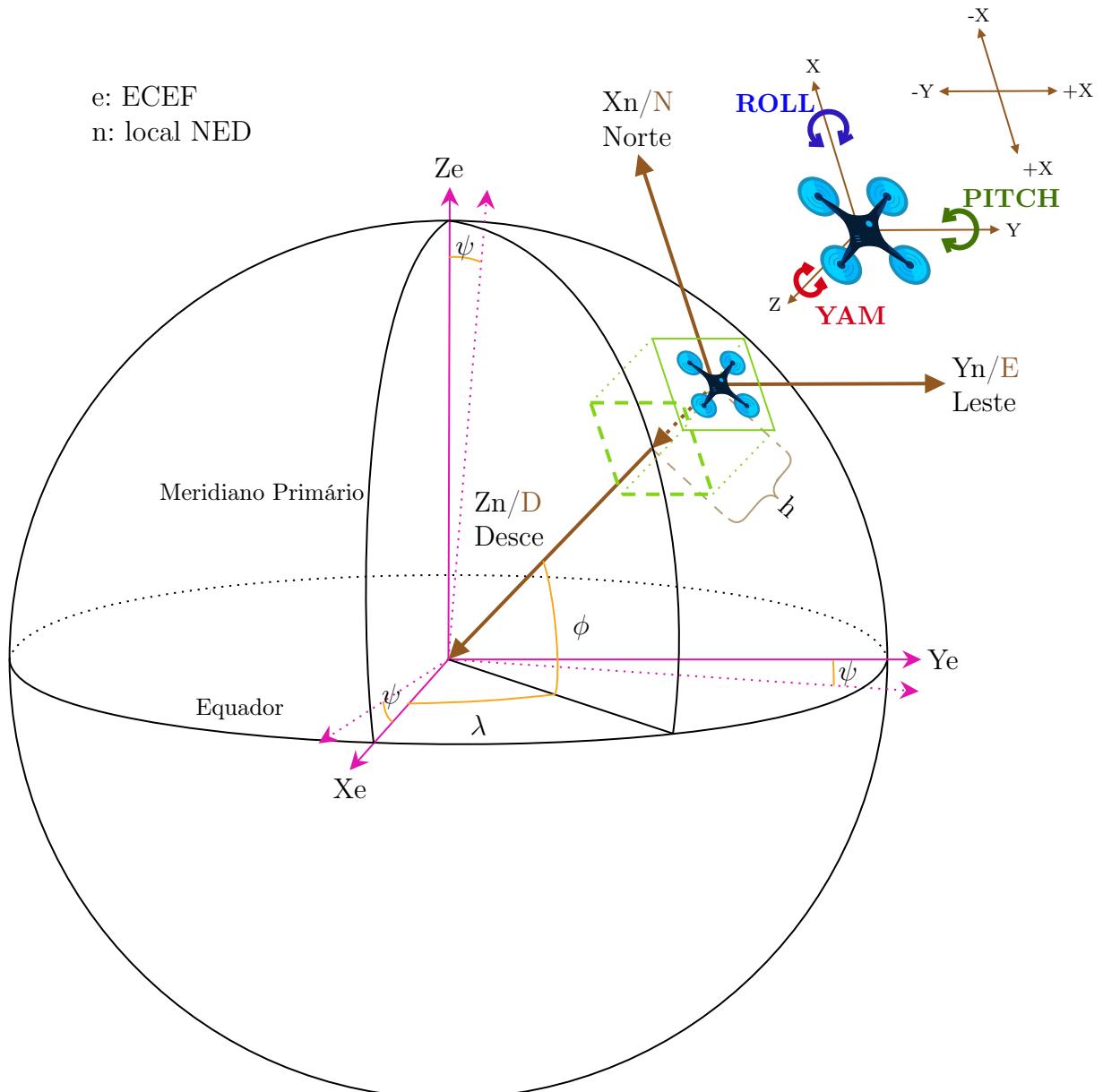
Existe uma relação entre o sistema NED e ECEF, aonde ECEF é o sistema padrão de coordenadas geodésicas e a relação de $n(X,Y,Z)$ com $e(X,Y,Z)$ é uma matriz de rotação do tipo 2.5.

$$R = \begin{pmatrix} -\sin(\phi)\cos(\lambda) & -\sin(\lambda) & -\cos(\phi)\cos(\lambda) \\ -\sin(\phi)\sin(\lambda) & \cos(\lambda) & -\cos(\phi)\sin(\lambda) \\ \cos(\phi) & 0 & -\sin(\phi) \end{pmatrix} \quad (2.5)$$

E a equação 2.6 relaciona ECEF com NED, aonde P é o ponto de intersecção do plano com a superfície da terra ou o centro do frame do VANT.

$$NED = R(ECEF - P) \quad (2.6)$$

Figura 32 – NED



Fonte: o autor com base em ([MUNGUÍA, 2014](#)), ([KISSAI; SMITH, 2019](#)), ([JSBSIM, 2020](#)).

Em NED "N"equivale ao eixo X (Norte ou Sul) direção frontal do VANT, "E"equivale ao eixo Y (Leste ou Oeste) direção lateral do VANT e "D"equivale ao eixo Z ou para baixo (aponta para o centro da terra), isso porque para uma aeronave o que interessa esta abaixo dela, e tem seu termino bem próximo ao centro da terra, h é a altura do VANT em relação ao solo ([KISSAI; SMITH, 2019](#)).

3 Metodologia

Segundo ([RAMPAZZO, 2005](#)) metodologia vem do grego (*methodos + logia*) ou "estudo do método", e método do grego (*methodos*) (*methà + odon*) quer dizer "o caminho para chegar", logo metodologia é o estudo de um conjunto de etapas dispostas em ordem para o estudo de uma determinada ciência e para alcançar um objetivo científico.

Neste capítulo e etapa do estudo serão explanadas as simulações que foram realizadas para chegar a uma conclusão sobre o funcionamento do protótipo.

Foi escolhido que os testes e simulação para validar o protótipo seria feito em módulos delimitados em funções específicas do protótipo, isso porque devido a algumas especificações como; tempo, custo verificou-se que não seria possível realizar um teste total do protótipo. Porem é devidamente valido e viável realizar testes modulares provando que a maior parte dos resultados esperados funciona.

O funcionamento total seria por exemplo um teste de campo utilizando um vant real em um ambiente real enfrentando condições reais de temperatura, pressão, vento, luminosidade entre várias outras condições adversas que poderiam afetar o funcionamento. Porem depois de toda a pesquisa descobriu-se que testes de funcionamento que envolvam robótica ou mais especificamente um vant, são realizados primeiramente com simulações utilizando softwares e ferramentas desenvolvidos para esse fim, logo optou-se por validar o protótipo utilizando-se estas ferramentas. Devido a alguns objetivos que especificam a utilização de um computador complementar (Raspberry Pi) que sera acoplado ao vant e sera responsável pela parte de visão computacional devido ao seu baixo peso e pequeno tamanho, foi feito um *benchmark* da execução do software de visão computacional comparando os resultados com o do computador utilizado para rodar os simuladores.

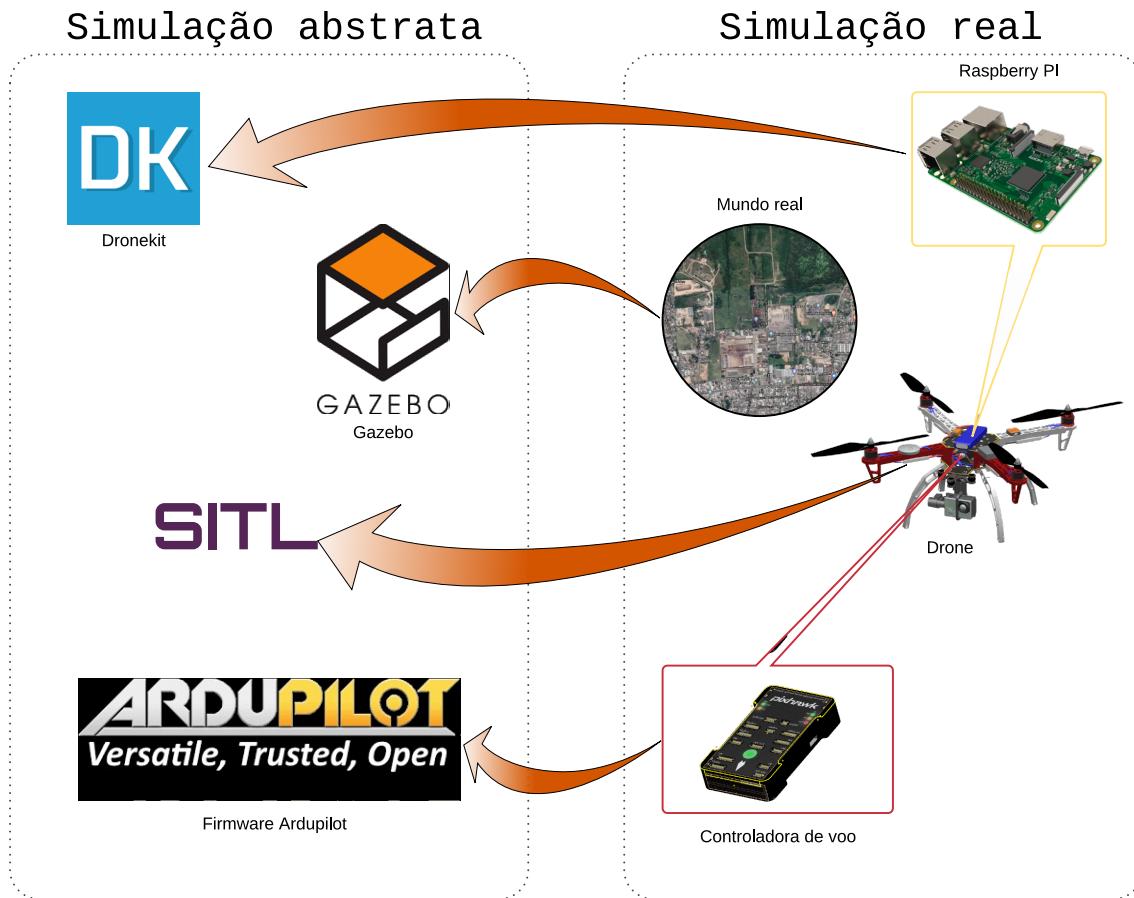
De maneira a auxiliar na compreensão de como funcionara o sistema a ilustração [33](#) representa a maneira como uma simulação real é projetada para o sistema de simulação abstrato, perceba que existe uma divisão entre cada ferramenta. Iniciando de cima para baixo nos temos:

- O computador complementar (Raspberry Pi) que comportou a ferramenta de programação para VANTs Drokit é abstraída pelo próprio Dronekit API que funcionara juntamente com o software de visão computacional.
- O ambiente físico, gravidade, vento, terreno e luminosidade no caso o local aonde o VANT seria pilotado, e abstraído pelo software Gazebo.
- O equipamento de voo quadricóptero é simulado pelo simulador de VANTs SITL

que vem embutido no firmware Ardupilot.

- A controladora de voo Pixhawk que executa o firmware Ardupilot é simulado pelo próprio, que já possui embutido um sistema que possibilita testes e simulações através de software.

Figura 33 – Simulação real para abstrata



Fonte: a autor.

3.1 Métodos

Para uma melhor compreensão do funcionamento do sistema foram desenvolvidos diagramas, fluxogramas e gráficos que explicam modularmente cada processo. Simulações com software demonstraram o funcionamento integrado de alguns desses módulos.

Uma benchmark de hardware também será realizado como comparação e validação do protótipo.

Os módulos foram subdivididos, um deles sera o de visão computacional que validara o reconhecimento facial, esse modulo vai validar as funções de distinção entre um

humano inserido no banco de busca e outros que não foram, numero de falsos positivos detectados.

3.2 Componentes Físicos

Os componentes físicos são o computador utilizado nas simulações, computador complementar (Raspberry Pi) e os dispositivos de captura de imagem utilizado nos testes.

3.2.1 Computador Utilizado nas Simulações

Devido ser necessário executar simultaneamente varias ferramentas de simulação que na sua maioria se utilizam de interfaces gráficas que necessitam de renderização em tempo real, e com um sistema de visão computacional que necessita de uma GPU e boa quantidade de memoria RAM para conseguir executar sua tarefa de reconhecimento facial, foi preciso um computador com hardware de tecnologia de vanguarda.

- Especificações do Computador:
 - Processador:
 - * Intel Core i7 8700
 - * 6 núcleos de processamento
 - * 12 threads
 - * 3.20GHz de frequência base
 - * 4.60GHz de frequência máxima
 - * 12MB de memória cache
 - L1 (Data) cache 6x32 KBytes
 - L1 (Instruction) cache 6x32 KBytes
 - L2 cache 6x256 KBytes
 - L3 cache 12 MBytes
 - Placa de Video:
 - * NVIDIA GFORCE RTX2070
 - * 2304 NVIDIA CUDA Cores
 - * 1620 MHz de clock máximo
 - * 1410 MHz de clock base
 - * 8GB de memoria GDDR6
 - * Interface de memoria de 256bit
 - * Taxa de transferencial 448GB/s

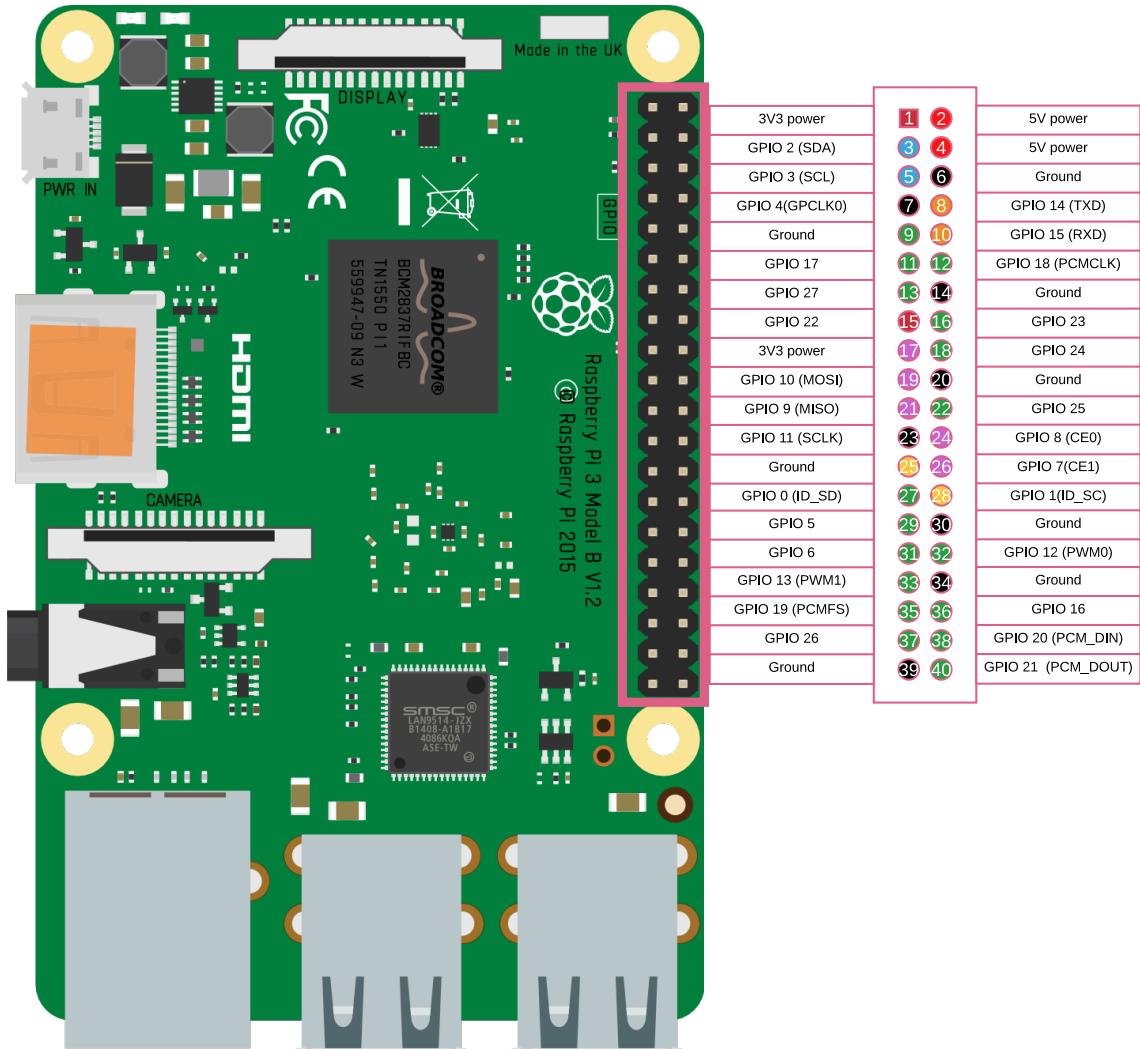
- Memória RAM:
 - * 16GB de capacidade
 - * Taxa de transferência 3200Mb/s
 - * Tecnologia DDR4

- Armazenamento:
 - * 1TB SSD
 - * Taxa de leitura gravação 500MB/s e 450MB/s
 - * Interface SATA 3

3.2.2 Computador Complementar

Devido ao fato da pesquisa propor um vante que acople um computador que comportaria o software de visão computacional, foi estipulado que este seria um Raspberry Pi, pelo seu reduzido tamanho e peso, Foi feita uma implementação do sistema de reconhecimento facial que foi utilizado na simulação, com o Raspberry Pi para provar seu funcionamento em uma possível simulação real.

Figura 34 – Raspberry Pi 3 B



Fonte: o autor com base em ([RASPBERRYPI.ORG](https://www.raspberrypi.org), 2020b).

