UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

JOÃO VICTOR MEDEIROS ROCHA

CONTROLE DE POSICIONAMENTO DE ROBÔ AÉREO AUTÔNOMO UTILIZANDO VISÃO COMPUTACIONAL

PATOS DE MINAS
2023

JOÃO VICTOR MEDEIROS ROCHA

CONTROLE DE POSICIONAMENTO DE ROBÔ AÉREO AUTÔNOMO UTILIZANDO VISÃO COMPUTACIONAL

Projeto Final de Curso apresentado à banca examinadora como requisito parcial de avaliação da disciplina de Projeto Final de Curso 2 da graduação em Engenharia Eletrônica e de Telecomunicações, da Faculdade de Engenharia Elétrica, da Universidade Federal de Uberlândia, Campus Patos de Minas.

Orientador: Prof. Me. Éder Alves de Moura

PATOS DE MINAS

2023

JOÃO VICTOR MEDEIROS ROCHA

CONTROLE DE POSICIONAMENTO DE ROBÔ AÉREO AUTÔNOMO UTILIZANDO VISÃO COMPUTACIONAL

Projeto Final de Curso apresentado à banca examinadora como requisito parcial de avaliação da disciplina de Projeto Final de Curso 2 da graduação em Engenharia Eletrônica e de Telecomunicações, da Faculdade de Engenharia Elétrica, da Universidade Federal de Uberlândia, Campus Patos de Minas.

Patos de minas, 30 de junho de 2023.

Banca Examinadora:

Prof. Me. Éder Alves de Moura
(Orientador)

Prof. Dr. Daniel Costa Ramos - UFU
Membro Avaliador

Prof ª.Dr ª.Gabriela Vieira Lima - UFU
Membro Avaliador

AGRADECIMENTOS

Agradeço primeiramente a Deus e aos meus pais, avós, irmã em especial aos meus pais, por acreditarem no meu potencial, além de serem os responsáveis para que fosse possível chegar até aqui. Sendo os responsáveis por fornecer todo apoio, zelo e carinho em cada etapa da minha vida e deste projeto. Agradeço ao suporte que me foi dado nos momentos em que mais precisei e graças a isto é pude me tornar o homem que sou hoje.

Agradeço ao Professor Éder Alves de Moura por ter aceitado ser meu orientador, além de se colocar à disposição para me atender e responder todas minhas dúvidas. Agradeço, também, à todo o corpo docente e aos técnicos da Universidade Federal de Uberlândia que me passaram todo o conhecimento, além de auxiliar em projetos trabalhos e dúvidas ao longo de toda caminhada, até aqui.

Por fim, agradeço aos meus amigos pelo companheirismo e por tornarem momentos difíceis, em risadas e momentos felizes, transformando essa caminhada em algo muito mais leve e fácil.

"A energia de um único pensamento pode determinar o movimento de um universo"
(TESLA, 1893)

Resumo

A utilização de robôs aéreos autônomos, popularmente conhecidos como drones, vêm crescendo e revolucionando a maneira como os seres humanos interagem com o espaço aéreo. Nos últimos anos, estes robôs estão ganhando em notoriedade e em possibilidades de uso, devido à melhoria de suas capacidades de atuação, como resultado da melhoria da microeletrônica, que têm conseguido realizar a miniaturização de componentes eletrônicos, sensores e câmeras. O crescente acesso aos drones, aliado à pesquisa acadêmica, tornou possível um cenário de aplicações comerciais, tais como mapeamento, monitoramento ambiental, entrega de pacotes, filmagens, dentre outras tantas. Para que estes drones naveguem em ambientes não estruturados, evitando a colisão com obstáculos e de maneira estável e autônoma, são necessárias diversas técnicas e tecnologias, tais como a Visão Computacional, os sistemas de controle realimentado e os sistemas embarcados, que serão abordadas neste trabalho. Deste modo, considerando este cenário, este trabalho propõe e desenvolve um sistema de rastreamento e acompanhamento autônomo de um drone em relação à um alvo, em que, a tarefa proposta consiste em processar as imagens da câmera embarcada do drone, identificar o rosto de uma pessoa a ser rastreado, e manter uma distância pré-determinada em relação à pessoa, que pode se mover livremente no ambiente, de forma autônoma. As imagens serão enviadas do drone (modelo DJI Tello) para um computador via comunicação WiFi e que serão processadas com a biblioteca de segmentação e classificação de imagens MediaPipe, determinando o alvo a ser rastreado e seguido. O sistema de controle automático de posição é implementado com um controlador Proporcional-Integrativo-Derivativo (PID). O trabalho apresenta o procedimento detalhado de configuração e montagem do projeto, além dos resultados práticos obtidos com a execução da tarefa proposta.

Palavras-chaves: rastreio visual; controle de posição; visão computacional; pid; drone, robótica aérea.

Abstract

The use of autonomous aerial robots, popularly known as drones, has been growing and revolutionizing the way humans interact with airspace. In recent years, these robots are gaining in notoriety and possibilities of use, due to the improvement of their performance capabilities, as a result of the improvement of microelectronics, which have managed to miniaturize electronic components, sensors and cameras. The growing access to drones, combined with academic research, has made possible a scenario of commercial applications, such as mapping, environmental monitoring, package delivery, filming, among many others. For these drones to navigate in unstructured environments, avoiding collision with obstacles and in a stable and autonomous way, several techniques and technologies are necessary, such as Computer Vision, feedback control systems and embedded systems, which will be addressed in this work. Considering this scenario, this work proposes and develops an autonomous tracking and following system for a drone in relation to a target, which consists of processing the images from the drone's on-board camera, identifying the face of a person to be tracked, and maintain a predetermined distance in relation to the person, who can move freely in the environment. Images are sent from the drone (DJI Tello model) to a computer via WiFi communication. These images are processed with the MediaPipe library (used for image segmentation and classification) which determines the target to be tracked and followed. The automatic position control system of the drone is implemented with a Proportional-Integrative-Derivative (PID) controller. The work presents the detailed configuration and assembly procedure of the project, in addition to the practical results obtained with the execution of the proposed task.

Key-words: visual tracking; position control; computer vision; pid; drone; aerial robotics.

Lista de Ilustrações

Figura 1 – Detecção: estruturas, algoritmo eigenfaces e detecção facial em tempo real. 15
Figura 2 – Detecção: estruturas, algoritmo eigenfaces e detecção facial em tempo de execução. 16
Figura 3 – “Patches” coletados de imagens áreas. 18
Figura 4 – Diagrama de blocos de um sistema de controle utilizando o controlador PID. 20
Figura 5 – Representação do drone utilizado para o projeto. 22
Figura 6 – Representação gráfica das hélices modelo 3044P e propulsores. 23
Figura 7 – Representação gráfica dos protetores de hélices removíveis do Tello. . . 24
Figura 8 – Especificações técnicas detalhadas do drone. 24
Figura 9 – Diagrama do drone Tello. 25
Figura 10 – Indicador de estado Aircraft. 26
Figura 11 – Indicadores de estado do drone. 27
Figura 12 – Sensores de fluxo óptico e Time of Flight. 27
Figura 13 – Sensor Time of Flight (TOF). 28
Figura 14 – Representação do sistema de posicionamento de visão. 29
Figura 15 – Ilustração da bateria do drone Tello. 30
Figura 16 – Ilustração do processo de carregamento da bateria do Tello. 30
Figura 17 – Tabela detalhada das características técnicas da bateria 31
Figura 18 – Especificações técnicas completas da câmera. 32
Figura 19 – Processador VPU da Intel Movidius Myriad 2 32
Figura 20 – Exemplo de funcionalidades da biblioteca DjiTelloPy 34
Figura 21 – Mecanismo de conexão drone - software 35
Figura 22 – Portas UDP, conexão drone - software 36
Figura 23 – Gerenciamento de versões utilizando o Conda 37
Figura 24 – Bloco de Conexão drone-software 38
Figura 25 – Bloco de detecção e rastreio 39
Figura 26 – Bloco de controle de posicionamento 40
Figura 27 – Simplificação da arquitetura do sistema de controle e posicionamento . 41
Figura 28 – Calibração do PID no eixo X 43
Figura 29 – Calibração do PID no eixo Y 43
Figura 30 – Calibração do PID no eixo Z 43
Figura 31 – Luzes piscando verde,vermelho,amarelo representando verificações internas. 44
Figura 32 – Conexão rede Tello - CF4886. 45

Figura 34 – Inicio do voo até atingir a altura do rosto. 45
Figura 33 – Script python que inicializa o sistema. 46
Figura 35 – Voo até uma altura média. 46
Figura 36 – Voo até a altura do rosto. 47
Figura 37 – Sistema sendo controlado pelo algoritmo de detecção. 47
Figura 38 – Sistema sendo controlado pelo algoritmo de detecção. 48
Figura 39 – Sistema sendo controlado pelo algoritmo de detecção. 49
Figura 40 – Algoritmo de detecção realizando o controle do drone. 50
Figura 41 – Erros de detecção do algoritmo. 50

Lista de abreviaturas e siglas

3D	3 dimensões
ABS	Acrilonitrila Butadieno Estireno
AP	*Access Point*
API	*Application Programming Interface*
ATTI	Modo atitude com quando sistema esta indisponível
CPU	*Central Processing Unit*
GPU	*Graphic Processing Unit*
IDE	*Integrated Development Environment*
PID	Proporcional integrador e derivativo
SARP	Sistemas de Aeronaves Remotamente Pilotadas
SDK	*Software Development Kit*
SLAM	*Simultaneous Localization and Mapping*
SoC	*Sistem on Chip*
ToF	*Time of Flight*
UAV	*Unmanned Aerial Vehicle*
VANT	Veículo Aéreo não Tripulado
VC	Visão computacional
VPU	*Visual Processing Unit*

Sumário

1 INTRODUÇÃO 11
1.1 Panorama geral dos drones 11
1.2 Justificativa 12
1.3 Objetivos 13
1.4 Organização do trabalho 13

2 REFERENCIAL TEÓRICO 14
2.1 Visão computacional 14
2.1.1 Detecção de objetos em imagens 15
2.2 Sistema de controle 18
2.2.1 O controlador PID 19

3 MATERIAIS E MÉTODOS 22
3.1 Descrição do hardware 22
3.1.1 Diagrama do drone 24
3.1.2 Modos de voo 25
3.1.3 Indicador de estado do drone 26
3.1.4 Sensor de fluxo óptico e sensor Time of Flight(TOF) 26
3.1.5 Bateria 30
3.1.6 Câmera 31
3.1.7 Processador 32
3.2 Descrição do software 33
3.2.1 Kit de desenvolvimento de software e biblioteca djitellopy 33
3.2.2 Mecanismo de comunicação drone - software 34
3.3 Preparação do ambiente para montagem do experimento 35
3.3.1 Arquitetura do sistema 38

4 RESULTADOS E DISCUSSÕES 42
4.1 Montagem do experimento e discussões 42
4.2 Resultados 44

5 CONCLUSÃO 52

REFERÊNCIAS 53

1 Introdução

Este trabalho apresenta o desenvolvimento de um sistema de controle de posicionamento de um robô aéreo autônomo utilizando Visão Computacional (VC). Neste capítulo introdutório, será apresentado o cenário geral para qual estes robôs estão sendo utilizados, discutindo sua importância e quais as justificativas e objetivos do uso destas pequenas máquinas.

1.1 Panorama geral dos drones

Os robôs aéreos denominados por Veículo Aéreo Não Tripulados (VANTs), ou Sistemas de Aeronaves Remotamente Pilotadas (SARP) ou, ainda, *Unmanned Aerial Vehicles* (UAVs), podem ser definidos como uma classe de pequenas máquinas voadoras altamente inteligentes (MICHELSON, 1998). Estes robôs aéreos são, também, popularmente conhecidos como drones e estão revolucionando a maneira como os humanos interagem com o espaço aéreo. Essas pequenas aeronaves, autônomas ou controladas remotamente, têm se tornado cada vez mais populares e estão desempenhando papéis importantes em diversos setores, desde o entretenimento até aplicações comerciais e militares.

A tecnologia da robótica aérea tem suas raízes nas primeiras décadas do século XX, quando experimentos com aeronaves controladas remotamente começaram a surgir. Também, foi somente nas últimas duas décadas que os robôs aéreos se tornaram acessíveis ao público em geral. O fácil acesso se deu graças aos avanços tecnológicos em diversos campos, em especial a microeletrônica que consegue realizar a miniaturização de componentes eletrônicos, tais como sensores, câmeras, processadores e baterias com maior densidade energética.

Com a popularização dos robôs aéreos, as aplicações vêm se multiplicando em diversos campos. Inicialmente, a aplicação se dava principalmente em atividades com fins militares, como vigilância e reconhecimento. Contudo, rapidamente ocorreu expansão para outras áreas, como mapeamento e monitoramento ambiental, entrega de pacotes, filmagens cinematográficas, inspeções de infraestruturas, agricultura de precisão e até mesmo para uso recreativo como uma substituição das práticas de aeromodelismo. Com o grande avanço e popularidade no mercado, aliada a facilidade de acesso a esta tecnologia, as mais diversas formas para estes robôs vêm sendo desenvolvidas. Albiero e Biasi (2016), classificam estas máquina divididas em dois tipos: os de uso militar, que são, geralmente, em formato de aviões e podem ser classificados como robôs aéreos de asa fixa; e robôs aéreos de asa rotativa, que são multirrotores, ou seja, possuem mais de duas hélices para levantar voo,onde os quadricópteros é o modelo mais famoso e popular que recebeu o nome

de drone que corrobora para ser a classe de robôs aéreos mais utilizados.

Sendo assim, outro motivo que se pode atribuir a esta grande ascensão se dá ao fato de estas pequenas e eficientes máquinas serem ótimas em atividades em que se é exigido a estabilidade de voo. Além disso, a possibilidade de pairar sob o ar se fez uma característica importante para a popularização dos drones, pois é possível controlar com precisão a velocidade de movimento, realizar curvas fechadas e até mesmo realizar manobras.

A crescente importância dos drones é evidente, ao se analisar o impacto positivo que têm gerado em diversas áreas. A sua versatilidade, combinada com a capacidade de operar em ambientes desafiadores e de difícil acesso, tem proporcionado avanços significativos em pesquisa científica, monitoramento ambiental, mapeamento geográfico, gerenciamento de desastres naturais, agricultura de precisão e muitos outros campos. Além disso, os drones têm demonstrado um potencial imenso para agilizar processos, reduzir custos e aumentar a eficiência em várias indústrias, como a entrega de pacotes, inspeções de infraestruturas e vigilância de segurança. Com a contínua evolução tecnológica e o aprimoramento das regulamentações, é esperado que os drones desempenhem um papel cada vez mais relevante em nossas vidas, impulsionando a inovação e contribuindo para soluções mais eficazes e sustentáveis.

1.2 Justificativa

A pesquisa sobre robôs aéreos autônomos desempenha um papel fundamental na busca por soluções tecnológicas inovadoras e no desenvolvimento de aplicações práticas que podem ter um impacto significativo em diversas áreas da sociedade. Uma das razões pelas quais a investigação nesse campo é justificada: como por exemplo, a pesquisa sobre robôs aéreos autônomos impulsionam o avanço tecnológico, levando ao desenvolvimento de aeronaves cada vez mais inteligentes e capazes de realizar tarefas complexas de forma autônoma. Sendo estas inovações tecnológicas capazes de melhorar muitas áreas e setores comerciais e civis.

Além disso, os robôs aéreos autônomos podem oferecer soluções para desafios atuais enfrentados por setores como logística, transporte, agricultura, inspeções industriais e monitoramento ambiental. A pesquisa nessa área busca desenvolver sistemas autônomos mais eficientes, econômicos e confiáveis, que possam lidar com problemas complexos e aumentar a eficiência das operações. À medida que essas tecnologias avançam, é necessário estabelecer diretrizes para o uso ético, a privacidade, a segurança e a integração segura dessas aeronaves em espaços aéreos compartilhados.Contribuindo para o entendimento dos desafios e oportunidades associados aos drones autônomos, auxiliando na criação de um ambiente regulatório adequado.

