DOI: 10.48011/asba.v2i1.1701

Desenvolvimento de um drone autônomo para tarefas de entrega de carga ⋆

Thiago L. Rocha * **Adriano M. C. Rezende** *
Victor R. F. Miranda * **Héctor Azpúrua** †
Gustavo M. Freitas *

* *Escola de Engenharia, Universidade Federal de Minas Gerais (thiagolages,adrianomcr,victormrfm, gustavomfreitas)@ufmg.br*
† *Dept. de Ciência da Computação, Universidade Federal de Minas Gerais (hector.azpurua@dcc.ufmg.br)*

Abstract: Drones are increasingly being used in logistics, due to its speed and practicality. This paper presents a methodology to make autonomous drone delivery feasible. Two path planning strategies are presented and the quadrotor's localization is done through GPS, IMU and barometer during flight. In the landing phase, the pose estimated by a camera with an ArUco marker or by Ultrawide Band devices are merged to GPS data using an Extended Kalman Filter. The robot's control is done by a Vector Fields technique such that the drone converges to the desired curve. The path planning strategies are validated through simulation and real experiments, and the different localization techniques are compared. Preliminary results show the viability of using drones to deliver packages.

Resumo: Drones são cada vez mais usados em aplicações de logística devido à sua rapidez e praticidade. Este artigo apresenta uma metodologia para a viabilização de entregas utilizando drones autônomos. Duas estratégias de planejamento de caminho são apresentadas e a localização do quadrirrotor é feita através do GPS, IMU e barômetro durante o voo. Na fase de pouso, a pose estimada por uma câmera junto a um marcador ArUco ou por dispositivos Ultrawide Band são fundidos ao GPS utilizando um Filtro de Kalman Estendido. O controle do robô é feito por uma técnica de Campos Vetoriais, de maneira que o drone convirja para a curva desejada. As estratégias de planejamento de caminho são validadas através de simulações e experimentos reais, e as diferentes técnicas de localização são comparadas. Os resultados preliminares mostram a viabilidade da utilização de drones para entregas de pacotes.

Keywords: Autonomous Drones; Path Planning; Vector Field Control; Autonomous Delivery.

Palavras-chaves: Drones Autônomos; Planejamento de caminho; Controle por Campos Vetoriais; Entrega Autônoma.

1 INTRODUÇÃO

Robôs autônomos são objeto de estudo há muito tempo, e a crescente demanda por soluções automatizadas de problemas do mundo real tem acelerado pesquisas na área. Tais problemas podem envolver tarefas domésticas (aspiradores de pó e cortadores de grama), militares (salvamento, patrulha e ataques), de segurança (exploração e resgate), industriais (produção e logística), dentre outros.

Nos últimos anos, o mercado de compras online vem crescendo cada vez mais e, desde então, empresas como *Amazon*, *UPS* [1] e *Alphabet* [2] buscam tornar realidade entregas autônomas com drones. Segundo Schneider (2020), a *FlightForward*, filial da *UPS* responsável pelos voos de drones, já conseguiu a certificação de transportadora aérea, o que a permite fazer entregas de pequenos pacotes com drones. Além disso, a *Wing*, uma divisão da *Alphabet*, lançou o primeiro serviço comercial de pequenas entregas dos Estados Unidos, na cidade de *Christiansburg*, Virgínia. Estas conquistas foram alcançadas no final de 2019 e mostram que o mercado de *delivery* utilizando drones está certamente ganhando espaço.

Dentro do mercado de compras online brasileiro, o *delivery* de comida movimentou R$ 15 bilhões em 2019 (ABRASEL, 2020) [3], o que representa um aumento de 20% em relação ao ano anterior. Dessa forma, os serviços de entrega vem enfrentando uma alta demanda de pedidos, e por vezes não conseguem manter um baixo tempo de entrega. Assim, é necessário que novas e inovadoras formas de *delivery* sejam propostas e validadas, a fim de torná-las mais rápidas, confiáveis e seguras.

Neste cenário, a entrega autônoma se apresenta como uma alternativa muito conveniente, principalmente em períodos de isolamento social como o vivido por vários países em 2020 em função do coronavírus. Em situações como essa,

⋆ Este artigo teve o apoio da Universidade Federal de Minas Gerais (UFMG), do Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq), e do Instituto Tecnológico Vale (ITV).

[1] *United Parcel Service* - Serviço de correios dos Estados Unidos

[2] Empresa-mãe da Google, Inc.

[3] Assossiação Brasileira de Bares e Restaurantes

o contato físico entre pessoas deve ser reduzido o máximo possível, principalmente com pessoas dos grupos de risco. Desta forma, uma entrega feita por um quadrirrotor autônomo impediria qualquer tipo de contato físico entre o entregador e o cliente, mantendo assim as recomendações da Organização Mundial da Saúde (OMS) e impedindo a proliferação do vírus.

Impulsionados pela demanda envolvendo o transporte aéreo autônomo de cargas em curtas distâncias, diversas pesquisas vem sendo desenvolvidas abordando diferentes estratégias para o transporte com carga acoplada e outros problemas comuns nesse tipo de tarefa, como localização, controle e planejamento de trajetória (Villa et al., 2019).

Este artigo apresenta uma aplicação real de algumas técnicas previamente utilizadas para uma competição de drones autônomos pela equipe XQuad-UFMG (Rezende et al., 2019). O objetivo é que os conhecimentos desenvolvidos na universidade sejam também utilizados para atender às demandas da sociedade.

Neste contexto, este artigo apresenta uma metodologia para a viabilização de entregas feitas por um drone autônomo. Após a geração de um caminho entre dois pontos de interesse, a localização do drone é obtida através do seu GPS (*Global Positioning System*), IMU (*Inertial Measuremnt Unit*) e barômetro. Uma vez obtida sua localização, o controle do quadrirrotor é realizado através de uma técnica de campos vetoriais, que faz com que o drone siga o caminho desejado. Para melhorar a precisão de pouso, dispositivos *Ultra-wideband (UWB)* ou uma câmera junto a um marcador ArUco[4] foram utilizados. As simulações e experimentos reais mostram que as estratégias utilizadas são viáveis para realização de entregas autônomas de pacotes.

