Arquitetura de Simulação de Missões com Drones com Aspectos Mais Realistas de Comunicação Ar-Terra*

Bruno A. Hilario[1], **Diego Passos**[1,2], **Raphael Guerra**[1]

[1] Instituto de Computação – Universidade Federal Fluminense (UFF)
24210-346 – Niterói – RJ – Brasil

[2]ISEL, Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa
Lisboa – Portugal

brunohilario@id.uff.br, dpassos@ic.uff.br, rguerra@ic.uff.br

Abstract. *Unmanned Aerial Vehicles (UAVs) have seen significant growth due to their affordable prices, availability, and ease of operation. These operability benefits have led to their introduction into many civilian applications, including surveillance, remote monitoring, relief operations, and recreational use. For critical applications, a realistic UAV mission simulation platform, including accurate models to predict the range and quality of communication between the UAV and the ground station, is of great value to ensure the effectiveness and safety of the flight. Our studies showed that simulation platforms are, in general, simplified in terms of communication models. In this article, we also propose a simulation architecture based on the SITL ArduPilot that introduces more accurate communication models.*

Resumo. *Os Veículos Aéreos Não Tripulados (ou UAV, do inglês Unmanned Aerial Vehicles) tiveram um crescimento significativo devido a seus preços acessíveis, disponibilidade e facilidade de operação. Esses benefícios de operabilidade levaram a sua introdução em muitas aplicações civis, incluindo vigilância, monitoramento remoto, operações de socorro, além do uso recreativo. Para aplicações críticas, uma plataforma realista de simulação de missão de UAV, incluindo modelos precisos para prever o alcance e a qualidade da comunicação entre o UAV e a estação terrestre, é de grande valia para garantir a eficácia e a segurança do voo. Nossos estudos mostraram que as plataformas de simulação são, em geral, simplificadas quanto aos modelos de comunicação. Neste artigo, propomos uma arquitetura de simulação baseada no SITL ArduPilot que introduz modelos de comunicação mais precisos.*

1. Introdução

Os *Unmanned Aerial Vehicles* (UAVs), também conhecidos como *drones*, são aeronaves não tripuladas que podem ser diferenciadas em termos do tipo (asa fixa, multi-rotor, etc), do tamanho, do peso e da sua fonte de energia [Vergouw et al. 2016]. *Drones* são uma tecnologia promissora para uma série de aplicações, como coletar dados de redes externas de sensores, monitorar em tempo real o tráfego rodoviário, além de fornecer

*Este trabalho recebeu apoio financeiro e operacional da Prefeitura Municipal de Niterói em parceria com a Universidade Federal Fluminense, sob interveniência da Fundação Euclides da Cunha.

cobertura sem fio. Eles podem atuar no apoio à segurança pública em operações de busca e resgate, segurança e vigilância [Bastos et al. 2022]. Podem também ser usados em aplicações civis como entrega de mercadorias e inspeção de infraestrutura civil [Shakhatreh et al. 2019]. O desenvolvimento de *drones*, juntamente com a presença da Internet das Coisas (IoT), permite a criação de sistemas que podem fornecer novos serviços de valor agregado de IoT quando equipados com dispositivos de comunicação adequados e controláveis remotamente (sensores, câmeras e atuadores).

O consumo de energia geralmente é uma preocupação para dispositivos equipados com bateria. Para *drones*, isso se torna mais significativo, pois a bateria é utilizada para manter a aeronave voando e manter esses dispositivos operando e conectados a rede [Motlagh et al. 2016]. Do ponto de vista da conexão de rede, por exemplo, uma série de tecnologias de comunicação sem fio derivadas do padrão IEEE 802.15.4 para as chamadas *Low Rate Personal Area Networks* (LR-WPANs) (*e.g.*, Zigbee, wirelessHART e Thread [Nekrasov et al. 2019]) foram consideradas para permitir comunicação de longa distância com baixo consumo de energia em *drones*.

Mesmo assim, a eficiência energética dos módulos de comunicação é apenas uma parte do desafio. De forma mais geral, o planejamento de missões é importante para evitar fracassos. Em missões de busca ou vigilância, por exemplo, o operador precisa otimizar o voo tendo em conta as limitações dos *drones*, não só em termos de autonomia, mas também de cobertura de sinal [Coopmans et al. 2015]. Fatores como orientação e posicionamento da antena, altitude e obstrução afetam a potência do sinal recebido e a taxa de recepção de pacotes. Por todos esses fatores, uma plataforma realista de simulação de missão de *drones*, incluindo modelos precisos para prever o alcance e a qualidade da comunicação entre o *drone* e a estação terrestre, é de grande valia para garantir o sucesso das missões.