Logo, este trabalho tem como motivação colaborar para o desenvolvimento e motivar

as pesquisas em robôs aéreos autônomos que irá corroborar para automatizar e melhorar processos em diversas áreas como por exemplo a agricultura de precisão.

1.3 Objetivos

Obtendo espaço de forma promissora, os robôs aéreos estão sendo aplicados em larga escala nas mais diversas áreas. Tornando-se, uma das linhas de pesquisa mais promissora dentro do campo de robótica móvel. Com o objetivo de fazer com que estas pequenas máquinas voadoras naveguem é necessário a utilização de técnicas como a VC. Sendo assim, o crescimento e desenvolvimento destas áreas potencializa a realização de tarefas antes inimagináveis, como receber a entrega de uma encomenda de um produto comprado na internet em pouquíssimos segundos ou até mesmo pulverizar dezenas de hectares de maneira inteligente e autônoma. Além destas aplicações, podemos pensar em diversas outras como por exemplo segurança e monitoramento de fronteiras. Logo, o uso de ambas as tecnologias operando de forma sinérgica automatiza e aceleram tarefas, diminuem riscos, e melhoram a qualidade vida da humanidade de diferentes formas.

Portanto, o objetivo geral deste trabalho será entregar um sistema controle de posicionamento de um robô aéreo utilizando VC embarcado, para ser base para construção de robôs aéreos para as mais diversas aplicações como monitoramento de perímetros, aplicações militares, aplicações na área agrícola e até mesmo entrega de alimentos e suprimentos;

1.4 Organização do trabalho

O restante deste trabalho será organizado da seguinte maneira:

- O Capítulo 2 irá abordar todo o referencial teórico, abordando os conceitos de Visão computacional e rastreio, a formação da imagem, o rastreio e conceitos de bouding box e a maneira como são capturadas e por fim o sistema de controle e controlador PID;
- O Capítulo 3 será um descritivo da metodologia de desenvolvimento do trabalho, abordando toda a descrição do hardware e do software utilizado bem como a apresentação da montagem do experimento;
- Já o Capítulo 4 será uma sessão destinada aos resultados e discussões onde serão apresentados os resultados obtidos os gráficos e imagens que irão ilustrar os testes feitos;
- Por fim, no Capítulo 5 estará sintetizada as conclusões do trabalho e uma proposta de trabalhos futuros.

2 Referencial Teórico

Nesta seção será feito todo o referencial teórico que fornecerá o embasamento necessário para auxiliar no desenvolvimento do trabalho, onde serão apresentados os conceitos de visão computacional, incluindo a formação da imagem definição do problema de detecção em imagens e uma abordagem acerca dos sistemas de controle.

2.1 Visão computacional

A VC é uma área multidisciplinar que combina técnicas de processamento de imagem, aprendizado de máquina e inteligência artificial que em conjunto são responsáveis por fornecer, através de uma câmera, e sensores a capacidade de um computador, robô ou qualquer dispositivo que possua um processador a capacidade de analisar e compreender imagens e vídeos (KARN et al., 2021). Isto torna possível, que um sistema desempenhe funções e realize atividades com base no que está sendo processado pela câmera e processador. A VC, pode também ser definida como uma transformação de dados ou informações provenientes de uma foto ou vídeo, para que seja feito uma decisão ou representação computacional, isto é, a execução de um algoritmo, para se atingir um objetivo ou realizar alguma tarefa (BRADSKI; KAEHLER, 2008).

Para ser possível desenvolver um sistema de VC seja ele para aplicar em robôs aéreos ou não, deve ser levado em consideração algumas funções que foram indicadas por (REHEM; TRINDADE, 2009) que são comuns a estes sistemas. A aquisição de uma imagem digital, pode ser produzida por um ou vários sensores, cujo resultado pode variar entre uma imagem bidimensional, uma cena tridimensional ou ainda uma sequência de imagens, dependendo do tipo do sensor.

Nessas imagens, os valores dos pixels geralmente indicam a intensidade da luz em uma ou várias faixas de cor. O pré-processamento da imagem, realizado antes da aplicação de um método de visão computacional em uma imagem é importante para ser possível extrair informação e assegurar que ela satisfaça às condições do método.

Após ser realizado um pré-processamento, há um processamento e coleta as características matemáticas da imagem em vários níveis de complexidade, tais como detecção de bordas, de cantos, de textura, de formato e de movimento. Já a detecção e segmentação referem-se à detecção da relevância de regiões da imagem para processamento posterior. Finaliza-se com o processamento de alto nível que inclui a verificação da satisfação dos dados, a estimativa de parâmetros sobre a imagem e a classificação dos objetos detectados em diferentes categorias.

Jähne e Haussecker (2000) sintetiza afirmando que o objetivo de sistemas de VC é converter as imagens captadas em sinais digitais para que possamos obter informações sobre onde ele está (geometria, posicionamento a partir de eixos cartesianos), e o que é (ser/objeto), ou quais suas propriedades.

De forma mais simplificada, um sistema de visão computacional requer uma entrada de dados obtida por meio sensores, câmeras, ou vídeos. Após a aquisição dos dados, segue-se o fluxo de processamento, no qual os dados originais são transformados em uma informação esperada. (REHEM; TRINDADE, 2009) exemplificam o sistema de VC com uma *pipeline* simples como: receber uma imagem colorida (dado), obter a representação binária da imagem (processamento), converter uma imagem preta e branca em níveis de cinza. A transformação da imagem ocorre a partir de um processo realizado por métodos contidos em bibliotecas de processamento gráfico como OpenCV e OpenGL.

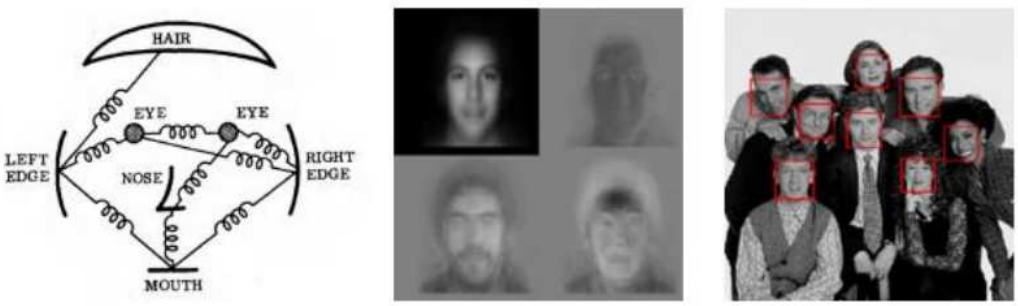
2.1.1 Detecção de objetos em imagens

A detecção de objetos é um problema fundamental na área visão computacional, sendo crucial em diversas aplicações, incluindo sistemas de vigilância, controle de tráfego e veículos aéreos não tripulados(SMEULDERS et al., 2013).

De maneira simplista, a dificuldade do rastreio, pode ser definido como um problema de verificar a posição e estimar a trajetória de um objeto no plano da imagem conforme ele se move ao longo de uma cena ou frame (YILMAZ; JAVED; SHAH, 2006).

O problema de detecção pode ser dividido em vários tipos. Por exemplo, quando se sabe o que esta procurando, o problema é puramente de detecção de objetos envolvendo a varredura rápida de uma imagem para determinar onde uma correspondência pode ocorrer como pode ser visto na Figura 1.

Figura 1 – Detecção: estruturas, algoritmo eigenfaces e detecção facial em tempo real.



Fonte: Turk e Petland (1991).

Entretanto se no problema existe um objeto rígido específico para ser detectado este trabalho é nomeado de reconhecimento de instância. Cuja as técnicas utilizam o algoritmo

para procurar por pontos característicos e verificar se os pontos encontrados se alinham em um formato geometricamente plausível.

Por fim, a versão mais desafiadora das tarefas de detecção ou reconhecimento de "classes", este método consiste em detectar e reconhecer instâncias de classes extremamente variadas, como animais, pessoas e móveis. Esta técnica envolve algoritmos de segmentação da imagens em regiões significativas.

Em muitos casos, a tarefa de detecção depende do contexto que se deseja detectar ou que esteja presente ao redor e também dos elementos de cena.

Portanto, podemos concluir que a detecção de objetos se dá através de uma imagem que é fornecida para analisar, como por exemplo uma fotografia de um grupo da faculdade ou de uma família por exemplo como mostra Figura 2.

Figura 2 – Detecção: estruturas, algoritmo eigenfaces e detecção facial em tempo de execução.

Fonte:Rowley, Baluja e Kanade (1998)

De início pode-se pensar na tentativa de aplicar um algoritmo de reconhecimento a todas as "sub janelas" possíveis nesta imagem. Entretanto, estes algoritmos tendem a ser lentos e propensos a erros. Uma alternativa mais eficaz se faz através da construção de detectores especializados, que trabalham para encontrar rapidamente regiões prováveis de ocorrência de objetos. Logo, para uma detecção mais eficaz neste projeto será utilizado um detector que realiza a detecção do rosto.

De uma maneira geral, é possível utilizar diversos tipos de detectores como por exemplo o detector de pedestres, que é um detector com uma gama mais geral de métodos para detecção de objetos pois a cena ou uma imagem urbana podem possuir "n" objetos simultâneos em um único quadro de vídeo ou imagem. Estes detectores mencionados não

se restringe apenas a aplicações de segurança automotiva, por exemplo ou detecção de pedestres, mas sim para diversos outros casos e situações(LEIBE et al., 2007).

Sendo assim, um dos principais algoritmos existente é o de detecção facial. pois de maneira geral uma face ou rosto em uma imagem pode ser considerada como um objeto de interesse a ser detectado. Antes que o reconhecimento facial possa ser aplicado a uma imagem geral, os limites ou locais e tamanhos de qualquer rosto devem primeiramente serem mapeados ou encontrados.

A princípio, segundo Moghaddam e Pentland (1997), é possível aplicar um algoritmo de reconhecimento facial em cada pixel e escala mas tal processo é muito lento e com um custo computacional elevado se colocado em prática. Ao longo dos anos, uma grande variedade de algoritmos de detecção rápida de face foram desenvolvidas (YANG; KRIEGMAN; AHUJA, 2002).

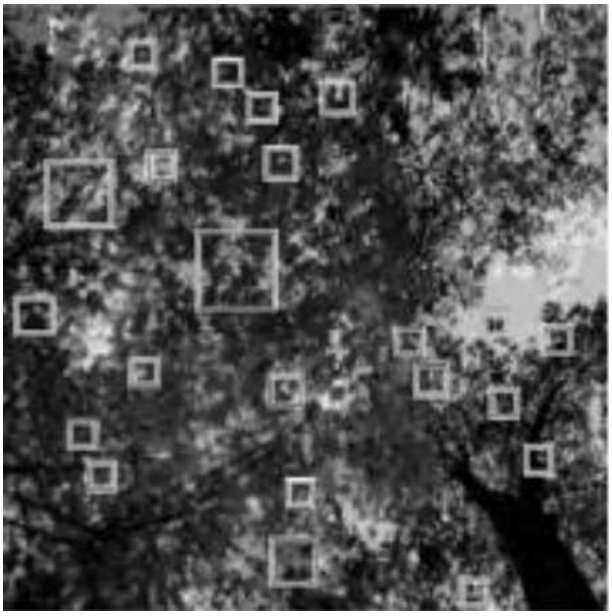
De acordo com a nomenclatura adotada por Yang, Kriegman e Ahuja (2002), as técnicas de detecção facial podem ser classificadas em três tipos: as baseadas em recursos, baseadas em modelos ou baseadas em aparência. As técnicas baseadas em recursos tentam encontrar os locais de características distintas da imagem, como os olhos, nariz e boca, e então verificar se as características estão em um arranjo geométrico plausível. Essas técnicas incluem algumas das primeiras abordagens de reconhecimento facial (FISCHLER; ELSCHLAGER, 1973), bem como abordagens mais recentes baseadas em auto espaços modulares (MOGHADDAM; PENTLAND, 1997), jatos de filtro local (LEUNG; BURL; PERONA, 1995) máquinas de vetor de suporte (HEISELE; SERRE; POGGIO, 2007), e boosting (SCHNEIDERMAN; KANADE, 2004).

A abordagem baseada em aparência escaneia pequenas manchas retangulares sobrepostas a imagem em busca de prováveis candidatos a representar aquele rosto, que posteriormente podem ser refinados usando uma cascata de mais algoritmos de detecção mais seletivos (SUNG; POGGIO, 1998).

Sendo assim, ambos os sistemas citados coletam um conjunto de patches de rostos rotulados bem como um conjunto de tirados de imagens que não contem rostos, como imagens aéreas ou vegetação. As imagens de rosto coletadas são aumentadas por espelhamento artificial, girando, dimensionando e traduzindo as imagens em pequenas quantidades para tornar os detectores de face menos sensíveis a tais efeitos como mostrado na Figura 3

Após a coleta de um conjunto inicial de imagens de treinamento, alguns pré processamentos opcionais podem ser realizado, como subtrair um gradiente médio (função linear) da imagem para compensar os efeitos de sombreamento global e usar a equalização do histograma para compensar a variação do contraste da câmera para que assim se realize a detecção.

Figura 3 – “Patches” coletados de imagens áreas.



Fonte: Schneiderman e Kanade (2004)

2.2 Sistema de controle

Os sistemas de controle são fundamentais para o funcionamento eficiente e seguro de uma ampla gama de processos e dispositivos.

Estes sistemas têm como objetivo monitorar e regular o comportamento de um sistema dinâmico, garantindo que ele opere de acordo com um conjunto de requisitos e metas pré-definidos. Estando presentes em diversos domínios, eles são utilizados desde processos industriais complexos até sistemas de navegação espacial, desempenhando um papel essencial na melhoria da eficiência, na redução de custos e no aumento da segurança e confiabilidade de operações.

Tais sistemas podem ser compostos de diversos elementos, entretanto comumente em sua estrutura mais básica possuem um sensor, um controlador e atuador. Os sensores realizam a medida do estado atual do sistema e fornece as informações coletadas ao controlador, que baseado em um conjunto de regras de controle, envia um sinal até os atuadores, que irão realizar ações necessária para ajustar o comportamento do sistema.

Entretanto, existem diversos tripos de sistema de controle sendo os mais conhecidos pela bibliografia os sistemas de malha aberta e malha fechada. Os sistemas de malha aberta não possuem realimentação, ou seja, não levam em consideração a saída real do sistema. Já os sistemas de controle em malha fechada utilizam a realimentação para comprar a saída com a entrada e realizar o controle.

Portanto, no sistemas de controle existem diversos tipos de controladores sendo possível desenvolver um controladores específicos para processos máquinas especificas,

contudo os sistemas de controle clássico como o controlador PID (proporcional, integrador e derivativo) é um dos mais utilizados e será utilizado para o projeto e será apresentado com maior detalhamento na Seção 2.2.1.

2.2.1 O controlador PID

O controlador PID (Proporcional Integrador Derivativo) é um dos muitos tipos de controladores existentes dentro do campo de sistemas de controle, possuindo, uma longa história no domínio da controle e automação. O motor vapor desenvolvido por James Watt em 1769 foi aceito como o marco inicial para os sistemas de controle. Em 1868, J. C. Maxwell desenvolveu um modelo matemático para realizar o controle da máquina a vapor.

Posteriormente após o desenvolvimento do modelo Maxwell os classificou em em duas categorias: máquinas moderadoras e máquinas com controle genuínos. De acordo com terminologias modernas, as máquinas ditas moderadoras onde se tem apenas com apenas ação de controle proporcional, enquanto as máquinas controle genuíno possuem o controle proporcional e integral.