O artigo está organizado da seguinte maneira: a Seção 2 descreve o problema da entrega com drones autônomos, e a Seção 3 apresenta as metodologias utilizadas para realizar tal tarefa. Os resultados obtidos são discutidos na Seção 4. As conclusões do trabalho são apresentadas na Seção 5, junto com trabalhos futuros.

2 ENTREGA AUTÔNOMA COM DRONES

Esta Seção apresenta em detalhes o problema de entrega autônoma com drones e especifica os equipamentos, software e simulador utilizados na metodologia.

2.1 Descrição do Problema

O problema abordado envolve o transporte de uma carga de pequeno porte entre dois pontos de interesse utilizando um drone autônomo, ou seja, sem que este receba comandos de um piloto humano. O caminho escolhido deve ser curto, tendo em vista a baixa autonomia deste tipo de robô, e livre de obstáculos, a fim de evitar colisões.

Além disso, é importante que o drone possa transportar objetos relativamente frágeis sem que haja movimentos bruscos, como é feito em (Raffo and de Almeida, 2016) com uma carga suspensa. Esta abordagem difere da utilizada aqui, visto que o pacote está acoplado ao corpo do drone e, portanto, não pode oscilar livremente.

[4] Marcador de Realidade Aumentada - `http://www.uco.es/investiga/grupos/ava/node/26`

Figura 1. Drone utilizado para a entrega autônoma e seus componentes.

A princípio, regiões residenciais serão o principal local de pouso em tarefas de entrega; portanto é importante que o drone tenha a capacidade de aterrissar em locais com área restrita. Dessa forma, os testes feitos consideram uma plataforma de dimensões 1x1m.

2.2 Recursos Utilizados

2.2.1 Hardware O drone utilizado é o DJI Matrice 100, ilustrado na Figura 1, por permitir a implementação de códigos próprios para controle do quadrirrotor. Todos os sensores que acompanham o drone por padrão são utilizados, sendo eles o GPS, IMU, magnetômetro de três eixos e um barômetro. Estes são responsáveis por ajudar na estimação da posição do drone em coordenadas globais (latitude e longitude), sua orientação e altura. A IMU é capaz de medir a aceleração linear do drone nos eixos **x**, **y** e **z** utilizando acelerômetros, bem como as velocidades angulares relativas a cada um deles, utilizando giroscópios. Junto a esses sensores, a placa de controle de voo do próprio drone também é utilizada para fazer a interface entre os sinais de controle enviados e a atuação nos motores *brushless*.

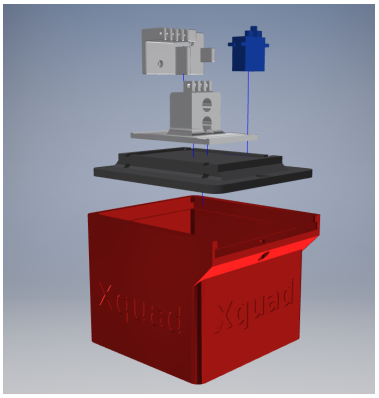

Figura 2. Mecanismo de acoplamento utilizando um servo motor, junto à caixa utilizada para transporte de carga.

Figura 3. Dispositivo de acoplamento e liberação da carga presente no drone.

Além da placa controladora de voo já disponível no drone, um kit de desenvolvimento Jetson Nano [5] é usado para o processamento de dados, planejamento de caminho e controle do quadrirrotor.

Um modelo 3D da caixa utilizada na entrega foi desenvolvido, como mostra a Figura 2. Seu mecanismo de acoplamento utiliza um servo motor, conforme ilustrado na Figura 3. Um Arduino Nano é conectado à placa Jetson para fazer o controle deste servo motor, colocando-o em posição de acoplameno ou desacoplamento da caixa no drone. O esquemático da Figura 4 ilustra as conexões entre os componentes efetivamente utilizados pelo drone durante os experimentos reais.

Considerando que o local de pouso utilizado nos experimentos possui 1x1m e que a precisão horizontal na localização do drone é de aproximadamente 2m, é necessário o uso de sensores adicionais para auxiliá-lo a fazer um pouso mais seguro e preciso. Em função da impossibilidade de realização de novos testes com o drone autônomo, a aplicação dos seguintes sensores foi verificada apenas em abiente simulado: (i) uma *RaspberryPi Camera v2.0* apontada para baixo juntamente a um marcador ArUco na plataforma ou (ii) dispositivos *Ultra-wideband (UWB)* ancorados no local de pouso, e um dispositivo do mesmo tipo acoplado ao drone. De ambas as formas, é possível obter informações adicionais da posição do drone em relação à plataforma durante o pouso. A validação experimental do uso destes sensores será feita assim que o drone estiver disponível para novos testes.

2.2.2 Software O Sistema Operacional utilizado é o Ubuntu 18.04 juntamente ao ROS (*Robot Operating System*). O *package* do ROS disponibilizado pela DJI, denominado Onboard-SDK-ROS [6], é utilizado para estabelecer a comunicação entre o drone e a Jetson Nano, responsável pelo processamento de dados. O pacote permite enviar comandos de velocidade ao sistema de controle embarcado e fornece dados dos sensores presentes no drone, como GPS, IMU e barômetro, além da estimativa de posição e orientação do drone em tempo real.

Para o uso de sensores adicionais como o ArUco e os dispositivos UWB, é necessário computar as informações fornecidas por cada um. Para a identificação e estimação de pose do ArUco, algoritmos específicos da biblioteca OpenCV [7] são utilizados. No caso dos dispositivos UWB, a diferença entre o tempo de chegada (TDoA) do sinal de cada um deles é utilizado para estimar a distância do drone aos dispositivos ancorados no local de pouso.