- CPU Broadcom BCM2837 de 64 bits Quad Core de 1.2GHz
- 1GB RAM
- LAN sem fio BCM43438 e Bluetooth Low Energy (BLE)
- 100 Base Ethernet
- Extensão GPIO de 40 pinos
- 4 portas USB 2
- Saída estéreo de 4 polos e porta de vídeo composto
- HDMI em tamanho real
- Porta de conexão para câmera CSI Raspberry Pi

- Porta de conexão para display DSI Raspberry Pi sensível ao toque
- Porta Micro SD para carregar seu sistema operacional e armazenar dados
- Fonte de alimentação Micro USB comutada atualizada até 2.5^a

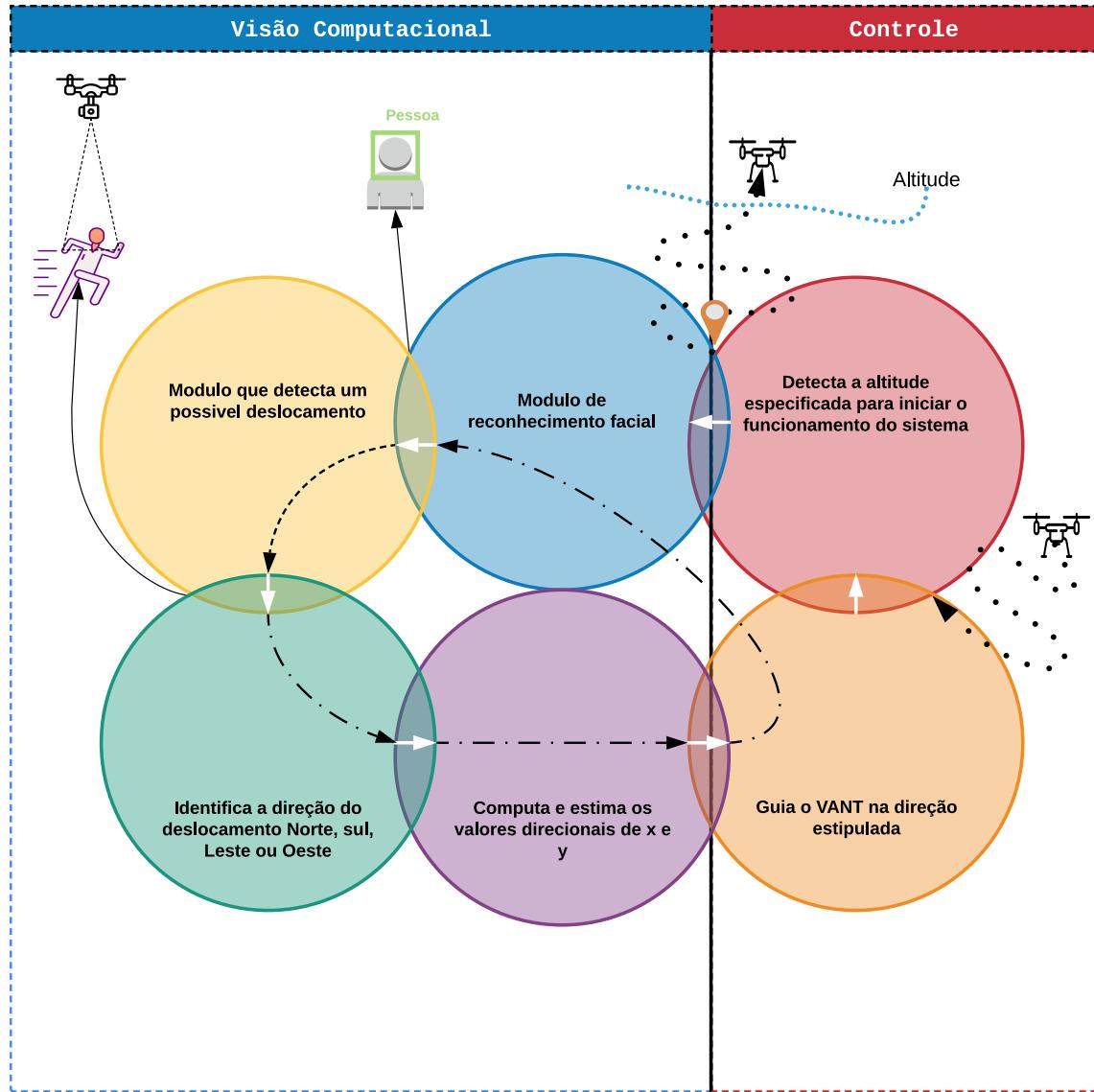
3.2.3 Dispositivo de Captura

3.3 Sistemas Operacionais

Para implementar as simulações no computador foi utilizado o sistema operacional linux Mint 19, baseado no linux Ubuntu Bionic. Para implementar o sistema de reconhecimento facial no computador complementar foi utilizado o sistema operacional linux Raspbian sem interface gráfica, que também é baseado no Ubuntu Bionic. A maioria das ferramentas são desenvolvidas para sistemas UNIX e que se baseiam no Linux Debian e Ubuntu por isso se chegou nessa escolha.

3.4 Fluxogramas dos Protótipos e Funcionamento do Sistema

Figura 35 – Diagrama de Funcionamento do Sistema

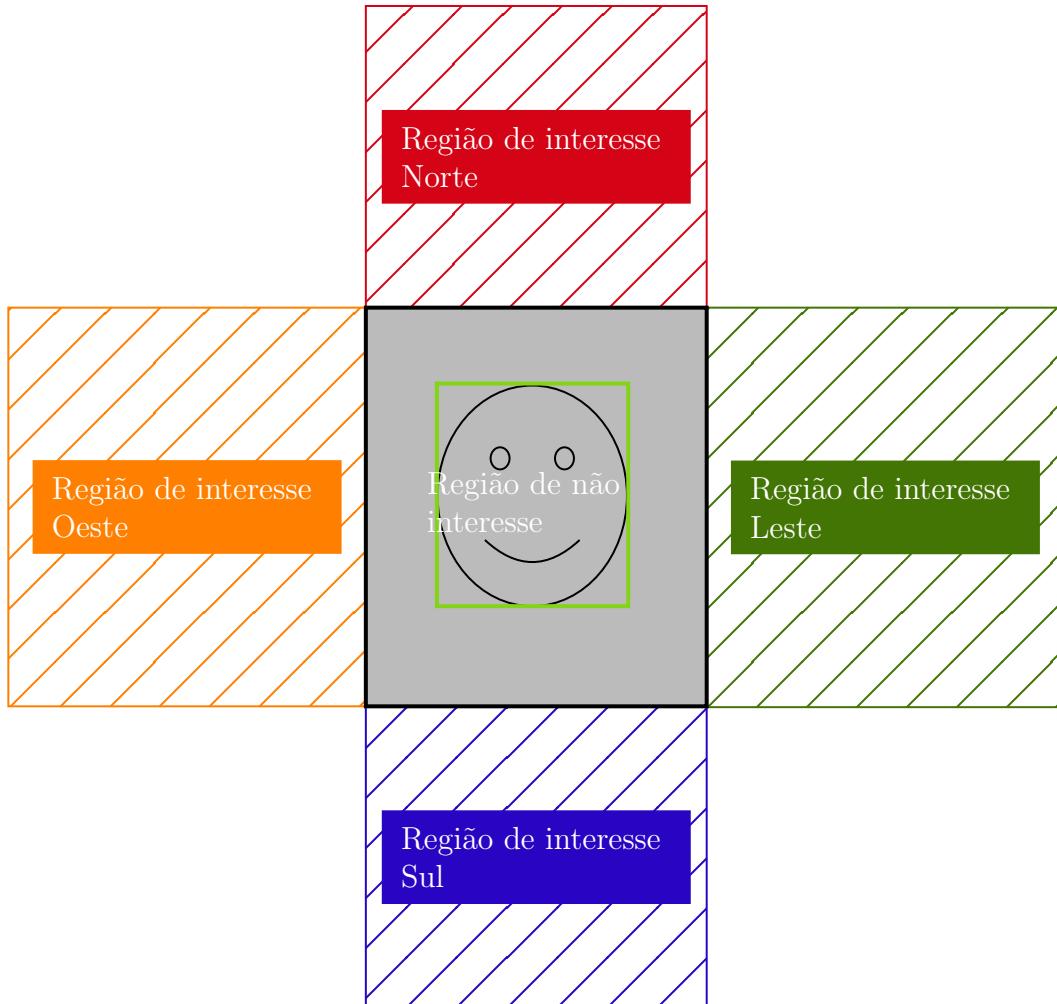


Fonte: a autor.