Após vários estudos Elmer Sperry, em 1911, desenvolveu o primeiro controlador PID para a Marinha dos Estados Unidos (BENNETT, 2000). Em 1939, no entanto, a Taylor Instrument Companies lançou uma versão totalmente revisada e adotaram o nome de controlador pneumático *Fulscope* (BENNETT, 1993).

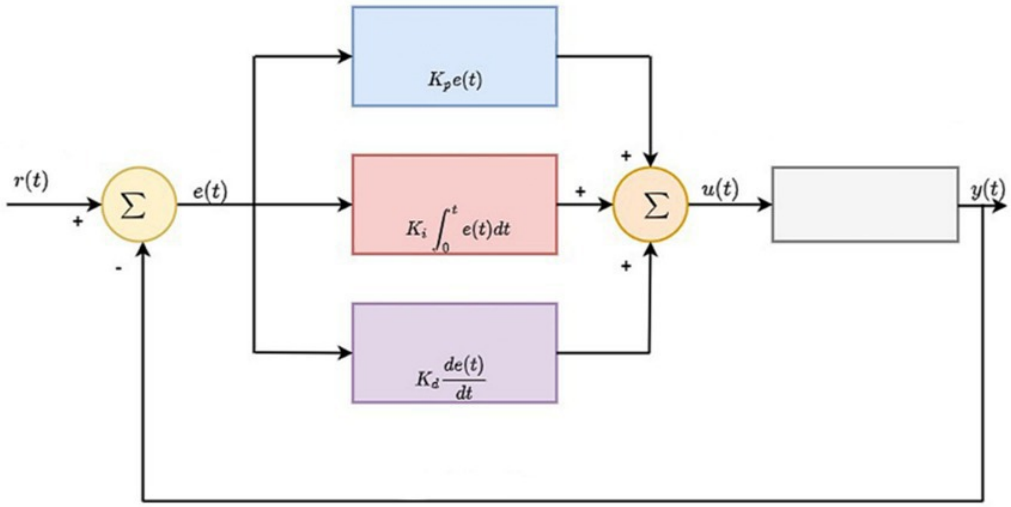
Este novo dispositivo era capaz de realizar uma ação denominada por “proatividade”. Da mesma forma, a empresa Foxboro seu controlador *Hyper-reset* que tinha controle proporcional, onde naquele mesmo ano surgiu o controlador “Stabilog” que tinha como objetivo fornecer um controle compatível com a derivada do sinal de erro. Esta implacável onda de surgimento de novos modelos de controladores foi o que fomentou o ecossistema para a melhoria dos controladores.

Após alguns anos, o problema do erro de estado estacionário no controlador proporcional foi minimizado ajustando o *setpoint* para algum valor arbitrário até que o erro chegasse a zero. Este ajuste “integrou” o erro e ficou conhecido como controlador integral proporcional.

O primeiro controlador pneumático com ação derivativa foi desenvolvida pelas empresas de instrumentos Taylor em 1940, o que minimizou os problemas de sobre sinal(*overshooting*).

Contudo, os engenheiros não foram capazes de identificar as constantes apropriadas para os controladores PID até 1942, quando finalmente Zieglers e Nichols encontraram e ajustaram estas constantes que possibilitaram o uso industrial de controladores PID na década de 50 (BENNETT, 1993).Observando a Figura 4, pode-se ver o diagrama de blocos que utiliza um controlador PID.

Figura 4 – Diagrama de blocos de um sistema de controle utilizando o controlador PID.



Fonte: Adaptado de Borase et al. (2020).

O controlador PID emprega três modos, ou seja, controle proporcional, integrador e derivativo. O termo proporcional incorpora mudanças proporcionais para erro (que é a diferença entre o ponto setpoint e variável de controle) para a saída.

O termo integral examina a variável de controle ao longo do tempo e corrige a saída reduzindo o deslocamento da variável de controle. O modo de controle derivativo monitora a taxa de alteração da variável de controle e, portanto, altera a saída quando há variações incomuns.

O projetista ou engenheiro tem liberdade de ajustar cada parâmetro das três funções de controle para obter o desejado desempenho do processo. Devido à sua estrutura simples, de fácil implementação e manutenção, os controladores PID são os controladores mais amplamente utilizados no controle de movimento, controle de processos, eletrônica de potência, hidráulica, pneumática, indústrias manufatureiras, sistemas robóticos e etc (ÅSTRÖM; HÄGGLUND, 2001).

O controlador PID de tempo contínuo é descrito, em forma paralela, pela seguinte equação (BORASE et al., 2020):

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt}, \tag{2.1}$$

onde a função $e(t)$ representa o sinal de erro e as constantes K_p, K_i e K_d são, respectivamente: os ganhos proporcional, integral e derivativo do controlador PID. Esta equação também pode ser convertida para a função de transferência de Laplace (WANG et al., 1999; MEZURA-MONTES; COELLO, 2011):

$$C(s) = K_p + \frac{K_i}{s} + K_d s. \tag{2.2}$$

Uma possível versão do controlador PID de tempo discreto é dada por (MEZURA-MONTES; COELLO, 2011):

$$u(k) = K_p e(k) + K_i T_s \sum_{i=0}^{k} e(i) + \frac{K_d}{T_s}\Big(e(k) - e(k-1)\Big). \tag{2.3}$$

onde $T_s \in \mathbb{R}$ é o tempo de amostragem do sistema digital embarcado. A constante T_s pode ser omitida da equação com o devido ajuste de K_i e K_d.

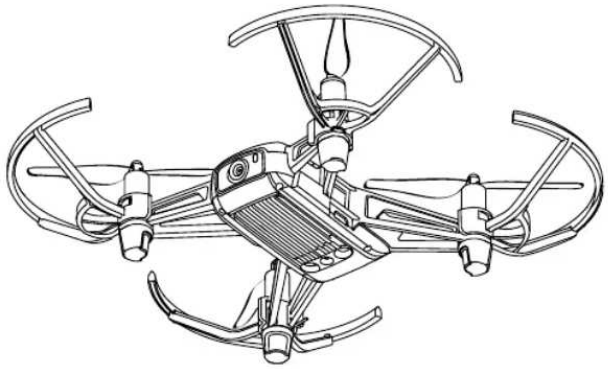
3 Materiais e Métodos

Este capítulo discorre sobre as características e aspectos técnicos de hardware do drone DJI Tello utilizado para o desenvolvimento do projeto. Será exposta a totalidade da estrutura do software que assume a tarefa de direcionar o drone de forma autônoma dentro de seu ambiente. Será discutida a seleção da linguagem de programação e das bibliotecas empregadas na criação do algoritmo do drone, justificando as decisões e escolhas com base em critérios técnicos considerados durante a implementação do software a ser embarcado no dispositivo. Por fim, será abordada a montagem do oi e a maneira pela qual foi possível confeccionar este protótipo, tornando-o apto para voar e permitindo a realização do controle por meio da visão computacional.

3.1 Descrição do hardware

O Tello como pode ser visto na Figura 5 é um pequeno quadrator projetado com um design compacto de estrutura leve, sem módulo GPS e outros sensores mais modernos. Sendo acessível se comparado a outros drones no mercado, podendo ser pilotado manualmente ou se transformar em um robô aéreo autônomo, com sistema de posicionamento e visão, além de possuir uma pequena câmera mono-ocular *onboard*.

Figura 5 – Representação do drone utilizado para o projeto.



Fonte: Ryze-Robotics (2018).

Será apresentado com uma maior riqueza de detalhes aspectos técnicos da construção

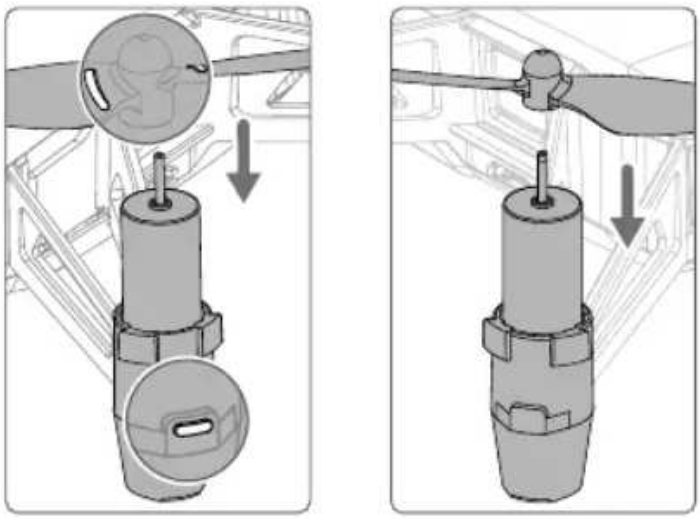
do veículo, características do hardware utilizado.

Por meio da utilização do sistema de visão e controladora de vôo, o Tello tem a capacidade de pairar no ar em lugares que sejam necessários. Logo, quando o assunto é portabilidade e facilidade a escolha do Tello é certeira se tratando de aplicações *indoor* e educacionais.

O Tello conta com dimensões aproximadas de 98 mm x 92.5 mm x 41 mm e peso de cerca de 80 gramas graças a toda sua estrutura principal ser construída com plástico ABS(Acrilonitrila butadieno estireno) durável, que oferece uma combinação de resistência e leveza.

Essa construção robusta e ao mesmo tempo leve e resistente permite que o drone resista a impactos moderados e colisões acidentais, aumentando sua durabilidade e longevidade. Muito deste vigor se dá ao eficiente conjunto de hélices e protetores de hélices do modelo 3044P que são hélices rígidas e que foram projetadas para rodarem no sentido contrario, observe a Figura 6, o tornando extremamente portátil, permitindo que os usuários o transportem facilmente para diferentes locais de voo.

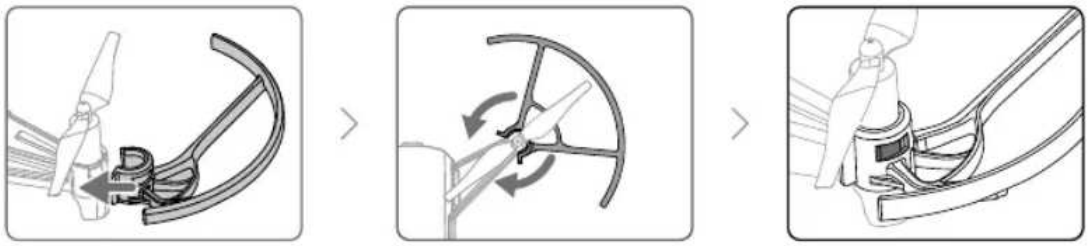
Figura 6 – Representação gráfica das hélices modelo 3044P e propulsores.



Fonte: Ryze-Robotics (2018).

O drone Tello possui uma ótima câmera que captura fotos com resolução de 5 megapixels e grava vídeo ao vivo a 720p estes e outros aspectos serão apresentados com maior riqueza de detalhes em sessões a seguir. O seu tempo de voo máximo é de aproximadamente 13 minutos. Entretanto, vale ressaltar que o tempo de voo máximo de 13 minutos foi testado em condições sem vento em ambientes onde a velocidade do vento era consistente se aproximando de 15[km/h], sendo assim este valor de tempo de voo deve ser levado em conta apenas para referência.

Figura 7 – Representação gráfica dos protetores de hélices removíveis do Tello.



Fonte: Ryze-Robotics (2018).

A distância máxima de voo é 328 pés equivalente 100 m. Mesmo sendo um drone bem compacto o Tello possui um sistema de proteção que o permite para realizar pousos emergenciais com segurança, quando a conexão é perdida e tendo também seus protetores de hélice que podem ser ótimos aliados no aspecto segurança.

Portanto, observe a Figura 8 a seguir que representa as especificações técnicas detalhadas do drone.

Figura 8 – Especificações técnicas detalhadas do drone.

Peso (incluindo a hélice Guarda)	87 g
Velocidade máxima	17,8 mph (28,8 kph)
Max Tempo de vôo	13 minutos (0 vento a uma consistente 9 mph (15 kph))
Faixa de temperatura operacional	32 ° a 104 ° F (0 ° a 40 ° C)
Faixa de freqüência operacional	2,4 a 2,4835 GHz
Transmissor (EIRP)	20 dBm (FCC) 19 dBm (EC) 19 dBm (SRRC)

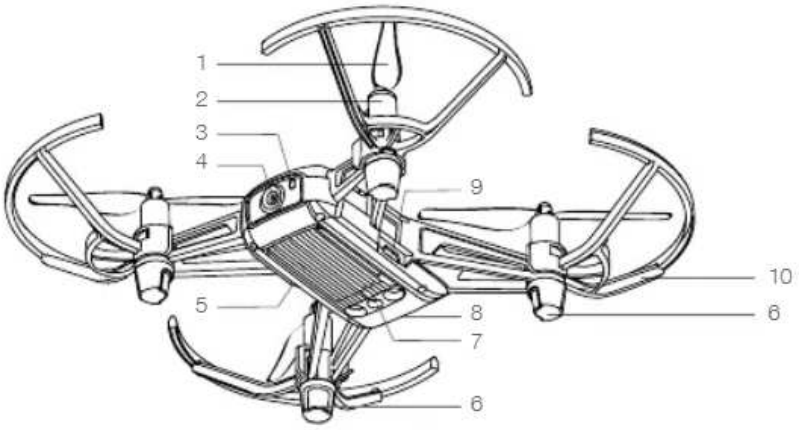
Fonte: Ryze-Robotics (2018).

3.1.1 Diagrama do drone

Para entender melhor a construção e o hardware presente no drone Tello observe o diagrama apresentado pela Figura 9 onde cada parte do drone, é representado por seu respectivo número:

1. Hélices;
2. Motores;
3. Indicador de estado;
4. Câmera;

Figura 9 – Diagrama do drone Tello.



Fonte: Ryze-Robotics (2018).

5. Botão de alimentação;
6. Antenas;
7. Sistema de posicionamento de visão;
8. Bateria;
9. Porta micro USB para carregamento; e
10. Protetores de hélice.

3.1.2 Modos de voo

O Tello possui diversos modos de voo, onde o mais utilizado é o modo de voo manual que se dá utilizando *joysticks* virtuais através de um dispositivo móvel, usando um controle remoto compatível, utilizando comandos de teclado de um computador. O modo de voo manual possui duas velocidades de voo para o drone sendo o modo Lento que é o padrão e o modo rápido. No modo lento, o máximo ângulo de atitude de voo é 9° e a velocidade máxima de voo é de 14 km/h. Já no modo rápido o drone atinge um ângulo de atitude de 25° e uma velocidade máxima de aproximadamente 29 km/h.

Ao voar no modo lento a aeronave utiliza de um sistema interno chamado de sistema de posicionamento de visão que utiliza sensores para estabilizar-se automaticamente, e caso este sistema esteja indisponível ou defeituoso a aeronave muda automaticamente para o modo atitude.

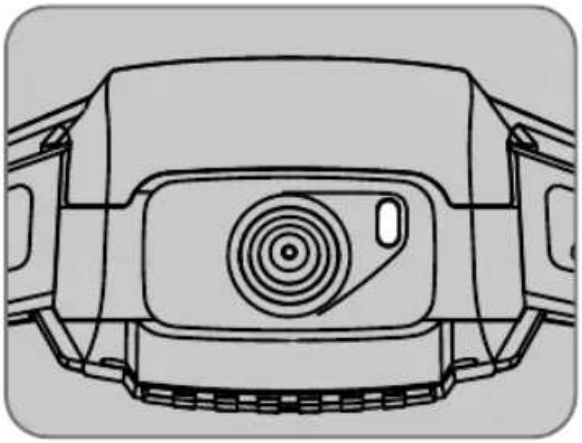
Quando o modo atitude é ativado a aeronave muda automaticamente para modo ATTI – modo de atitude quando o sistema de posicionamento de visão não está disponível (RYZE-ROBOTICS, 2018).