[5] Jetson Nano - `https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/`

[6] Onboard-SDK-ROS - `https://github.com/dji-sdk/Onboard-SDK-ROS`

[7] Biblioteca de Visão Computacional - `https://opencv.org/`

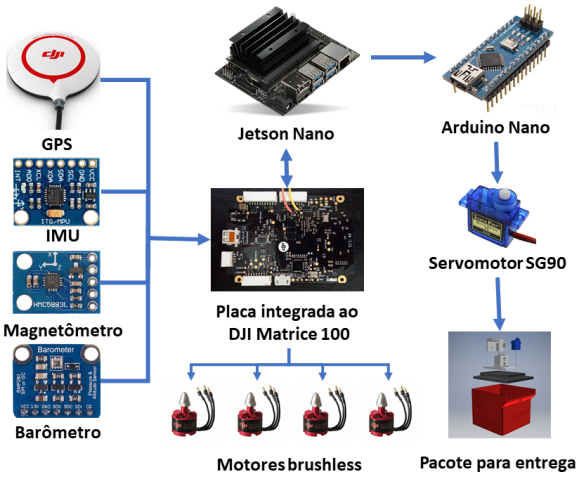

Figura 4. Representação em alto nível das ligações dos componentes fisicamente utilizados no drone.

2.2.3 Simulação O simulador *CoppeliaSim* [8], sucessor do V-REP, é utilizado por ser bastante conhecido, largamente utilizado na área de robótica, e por apresentar compatibilidade com o ROS. Além de emular a dinâmica do drone, o simulador conta com alguns sensores disponíveis por padrão, como GPS e IMU, e ainda permite que erros associados a esses sensores sejam incluídos, o que é usado para testar a robustez do método de controle proposto.

Sensores extras também foram adicionados, tanto os embarcados no drone, quanto os instalados no ambiente. Assim, foi possível embarcar uma câmera e um dispositivo UWB ao drone, e manter o marcador ArUco e outros dispositivos UWB na plataforma de pouso da simulação.

Os mesmos algoritmos de Visão Computacional utilizados para identificação do marcador são utilizados com as imagens geradas pela câmera simulada. Pelo fato de não ser possível emular diretamente a comunicação via rádio-frequência entre os dispositivos UWB, o cálculo da posição relativa do dispositivo de referência em relação às âncoras foi obtido através de transformações entre sistemas de coordenadas adicionando-se um ruído branco ± 1m de amplitude.

3 METODOLOGIAS EMPREGADAS

A solução do problema é dividido em três tarefas distintas: planejamento de caminho, localização e controle. O desenvolvimento da solução foi feito de maneira a atender aos requisitos do problema, como rapidez na entrega, porém evitando movimentos bruscos e colisões.

3.1 Planejamento de caminho

A estratégia de planejamento de caminho é feita de maneira a simplificar entregas autônomas com o drone. A sua definição começa por entender que a localização geográfica é inicialmente fornecida através da altitude e dos ângulos de latitude e longitude. Para os propósitos dos testes, as distâncias a serem percorridas são pequenas a ponto de um modelo de terra plana poder ser considerado. Assim, os ângulos de latitude e longitude são transformados em medidas de distância. A posição do drone é, então, inicialmente representada no referencial da terra, $\mathcal{F}_E$.

[8] CoppeliaSim - `https://www.coppeliarobotics.com/`

Considere que o ponto de partida do drone é $\mathbf{p}_s \in \mathbb{R}^3$ e que a carga deve ser entregue no ponto $\mathbf{p}_f \in \mathbb{R}^3$. Sem perda de generalidade, é possível considerar um sistema de coordenadas inercial $\mathcal{F}_I$, que respeita duas condições:

(1) O ponto de partida é a origem, ou seja, $\mathbf{p}_s = \mathbf{0}$;
(2) O eixo $\mathbf{x}$ do sistema de coordenadas $\mathcal{F}_I$ está no plano horizontal, apontando na direção do ponto final, ou seja, $\hat{x} \parallel \Pi_{xy}(\mathbf{p}_f - \mathbf{p}_s)$, onde $\Pi_{xy}(\cdot)$ representa a projeção no plano $\mathbf{xy}$.

O referencial inercial $\mathcal{F}_I$ é facilmente obtido através de duas operações: uma translação em relação a $\mathcal{F}_E$, a fim de satisfazer a consideração (1); e uma simples rotação em relação ao eixo $\mathbf{z}$ para satisfazer a consideração (2).

Duas estratégias de planejamento de caminho são apresentadas neste artigo. A primeira é mais simples e foi testada em experimentos reais; a segunda é mais elaborada e, em função da indisponibilidade do drone para a realização de novos experimentos, foi testada apenas em simulações.

A primeira estratégia consiste em dividir o caminho principal em três sub-caminhos: (i) uma reta vertical para a subida do drone até uma certa altura h; (ii) uma reta horizontal até a posição da plataforma de pouso; (iii) outra reta vertical para a descida até a plataforma. Uma máquina de estados inicializa no trecho (i) e evolui através dos trechos (ii) e (iii). A desvantagem desta abordagem é a existência de descontinuidades de primeira ordem no caminho, representadas por "quinas vivas" associadas às mudanças de estado.

Uma possível melhoria consiste em computar um caminho de referência passando pelos pontos $\mathbf{p}_s$ e $\mathbf{p}_f$ de forma contínua e suave. A segunda estratégia é então desenvolvida de maneira a criar 5 sub-caminhos: (i) reta vertical ascendente; (ii) arco de círculo; (iii) reta horizontal em direção à plataforma; (iv) arco de círculo; (v) reta vertical descendente. Ao invés de definir uma máquina de estados como no caso anterior, o espaço foi dividido em 5 setores $\mathcal{S}_i,\ i = 1, 2, 3, 4, 5$. Cada um deles possui um sub-caminho associado que será a referência para o drone, dependendo de sua localização. A definição dos setores é apresentada a seguir:

$$\begin{aligned}
\mathcal{S}_1 &= \left\{(x, y, z) \in \mathbb{R}^3 : z \leq h - r,\ x \leq d/2\right\}, \\
\mathcal{S}_2 &= \left\{(x, y, z) \in \mathbb{R}^3 : z > h - r,\ x < r\right\}, \\
\mathcal{S}_3 &= \left\{(x, y, z) \in \mathbb{R}^3 : z > h - r,\ r \leq x \leq d - r\right\}, \\
\mathcal{S}_4 &= \left\{(x, y, z) \in \mathbb{R}^3 : z > h - r,\ x > d - r\right\}, \\
\mathcal{S}_5 &= \left\{(x, y, z) \in \mathbb{R}^3 : z \leq h - r,\ x > d/2\right\}. \qquad (1)
\end{aligned}$$

onde h é a altura em que o drone irá trafegar (em relação à $\mathcal{F}_I$), r é o raio de curvatura dos círculos de transição e d é a separação horizontal entre $\mathbf{p}_s$ e $\mathbf{p}_f$. Os pontos de partida e chegada, os setores, bem como as variáveis aqui definidas são ilustrados na Figura 5.