3.4.1 Transformação do Sistema de Coordenadas do OpenCV

Essa etapa foi fundamental para a implementação e funcionamento do protótipo, a ideia é simples foram delimitadas regiões de interesse na utilizando o OpenCV, a ilustração 36 mostra todas as cinco áreas. No centro da tela se situa a região de não interesse e como o sugere o nome dentro desse retângulo não existe reação do sistema de estreamento, porem caso a pessoa se move e saia de dentro o sistema detecta para qual direção esta se dirigindo. A simplicidade vem do fato de o objetivo ou a lógica principal ser sempre manter o alvo (retângulo verde criado pelo OpenCV em torno da face humana) sempre dentro da região de não interesse.

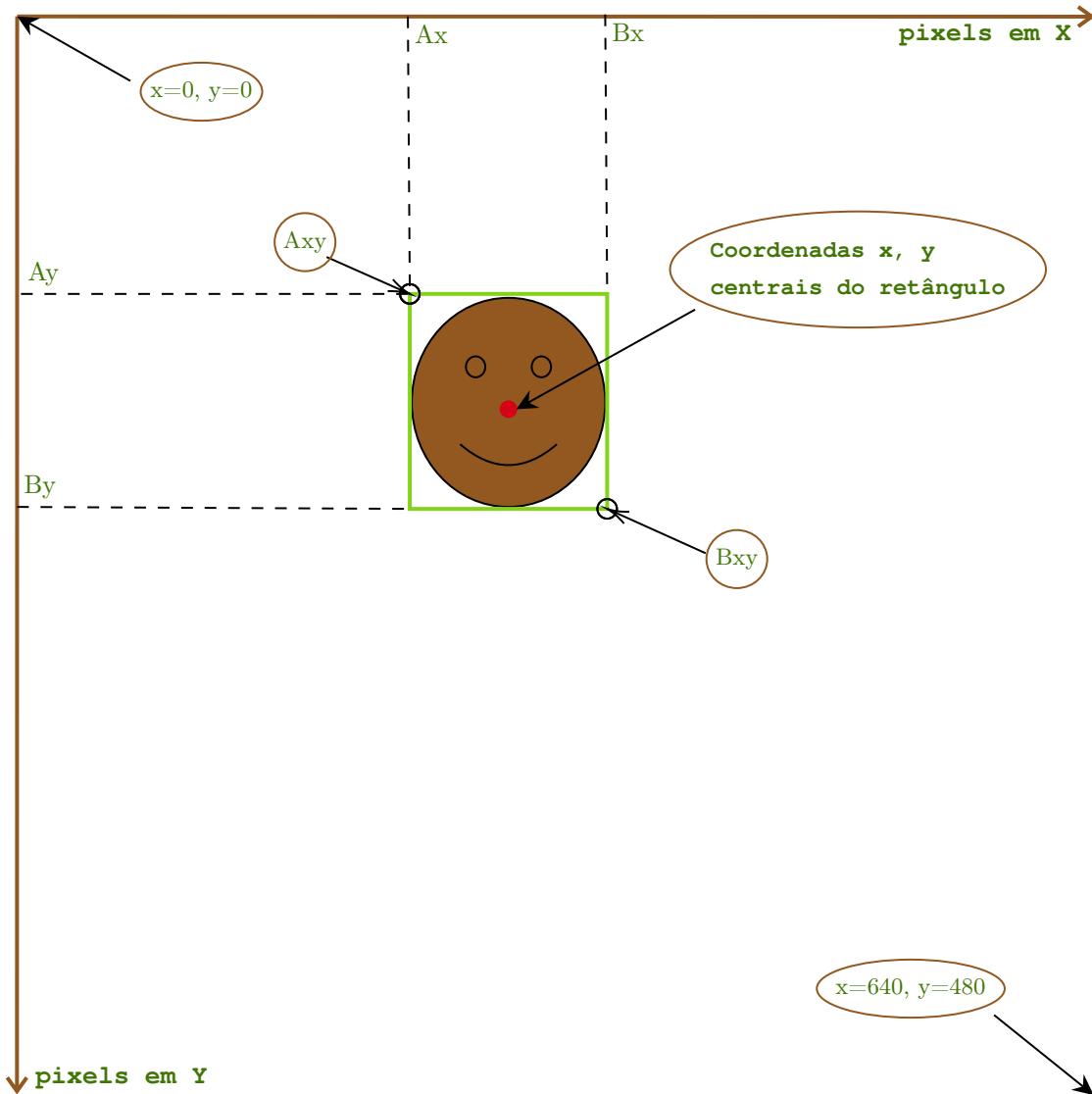
Figura 36 – Regiões de Interesse



Fonte: do autor

Porem a ideia das regiões de intersetar não trata todas as possibilidades, como uma movimentação na diagonal, e dessa forma foi necessário desenvolver outras etapas. Uma das primeiras etapas da implementação foi modificar o sistema de coordenadas de pixels no qual o OpenCV se baseia, para tornar conveniente sua utilização no sistema que detecta em qual direção o VANT tem que se movimentar. A ilustração 37 representa como o OpenCV trabalha com coordenadas de pixels, perceba que o ponto de origem das coordenadas verticais e Horizontais se situam na parte superior esquerda, logo não é possível identificar o sentido direcional em que o ponto na cor vermelha no centro da face esta se movendo. Para fins de orientação se estipulou que as linhas de pixels na horizontal pertencem a X e os na vertical a Y, e como é possível identificar com uma ferramenta do OpenCV o centro do retângulo desenhado em torno da face através de A_{xy} e B_{xy} , então se encontrou as coordenadas X e Y do ponto vermelho central.

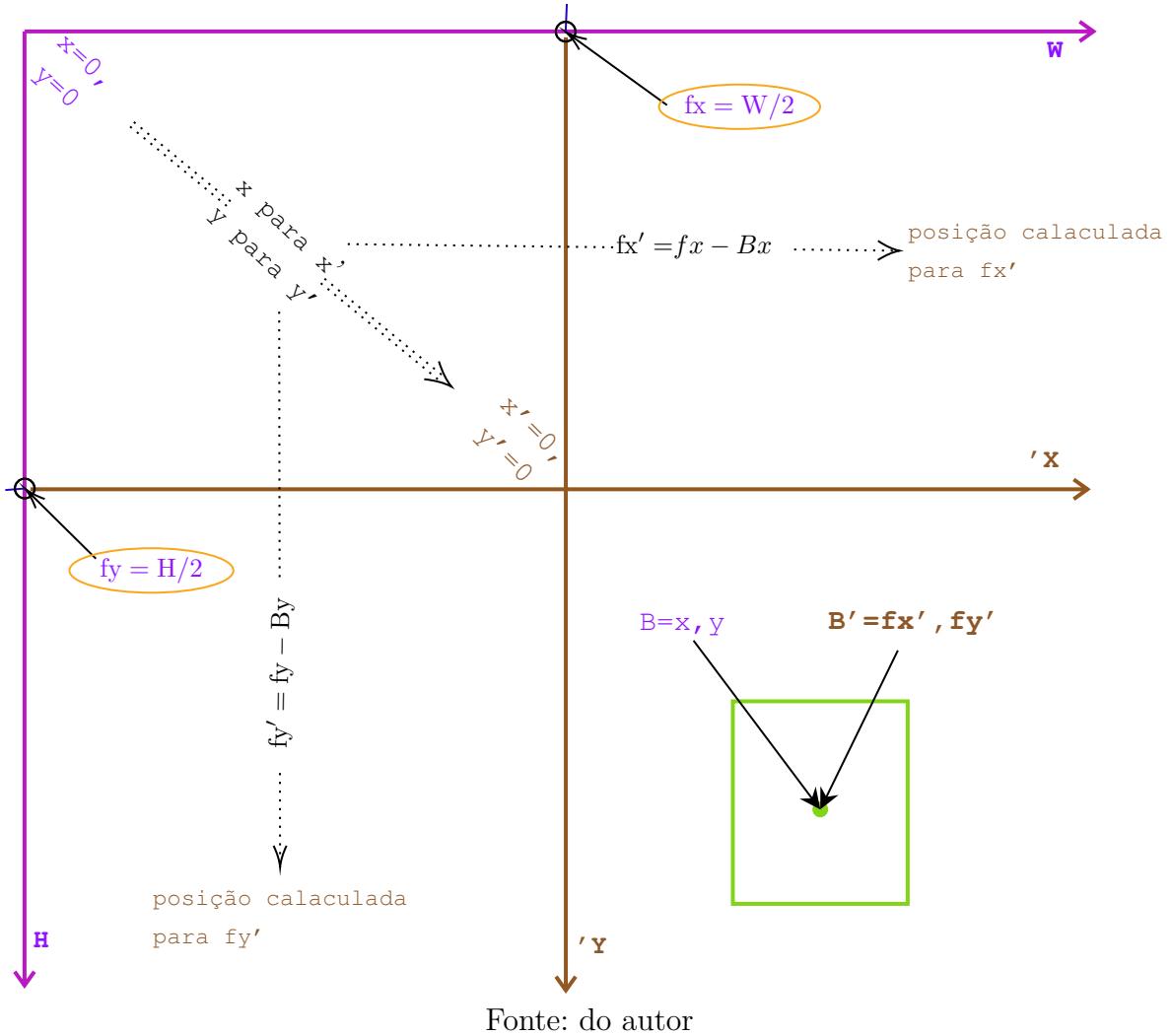
Figura 37 – Sistema de Coordenadas do OpenCV



Fonte: do autor

Para que o sistema funcione-se as coordenadas X=0 e Y=0 foram transportadas para um ponto central da tela, portanto foram realizados cálculos para determinar dois eixos um na horizontal e outro na vertical e sua intersecção ou ponto médio é o epicentro do novo sistema de coordenadas. Na ilustração 38 as duas retas na cor marrom representam o novo sistema, assim foi possível equacionar um novo ponto central para o retângulo que é desenhado em torno da face, esse novo ponto é dado como B' e ele recebe os valores calculados para fx' e fy'.