Neste modo, o drone não é capaz de se posicionar e assim é facilmente afetado por seus arredores tornado perigoso voar no ambiente pois pode facilmente ser afetado por fatores ambientais tais como o vento que pode resultar em deslocamento horizontais, apresentando riscos a aeronave, especialmente quando voando em espaços confinados. Se este estágio do modo atitude for atingido o drone ativa seu sistema de voo, que retorna a aeronave ao solo em local seguro assim que possível para evitar colisões e danos imprescindível determinadas circunstâncias.

3.1.3 Indicador de estado do drone

O indicador de estado do Tello é uma interface "homem-máquina" que comunica os status do sistema de controle de voo da aeronave e da bateria. O indicador de estado do drone está localizado na parte frontal, ao lado da câmera, como mostrado na Figura 10.

Figura 10 – Indicador de estado Aircraft.



Fonte: Ryze-Robotics (2018).

Sendo assim, como o indicador de estado tem a função de comunicar os status do sistema antes do voo foi padronizado em uma tabela que pode ser vista na Figura 11 que representa cada status luminoso fornecido pelo indicador.

3.1.4 Sensor de fluxo óptico e sensor Time of Flight(TOF)

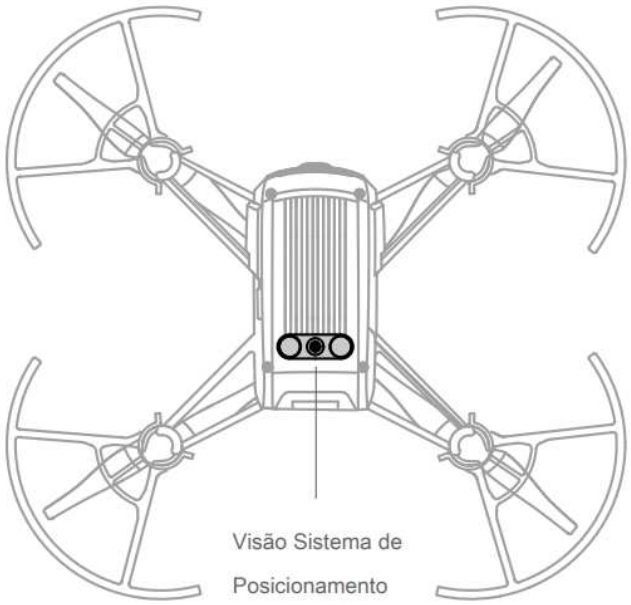
Os sensores de fluxo óptico e Time of Flight são um conjunto de sensores que ficam localizados na parte inferior do drone, este conjunto de sensores formam o sistema de posicionamento de visão como pode ser observado na Figura 12.

Figura 11 – Indicadores de estado do drone.

	Cor	padronizar	Estado aeronaves
estados normais			
	Alternando vermelho, verde e amarelo	piscando	Ligando e executando testes de diagnóstico de auto-
	Verde	Periodicamente pisca duas vezes	Visão Sistema de Posicionamento ativa
	Amarelo	piscando lentamente	Visão Positioning System indisponíveis, aeronave está em modo de Atitude
Unidos de carregamento Azul			
		Sólido	O carregamento está completo
	Azul	piscando lentamente	carregamento
	Azul	piscar rapidamente	erro de carregamento
Unidos Aviso Amarelo			
		piscar rapidamente	sinal de controlo remoto perdido
	Vermelho	piscando lentamente	Bateria Fraca
	Vermelho	piscar rapidamente	Criticamente baixo bateria
	Vermelho	Sólido	Erro crítico

Fonte: Ryze-Robotics (2018).

Figura 12 – Sensores de fluxo óptico e Time of Flight.



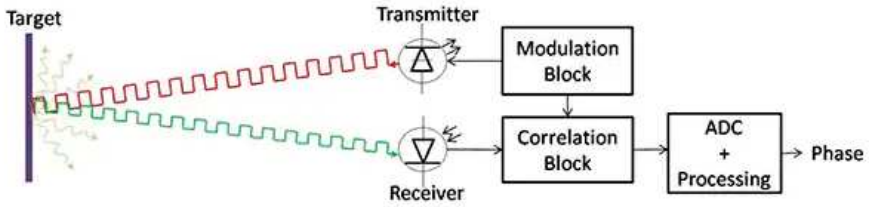
Fonte: Ryze-Robotics (2018).

O sensor de fluxo óptico, localizado na parte inferior do drone é voltado para o solo, para ser possível calcular a mudança nas coordenadas X e Z ao longo do tempo de voo. Este sensor é um sensor de imagem de baixa resolução, tirando imagens a cada intervalo de tempo comparando quadros consecutivos e derivando a mudança em cada coordenada. O sensor de fluxo óptico auxilia o drone a pairar sobre um determinado local sem deriva sendo utilizado para calcular a velocidade relativa do drone porém usado apenas internamente pelo drone para calcular a velocidade e não pode ser acessado diretamente.

O sensor TOF fornece medição rápida de distância para diversas aplicações, incluindo sistemas de assistência ao motorista automotivo, drones e até mesmo interfaces de usuário. Embora difundidos, estes sensores são tecnicamente exigentes, onde se faz necessário equilibrar requisitos como precisão, alcance, tempo de resposta e resolução com custo, consumo de energia e espaço disponível.

O princípio de funcionamento do TOF é simples, sendo extremamente popular na detecção remota de objetos usando ondas de luz, ultrassom e tecnologias de radar. O sistema consiste em basicamente em um módulo infravermelho transmissor que emite um sinal modulado ao alvo que reflete uma parte do sinal de volta para um receptor. Este sinal recebido é correlacionado com o sinal transmitido por um processador dedicado que mede o tempo de voo, e calcula o alcance correspondente ao alvo Figura 13.

Figura 13 – Sensor Time of Flight (TOF).



Fonte: Rocha (2023a).

Assim como o sensor de fluxo óptico o TOF também é localizado na parte inferior do drone e voltado para o solo, onde é utilizado para determinar a altura relativa do drone a partir de um objeto abaixo dele, o que é essencial para evitar obstáculos por baixo que em conjunto com um barômetro, é possível detectar uma altitude relativa em relação ao ponto de decolagem permitindo realização de voos mais precisos em ambientes fechados ou ao ar livre em condições sem vento.

Como visto anteriormente estes dois sensores compõe o chamado sistema de posicionamento de visão. Este sistema é ativado automaticamente quando o drone está ligado, não sendo necessário nenhuma ação adicional.

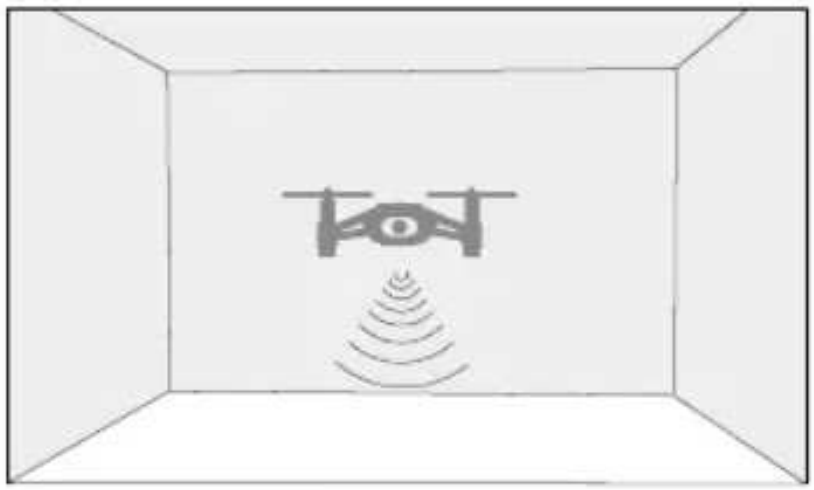
O Sistema só é eficaz quando a aeronave está em altitudes de 0,3 a 10 m e funciona melhor em altitudes de 0,3 a 6 m veja a Figura 14.

Se o drone está além deste intervalo de altura o sistema pode ser afetado causando possíveis problemas. Portanto nestas situações é necessário maior cautela.

O desempenho do sistema de posicionamento visão pode ser afetado através das seguintes superfície a ser sobrevoada:

1. Voando a altas velocidades abaixo de 0,5m;

Figura 14 – Representação do sistema de posicionamento de visão.



Fonte: Ryze-Robotics (2018).

2. Voando sobre superfícies monocromáticas (preto por exemplo, puro, branco puro, puro vermelho, verde puro);
3. Voando sobre superfícies altamente refletoras;
4. Voando sobre água ou superfícies transparentes;
5. Voando sobre superfícies ou objetos em movimento;
6. Voando em uma área onde a iluminação muda com frequência ou drasticamente;
7. Sobrevoando extremamente escuro (<10 lux) ou brilhantes (> 100.000 lux) superfícies ou para fontes de luz brilhantes (por exemplo, para a luz do sol);
8. Voando sobre superfícies sem padrões ou textura claras. E voando sobre superfícies com padrões idênticos de repetição ou textura (por exemplo, ladrilhos);
9. Voando sobre pequenas e finas objetos (por exemplo, galhos de árvores ou linhas de energia); e
10. Voando a uma velocidade de mais de 18 km/h a 1 m de altura ou inferior.

Logo, quando algum dos casos mencionados anteriormente ocorre o drone altera automaticamente para o modo de atitude. Isto ocorre pois, no modo atitude a aeronave não é capaz de se posicionar.

3.1.5 Bateria

A fonte de alimentação do Tello é composta de uma bateria de íons de lítio removível, com uma capacidade de 1100 [mAh] tendo uma tensão de 3,8 [V].

Figura 15 – Ilustração da bateria do drone Tello.

Fonte: Ryze-Robotics (2018).

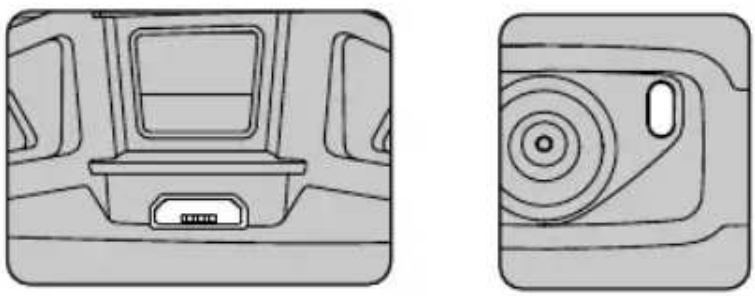
Esta bateria oferece energia suficiente para permitir um tempo de voo de aproximadamente 13 minutos, embora a duração exata dependa das condições de voo e do estilo de pilotagem.

A bateria tem algumas características próprias como por exemplo proteção contra sobrecorrente/sobretensão, onde nesta situação a bateria interrompe o fluxo de carga se uma corrente e ou tensão excessiva for detectada.

Além disso, possui um sistema de *Overdischarge* que realiza uma descarga automaticamente evitando a descarga excessiva e também uma contra curto circuito em situações onde a fonte de alimentação é cortada automaticamente se um curto-circuito é detectado.

Quando totalmente descarregada, basta conectar a bateria na porta Micro USB existente no drone.

Figura 16 – Ilustração do processo de carregamento da bateria do Tello.



Fonte: Ryze-Robotics (2018).

Uma outra alternativa é utilizar um carregador compatível com a bateria, utilizando a porta Micro USB o tempo de carregamento leva em torno 1 hora e 30 minutos, enquanto com o carregador compatível pode-se carregar até 3 baterias em um intervalo de 1 hora (RYZE-ROBOTICS, 2018).

O próprio Tello emite uma luz piscando lentamente durante o processo de carregamento demonstrando e uma luz azul sólida quando bateria está totalmente carregada.

Segundo a Ryze Tech, é necessário respeitar as regras de segurança durante o processo de bateria descrito abaixo:

1. Sempre usar um adaptador USB juntamente com uma fonte certificada que suporte de 5 V a 1,5 A ou acima;
2. Garantir a aeronave está desligado antes de carregar. Ele não pode ser cobrado quando ele é ligado;
3. Não carregue o voo da bateria imediatamente após o voo, pois a temperatura pode ser muito alta; e
4. Carregue a bateria de voo em um intervalo de temperatura de 5°a 45°C. Entretanto, a gama de temperatura ideal de carregamento é de 22° a 28°C).

Figura 17 – Tabela detalhada das características técnicas da bateria

bateria de vôo	
Capacidade	1100 mAh
Voltagem	3,8 V
Tipo de Bateria	Lipo
Energia	4,18 Wh
Peso líquido	25 ± 2 g
Carregar Faixa de temperatura	41 ° a 113 ° F (5 ° a 45 ° C)
Max carga de energia	10 W

Fonte: Ryze-Robotics (2018).

3.1.6 Câmera

Uma das características notáveis do DJI Tello é sua excelente câmera integrada, posicionada na parte frontal do drone. A câmera possui um sensor CMOS de 1/2.3 com 5 megapixels efetivos, permitindo que os usuários capturem imagens e gravem vídeos durante o voo.

A câmera tem capacidade de capturar imagens com uma resolução de 2592 x 1936 pixels garantindo imagens de alta qualidade, com detalhes nítidos e cores vibrantes.

Além disso, a função de estabilização eletrônica de imagem ajuda a reduzir as vibrações e tremores, resultando em imagens mais suaves e vídeos estáveis (RYZE-ROBOTICS, 2018).

Em termos de gravação de vídeo, a câmera do Tello suporta resolução HD de 720p a 30 quadros por segundo. Essa qualidade de vídeo é adequada para a maioria das necessidades recreativas e permite capturar momentos memoráveis com clareza e fluidez. Portanto veja a Figura 18 abaixo que representa as especificações técnicas detalhadas da câmera.

Figura 18 – Especificações técnicas completas da câmera.

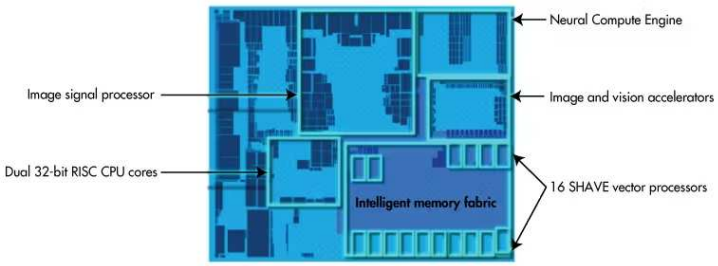
Câmera	
Max Tamanho da Imagem	2592 × 1936
Modos de gravação de vídeo	HD: 1280 × 720 30p
Formato de vídeo	MP4

Fonte: Ryze-Robotics (2018).

3.1.7 Processador

Uma das grandes vantagens de se utilizar o Tello além de sua grande portabilidade se da ao fato de o pequeno drone ser construído com o poderoso processador Intel Movidius Myriad 2. O Myriad 2, além de ser utilizado no Tello é encontrado em outros drones mais robustos e modernos como por exemplo o DJI Spark. Esta poderosa VPU (Unidade de processamento visual) trata-se de um processador de visão computacional e deep learning, projetado especificamente para aplicações de inteligência artificial e processamento de imagem em tempo real. Esta VPU fornece ao Tello a capacidade de realizar inferências através da execução de algoritmos de detecção de objetos, reconhecimento facial e outras tarefas computacionalmente custosas observe Figura 19.

Figura 19 – Processador VPU da Intel Movidius Myriad 2



Fonte: Rocha (2023a).

O Myriad 2, é o modelo de entrada para a série VPU da Intel, além disso este poderoso chip possui aceleradores de hardware de codificação e decodificação como por exemplo H.264 e Motion JPEG, bem como um mecanismo para tratar distorções de

vídeos e imagens capturadas por lentes olho de peixe, fluxo óptico denso e percepção de profundidade de câmeras estéreo, podendo operar com fluxos de 720p a 180 quadros/s.

Possuindo uma malha de memória inteligente de 450 Gbyte/s e 2,5 Mbytes de memória on-board formando um sistema de memória multi porta vinculado a todas as principais funções do sistema. Isso é fundamental para minimizar a movimentação de dados que é frequentemente usada em outros sistemas onde os componentes são encontrados em outros chips. Isso permite que os dados sejam despejados na memória de até 8 câmeras HD a uma taxa de até 700 Mpixels/s (RYZE-ROBOTICS, 2018).