3.2 Localização

O pacote Onboard-SDK-ROS fornece uma estimativa georeferenciada de orientação e posição global do drone. Estas informações são geradas a partir da fusão sensorial do GPS, IMU e barômetro disponíveis, que resultam em uma precisão de aproximadamente 2 m. Tal estimativa é suficientemente boa quando em voo de cruzeiro e, portanto,

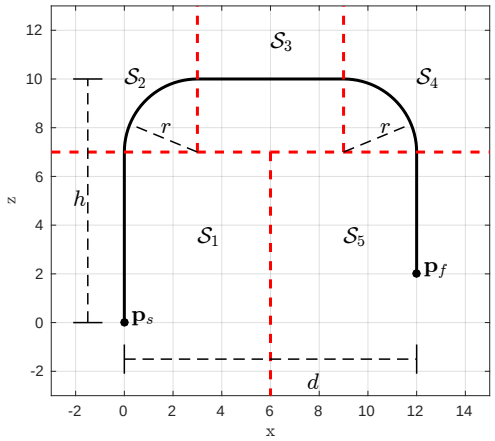

Figura 5. Setores $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, $\mathcal{S}_4$ e $\mathcal{S}_5$ (equação (1)).

foi utilizada durante todo o trajeto. No entanto, esta é insuficiente quando é preciso aterrissar em uma plataforma de 1x1m como a utilizada, cujas dimensões são inferiores à precisão da estimativa de posição. Além disso, caso a plataforma de pouso tenha sido movida, o GPS não será capaz de fazer a aterrissagem no local certo.

Por esses motivos, é necessária a obtenção de uma informação adicional que permita melhorar a estimativa de posição do drone em relação ao local de pouso. Este artigo apresenta duas abordagens para esta localização: (i) utilização de um marcador ArUco; (ii) utilização de dispositivos de comunicação *Ultra-wideband* (UWB) na área de pouso.

3.2.1 ArUco Para o uso do ArUco, um marcador de tamanho 0.8x0.8m foi impresso e colocado em cima da plataforma de pouso. Uma câmera acoplada ao drone, apontada para baixo, foi utilizada para visualizar o marcador durante o pouso. Os ArUcos possuem características que facilitam sua identificação na imagem, como bordas bem definidas e alto contraste de cores. Além disso, os marcadores são criados de maneira a não gerar ambiguidades em sua orientação.

Assim, algoritmos da biblioteca OpenCV específicos para identificação de ArUcos foram utilizados, bem como para estimar a pose relativa da câmera em relação ao marcador. Esta última etapa é feita através da solução do problema de *PnP (Perspective-n-Point)*, que se propõe a estimar a pose tridimensional de uma câmera calibrada dado um conjunto de pontos 3D e suas correspondentes projeções 2D no plano da câmera.

Sabendo o tamanho real do marcador e os parâmetros de calibração da câmera, é possível encontrar a pose que minimiza os erros de projeção dos pontos no plano da câmera. Estes pontos devem ser distinguíveis entre si, e neste caso, as quinas do ArUco são utilizadas para solucionar o problema de PnP. Como dito anteriormente, é possível determinar a orientação do marcador, o que permite diferenciar cada uma de suas quatro quinas. Assim, algoritmos que solucionam o problema de PnP (Lepetit et al., 2009; Hesch and Roumeliotis, 2011) podem ser utilizados.

É possível estimar a pose do drone em relação ao ArUco sabendo-se as poses relativas da câmera em relação ao marcador e do drone em relação à câmera. Esta informação, juntamente à posição e orientação esperada do

ArUco, é usada para estimar a localização do drone com respeito ao referencial inercial $\mathcal{F}_{\mathcal{I}}$. Na Seção 3.2.3 esta estimativa é usada para corrigir a localização do drone e, consequentemente, aumentar a precisão durante o pouso.

3.2.2 Ultra Wideband (UWB) Dispositivos que usam a tecnologia *ultra wideband* são comumente utilizados como um método de localização. Essa tecnologia utiliza ondas de rádio com uma largura de banda superior a $500MHz$, o que reduz a perda por obstruções e consequentemente aumenta a segurança das transmissões (Sahinoglu, 2008).

A localização por UWB pode ser utilizada em ambientes externos e internos, com uma precisão de até $20cm$ (segundo alguns fabricantes). Neste método são utilizados algoritmos de multilateração para estimar a posição x_T, y_T e z_T de um dispositivo móvel (chamado de *tag*) em relação a um referencial fixo, onde estão localizados outros dispositivos (chamados *anchors*, ou âncoras).

Neste artigo, são utilizados dispositivos UWB do modelo Decawave DWM1001 para estimar a posição do drone com maior precisão ao se aproximar da plataforma de pouso. São necessários um mínimo de cinco dispositivos para funcionamento do algoritmo, uma *tag* embarcada no drone e quatro âncoras em posições conhecidas, sendo uma escolhida como âncora base.

O cálculo da posição é feito com base na distância da *tag* em relação às âncoras, que é obtida através da diferença entre o tempo de chegada (TDoA) do sinal transmitido, multiplicado pela velocidade de propagação do sinal (velocidade da luz). Assim como apresentado em (Sayed et al., 2005):

$$d_{i1} = (t_i - t_1)c, \; i = 2, ..., N, \tag{2}$$

onde d_{i1} é a distância entre a âncora i e a âncora base, t_i é o instante de tempo em que o sinal enviado pela *tag* chega à âncora i e t_1 é o instante de tempo que este sinal chega à âncora base. A velocidade da luz é c e o número de âncoras é N. As distâncias em (2) resultam em uma região de interseção que representa a posição da *tag*, e pode ser obtida resolvendo o seguinte conjunto de equações:

$$\begin{aligned} (d_{21}^2 + d_1^2)^2 &= (x_2 + y_2 + z_2)^2 - 2J_2 + d_1^2, \\ (d_{31}^2 + d_1^2)^2 &= (x_3 + y_3 + z_3)^2 - 2J_3 + d_1^2, \\ &\vdots \\ (d_{N1}^2 + d_1^2)^2 &= (x_N + y_N + z_N)^2 - 2J_N + d_1^2, \end{aligned} \tag{3}$$

onde $J_i = (x_i x_T + y_i y_T + z_i z_T)$ e d_1 obtido através da equação abaixo, considerando t^0 o instante em que o sinal é enviado pela *tag* e t_1 o instante de tempo que este sinal chega à âncora base:

$$d_1 = (t_1 - t^0)c. \tag{4}$$

Para utilização dos dispositivos Decawave, uma distância máxima de 10m deve ser mantido entre as âncoras e a tag. Esta metodologia não estima orientação, diferente do método pela identificação visual de um marcador ArUco.