Existem simuladores comerciais, como o *Flight simulator* da DJI [DJI 2022] e o *Real Flight* [Software 2022], onde o foco é aprimorar as habilidades de pilotagem sem os riscos e custos potenciais associados ao treinamento na vida real. Eles possuem tutorais básicos, treinamento de voo pairado e de rota, além de permitir simular efeitos de vento, solo e falhas. Porém, esses simuladores não consideram consumo de energia nem possibilidade de falha na comunicação. Logo, não são apropriados para avaliar a viabilidade de missões específicas.

Há também simuladores não comerciais de código aberto que são facilmente extensíveis para incluir novas funcionalidades. O Simulador SITL (*Software in The Loop*) ArduPilot [ArduPilot 2022c] permite pilotar em um ambiente simulado. Ele é uma compilação do código do piloto automático ArduPilot que permite testar o comportamento do código sem um *hardware* de controlador de voo. O ArduPilot é um piloto automático portátil que pode ser executado em uma ampla variedade de plataformas, incluindo computadores pessoais. É o principal sistema de piloto automático de código aberto que suporta multicópteros, helicópteros e aeronaves de asa fixa. O Gazebo [Gazebo 2022] é um poderoso simulador 3D, capaz de ser integrado em diversas plataformas robóticas. É um simulador de dinâmica tridimensional de código aberto para mecanismos de robôs e multi-robôs, para ambientes internos e externos. O Gazebo tenta criar mundos realistas para os robôs, modelando características físicas. Em conjunto, SITL Ardupilot e Gazebo permitem uma simulação como as dos simuladores comerciais,

permitindo o treinamento de pilotos em um ambiente 3D. Além do Gazebo, o Ardupilot SITL também pode ser usado com outro simulador de mundo virtual, o AirSim. O AirSim [Research 2022] é um simulador para *drones* e carros, construído no *Unreal Engine*. É de código aberto, multiplataforma e oferece excelentes simulações física e visualmente realistas. Ele foi desenvolvido para se tornar uma plataforma de pesquisa de IA para experimentar algoritmos de aprendizado profundo, visão computacional e aprendizado por reforço para veículos autônomos. Além de poder treinar e aprimorar as habilidade de pilotagem, a vantagem desses simuladores em relação aos comerciais é poder testar novas funcionalidades antes de implementá-las em *drones* reais que utilizem o ArduPilot.

Estudos preliminares, executando missões tanto autônomas quanto manuais, sugerem que as plataformas de simulação disponíveis utilizam modelos muito simplificados para simular os aspectos de comunicação. Implementar modelos de propagação e de erros na simulação da comunicação pode ajudar a testar os limites de conexão e impedir imprevistos nas missões. Tais imprevistos incluem perda de sinal e, consequentemente, perda de dados, no caso de uma busca em tempo real, ou até mesmo a perda de controle do equipamento.

Neste artigo, fazemos um levantamento do estado da arte relacionado a modelos de propagação e modelos de erros para a comunicação entre a estação terrestre e o *drone*. Com base nisso, propomos uma arquitetura que permite o refinamento dos modelos de comunicação utilizados em simulações com o simulador Ardupilot SITL, juntamente com o *software* de controle MAVProxy [ArduPilot 2022a] e o *Mission Planner* [ArduPilot 2022b] para programar as missões automáticas. A arquitetura proposta faz uso de um *proxy* que intermedeia a comunicação entre o *drone* (SITL) e a estação terrestre (MAVProxy), adicionando modelos de propagação e erro na decisão de se as mensagens são repassadas ou não de uma ponta à outra da comunicação. Isso permite simular falhas na comunicação e avaliar seus impactos na missão. Nossos resultados experimentais demonstram que a arquitetura proposta é, de fato, mais realística na sua modelagem da comunicação, obtendo o efeito esperado de o sinal da comunicação na simulação se degradar com o aumento da distância e altura do *drone* em relação à estação terrestre. Além disso, os resultados mostram que essa degradação varia de acordo com o modelo de propagação escolhido, ilustrando a capacidade modular e extensível da arquitetura proposta.