Os aceleradores integrados o Movidius permite que os dados transportados logicamente entrem na pipeline do processo de maneira optimizada. A optimização reduz a latência e o consumo de energia, união perfeita que garante ao embarcado características para se operar com alto desempenho e nas condições desejadas.

Portanto, graças ao processador Myriad 2 e a excelente controladora de voo produzida pela empresa Chinesa DJI é possível transformar o Tello em um robô aéreo autônomo.

3.2 Descrição do software

Mesmo que o Drone DJI Tello, não seja um drone sofisticado com tantos recursos como GPS, bussola e outros sensores é notório a sua qualidade.

Para que seja possível transformar o Tello em um drone autônomo é necessário a construção de um software robusto, eficiente e que consiga se comunicar com o embarcado com baixa latência, grande largura de banda e bom *throughput* de dados pois é através deste canal de comunicação que irá trafegar todos os metadados responsáveis por enviar a informação resultante do processo de inferência para o drone.

3.2.1 Kit de desenvolvimento de software e biblioteca djitellopy

Pensando em todas as características para se implementar um robô aéreo autônomo, a DJI em parceria com Chinesa Ryze Robotics, desenvolveu um SDK (*"Kit de desenvolvimento de software"*) Open Source que posteriormente através da comunidade acabou sendo portado para as linguagens de programação Python, Swift e Scratch.

Posteriormente após a popularização do SDK, atualmente pode-se encontrar suporte a bibliotecas das mais diversas desde bibliotecas para criação de aplicativo como Flutter, React até mesmo para programação em celulares como a chamada "Drone Blocks". Este sucesso do SDK permite que desenvolvedores criem aplicativos e recursos adicionais para outras linguagens de programação através da criação de APIs(Interface de programação de aplicativos) de alto nível.

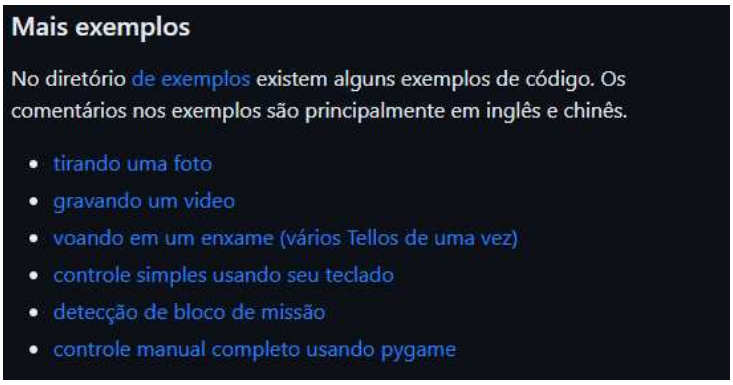
O desenvolvimento de novas funcionalidades advindas da comunidade Open Source, transformam positivamente a experiência de se trabalhar com robôs aéreos autônomos tornando amigável, além de expandir as funcionalidades.

Pensando no suporte a linguagem Python, foi desenvolvida uma biblioteca robusta nomeada de djitellopy. Esta biblioteca, possui uma série de funcionalidades para realizar a interação entre o drone e o software. Permitindo que desenvolvedores e pesquisadores criem aplicativos e recursos adicionais para expandir as funcionalidades do drone.

A biblioteca djitellopy, é resultado da implementação do desenvolvedor Damia Fuentes em conjunto com a comunidade OpenSource baseado no SDK oficial.

Esta poderosa biblioteca conta com diversas funcionalidades interessantes extras além das fornecidas pelo SDK oficial, como por exemplo a funcionalidade (*Swarm*) que é a capacidade de diversos drones realizarem voos em enxame entre outras diversas funcionalidade como mostra a Figura 20.

Figura 20 – Exemplo de funcionalidades da biblioteca DjiTelloPy



Fonte: Fuentes (2019).

Sendo assim, através do SDK e da biblioteca serem Open Source e flexíveis foi possível desenvolver este projeto realizar o controle e posicionamento do Tello utilizando VC.

3.2.2 Mecanismo de comunicação drone - software

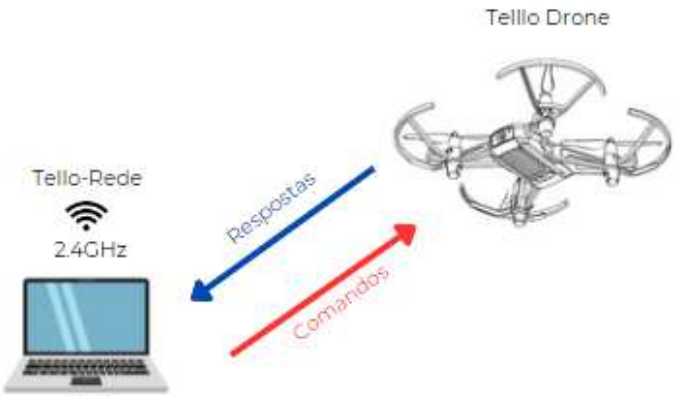
Para que o drone se comunique com o software e receba as instruções a serem executadas é necessário a comunicação com algum dispositivo. Logo, o Tello cria uma rede uma rede Wi-fi Direct. Esta tecnologia permite a comunicação direta entre dispositivos compatíveis, sem a necessidade de um ponto de acesso, ou *Access Point* (AP), intermediário.

Diferente de uma conexão tradicional de Wi-Fi, em que os dispositivos se conectam a um ponto de acesso para estabelecer uma rede local ou se conectarem à Internet, o Wi-Fi Direct permite que os dispositivos se comuniquem diretamente entre si, formando uma rede AD-HOC, também conhecida como rede ponto a ponto. Esta conexão permite uma

comunicação direta entre dois dispositivos, sem a necessidade de uma rede Wi-Fi externa, entretanto é operado na mesma frequência de 2.4 GHz.

Portanto, através do Wi-Fi Direct embarcado no drone gerado a partir do módulo de rede, é possível a troca de pacotes contendo metadados através da conexão entre a rede gerada e um computador veja a Figura 21.

Figura 21 – Mecanismo de conexão drone - software



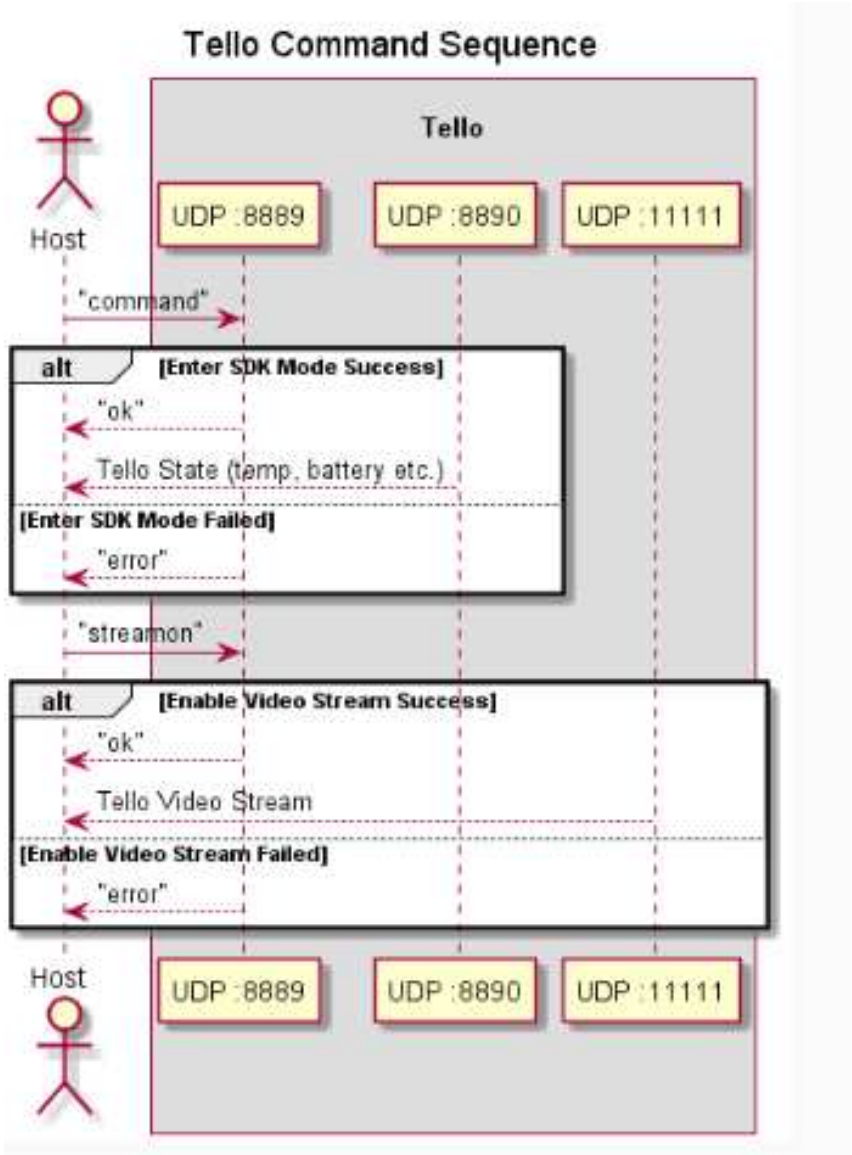
Fonte: Rocha (2023a).

Como mostrado na Figura 21 através do Wi-Fi gerado pelo drone, é possível realizar a troca de mensagens toda esta interface de comunicação entre o computador e o drone é feita através do protocolo de comunicação UDP. Nestas mensagens contém comandos como por exemplo decolagem, aterrissagem, *streaming* de vídeo ligado/desligado, mover para cima/baixo e etc. Todos estes comandos são enviados de um computador para o drone através da porta UDP nº 8889. Por outro lado, as respostas e o *streaming* de vídeo são enviados de volta do drone Tello EDU para o computador através das portas UDP nº `11111` e `8890` respectivamente veja a Figura 22.

3.3 Preparação do ambiente para montagem do experimento

Com o objetivo de implementar o controle de posicionamento de um robô aéreo autônomo utilizando visão computacional, foi necessário levantar todos os requisitos

Figura 22 – Portas UDP, conexão drone - software



Fonte: Ryze-Robotics (2018).

necessários para o projeto para pensar em como seria a modelagem do sistema e sua arquitetura.

Toda a montagem da arquitetura do sistema foi feita sob a plataforma do computador Apple M1.

O Apple M1 é um processador sistema-em-um-chip, ou *System on Chip* (SOC), o que significa que a CPU, GPU e memória RAM ficam todos contidos em um único chip. Este chip utiliza a arquitetura arm64 contendo CPU de 8 núcleos sendo seis núcleos de desempenho e dois núcleos de eficiência. Possui também para tarefas de inferência uma GPU de 14 núcleos, além de conter uma tecnologia nomeada neural *engine* de 16 núcleos com largura de banda de memória de 200 Gbps. Toda essa robustez permite que cada componente se comunique com mais eficiência para tempos de processamento mais rápidos

e menor latência, reduzindo o consumo de energia.

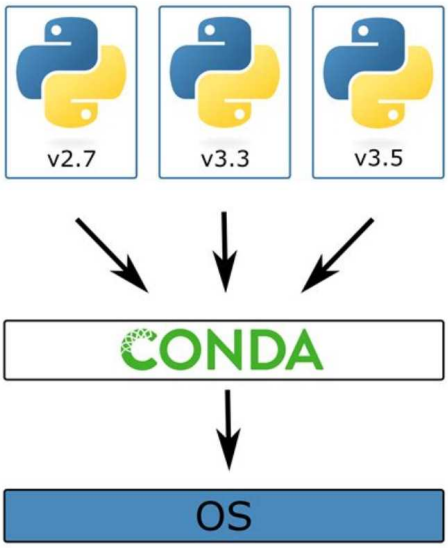
Relacionado a arquitetura de software , a mesma foi construída sob a linguagem Python na versão 3.8. A linguagem Python é uma linguagem de propósito geral, ou seja, pode ser utilizada para as mais diversas aplicações. Em termos técnicos, Python é uma linguagem multiparadigma, ou seja , interpretada, orientada a objetos, funcional, tipada, imperativa e de *script* , sendo a responsável por construir todo o *script* que será executado dando início a arquitetura do sistema.

Para poder construir o algoritmo de detecção e rastreio foram utilizados as bibliotecas OpenCV, MediaPipe e CVZone. Entretanto essas bibliotecas possuem diversas versões podendo apresentar problemas ao se trabalhar com diferentes versões.

Logo, para solucionar este problema foi utilizado a IDE Anaconda, o Anaconda vem com o Conda, uma ferramenta para gerenciamento de projetos e ambientes operada exclusivamente a partir da linha de comando.

O Conda permite a criação de ambientes virtuais sendo possível manter pacotes Python e versões de bibliotecas em versões específicas, sendo capaz de lidar com diversas versões de Python e das bibliotecas OpenCV, MediaPipe como mostra a Figura 23.

Figura 23 – Gerenciamento de versões utilizando o Conda



Fonte: Rocha (2023a)

Para ser possível desenvolver o algoritmo de rastreio foi necessário o MediaPipe na versão 0.6.1. O Mediapipe como descrito na seção 2 é um *framework* para construção de pipelines com o objetivo de realizar inferência sobre dados arbitrários. Através deste poderoso *framework* de VC juntamente com o OpenCV na versão 4.7 e CVZone 0.7.1 foi possível construir o algoritmo de detecção responsável por realizar a inferência.

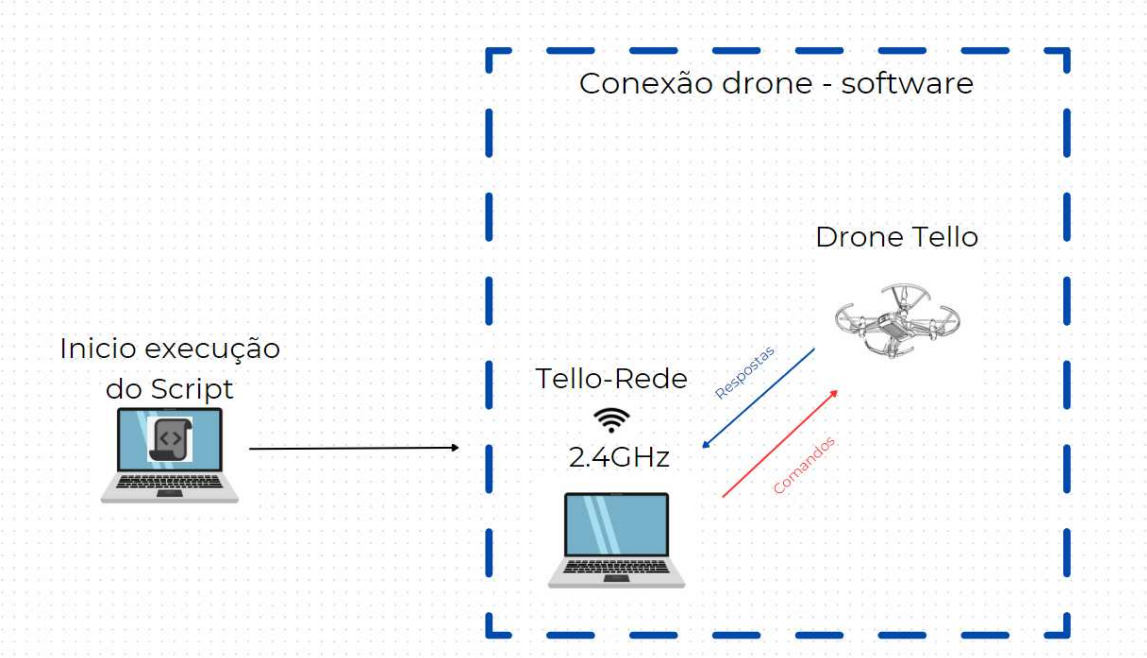
3.3.1 Arquitetura do sistema

Pensando em uma arquitetura modular não monolítica, o sistema foi montado dividido em três blocos pensados para suportar manutenções e possíveis atualizações:

1. Bloco de Conexão drone-software;
2. Bloco de detecção e rastreio; e
3. Bloco de controle de posicionamento.

Na Etapa 1, ou também nomeada de bloco conexão drone-software é a etapa inicial, ou seja, o ponto de partida do experimento. Nesta etapa, inicialmente o drone é ligado gerando uma rede Wi-Fi direct nomeada Tello-Rede, sendo assim o computador se conecta a Tello-Rede. Estando tudo certo com a conexão agora o drone e o computador conseguem realizar a troca de dados e portanto o *script* é executado dando início ao experimento veja a Figura 24.