Figura 38 – Conversão do Sistema de Coordenadas OpenCV para Coordenadas Cartesianas



O objetivo de obter um sistema de coordenadas que pudesse ser utilizado para orientar o direcionamento do VANT foi alcançado e ele pode ser observado na ilustração 39, esse sistema de coordenadas é denominado de cartesiano e possui quatro quadrantes distintos e bem definidos, é o mesmo utilizado para localização geográfica na superfície terrestre. Da mesma forma que aeronaves conseguem se orientar na superfície terrestre usando coordenadas cartesianas, também é possível guiar um VANT para chegar ao objetivo do protótipo.

Com o novo sistema de coordenadas foi possível obter valores para X e Y com uma maior precisão (0.005m/s a 5m/s), assim como os quatro quadrantes conseguem indicar para qual direção a aeronave tem que se deslocar. Esses valores de velocidade posteriormente serão enviados para a controladora de voo do VANT.

A interpretação das informações da ilustração 38 pode ser feita através das seguintes sintaxes matemáticas;

- Sintaxe do quadrante I:

$$B' \in (\text{Quadrante I}) \mid \left(\frac{W}{2} - Bx > 0 \right) \wedge \left(\frac{H}{2} - By > 0 \right) \quad (3.1)$$

- Sintaxe do quadrante II:

$$B' \in (\text{Quadrante II}) \mid \left(\frac{W}{2} - Bx < 0 \right) \wedge \left(\frac{H}{2} - By > 0 \right) \quad (3.2)$$

- Sintaxe do quadrante III:

$$B' \in (\text{Quadrante III}) \mid \left(\frac{W}{2} - Bx < 0 \right) \wedge \left(\frac{H}{2} - By < 0 \right) \quad (3.3)$$

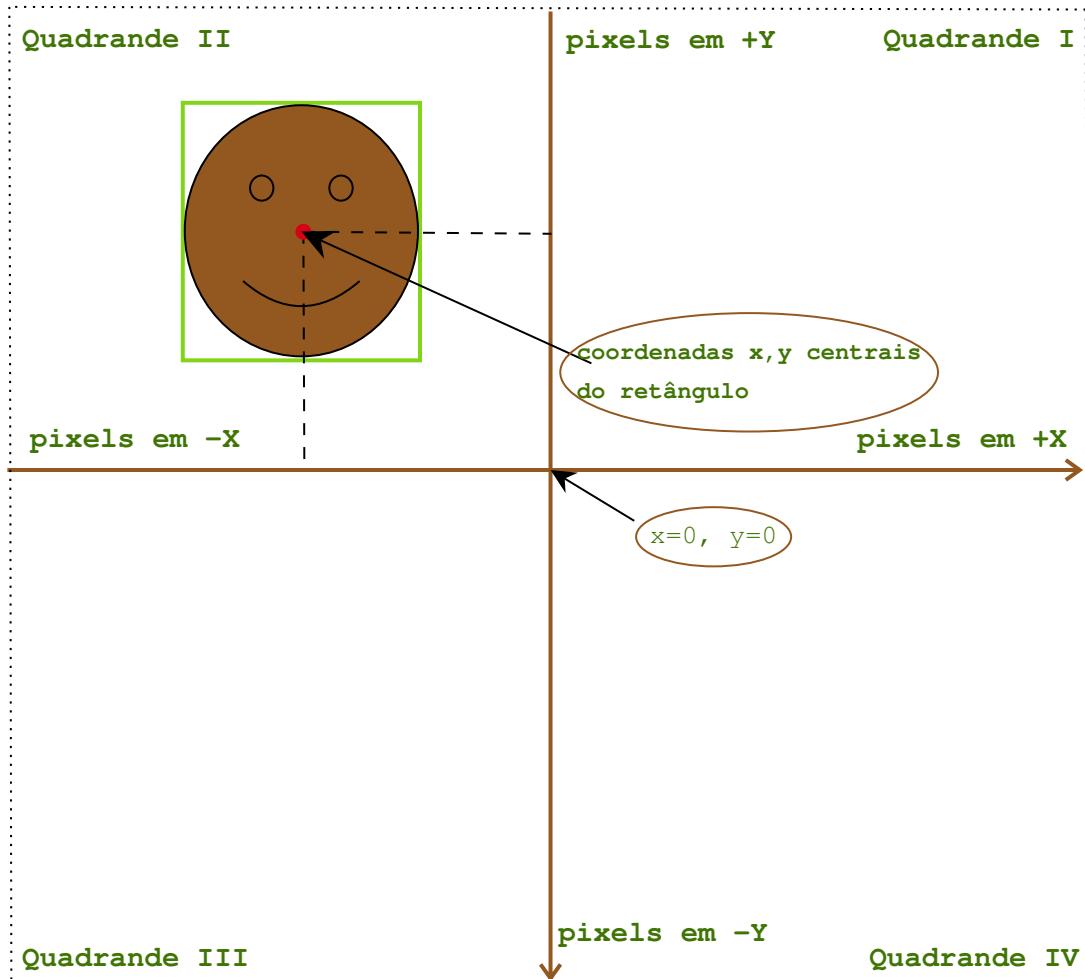
- Sintaxe do quadrante IV:

$$B' \in (\text{Quadrante IV}) \mid \left(\frac{W}{2} - Bx > 0 \right) \wedge \left(\frac{H}{2} - By < 0 \right) \quad (3.4)$$

- Sintaxe quando não pertence a nenhum quadrante:

$$B' \notin (\text{Quadrante I} \vee \text{II} \vee \text{III} \vee \text{IV}) \mid \left(\frac{W}{2} - Bx = 0 \right) \vee \left(\frac{H}{2} - By = 0 \right) \quad (3.5)$$

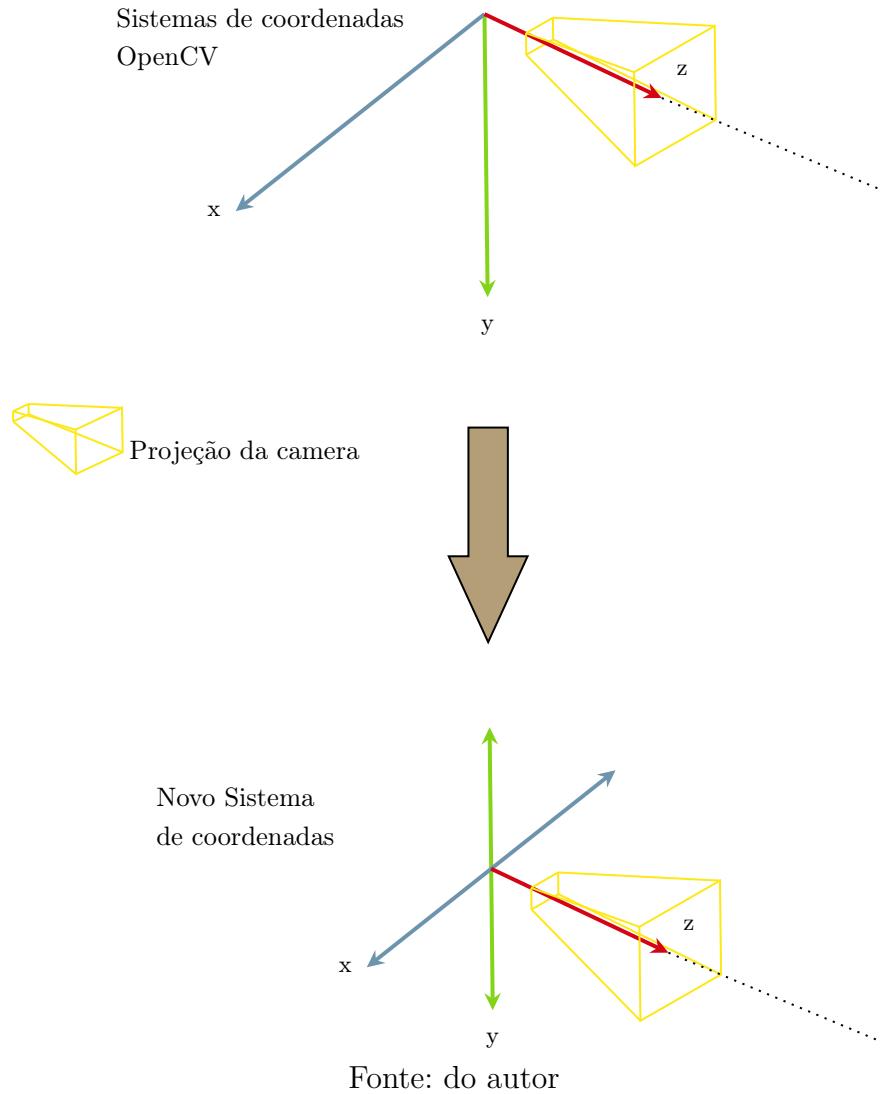
Figura 39 – Sistema de Coordenadas Cartesianas em Visão Computacional



Fonte: do autor

Para concluir essa parte a ilustração 40 representa a projeção em 3D do sistema de coordenadas original que o OpenCV utiliza e como fica apos a transformação. O OpenCV possui o eixo Z (profundidade), ele é utilizado para calcular a posição em um sistema 3D ou adquirir a velocidade de deslocamento, porem essa etapa não sera abordada mas pode ser um bom tema para trabalhos futuros ou aperfeiçoamento do sistema.