O restante do artigo está organizado da seguinte forma. A Seção 2 apresenta os estudos e trabalhos sobre modelos de propagação e modelos de erros de comunicação entre *drones* e a estação terrestre. Na Seção 3, propomos uma arquitetura modular para a introdução de modelos mais realísticos de comunicação apresentados na Seção 2. Na Seção 4, são discutidos os resultados de experimentos realizados para a validação da arquitetura proposta. A Seção 5 conclui o artigo e discute trabalhos futuros.

2. Conceitos Fundamentais

Esta seção descreve os conceitos fundamentais da arquitetura proposta para tornar a simulação mais realista em relação à comunicação entre o *drone* e a estação terrestre. A Seção 2.1 descreve o protocolo MAVLink. A Seção 2.2, apresenta os modelos de propagação e a Seção 2.3 os modelos de erro encontrados na literatura.

2.1. Protocolo MAVLink

MAVLink é um protocolo de mensagens leve para comunicação bidirecional entre a estação terrestre e o *drone*. O MAVLink segue uma arquitetura híbrida moderna de publicação-assinatura (*publish-subscribe*) e ponto-a-ponto. Os fluxos de dados são enviados/publicados como tópicos, enquanto os sub-protocolos de configuração, como o protocolo de missão ou o protocolo de parâmetro, são ponto-a-ponto com retransmissão. Este protocolo é usado nos principais sistemas de piloto automático, em particular no ArduPilot, e fornece recursos poderosos não apenas para monitorar e controlar missões de *drones*, mas também para sua integração à Internet [Dronecode 2022]. Ele serializa as mensagens dos estados do sistema e os comandos que o *drone* deve executar em um formato binário específico (ou seja, um fluxo de bytes), independente da plataforma. Como suas mensagens são tipicamente pequenas, podem ser transmitidas de forma confiável por diferentes meios sem fio, incluindo WiFi ou mesmo dispositivos de telemetria serial com baixas taxas de dados.

Uma mensagem na versão 1.0 do MAVLink, como mostrado na Figura 1, possui oito campos. O primeiro campo é o STX, um marcador que sempre tem o valor 0XFE. O segundo byte (LEN) representa o comprimento da mensagem em bytes. O terceiro byte (SEQ) denota o número de sequência da mensagem, permitindo valores de 0 a 255. Ao atingir 255, o número de sequência é redefinido para 0. O quarto byte (SYS) representa o ID do sistema (todo *drone* deve ter seu ID de sistema). O quinto byte (COMP) é o ID do componente e identifica o componente do sistema que está enviando a mensagem. O sexto byte (MSG) representa o ID da mensagem, que se refere ao tipo de mensagem embutida na carga útil. Por exemplo, o ID da mensagem igual a 0 se refere a uma mensagem do tipo *HEARTBEAT*, que indica que o sistema está ativo e é enviada a cada segundo. Já a mensagem com ID igual a 33, por exemplo, se refere à mensagem que reporta a coordenada do GPS do *drone*. O ID da mensagem é a informação essencial que permite analisar a carga útil e extrair a informação dela, com base no tipo de mensagem. O *payload* está localizado logo após o ID da mensagem e pode ter no máximo 255 bytes. E, finalmente, os dois últimos bytes são para a soma de verificação.

Figura 1: Formato da mensagem MAVLink v1

Já a mensagem da versão 2.0 do MAVlink possui todos os campos que já existiam no MAVLink 1.0 adicionando dois conjuntos de novas *flags* antes do número de sequência (SEQ): uma de incompatibilidade e outra de compatibilidade. Ambos os campos indicam a presença de certos tipos especiais de informação no corpo do pacote. Destas informações especiais, aquelas que são fundamentais para o processamento do pacote são indicadas nas *flags* de incompatibilidade (*i.e.*, um receptor incapaz de entender estas informações é incompatível com a mensagem). Por outro lado, algumas dessas informações especiais são não-essenciais para a compreensão do pacote e, neste caso, são informadas nas *flags* de compatibilidade (*i.e.*, o receptor é compatível com a mensagem mesmo que não seja capaz de entender essas informações). É adicionado também um campo opcional de assinatura para garantir que o enlace seja a prova de violação. Além de novos campos, o campo do identificador da mensagem (MSGID) é agora codificado em

24 bits, ao invés dos 8 bits da versão anterior. A Figura 2 mostra a estrutura do cabeçalho do MAVLink 2.0.