Figura 24 – Bloco de Conexão drone-software



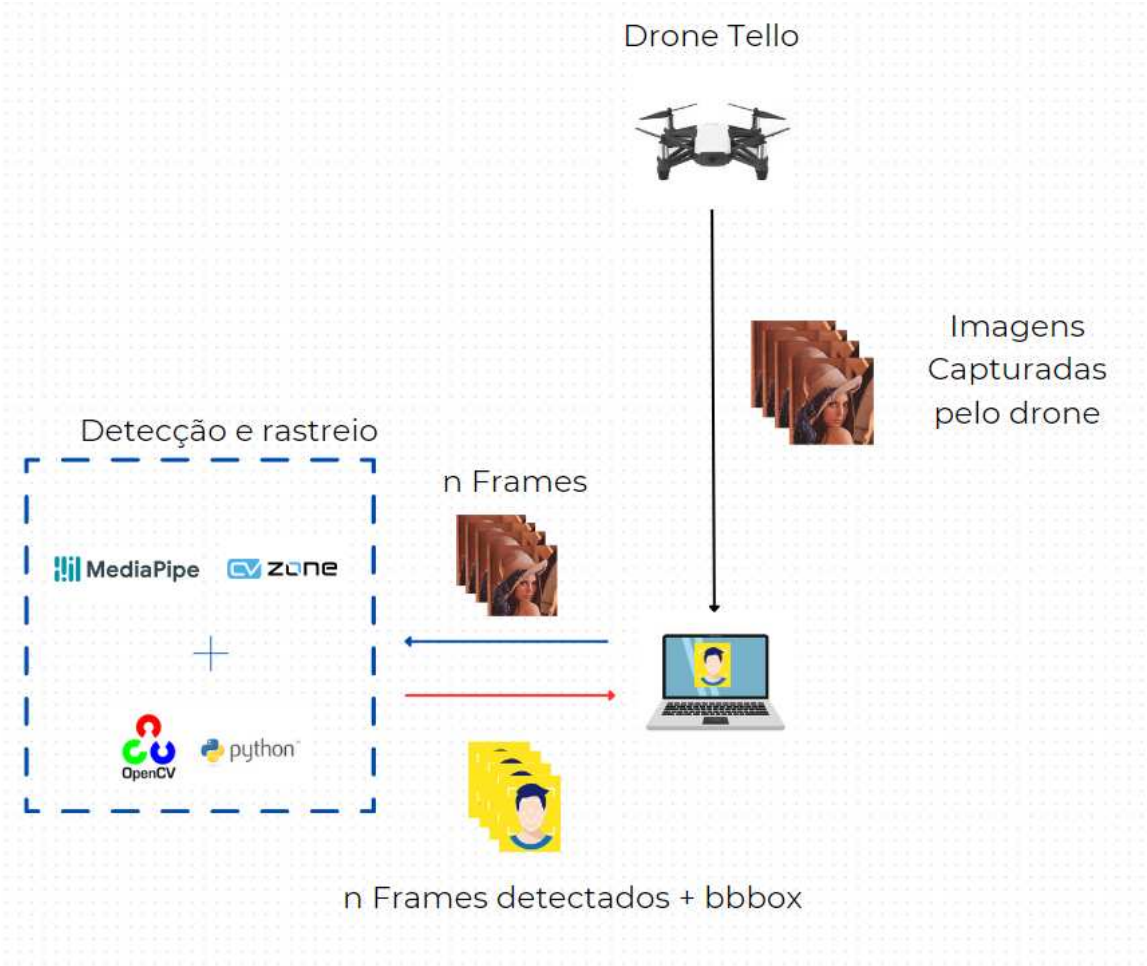
Fonte: Rocha (2023a).

Já na Etapa 2, têm-se o bloco de detecção e rastreio, com a conexão drone-software estabelecida e *script* iniciado, o drone começa a captura de vídeo através do método streamon() presente no *script*. Este método é responsável por ligar a câmera do drone e capturar "*n*-frames". Os frames capturados são enviados para o computador que irá iniciar o *pipeline* de detecção ou bloco de detecção.

Dentro do pipeline de detecção existe uma sequência de processamento que é realizada da entrada até a saída, contendo as informações que irão ser fornecidas para ser

possível realizar o controle de posicionamento do robô aéreo e, sendo assim, essa sequência, ilustrada na Figura 25, o seguinte conjunto de procedimentos é realizado para cada frame capturado:

Figura 25 – Bloco de detecção e rastreio



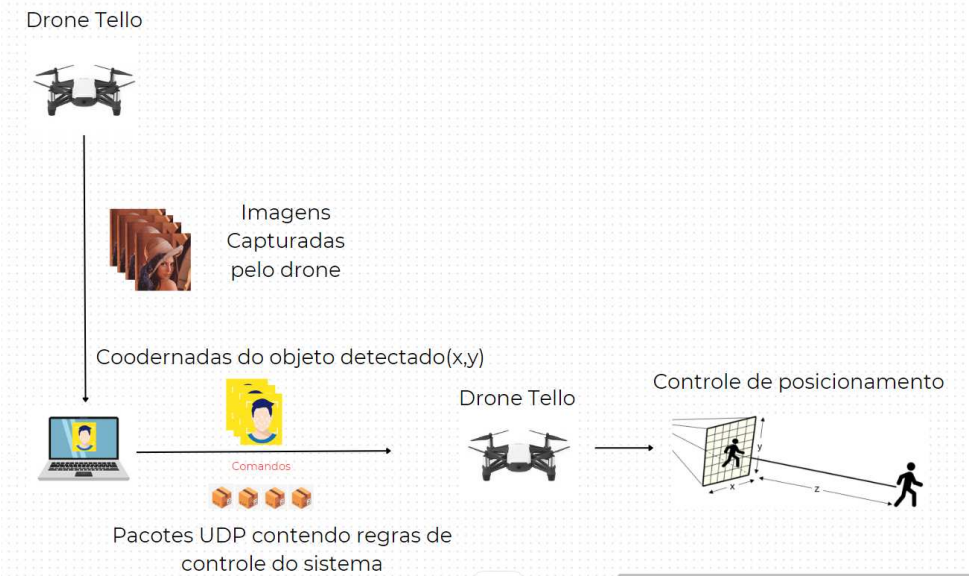
Fonte: Rocha (2023a).

- Observando a Figura 25, inicialmente os nframes capturados pelo drone serão a entrada de dados (input);
- Feito a entrada de dados, a biblioteca OpenCV irá realizar um pré-processamento dos frames os convertendo em escala de cinza para diminuir o tempo de processamento;
- Feito o pré-processamento através do OpenCV, o MediaPipe Object Detector realizará a detecção e localização da única classe do modelo que é o rosto dentro dos n- frames. Feita a detecção teremos então um tensor (matriz com "n"dimensões) contendo as coordenadas de todos os objetos detectados que possuem a classe rosto; e
- Com isso, serão atribuídas as caixas delimitadoras (*bouding box*) a cada rosto presente nos frames capturados. Logo, para representar as caixas delimitadoras nos frames

detectados é feita uma análise nos pixels partindo das coordenadas de origem (0,0) que neste caso correspondem ao centro do objeto detectado ou seja o centro do rosto. Sendo assim, através das coordenadas do centroide do frame detectado $[c_x, c_y]$ e possível desenhar através da biblioteca OpenCV os retângulos que representam as caixas delimitadoras (*bouding box*). Entretanto, neste caso específico entra a biblioteca CVZone pois as caixas delimitadores devem respeitar os coeficientes K_p , K_i, K_d do controlador PID, que serão utilizados para realizar o controle de posicionamento. A biblioteca CVZone, contém as equações do controlador PID, que foram ajustados empiricamente afim de se obter um sistema estável sem muitas oscilações.

E assim, na Etapa 3, com as matrizes de coordenadas contendo a localização da classe rosto que é o objeto em questão, juto com as caixas delimitadoras devolvidas pelo *script* e renderizados na tela do computador, serão enviadas pelo protocolo UDP, na porta `8889`, a informação de todas as coordenadas do objeto para o drone, o qual respeitando as regras do controlador PID irá realizar os cálculos irá ajustar o seu erro de posicionamento $e(t)$ afim de realizar o controle de posicionamento através das coordenadas mais caixas delimitadoras veja a Figura 26, que ilustra este processo.

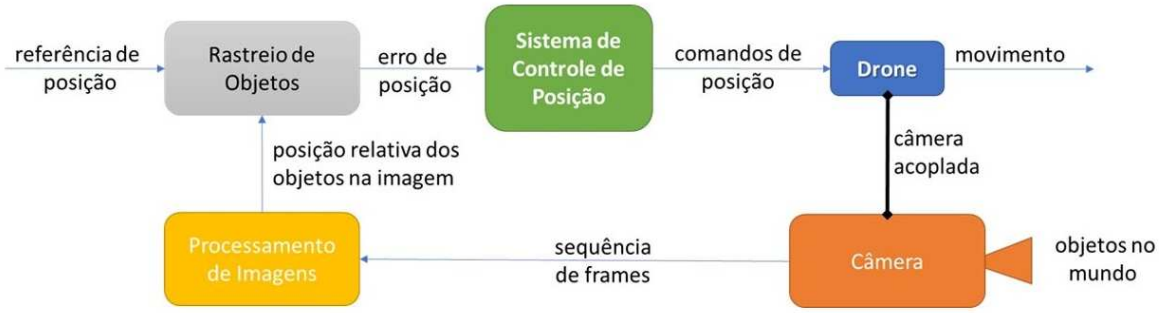
Figura 26 – Bloco de controle de posicionamento



Fonte: Rocha (2023a).

Portanto, a Figura 27 exibe um diagrama de blocos que generaliza e simplifica toda a arquitetura do sistema.

Figura 27 – Simplificação da arquitetura do sistema de controle e posicionamento



Fonte: Rocha (2023a).

4 RESULTADOS E DISCUSSÕES

Este capítulo apresenta os resultados obtidos após a construção do algoritmo responsável por realizar o controle de posicionamento do robô aéreo autônomo, além de discutir as dificuldades encontradas na etapa de testes.

4.1 Montagem do experimento e discussões

Para ser possível concluir o objetivo e construir um sistema de controle de posicionamento, esquematizado na Figura 27

Contudo, quando o tempo $t \neq 0$ e a posição $p \neq 0$, têm se que $[c_x, c_y] \neq 0$. logo, partindo deste princípio, como mostra a Figura 27, a medida em que t e p são alterados têm-se para malha de controle e consequentemente para o drone (variável controlada) a representação de erro de posicionamento $e(t)$. Sendo assim, o erro de posicionamento deve ser tratado por um controlador PID, para que o drone faça uma comparação entre a referência de posição a posteriori (último frame capturado) e faça uma comparação entre a posição relativa do objeto na cena e as caixas delimitadoras atribuídas pelo bloco de detecção. Partindo desta comparação disso sinal de controle ou comando de posição será enviado ao drone para que seja feita, uma correção no posicionamento e realiza o rastreio ou seja, acompanhar, as coordenadas $[c_x, c_y]$, consequentemente realizando o controle de posicionamento.

A biblioteca CVZone possui um método `pid()` que implementa as equações do controlador PID. Porém, para o sistema operar com uma resposta temporal adequada, fez-se necessário realizar a sintonia dos ganhos do controlador PID, este procedimento consiste em encontrar os coeficientes K_p, K_i, K_d. Este processo exigiu grande esforço para encontrar os coeficientes adequados à resposta temporal desejada. A sintonia foi realizada por meio de um processo empírico, de ajuste recursivo dos valores. Para facilitar esta etapa de calibração foi construído um gráfico para representar a resposta do sistema e ser possível calibrar o PID nos eixos X, Y e Z, partindo das coordenadas das caixas delimitadoras como mostram respectivamente as Figuras 28, 29 e 30.

Para ser possível concluir a calibração do PID muitas dificuldades foram encontradas em especial para a calibração do PID no eixo Z. O eixo Z, é o eixo responsável pela profundidade, entretanto para ser possível coletar informações de profundidade ou *depth mode* se faz necessário de uso de câmeras stereo por exemplo ou a utilizando de lidar que fornecem uma maneira de ser trabalhar com o conceito de points cloud ou a utilização de SLAM e assim ser possível trabalhar perfeitamente em 3D.

Figura 28 – Calibração do PID no eixo X

Fonte: Rocha (2023a).

Figura 29 – Calibração do PID no eixo Y

Fonte: Rocha (2023a).

Figura 30 – Calibração do PID no eixo Z

Fonte: Rocha (2023a).

Logo, para solucionar o problema da calibração no eixo Z, foi observado de que a medida em que o objeto rosto se afastava ou aproximava da câmera do drone a caixa delimitadora alterava sua área, portanto o erro de posicionamento é um erro da área da caixa delimitadora, concluindo que ao passo em que a área da caixa delimitadora aumenta ou diminui o erro de posicionamento é dado em função desta área. Logo, o cálculo do erro do posicionamento do drone foi feito em relação a esta metodologia. Porém não é a solução

mais adequada pois para uma melhor precisão e controle o melhor procedimento se dá através da utilização de uma câmera stereo.

4.2 Resultados

Com o objetivo de entregar um robô aéreo autônomo, após diversos testes e ajustes na calibração do PID nos eixos X, Y e Z têm-se um sistema preparado para iniciar o voo do robô e realizar os testes finais do protótipo. Sendo assim, o robô segue uma "pipeline" de execução como mencionado no Capítulo 3.

Primeiramente, o drone é ligado e apresenta no indicador de estado com uma luz alternando entre verde, vermelho e amarelo piscando que como exibido na Figura 11 representa o processo de inicialização do drone e a execução de testes de diagnóstico internos para verificar se existe algum erro antes da decolagem.

Figura 31 – Luzes piscando verde,vermelho,amarelo representando verificações internas.

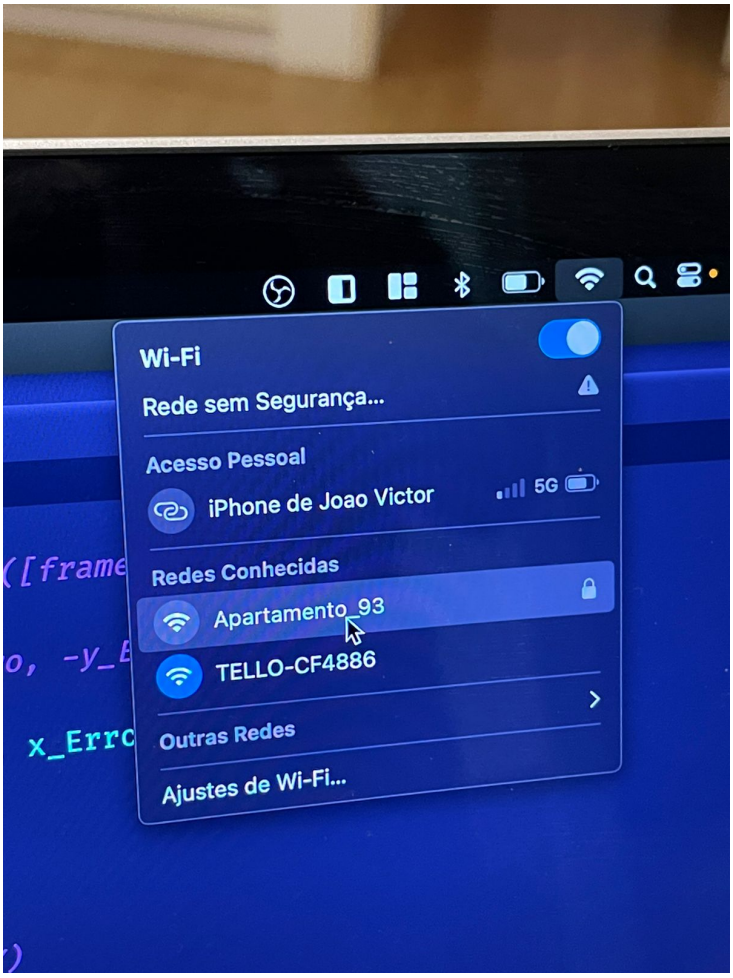
Fonte: Rocha (2023a).