3.2.3 Fusão Sensorial Uma forma de aprimorar a localização e estimar de forma mais precisa os estados de posição, orientação e velocidade do drone é utilizar informações de diversos sensores. Os métodos de fusão sensorial utilizam dados dos dispositivos de localização para obter uma informação mais precisa dos estados de interesse. O software presente no DJI Matrice 100 utiliza a informação do GPS, da IMU e do barômetro para fornecer a pose e velocidades lineares e angulares do drone.

Um dos métodos de fusão mais comuns é o filtro de Kalman Estendido (EKF) (Thrun et al., 2000). Com o objetivo de realizar um pouso mais preciso, os dados de localização fornecidos pelo algoritmo PnP durante a detecção do ArUco ou pelo Decawave foram fundidos com as informações do software da DJI. Como os dados dos sensores transmitem informação referente a estados coincidentes, um *bias* foi considerado na posição informada através da detecção do ArUco ou pelo Decawave.

É possível dividir o método estendido de fusão e filtragem de Kalman em duas etapas: Predição e Correção. Por simplicidade, as equações do filtro serão apresentadas com a notação $b \leftarrow a$ indicando que b é atualizado com o valor de a.

A etapa de predição no EKF discreto envolve o vetor de estados $\bar{\mathbf{x}}$ e a matriz de covariância P:

$$\bar{\mathbf{x}} \leftarrow f(\bar{\mathbf{x}}, \mathbf{u}, \Delta t), \tag{5}$$

$$P \leftarrow FPF^T + GQG^T, \tag{6}$$

onde f representa o modelo de propagação dos estados, que envolve a estimativa atual $\bar{\mathbf{x}}$, o vetor de entradas $\mathbf{u}$ e o passo de tempo Δt. A matriz $F \equiv F(\bar{\mathbf{x}}, \mathbf{u}, \Delta t)$ é a derivada parcial de f em relação à $\bar{\mathbf{x}}$, a matriz G é a derivada parcial de f em relação a $\mathbf{u}$ e a matriz Q é a matriz de covariância associada ao vetor de entradas $\mathbf{u}$.

A etapa de correção é definida da seguinte forma:

$$\bar{\mathbf{x}} \leftarrow \bar{\mathbf{x}} + K\left(\mathbf{w} - h(\bar{\mathbf{x}})\right), \tag{7}$$

$$P \leftarrow (I - KH)\, P, \tag{8}$$

onde $\mathbf{w}$ é o vetor de medição e $h(\bar{\mathbf{x}})$ o modelo de medição, que representa o valor esperado de $\mathbf{w}$ dado o estado estimado atual $\bar{\mathbf{x}}$. A matriz H é a Jacobiana de $h(\bar{\mathbf{x}})$ e I uma matriz identidade. A matriz K representa o ganho de Kalman, dado por:

$$K = PH^T \left(HPH^T + R\right)^{-1}, \tag{9}$$

onde R é a covariância dos dados de medição $\mathbf{w}$.

A estratégia utilizada define como vetor de entrada as velocidades lineares providas pelo sistema do drone DJI, sendo $\mathbf{u} = [v_x \; v_y \; v_z]$. As medições de posição são utilizadas na etapa de correção, incluindo os dados fornecidos pelo GPS DJI, Decawave ou através da identificação do ArUco.

Em geral, os dados coletados pelo GPS, em comparação com os dados do ArUco e do UWB, possuem um erro desconhecido de média não nula. Portanto, a utilização direta dessas medições na etapa de correção ocasiona uma oscilação na estimação. Outro fato é que, apesar de conter um deslocamento, o GPS é o único sensor disponível durante todo o experimento, não podendo ser descartado. Para fundir os dados destes dois sensores de forma apropriada, a estratégia considera estados extras que representam um *bias* associado às medições do GPS em relação à localização da plataforma de pouso (ArUco ou Decawave). Portanto, o vetor de estados é definido como:

$$\bar{\mathbf{x}} = [\bar{\mathbf{p}}\ \bar{\mathbf{p}}_b]^T = [\underbrace{\bar{x}\ \bar{y}\ \bar{z}}_{\text{posição}}\ \underbrace{\bar{b}_x\ \bar{b}_y\ \bar{b}_z}_{bias}]^T, \tag{10}$$

onde os primeiros três estados são de posição e os três subsequentes são *bias* do GPS em relação à localização (pré-definida) da plataforma de pouso. Note que o filtro não conta com estados de *bias* relacionados à orientação do ArUco. Por isso, sua orientação geográfica deve ser conhecida com relativa precisão.

Sendo $\bar{\mathbf{p}} = [\bar{x}\ \bar{y}\ \bar{z}]^T$ e $\bar{\mathbf{p}}_b = [\bar{b}_x\ \bar{b}_y\ \bar{b}_z]^T$, temos:

$$f(\bar{\mathbf{x}}, \mathbf{u}, \Delta t) = \begin{bmatrix} \bar{\mathbf{p}} + \mathbf{u}\Delta t \\ \bar{\mathbf{p}}_b \end{bmatrix}. \tag{11}$$

Na etapa de correção associada à medição de posição do GPS, é considerada uma variável binária que determina se dados de localização da plataforma de pouso já estão sendo coletados ou não. Essa variável ξ determina a inclusão do *bias* no modelo de medição do GPS, valendo 1 quando há dados de localização em relação à plataforma de pouso e 0 em caso negativo. Assim, temos:

$$h_1(\bar{\mathbf{x}}) \equiv h_1(\bar{\mathbf{p}}, \bar{\mathbf{p}}_b) = \bar{\mathbf{p}} + \xi\bar{\mathbf{p}}_b. \tag{12}$$

Por fim, o modelo de medição relacionado à plataforma de pouso é descrito como:

$$h_2(\bar{\mathbf{x}}) \equiv h_2(\bar{\mathbf{p}}) = \bar{\mathbf{p}}. \tag{13}$$

Dessa forma, a estimação da pose do drone melhora, aumentando a segurança e precisão durante o pouso. Os resultados na Seção 4 demonstram a eficácia da metodologia.