Figura 40 – Projeção do Sistema de Coordenadas OpenCV para Cartesianas



Fonte: do autor

Para demonstrar como o VANT interage e interpreta os dados que recebe do sistemas de visão computacional e atua para se orientar, foram criados dois modelos de gráfico do tipo vetor gradiente, aonde o tom de cor mais forte representa o valor máximo e consequentemente o tom mais claro o mínimo valor. Foi estipulado que o valor máximo seria de $\approx 5/m_s$, observando que segundo o site Ardupilot.org-docs¹ um VANT do tipo quadricóptero atinge de $10m/s \approx 13m/s$ no máximo, antes de se tornar incapaz de manter a altitude e a velocidade horizontal.

¹ <<https://ardupilot.org/copter/docs/auto-mode.html>>

O primeiro gráfico ilustrado pela figura 41 tem como objetivo interpretar como o sistema de visão computacional atua junto com o Dronekit. Se o gráfico 41 for sobreposto pela tela do OpenCV a parte aonde o gradiente é mais fraco fica no centro, ou seja, local aonde o software de visão computacional definiu como região de não interesse (não existe velocidade ou é insignificante), ja nas bordas a coloração é mais forte, isso significa que quando a pessoa que esta sendo monitorada pelo software estiver se aproximando das bordas da interface de captura da câmera, a velocidade aumenta gradualmente, tendo como objetivo sempre manter o alvo no centro da tela, logo conforme o alvo estiver se aproximando do centro a velocidade gradualmente vai diminuindo até parar assim que chegar ao centro e estiver na região de não interesse.

As escalas nos eixos X e Y contem os possíveis valores de velocidade em m/s que podem ser enviados para o VANT, como ja foi abordada no referencial teórico é utilizado o sistema de orientação por coordenadas NED e por regra nesse sistema de coordenadas o X fica no eixo das ordenadas e Y no eixo das abcissas.

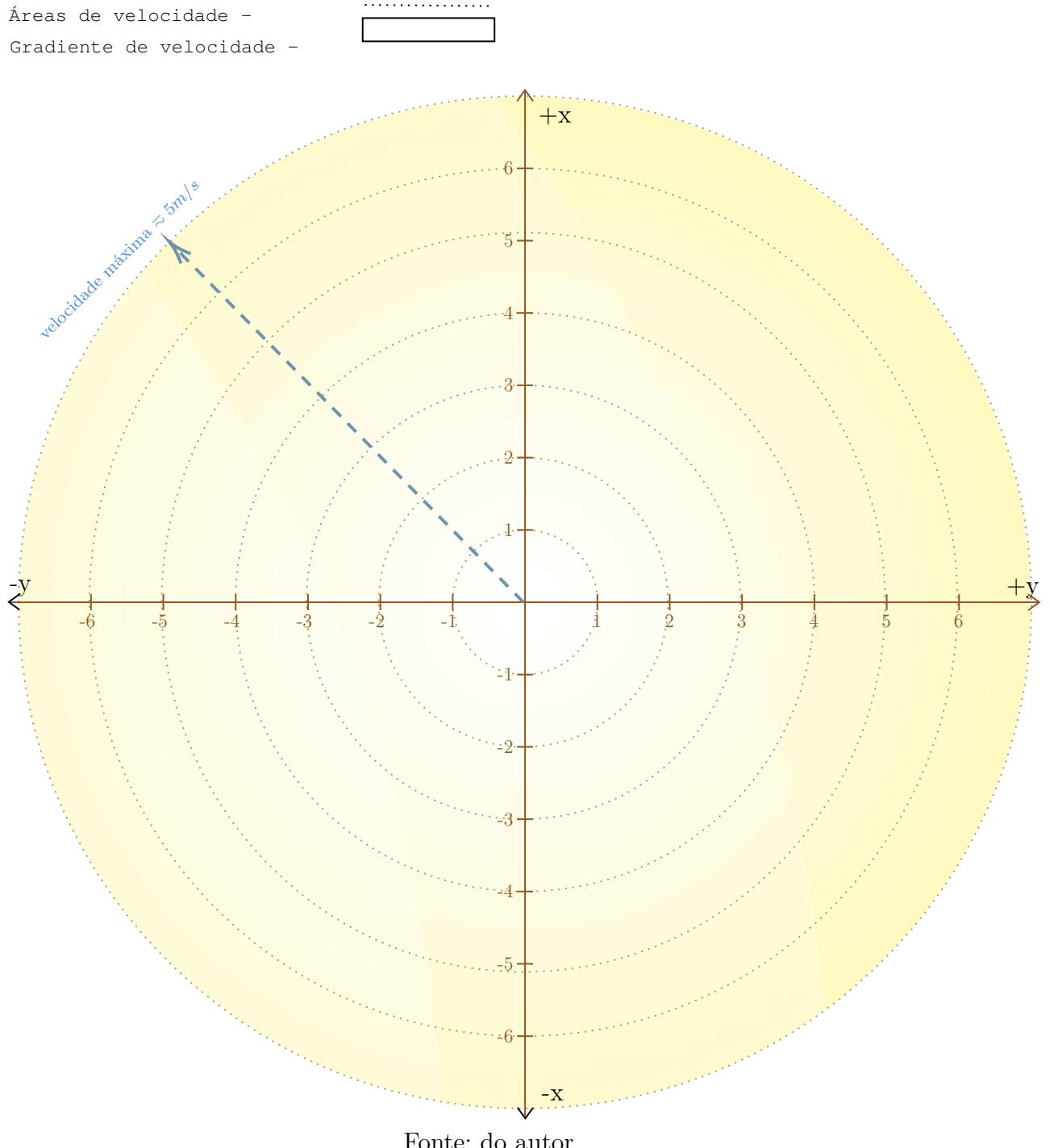
No gráfico 42 foram adicionadas duas cores para distinguir as velocidades em X e Y, os VANTS representam graficamente algumas posições dentro do sistema de coordenadas com seus respectivos valores de velocidade nos vetores X e Y assim como qual direção eles toram ao receber os valores de velocidade. A cor verde representa o gradiente de velocidade no eixo X e a cor vermelha no eixo Y e igualmente como no gráfico 41 nas bordas tem sua maior intensidade de velocidade e no centro não existe ou é insignificante as forças de velocidade. Nos eixos diagonais a X e Y pontos (A,B,C,D) as cores se sobrepõem, isso significa que os valores de velocidade serão iguais ou próximos para X e Y nesse ponto.

Analizando todos os pontos no gráfico 42 aonde estão os VANTS:

- VANT 1: Nesse ponto $x=-0.9\text{ms}$ e $y=1.9\text{m/s}$, e de acordo com a regra 3.2 pertence ao quadrante II, tem direção Noroeste tendendo a se aproximar mais do eixo X por ter uma maior velocidade na componente X.
- VANT 2: Nesse ponto $x=5\text{ms}$ e $y=0\text{m/s}$, e de acordo com a regra 3.5 não pertence a nenhum quadrante por ter uma das componentes de velocidade com valor nulo, porém podemos dizer que tem direção Norte de acordo com a componente x.
- VANT 3: Nesse ponto $x=-3.95\text{ms}$ e $y=-6.5\text{m/s}$, e de acordo com a regra 3.3 pertence ao quadrante III, tem direção Sudoeste e tendendo a se aproximar mais do eixo Y por ter uma maior velocidade na componente Y.
- VANT 4: Nesse ponto $x=1.5\text{ms}$ e $y=3.8\text{m/s}$, e de acordo com a regra 3.1 pertence ao quadrante I, tem direção Nordeste tendendo a se aproximar mais do eixo Y por ter uma maior velocidade na componente Y.