Após todos a execução bem sucedida de todos os testes o robô então está pronto para decolar e se conectar com o computador que irá dar o início da pipeline do processo de execução. Logo o drone gera uma rede Wi-Fi Direct para que o computador se conecte a sua rede para troca de informações como mostra a Figura 32.

Com a rede conectada entre drone e computador como exibe a Figura 32 é dado início a execução do *script* Python que inicializa o sistema como mostra a Figura 33.

Após a execução do *script* o drone inicia o voo até atingir a altura do rosto para realizar a coleta do primeiro frame de detecção como mostrado nas Figuras 34, 35 e 36.

Figura 32 – Conexão rede Tello - CF4886.

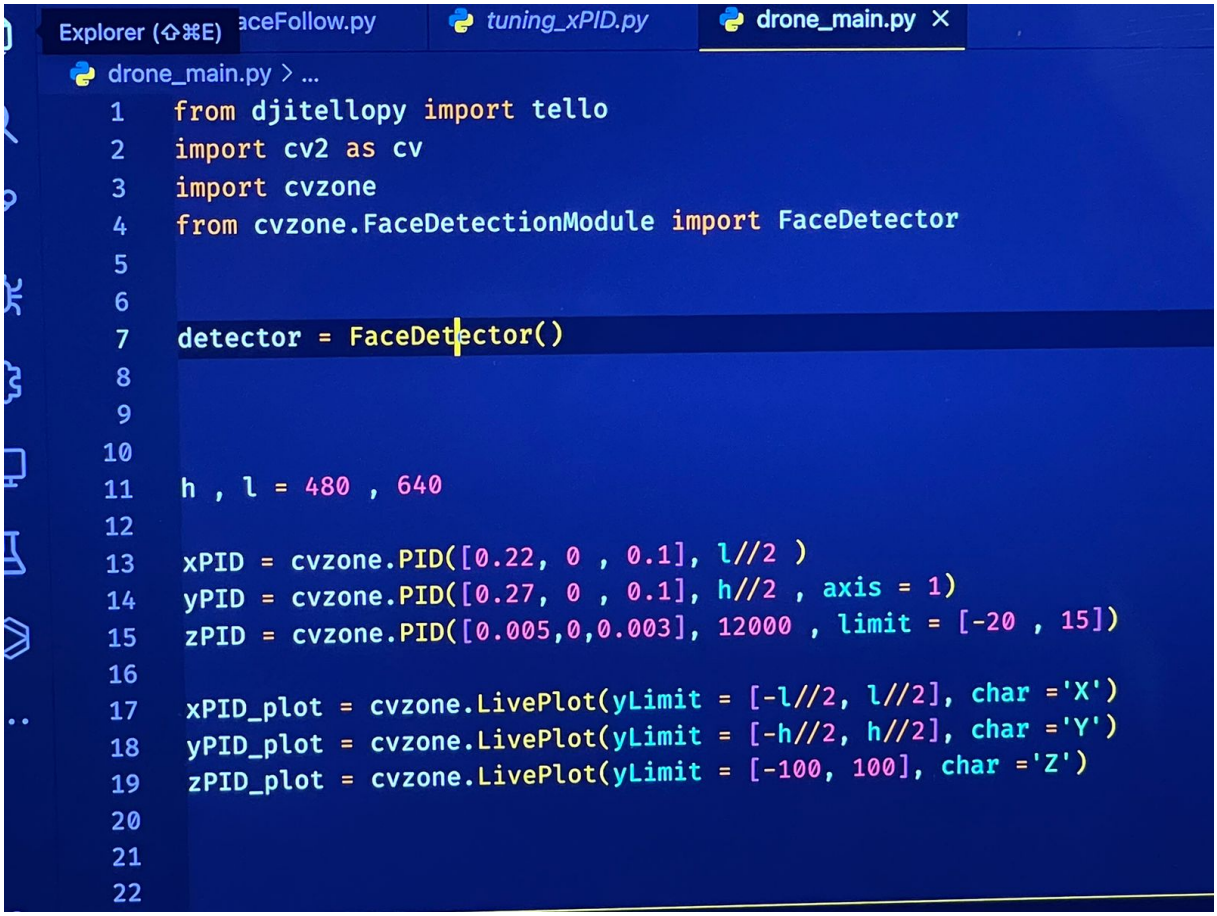


Fonte: Rocha (2023a).

Figura 34 – Inicio do voo até atingir a altura do rosto.

Fonte: Rocha (2023a).

Figura 33 – Script python que inicializa o sistema.

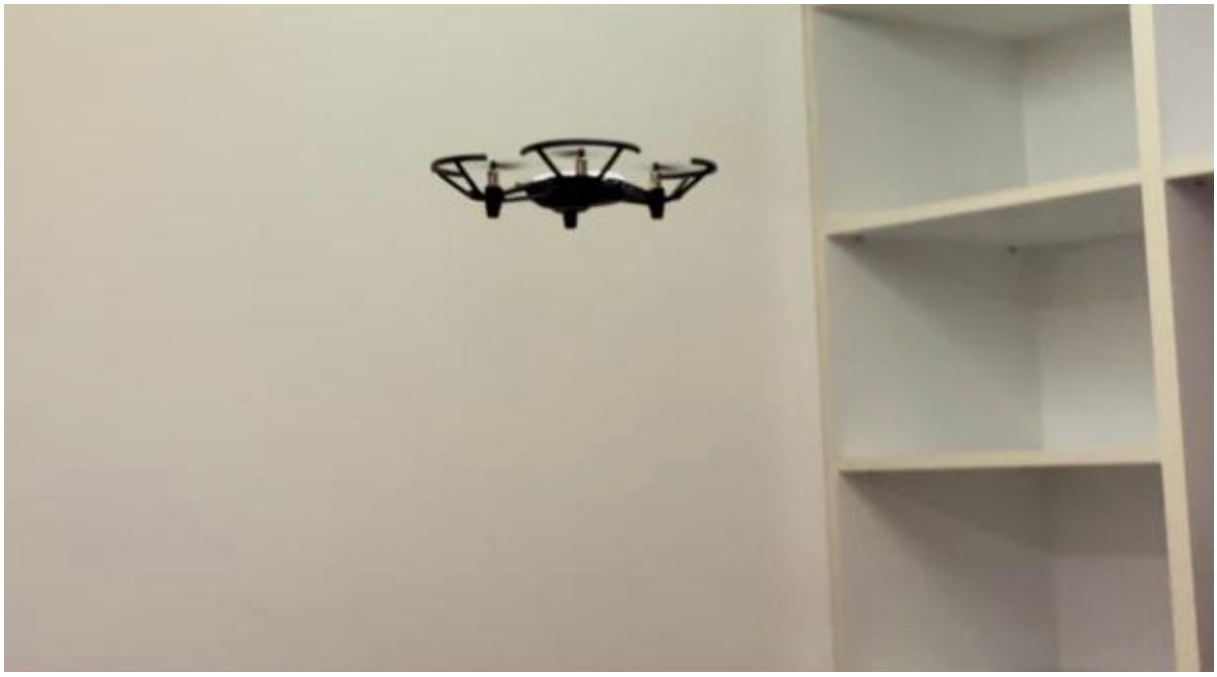


Fonte: Rocha (2023a).

Figura 35 – Voo até uma altura média.

Fonte: Rocha (2023a).

Figura 36 – Voo até a altura do rosto.



Fonte: Rocha (2023a).

Com o drone até a altura do rosto inicia-se a detecção e sendo assim, já é possível realizar o controle como pode ser observado na Figura 37.

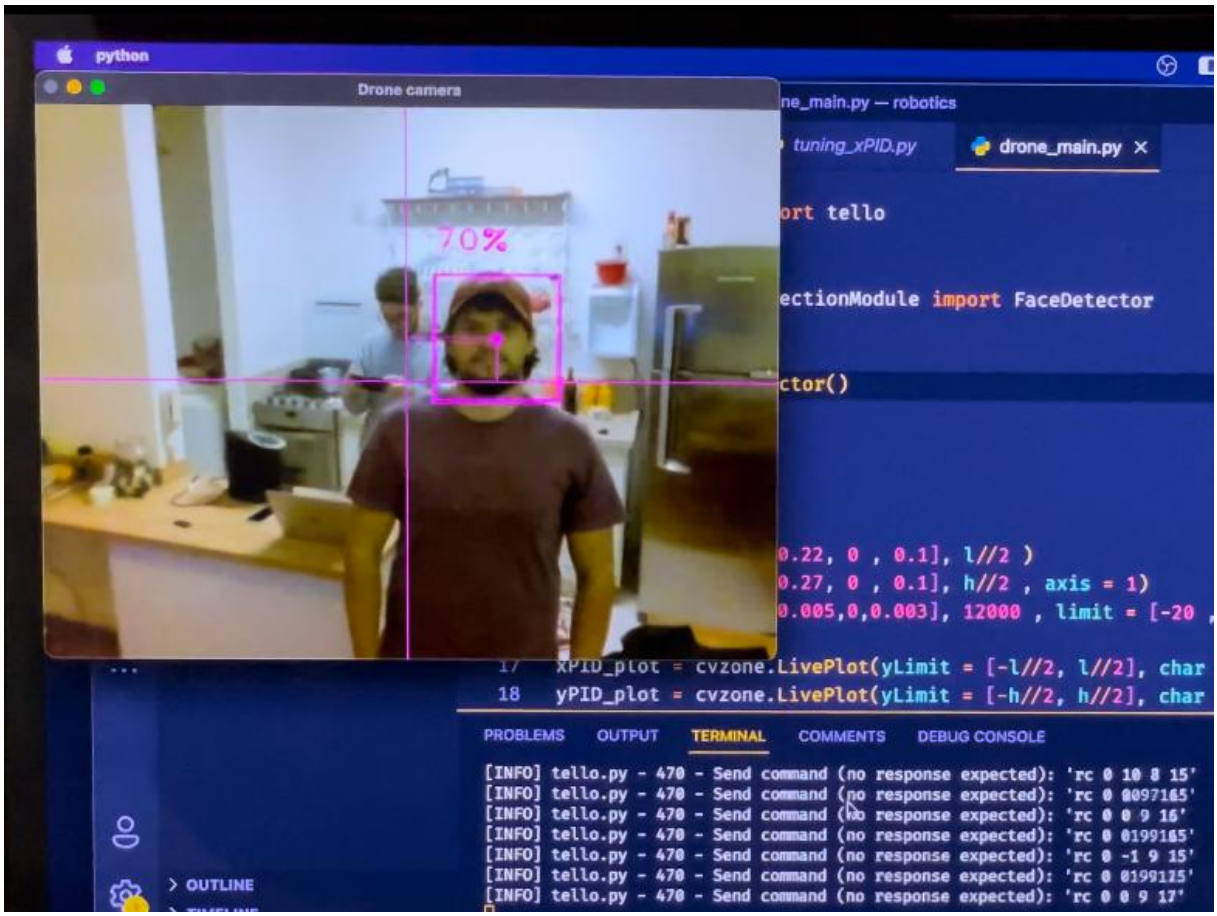
Figura 37 – Sistema sendo controlado pelo algoritmo de detecção.

Fonte: Rocha (2023a).

Entretanto, ao mover o rosto, ou seja, mudar a posição do rosto ao longo do tempo as coordenadas do objeto detectado, são alteradas isso para a malha do sistema de controle

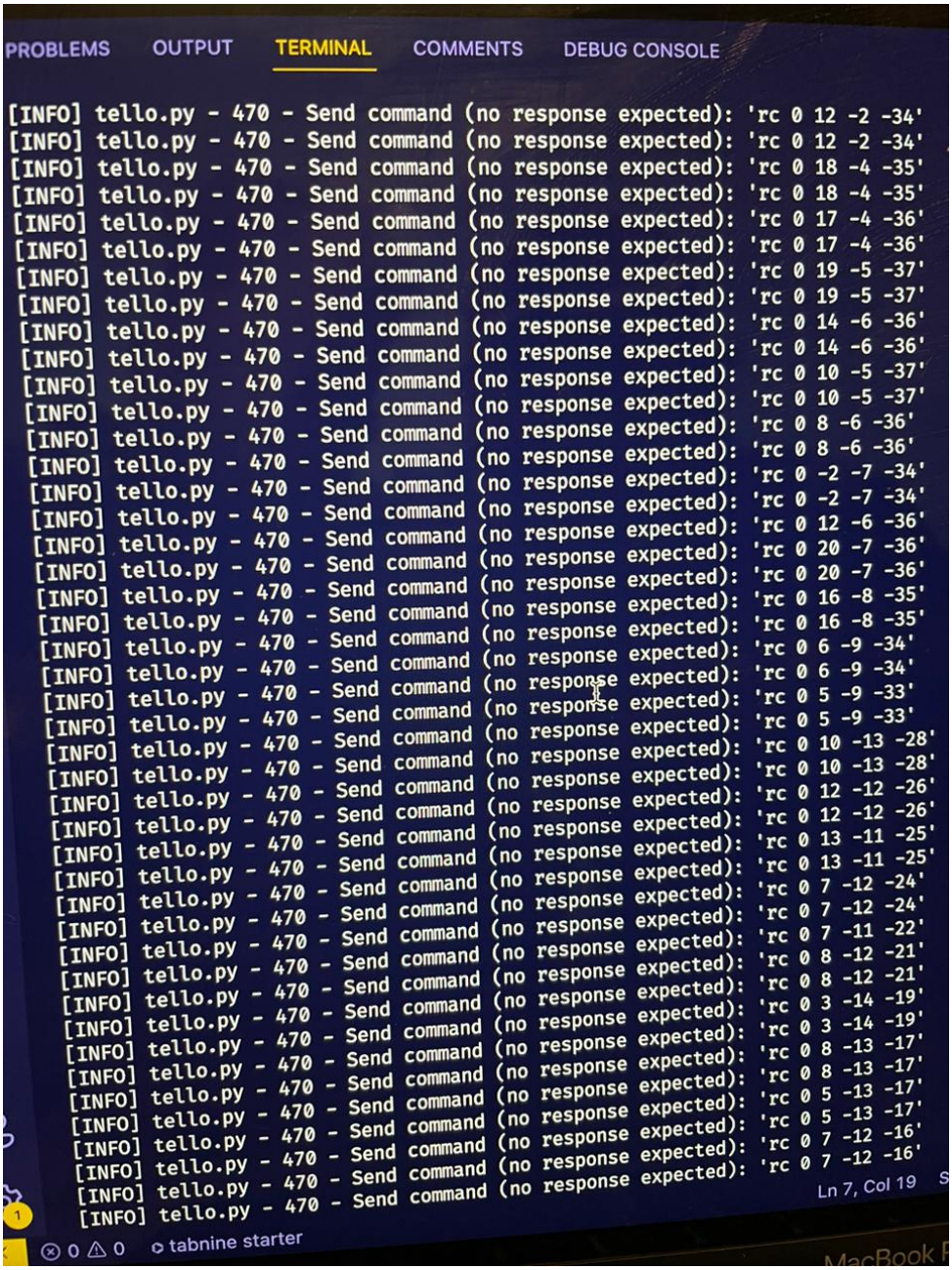
representa um erro de posição e portanto o drone precisa alterar sua posição para conseguir acompanhar o centroide da caixa delimitadora ou seja, as coordenadas $[c_x, c_y]$. Porém, internamente, o que ocorre é o envio de um sinal para os motores do drone para que o mesmo se reposicione para as novas coordenadas $[c_x, c_y]$ da caixa delimitadora da posição atual do rosto, veja as Figuras 38 e 40.