3.3 Controle por Campos Vetoriais

O controle do quadrirrotor pode ser abstraído em dois níveis: alto e baixo. O controlador de baixo nível já está implementado no próprio drone, e seu papel é enviar sinais de atuação nos motores *brushless* a partir de informações de alto nível. No caso desta implementação, estas informações são as velocidades lineares em $\mathbf{x}$, $\mathbf{y}$ e $\mathbf{z}$, já que o modo de voo *Position Mode* é utilizado. O uso deste controlador em conjunto com a interface disponibilizada pelo pacote Onboard-SDK-ROS permite que o modelo do integrador simples seja considerado para o robô. Isso pode ser feito dado que o drone não realizará movimentos abruptos, como acrobacias. Assim, o seguinte modelo é considerado:

$$\dot{\mathbf{p}} = \mathbf{u}, \tag{14}$$

onde $\mathbf{p} = [x,\ y,\ z]$ representa a posição do drone em um referencial inercial, e $\mathbf{u} = [u_x,\ u_y,\ u_z]$ é o sinal de controle, ou seja, as velocidades de comando.

O controle de alto nível lida apenas com velocidades lineares e considera que o modelo na equação (14) é válido. Assim, uma estratégia baseada em campos vetoriais é utilizada (Gonçalves et al., 2010). Um campo cujas linhas integrais convergem para uma curva $\mathcal{C}$, definida adiante, deve ser utilizado.

Técnicas tradicionais de controle, ou até modos de voo que utilizam *waypoints*[9] no mapa, poderiam ter sido utilizados, porém com desvantagens. Na técnica utilizada, o caminho gerado previamente é suave, o que elimina efeitos indesejados causados pelo chaveamento entre leis de controle a cada mudança de *waypoint*. Além disso,

[9] Ponto intermediário em uma rota, onde há parada ou mudança de curso.

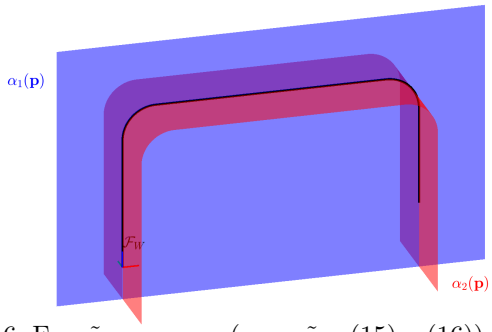

Figura 6. Funções α_1 e α_2 (equações (15) e (16)).

trechos como os dos setores S_2 e S_4 da Figura 5 podem ser otimizados a fim de, por exemplo, gerar menos movimentos bruscos com a carga, a partir de ajuste de parâmetros.

O controle por campos vetoriais é utilizado para seguir caminhos, e não trajetórias. A lei de controle que será proposta é uma função apenas do estado $\mathbf{p}$ do drone, portanto, independe diretamente do tempo t. Esta propriedade é particularmente interessante pois não possui dois problemas do controle de trajetória: (i) caso a referência comece bem distante da posição inicial do drone, este poderá passar por um transiente agressivo, o que é indesejado; (ii) em caso de falha temporária no sistema que faça o drone parar de responder por um tempo, quando este voltar a funcionar, a referência pode estar afastada, gerando um estado transiente adicional.

Para representar a curva $\mathcal{C}$, é preciso definir funções escalares $\alpha_i : \mathbb{R}^3 \to \mathbb{R}$, $i = 1, 2$, para cada cada setor $\mathcal{S}_j$, $j = 1, 2, 3, 4, 5$, de maneira que a interseção entre suas superfícies de nível zero, $\alpha_i(\mathbf{p}) = 0$, forme a curva desejada (Gonçalves et al., 2010). Ou seja, define-se $\mathcal{C}$ por $\mathcal{C} = \{\mathbf{p} \in \mathbb{R}^3 : \alpha_1(\mathbf{p}){=}0 \wedge \alpha_2(\mathbf{p}){=}0\}$. A Figura 6 ilustra esta representação. A definição das funções α_1 e α_2 utilizadas é apresentada a seguir:

$$\alpha_1 = \begin{cases} y, & \text{partindo de } \mathbf{p}_s \\ -y, & \text{partindo de } \mathbf{p}_f \end{cases} \tag{15}$$

$$\alpha_2 = \begin{cases} -x, & \text{se } \mathbf{p} \in \mathcal{S}_1 \\ r^{-1}\sqrt{[x-r]^2 + [z-(h-r)]^2} - 1, & \text{se } \mathbf{p} \in \mathcal{S}_2 \\ z - h, & \text{se } \mathbf{p} \in \mathcal{S}_3 \\ r^{-1}\sqrt{[x-(d-r)]^2 + [z-(h-r)]^2} - 1, & \text{se } \mathbf{p} \in \mathcal{S}_4 \\ x - d, & \text{se } \mathbf{p} \in \mathcal{S}_5 \end{cases} \tag{16}$$

Dessa forma, para cada ponto no espaço, é definida uma componente convergente e outra tangencial à curva.

$$F_{conv} = \frac{\nabla V}{\|\nabla V\|}, \qquad F_{tang} = \frac{\nabla\alpha_1 \times \nabla\alpha_2}{\|\nabla\alpha_1 \times \nabla\alpha_2\|}, \tag{17}$$

onde $V = \frac{1}{2}\alpha_1^2 + \frac{1}{2}\alpha_2^2$.