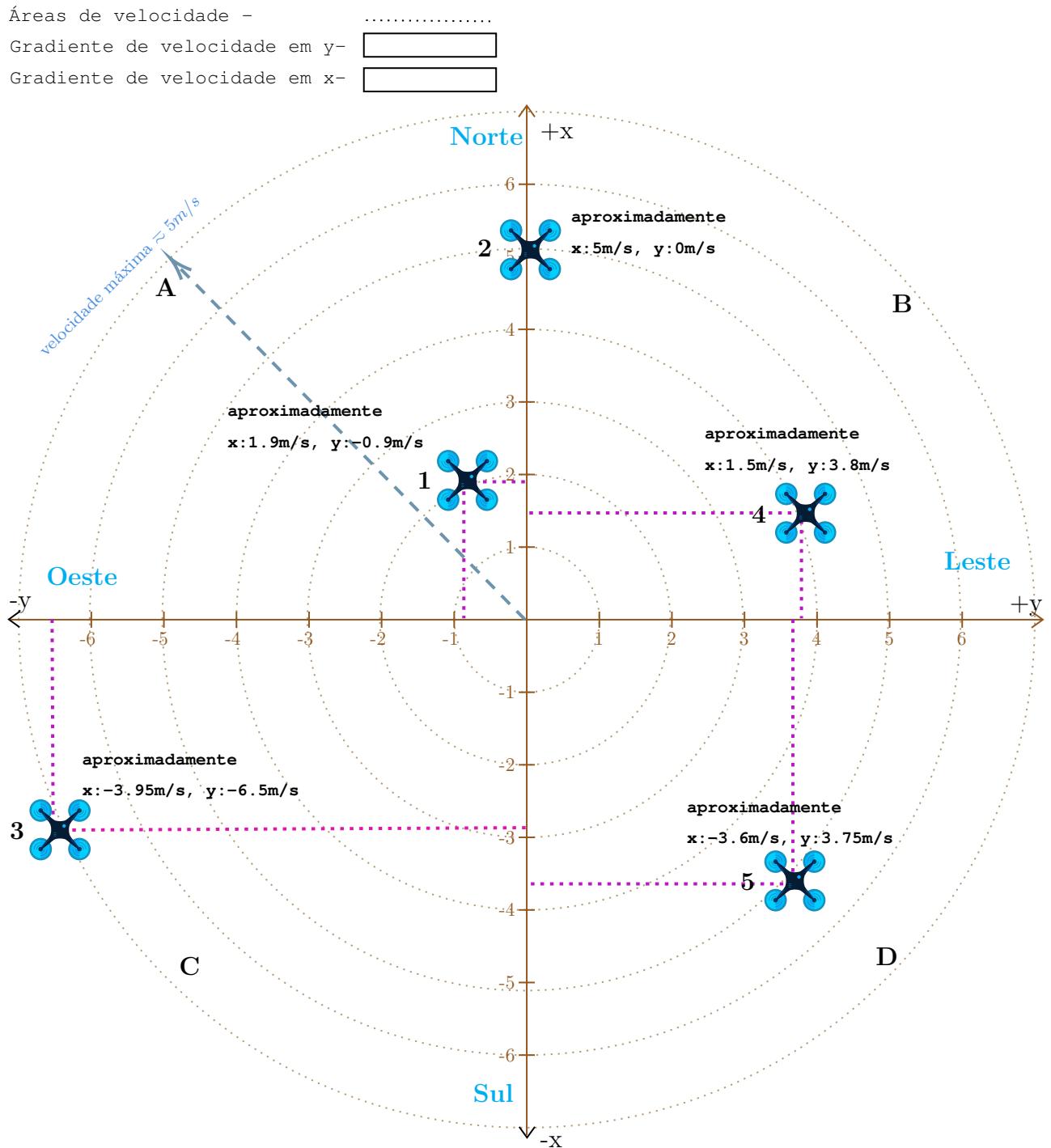
- VANT 5: Nesse ponto $x=-3.6\text{ms}$ e $y=3.75\text{m/s}$, e de acordo com a regra 3.4 pertence ao quadrante IV, tem direção Sudeste e nesse caso tem diferença de valores nas componentes X e Y insignificante não tendendo a se direcionar a nenhum dos eixos.