Figura 38 – Sistema sendo controlado pelo algoritmo de detecção.



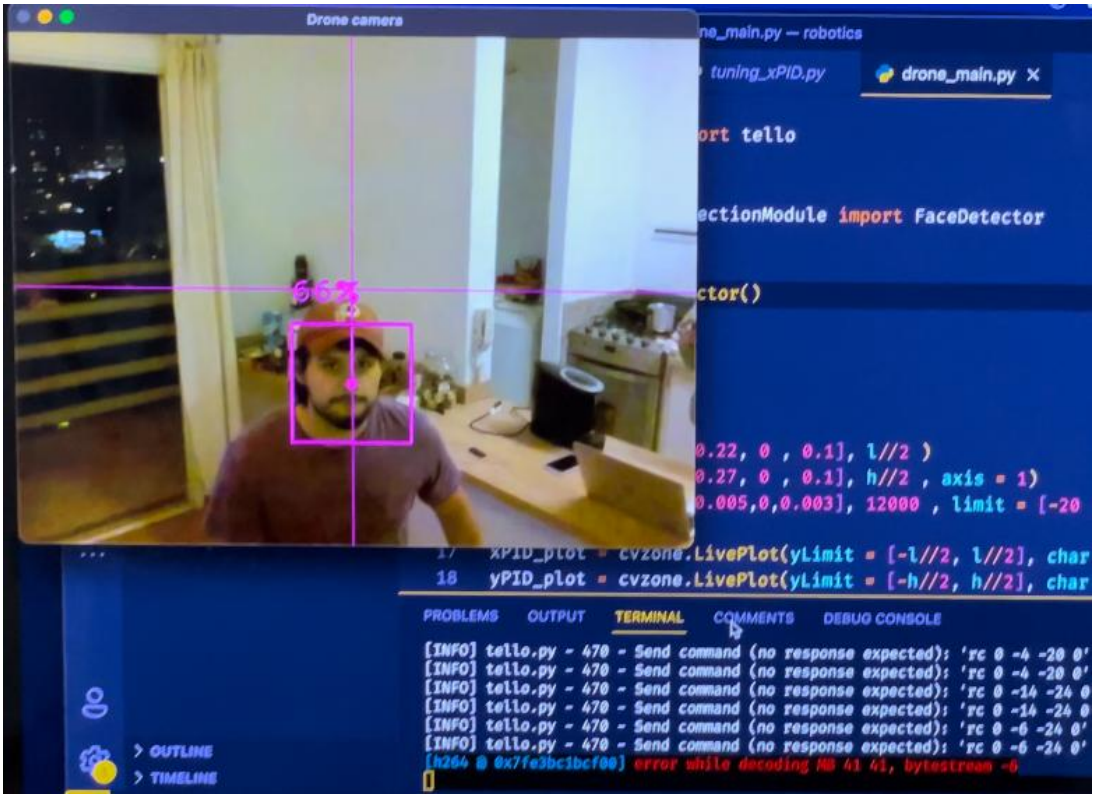
Fonte: Rocha (2023a).

Figura 39 – Sistema sendo controlado pelo algoritmo de detecção.



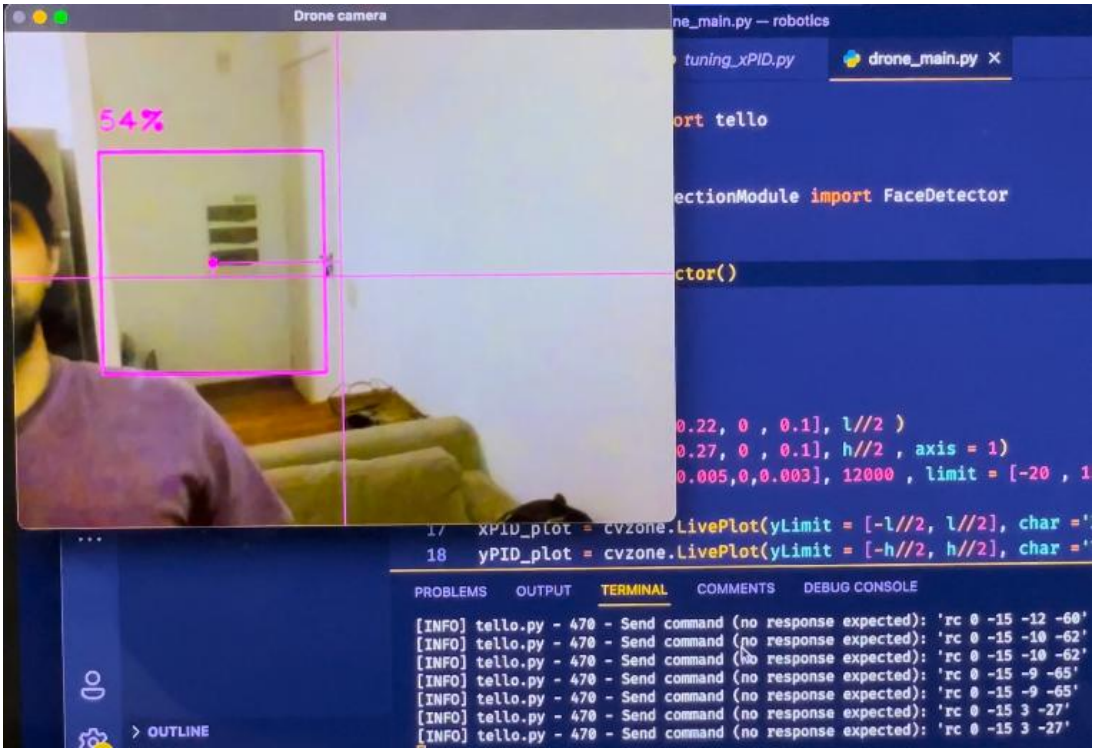
Fonte: Rocha (2023a).

Figura 40 – Algoritmo de detecção realizando o controle do drone.



Fonte: Rocha (2023a).

Figura 41 – Erros de detecção do algoritmo.



Fonte: Rocha (2023a).

Portanto, é possível ver que o sistema funciona de maneira adequada e finalmente podemos realizar o controle de posicionamento de um robô aéreo autônomo utilizando visão computacional cumprindo o objetivo. Porém como pode ser visto na Figura 41, em alguns momentos o algoritmo de detecção o objeto rosto na cena não é detectado corretamente por problemas do modelo de detecção do rosto, que serão melhorados em projetos futuros.

Além do desenvolvimento prático realizado, durante a execução de cada etapa, diversas dificuldades foram encontradas como, por exemplo, a falta de documentação. Mesmo com grande avanço no setor de robôs aéreos autônomos, ainda é trabalhoso encontrar documentação e bibliografias disponíveis acerca do tema de robótica aérea. Portanto, pensando na falta de documentação aliado as dificuldades enfrentadas em cada etapa de desenvolvimento do algoritmo de controle de posicionamento até a prototipação motivou o desenvolvimento de um repositório no Github intitulado "Teaching-aerial-vehicles" (ROCHA, 2023b). Sendo que, este repositório tem como o objetivo ser um ponto de partida para entusiastas e aspirantes a roboticistas que queiram trabalhar com a plataforma do Tello. Esse repositório conta com uma ótima documentação, contendo o passo a passo partindo do absoluto zero demonstrando desde a configuração do ambiente de desenvolvimento até o *deploy* na plataforma do Tello.

5 Conclusão

O objetivo deste trabalho consistiu em realizar o controle autônomo da posição relativa de um robô aéreo em relação a um alvo desejado (o rosto de uma pessoa), utilizando técnicas de visão computacional para a estimação de distância e o controle de posição com uma implementação de um controlador PID. O protótipo desenvolvido foi capaz de operar de maneira fluída e estável e foi apresentado um roteiro detalhado do planejamento, da configuração e do funcionamento, com a execução de um experimento prático.

O sucesso da execução da proposta fica evidenciado com os resultados apresentados, onde o drone conseguiu manter a distância relativa em um voo autônomo com relação a uma pessoa que estivesse em seu campo de visão, mesmo sob a movimentação livre nos três eixos e, até mesmo, sob pequenas rotações. Apesar de ter alcançado o desempenho almejado para esta proposta, existem alguns ajustes que ainda podem ser realizados, como, por exemplo: em cenas onde mais de um rosto está sendo capturado pela câmera, o algorítimo apresenta dificuldade de selecionar um único alvo para o rastreio; a mudança da estratégia de sintonia do controlador de posição pode ser melhorada, com a adoção de alguma estratégia sistematizada; o processo de medição da distância também pode ser refinado, utilizando alguma estratégia de filtragem para a suavização das medições; e a mudança na seleção do objeto rastreado, monitorando o corpo todo, ao invés de, apenas, o rosto. Além disso, para trabalho futuros, sugere-se: atualizações e melhorias do protótipo e algoritmo, por meio da implementação de novas técnicas de detecção de objetos como, por exemplo, a YoloV8; construir uma interface gráfica de interação com o usuário para facilitar a interação com o sistema; implementar a presente proposta em um drone com um sistema de sensores e processamento mais capacitado, como o uso de um sistema de posicionamento global e um sistema embarcado com maior capacidade de processamento; e projetar um sistema de rastreamento de múltiplos alvos.

Por fim, entendemos que o desenvolvimento deste trabalho demonstrou a possibilidade da configuração de um drone de relativo baixo custo para a operação autônoma e, pela descrição detalhada do desenvolvimento do projeto, abre diversas possibilidades de trabalhos futuros, utilizando o material exposto.

REFERÊNCIAS

ALBIERO, V.; BIASI, H. H. de. Drone autônomo guiado através de templates utilizando visão computacional. *Anais SULCOMP*, v. 8, 2016. Citado na página 11.

ÅSTRÖM, K. J.; HÄGGLUND, T. The future of pid control. *Control engineering practice*, Elsevier, v. 9, n. 11, p. 1163–1175, 2001. Citado na página 20.

BENNETT, S. Development of the pid controller. *IEEE Control Systems Magazine*, IEEE, v. 13, n. 6, p. 58–62, 1993. Citado na página 19.

BENNETT, S. The past of pid controllers, preprints of ifac workshop on digital control. *Past, Present and Future of PID Control, Terrassa, Spain*, p. 13, 2000. Citado na página 19.

BORASE, R. P. et al. A review of PID control, tuning methods and applications. *International Journal of Dynamics and Control*, Springer Science and Business Media LLC, v. 9, n. 2, p. 818–827, jul. 2020. Disponível em: <https://doi.org/10.1007/s40435-020-00665-4>. Citado na página 20.

BRADSKI, G. R.; KAEHLER, A. *Learning OpenCV*. Sebastopol, CA: O'Reilly Media, 2008. Citado na página 14.

FISCHLER, M. A.; ELSCHLAGER, R. A. The representation and matching of pictorial structures. *IEEE Transactions on computers*, IEEE, v. 100, n. 1, p. 67–92, 1973. Citado na página 17.

FUENTES, D. 2019. Disponível em: <https://github.com/damiafuentes/DJITelloPy>. Citado na página 34.

HEISELE, B.; SERRE, T.; POGGIO, T. A component-based framework for face detection and identification. *International Journal of Computer Vision*, Springer, v. 74, p. 167–181, 2007. Citado na página 17.

JÄHNE, B.; HAUSSECKER, H. *Computer vision and applications: a guide for students and practitioners*. [S.l.]: Academic Press, Inc., 2000. Citado na página 15.

KARN, A. et al. Artificial intelligence in computer vision. *International Journal of Engineering Applied Sciences and Technology*, v. 6, n. 1, p. 2455–2143, 2021. Citado na página 14.

LEIBE, B. et al. Dynamic 3d scene analysis from a moving vehicle. In: IEEE. *2007 IEEE conference on computer vision and pattern recognition*. [S.l.], 2007. p. 1–8. Citado na página 17.

LEUNG, T. K.; BURL, M. C.; PERONA, P. Finding faces in cluttered scenes using random labeled graph matching. In: IEEE. *Proceedings of IEEE International Conference on Computer Vision*. [S.l.], 1995. p. 637–644. Citado na página 17.

MEZURA-MONTES, E.; COELLO, C. A. C. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation*, v. 1, n. 4, p. 173–194, 2011. ISSN 2210-6502. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210650211000538>. Citado 2 vezes nas páginas 20 e 21.

MICHELSON, R. International aerial robotics competition-the world's smallest intelligent flying machines, 13th rpvs. In: *UAVs International Conference, UK*. [S.l.: s.n.], 1998. p. 31–1. Citado na página 11.

MOGHADDAM, B.; PENTLAND, A. Probabilistic visual learning for object representation. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 19, n. 7, p. 696–710, 1997. Citado na página 17.

REHEM, A.; TRINDADE, F. H. Técnicas de visão computacional para rastreamento de olhar em vídeos. *Publicado em*, v. 3, n. 02, 2009. Citado 2 vezes nas páginas 14 e 15.

ROCHA, J. V. M. Imagens ilustrativas do trabalho controle de posicionamento de robô aéreo autônomo utilizando visão computacional. 2023. Citado 16 vezes nas páginas 28, 32, 35, 37, 38, 39, 40, 41, 43, 44, 45, 46, 47, 48, 49 e 50.

ROCHA, J. V. M. *Teaching UAV Aerial Robotics*. 2023. Disponível em: <https://github.com/joaomedeirosr/teaching-uav-aerial-robotics>. Citado na página 51.

ROWLEY, H. A.; BALUJA, S.; KANADE, T. Rotation invariant neural network-based face detection. In: IEEE. *Proceedings. 1998 IEEE computer society conference on computer vision and pattern recognition (Cat. No. 98CB36231)*. [S.l.], 1998. p. 38–44. Citado na página 16.

RYZE-ROBOTICS. *Tello - User Manual*. 2018. Disponível em: <https://www.droneweb.com.br/downloads/Manual_DJI_Tello_Ryze_BR_1.0.pdf>. Citado 12 vezes nas páginas 22, 23, 24, 25, 26, 27, 29, 30, 31, 32, 33 e 36.

SCHNEIDERMAN, H.; KANADE, T. Object detection using the statistics of parts. *International Journal of Computer Vision*, v. 56, n. 3, 2004. Citado 2 vezes nas páginas 17 e 18.

SMEULDERS, A. W. et al. Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 36, n. 7, p. 1442–1468, 2013. Citado na página 15.

SUNG, K.-K.; POGGIO, T. Example-based learning for view-based human face detection. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 20, n. 1, p. 39–51, 1998. Citado na página 17.

TURK, M.; PETLAND, A. Eigenfaces for fisherfaces. *Vision and Modeling Group. The Media Laboratory. Massachusetts Institute of Technology*, 1991. Citado na página 15.

WANG, Q.-G. et al. Pid tuning for improved performance. *IEEE Transactions on Control Systems Technology*, v. 7, n. 4, p. 457–465, 1999. Disponível em: <https://doi.org/10.1109/87.772161>. Citado na página 20.

YANG, M.-H.; KRIEGMAN, D. J.; AHUJA, N. Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 24, n. 1, p. 34–58, 2002. Citado na página 17.

YILMAZ, A.; JAVED, O.; SHAH, M. Object tracking: A survey. *Acm computing surveys (CSUR)*, Acm New York, NY, USA, v. 38, n. 4, p. 13–es, 2006. Citado na página 15.