Assim como observado em (Gonçalves et al., 2010), funções $G \equiv G(V) = -(2/\pi)\arctan(k_f\sqrt{V})$ e $H \equiv H(V) = \sqrt{1 - G^2}$ são definidas, onde $k_f > 0$ é um peso de convergência. Estas funções fazem parte da definição do campo vetorial $F(\boldsymbol{p})$ utilizado na estratégia de controle, como visto a seguir:

$$F(\boldsymbol{p}) = v_r(GF_{conv} + HF_{tang}), \tag{18}$$

onde $v_r \equiv v_r(\mathbf{p}) > 0$ é a velocidade desejada para o robô. Definida por $v_r(\mathbf{p}) = v_i$ para $\mathbf{p} \in \mathcal{S}_i$, $i = 1, 2, 3, 4, 5$. Ou seja, a velocidade de referência para o drone é dependente do setor em que ele se encontra.

A orientação do drone foi controlada de maneira a manter seu ângulo ψ em torno do eixo $\mathbf{z}$ em 0° em relação ao referencial $\mathcal{F}_I$. Dessa forma, o robô fica alinhado com o vetor $(\mathbf{p}_f - \mathbf{p}_s)$, que vai do ponto de partida ao ponto de chegada. Assim, a velocidade angular $\dot{\psi}$ é definida como:

$$\dot{\psi} = -k_\psi \psi. \tag{19}$$

Uma vez inicializado, o drone segue a lógica demonstrada no Algoritmo 9, de maneira a completar a tarefa de modo autônomo.

Algoritmo 1 - Controle de alto nível do quadrirrotor

```
entregue ← falso
pousoFeito ← falso
tarefaTerminada ← falso
enquanto pousoFeito == falso faça
    posicao ← posAtual()
    setor ← identificarSetor(posicao)
    vel ← calcularCampo(posicao, setor)
    enviarVelocidade(vel)
    se pos == posPouso então
        pousoFeito ← verdadeiro
enquanto entregue == falso faça
    liberarPacote()
    entregue ← verdadeiro
    decolar()
tarefaTerminada ← verdadeiro
```

4 RESULTADOS

Nesta Seção são apresentados resultados de simulações e experimentos reais, os quais representam tarefas de entrega utilizando um drone em operação autônoma.

4.1 Simulações

As simulações realizadas têm como objetivo comparar os métodos de localização utilizados na fase de pouso, bem como as duas estratégias de planejamento de caminhos apresentadas na Seção 3.

Para a verificação dos métodos de localização, duas situações foram emuladas no *CoppeliaSim*: uma onde a plataforma de pouso estava exatamente no ponto esperado, e outra onde a plataforma estava deslocada da referência em 3m no eixo $\mathbf{y}$. Foram feitas 15 repetições do experimento para cada uma das duas configurações da plataforma de pouso e para cada uma das três combinações de sensores, totalizando 90 repetições. A Figura 7 representa os pontos de pouso do drone na situação sem deslocamento, em que o centro do alvo, dado pelo ponto $[150\text{m}, 0\text{m}]$, define o local de pouso.

Para comparação entre os métodos de pouso utilizando diferentes sensores, o erro associado é definido como a

	Sem deslocamento		Com deslocamento	
	Média	Desvio	Média	Desvio
GPS	0.735m	0.301m	3.178m	0.682m
GPS+ArUco	0.235m	0.140m	0.220m	0.070m
GPS+UWB	**0.101m**	**0.007m**	**0.173m**	**0.036m**

Tabela 1. Média e Desvio Padrão do erro associado a cada conjunto de experimentos.

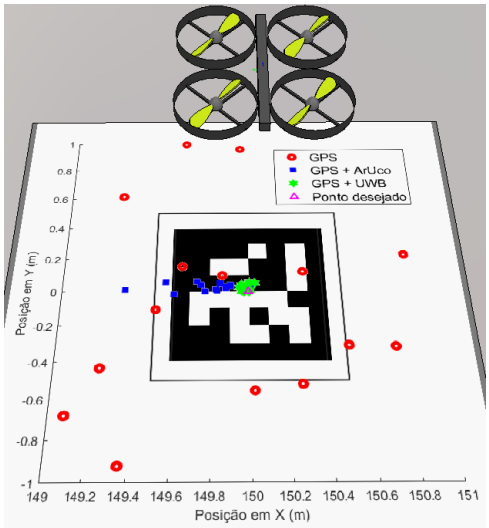

Figura 7. Comparação entre conjuntos de pouso feitos utilizando diferentes sensores.

distância entre o ponto esperado de pouso e o ponto efetivamente atingido. A Tabela 1 mostra a média e o desvio padrão do erro para cada conjunto de resultados.

A precisão associada ao GPS se mostra insuficiente ao realizar o pouso em uma plataforma de 1x1m como a utilizada. Além disso, caso a plataforma de pouso sofra algum deslocamento inesperado, o GPS sozinho, mesmo em um caso idealizado com altíssima precisão, não é capaz de corrigir sua posição e aterrissar no local correto. A utilização de sensores extras é importante para aumentar a precisão de pouso, principalmente no caso com deslocamento da plataforma, como fica evidente na Tabela 1.

O uso da câmera junto ao ArUco diminui o erro médio em mais de três vezes, enquanto o uso de dispositivos UWB é cerca de sete vezes mais preciso, se comparado ao uso do GPS de forma isolada. No caso do uso do GPS+UWB, há aumento do erro médio com a plataforma deslocada, se comparado à situação sem deslocamento. Isto se deve ao fato de que os dispositivos UWB só operam quando a *tag* está a uma distância mínima de 10m das âncoras, o que faz com que o drone tenha menos tempo para corrigir seu caminho e fazer o pouso no local correto. Dado que o drone parte de um ponto cuja coordenada em $\mathbf{y}$ está alinhada com a posição do ArUco, o erro observado no eixo $\mathbf{y}$ é notoriamente menor do que o erro em $\mathbf{x}$, conforme ilustrado na Figura 7.

Com respeito ao planejamento de caminhos, a Figura 8 apresenta, em verde, o percurso realizado em simulação pelo drone utilizando como referência o caminho computado pela segunda estratégia de planejamento. A simulação considerou uma altura de voo de $h = 47m$, distância entre os pontos de partida e chegada de $d = 150m$ e arcos de circunferência de raio $r = 15m$. A Figura 9 ilustra os erros ao longo de um percurso executado pelo drone utilizando GPS e UWB.