Figura 41 – Gráfico que Representa o Gradiente de Velocidade



Fonte: do autor

Figura 42 – Gráfico que Representa o Gradiente de Velocidade para X e Y

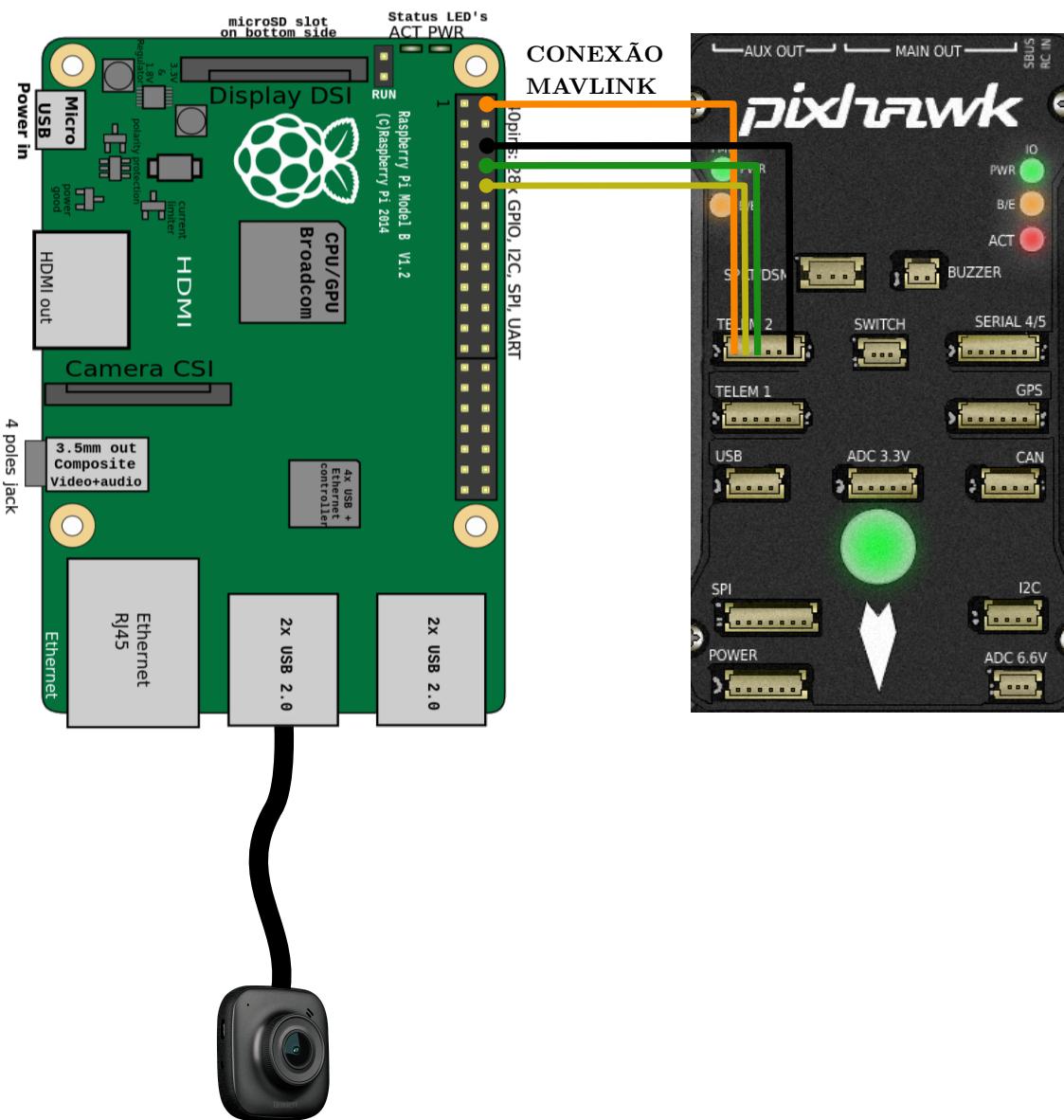


Fonte: do autor

3.5 Simulações com Software

3.6 Simulações com Hardware

Figura 43 – Conexão MAVLink da Raspberry Pi com Pixhawk



Fonte: Ardupilot Documents

4 Analise dos Resultados e Considerações Finais

4.1 Resultados

4.2 Considerações Finais

Referências

- ARDUPILOT. *ArduPilot Documentation*. [S.l.], 2019. Disponível em: <<https://ardupilot.org/ardupilot/>>. Acesso em: 2019. Citado 7 vezes nas páginas 24, 28, 29, 30, 31, 32 e 33.
- ARDUPILOT.ORG. *Communicating with Raspberry Pi via MAVLink*. 2020. Disponível em: <<https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html>>. Citado 2 vezes nas páginas 2 e 3.
- BRUM, H. K. M. *Brasil não soluciona nem 10% dos seus homicídios*. Paraná, 2018. Disponível em: <<https://www.gazetadopovo.com.br/ideias/brasil-nao-soluciona-nem-10-dos-seus-homicidios-d726kw8ykpwh6xm41zakgzoue/>>. Acesso em: 17 set. 2019. Citado na página 1.
- BURGGRÄF ALEJANDRO RICARDO PÉREZ MARTÍNEZ, H. R. J. W. P. uadrotors in factory applications: design and implementation of the quadrotor's p-pid cascade control system. *SN Applied Sciences*, Springer, v. 1, n. 722, 2019. Disponível em: <<https://doi.org/10.1007/s42452-019-0698-7>>. Citado 2 vezes nas páginas 38 e 41.
- CYPRIANO, R. M. M. I. M. A. *ESTUDO DAS FORÇAS GERADAS POR UMA HÉLICE*. [S.l.], 2014. Disponível em: <http://www.engenhariamecanica.ufes.br/sites/engenhariamecanica.ufes.br/files/field/anexo/2015.1_-_marcos_cypriano_roberto_makio.pdf>. Citado na página 36.
- DRONECODE. *mRo Pixhawk Flight Controller (Pixhawk 1)*. 2020. Disponível em: <https://docs.px4.io/v1.9.0/en/flight_controller/mro_pixhawk.html>. Citado 2 vezes nas páginas 3 e 26.
- GEITGEY, A. Machine learning is fun! part 4: Modern face recognition with deep learning. Medium.com, 2016. Disponível em: <<https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>>. Citado 3 vezes nas páginas 14, 15 e 16.
- HE, K. et al. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. Disponível em: <<http://arxiv.org/abs/1512.03385>>. Citado na página 15.
- JSBSIM. *JSBSim Reference Manual*. 2020. Disponível em: <<https://jsbsim-team.github.io/jsbsim-reference-manual/mypages/user-manual-frames-of-reference/>>. Citado na página 46.
- KING, D. High quality face recognition with deep metric learning. 2017. Disponível em: <<http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html>>. Citado na página 16.
- KISSAI, A.; SMITH, M. Electronic navigation system based on the use of alternate coordinate system and polar stereographic projection for uavs operating in polar regions. *International Journal of Aeronautics and Aerospace Engineering*, v. 1, p. 46–53, 05 2019. Citado na página 46.

- LONGARINI, A. *Redes Neurais Artificiais / As máquinas pensam?* [S.l.], 2019. Disponível em: <<https://www.lambda3.com.br/2019/09/redes-neurais-artificiais-as-maquinas-pensam/>>. Acesso em: 2019. Citado na página 11.
- MENG, X. et al. Contact force control of an aerial manipulator in pressing an emergency switch process. In: . [S.l.: s.n.], 2018. p. 2107–2113. Citado na página 39.
- MSARDONINI. *The Basic Physics of the Quad-Copter*. [S.l.], 2015. Disponível em: <<https://beaglequad.wordpress.com/2015/10/03/the-basic-physics-of-the-quad-copter/>>. Acesso em: 2020. Citado na página 42.
- MUNGUÍA, R. A gps-aided inertial navigation system in direct configuration. *Journal of Applied Research and Technology*, v. 12, p. 803–814, 08 2014. Citado na página 46.
- PX4.IO. *Px4 Documentation*. [S.l.], 2020. Disponível em: <<https://px4.io/>>. Acesso em: 2019. Citado na página 27.
- RAMPAZZO, L. *Metodologia científica*. [S.l.]: Edições Loyola, 2005. Citado na página 47.
- RASPBERRYPI.ORG. *Raspberry Pi 3 Model B*. 2020. Disponível em: <<https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>>. Citado na página 3.
- RASPBERRYPI.ORG. *Raspberrypi documentation*. 2020. Disponível em: <<https://www.raspberrypi.org/documentation/>>. Citado na página 51.
- ROSEBROCK, A. Face recognition with opencv, python, and deep learning. pyimagesearch.com, 2018. Disponível em: <<https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/>>. Citado na página 14.
- SANTOS, A. M. S. P. *Políticas sociais em xeque: impactos da crise sobre as finanças municipais*. [S.l.], 2018. 35-46 p. Disponível em: <<https://royaltiesdopetroleo.ucam-campos.br/wp-content/uploads/2018/12/boletim-royalties-N61-dezem-2018-artigo-4.pdf>>. Citado na página 5.
- SANTOS, L. A. *Artificial intelligence and machine learning*. GitBook, 2018. 65 p. Disponível em: <https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/content/linear_algebra.html>. Citado na página 8.
- SILVA, V. H. *Rio de Janeiro identificou 8 mil pessoas com reconhecimento facial no Carnaval*. 2019. Disponível em: <<https://tecnoblog.net/289696/rio-de-janeiro-identificou-8-mil-reconhecimento-facial/>>. Citado na página 5.
- SOUZA, D. A. de et al. Câmeras de segurança e seus sistemas tecnológicos: Percepções sobre os motivos da utilização. p. 1–13, 2016. Disponível em: <<https://www.aedb.br/seget/arquivos/artigos17/12425136.pdf>>. Citado na página 5.
- TEFAY, B. et al. Design of an integrated electronic speed controller for compact robotic vehicles. In: *Proc. of Australasian Conf. Robotics and Automation*. [S.l.: s.n.], 2011. p. 1–8. Citado na página 31.

- TIERNEY ORACLE GROUNDBREAKER AMBASSADOR, O. A. D. B. *Understanding, Building and Using Neural Network Machine Learning Models using Oracle 18c.* [S.l.], 2018. Disponível em: <<https://developer.oracle.com/databases/neural-network-machine-learning.html>>. Acesso em: 2019. Citado na página 10.
- TURING, A. M. Computer machinery and intelligence. *Parsing the Turing Test*, Springer, DORDRECHT, p. 23–65, 2009. Citado na página 6.
- ZMOGINSKI, F. *A sociedade mais vigiada do mundo: como a China usa o reconhecimento facial.* [S.l.], 2019. Disponível em: <<https://www.uol.com.br/tilt/noticias/redacao/2019/01/19/a-sociedade-mais-vigiada-do-mundo-como-a-china-usa-o-reconhecimento-facial.htm>>. Acesso em: 20 set. 2019. Citado na página 1.

Apêndices

APÊNDICE A – Primeiro Apêncice

Quisque facilisis auctor sapien. Pellentesque gravida hendrerit lectus. Mauris rutrum sodales sapien. Fusce hendrerit sem vel lorem. Integer pellentesque massa vel augue. Integer elit tortor, feugiat quis, sagittis et, ornare non, lacus. Vestibulum posuere pellentesque eros. Quisque venenatis ipsum dictum nulla. Aliquam quis quam non metus eleifend interdum. Nam eget sapien ac mauris malesuada adipiscing. Etiam eleifend neque sed quam. Nulla facilisi. Proin a ligula. Sed id dui eu nibh egestas tincidunt. Suspendisse arcu.

APÊNDICE B – Perceba que o texto do título desse segundo apêndice é bem grande

Maecenas dui. Aliquam volutpat auctor lorem. Cras placerat est vitae lectus. Curabitur massa lectus, rutrum euismod, dignissim ut, dapibus a, odio. Ut eros erat, vulputate ut, interdum non, porta eu, erat. Cras fermentum, felis in porta congue, velit leo facilisis odio, vitae consectetur lorem quam vitae orci. Sed ultrices, pede eu placerat auctor, ante ligula rutrum tellus, vel posuere nibh lacus nec nibh. Maecenas laoreet dolor at enim. Donec molestie dolor nec metus. Vestibulum libero. Sed quis erat. Sed tristique. Duis pede leo, fermentum quis, consectetur eget, vulputate sit amet, erat.

Donec vitae velit. Suspendisse porta fermentum mauris. Ut vel nunc non mauris pharetra varius. Duis consequat libero quis urna. Maecenas at ante. Vivamus varius, wisi sed egestas tristique, odio wisi luctus nulla, lobortis dictum dolor ligula in lacus. Vivamus aliquam, urna sed interdum porttitor, metus orci interdum odio, sit amet euismod lectus felis et leo. Praesent ac wisi. Nam suscipit vestibulum sem. Praesent eu ipsum vitae pede cursus venenatis. Duis sed odio. Vestibulum eleifend. Nulla ut massa. Proin rutrum mattis sapien. Curabitur dictum gravida ante.

Phasellus placerat vulputate quam. Maecenas at tellus. Pellentesque neque diam, dignissim ac, venenatis vitae, consequat ut, lacus. Nam nibh. Vestibulum fringilla arcu mollis arcu. Sed et turpis. Donec sem tellus, volutpat et, varius eu, commodo sed, lectus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque enim arcu, suscipit nec, tempus at, imperdiet vel, metus. Morbi volutpat purus at erat. Donec dignissim, sem id semper tempus, nibh massa eleifend turpis, sed pellentesque wisi purus sed libero. Nullam lobortis tortor vel risus. Pellentesque consequat nulla eu tellus. Donec velit. Aliquam fermentum, wisi ac rhoncus iaculis, tellus nunc malesuada orci, quis volutpat dui magna id mi. Nunc vel ante. Duis vitae lacus. Cras nec ipsum.