4.2 Validação Experimental

A aplicação de drones em tarefas de *delivery* autônomo é demonstrada por um experimento preliminar utilizando a primeira estratégia de planejamento de caminhos, por esta

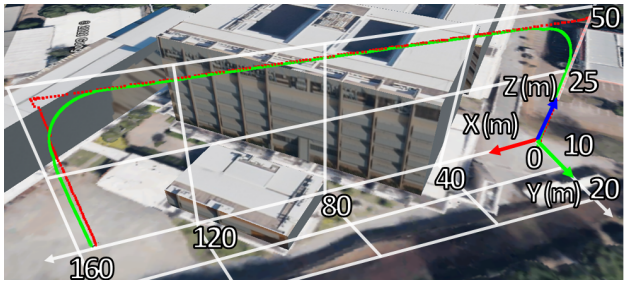

Figura 8. Representação dos resultados obtidos em experimento simulado (em verde) e real (em vermelho) da tarefa de *delivery* autônomo.

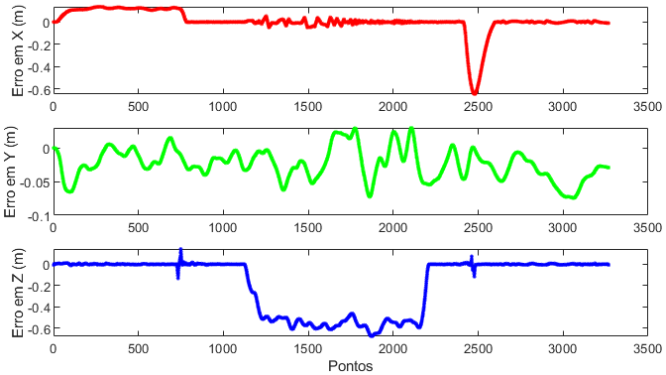

Figura 9. Erro em **x**, **y** e **z**, respectivamente, ao longo do caminho percorrido pelo drone.

ter sido extensivamente testada em ambiente simulado. A Figura 8 ilustra o caminho percorrido pelo drone, em vermelho; os resultados foram obtidos com os mesmos valores de referência para altura de voo ($h = 47m$) e distância entre os pontos de partida e chegada ($d = 150m$) utilizados na simulação. Os arcos de circunferência de raio $r = 15m$ presentes na segunda estratégia foram responsáveis por diminuir a distância total percorrida pelo drone de 272 m para 228 m, o que representa uma redução de mais de 15%.

As imagens da Figura 10 ilustram a sequência de etapas de entrega. Após definir o ponto de pouso e acoplar a caixa ao drone, a tarefa foi feita em modo autônomo e a caixa foi deixada no local combinado. Depois de decolar novamente e ir para um lugar mais afastado, uma pessoa que aguardava a entrega pegou a caixa e o objeto que estava dentro dela. O vídeo completo do *delivery* autônomo pode ser visto no canal do YouTube da equipe XQuad [10].

5 CONCLUSÕES E TRABALHOS FUTUROS

Este artigo apresentou técnicas de planejamento de caminho, localização e controle por campos vetoriais, que foram aplicadas em tarefas reais de entrega utilizando um drone autônomo. Algoritmos de visão computacional e dispositivos UWB forneceram estimativas de pose do quadrirrotor para um Filtro de Kalman Estendido, que realizou a fusão sensorial com dados do GPS. Os resultados obtidos em simulação demonstram uma melhora significativa na etapa de pouso.

[10] YouTube XQuad - `https://bit.ly/311KU4p`

Figura 10. Sequência de imagens que representa a entrega autônoma feita na UFMG.

A realização de experimentos reais com os sensores extras para auxílio no pouso será abordado em trabalhos futuros, avaliando também a influência de distúrbios externos. Além disso, melhorias podem ser implementadas na localização, em comunicações de longa distância e principalmente na detecção e desvio de obstáculos com o uso de mais câmeras nas laterais e no topo do drone.

REFERÊNCIAS

ABRASEL (2020). Do celular à mesa: como os apps de delivery transformam o mercado de bares e restaurantes. URL `https://bit.ly/2N6WgvA`.

Gonçalves, V.M., Pimenta, L.C.A., Maia, C.A., Dutra, B.C.O., and Pereira, G.A.S. (2010). Vector Fields for Robot Navigation Along Time-Varying Curves in *n*-Dimensions. *IEEE Trans. on Robotics*, 26(4), 647–659.

Hesch, J.A. and Roumeliotis, S.I. (2011). A Direct Least-Squares (DLS) method for PnP. In *2011 Int. Conf. on Computer Vision*, 383–390.

Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). EPnP: An accurate O(n) solution to the PnP problem. *Int. Journal of Computer Vision*, 81(2), 155.

Raffo, G.V. and de Almeida, M.M. (2016). Nonlinear robust control of a quadrotor uav for load transportation with swing improvement. In *2016 American Control Conference (ACC)*, 3156–3162. doi:10.1109/ACC.2016.7525403.

Rezende, A.M.C., Miranda, V.R.F., Machado, H.N., Chiella, A.C.B., Gonçalves, V.M., and Freitas, G.M. (2019). Autonomous System for a Racing Quadcopter. In *2019 19th Int. Conf. on Advanced Robotics (ICAR)*, 1–6.

Sahinoglu, Z. (2008). *Ultra-wideband positioning systems*. Cambridge university press.

Sayed, A.H., Tarighat, A., and Khajehnouri, N. (2005). Network-based wireless location: challenges faced in developing techniques for accurate wireless location information. *IEEE Signal Processing Magazine*, 22(4), 24–40.

Schneider, D. (2020). The delivery drones are coming. *IEEE Spectrum*, 57(1), 28–29. doi:10.1109/mspec.2020.8946304.

Thrun, S., Burgard, W., and Fox, D. (2000). *Probabilistic robotics*, volume 1. MIT press Cambridge.

Villa, D.K., Brandao, A.S., and Sarcinelli-Filho, M. (2019). A survey on load transportation using multirotor uavs. *Journal of Intelligent & Robotic Systems*, 1–30.