Figura 2: Formato da mensagem MAVLink v2

As mensagens MAVLink são classificadas em mensagens de estado e de comando. As mensagens de estado, que são enviadas do *drone* para a estação terrestre, contêm informações sobre o *drone*, como ID, localização, velocidade e altitude. As mensagens de comando são enviadas pela estação terrestre ao *drone* para execução de ações pelo piloto automático como decolar, pousar ou ir até um ponto qualquer.

O MAVLink possui vários tipos de mensagens de estado e de comando. A mensagem de estado de ID igual a 33 é a mensagem de posicionamento global (*GLOBAL POSITION MESSAGE*), ilustrada na Figura 3. Esta mensagem, que será utilizada na arquitetura proposta neste artigo, carrega informações importantes do *drone* relacionadas às suas coordenadas, latitude (lat), longitude (lon) e também altitude absoluta (alt). Esses três valores são codificados em quatro bytes cada. A mensagem contém também um campo de altitude relativa (*relative alt*), que representa a altitude em relação ao ponto de solo de decolagem do *drone* [Koubâa et al. 2019]. Além desses, a mensagem também carrega informações sobre a velocidade linear do *drone* ao longo dos três eixos (x,y,z) e da orientação (referida como *heading*). Essas informações são coletadas a partir do sensor GPS.

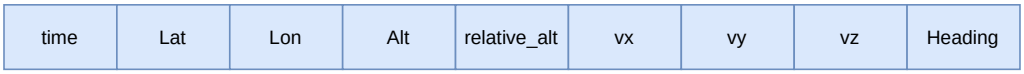


Figura 3: Mensagem de Posicionamento Global

Uma descrição mais completa e abrangente do protocolo pode ser encontrada em [Koubâa et al. 2019].

2.2. Modelos de Propagação

Embora calcular exatamente o valor da atenuação sofrida por um sinal de rádio em um caso concreto seja difícil, devido ao grande número de fatores envolvidos, existem diversos modelos na literatura que tentam capturar o comportamento médio da atenuação com base em fatores como a distância entre os nós e a frequência. Nas próximas seções, são apresentados alguns modelos genéricos populares e outros especificamente propostos para o cenário de comunicação entre estações terrestres e *drones*.

2.2.1. *Friis*

A Equação de *Friis* relaciona a potência de sinal recebido à potência do sinal transmitido, à distância entre as antenas, aos seus ganhos e à frequência do sinal em um *link* de comunicação no espaço livre [Lassabe et al. 2005]. Ela é descrita por:

$$\frac{Pr}{Pt} = GtGr\left(\frac{\lambda}{4\pi d}\right)^2, \tag{1}$$

onde Pr e Pt são, respectivamente, a potência recebida na antena receptora e a potência transmitida a partir da antena transmissora, Gr e Gt são, respectivamente, o ganho da antena do receptor e o ganho da antena do transmissor, λ é o comprimento de onda e d é a distância entre o transmissor e o receptor [Lassabe et al. 2005].

2.2.2. *Log-Distance*

O Modelo de propagação *log-distance* indica que a potência média do sinal em dBm recebido diminui logaritmicamente com a distância dos nós transmissor e receptor. A equação para este modelo é:

$$PL(dB) = PL(d_0) + 10n \cdot \log\left(\frac{d}{d_0}\right) \tag{2}$$

Na equação, n é o expoente de perda de caminho, d é a distância entre transmissor e receptor em metros e d_0 é uma distância de referência em metros – em geral, pequena – para qual a perda é $PL(d_0)$ [Siddiqui et al. 2019].

2.2.3. Modelos para Canais de Rádio Ar-Terra em Ambientes Urbanos

Em [Feng et al. 2006], são fornecidos novos modelos estatísticos para canais de rádio Ar-Terra em um ambiente urbano. O modelo é derivado para operar em frequências de 200 MHz a 5 GHz. Problemas como atenuação e sombreamento são avaliados em função do ângulo de elevação do *drone*, em oposição à distância usada em comunicações terrestres. Dois tipos principais de obstáculo no ambiente são considerados: edifícios e folhagens. De acordo com o bloqueio, o canal de rádio é classificado como Linha de Visão — ou LoS (*Line of Sight*) — e Linha de Visão Obstruída — ou OLoS (*Obstructed Line of Sight*). Dadas estas premissas, os autores adaptam o modelo *Log-distance* para o caso particular da comunicação entre o *drone* e a estação terrestre. Neste cenário, eles definem d_0 como a altura relativa do *drone* diretamente acima da estação terrestre, ou seja, $d_0 = h_t - h_r$, onde h_t e h_r são as alturas do *drone* e da estação terrestre, respectivamente, em relação ao solo. Assumindo que o ângulo de elevação é θ, $d/d_0 = 1/\sin\theta$. Assim, a Equação (2) torna-se:

$$PL(\theta)[dB] = PL(d_0) - 10n \cdot \log(\sin\theta), \tag{3}$$

onde $PL(d_0)$ é a perda no espaço livre dada pela Equação (1) de *Friis* para a distância de referência d_0. Quando o ângulo de elevação $\theta > 10$º, com base em dados empíricos, os autores concluem que:

$$-10n \cdot \log\sin\theta \approx -0,3115 + 0,2656e^{(90-\theta)/23,8} \tag{4}$$

Tabela 1: Parâmetros para modelos de perda de caminho médio — OLoS

Frequência	α_0	α_1	β
200 MHz	2,11	0,4125	22,07
1000 MHz	3,76	0,3724	21,38
2000 MHz	4,77	0,3530	21,04
2500 MHz	5,12	0,3895	21,58
5000 MHz	6,23	0,4787	22,65

De forma mais genérica, ao modelar o canal de rádio entre o *drone* e a estação terrestre em termos de altura e ângulo de elevação, os autores propõem que a Equação (3) para $\theta > 10^{\text{o}}$ pode ser modificada para:

$$PL(\theta)[dB] = PL(d_0) + L_2(\theta), \tag{5}$$

onde:

$$L_2(\theta)[dB] = \alpha_0 + \alpha_1 e^{(90-\theta)/\beta} \tag{6}$$

Depois de compensar os ganhos da antena e a remoção de $PL(d_0)$, a perda média de caminho para os canais LoS pode ser modelada usando as Equações (5) e (6). Após a análise de dados empíricos (usando ajuste de curva), os autores concluem que $L_2(\theta)$ pode ser considerado independente da frequência e altura da antena. A equação geral fica:

$$L_2(\theta)[dB] = -0,58 + 0,5496e^{(90-\theta)/24} \approx -20 \log \sin \theta \tag{7}$$

Da mesma forma, a comunicação OLoS pode ser modelada usando as Equações (5) e (6), mas, neste caso, os autores concluem que a frequência é um fator significativo. Em [Feng et al. 2006], são fornecidos valores para os coeficientes α_0, α_1 e β da equação para diferentes cenários OLoS, reproduzidos na Tabela 1. Na Equação (3), o expoente de perda de caminho n tem maior probabilidade de variar com a altura da antena, enquanto α_0, α_1 e β na Equação (6) são considerados independentes da altura da antena para a maioria dos casos estudados no artigo citado. Portanto, o modelo proposto pelos autores seria mais geral para várias alturas.

2.3. Modelos de Erro

Em redes de comunicação sem fio, o desvanecimento do sinal e as fontes de interferência e ruído causam variações na relação sinal-ruído (SNR – *Signal-to-Noise Ratio*). Tal variação também causa variações na taxa de erro de bit (BER – *Bit Error Rate*), pois, em geral, quanto menor o SNR, mais difícil é decodificar o sinal recebido [Lee et al. 2010]. Em [Lacage and Henderson 2006], os autores apresentam um modelo para estimar o BER na modulação BPSK (*Binary Phase Shift Keying*) em função do SNR:

$$BER = 0,5 \cdot erfc\left(\sqrt{SNR}\right). \tag{8}$$

Dado o BER, é possível calcular a probabilidade de que o pacote seja recebido com algum erro, denotada por PER (*Packet Error Rate*). A Equação (9) mostra

o cálculo do PER com base no BER, assumindo-se independência entre os símbolos [Lacage and Henderson 2006]:

$$PER(k) = 1 - (1 - BER)^s, \tag{9}$$

onde s é o tamanho do pacote em bits.

3. Descrição do Fluxo Geral da Solução Proposta

O SITL Ardupilot, como mostrado na Figura 4, conecta um *drone* simulado ao *MAVProxy* através de *sockets* TCP para a troca de mensagens entre eles. Outras estações terrestres como *Mission Planner*, podem ser conectadas via UDP ou TCP. Por se tratar de uma conexão TCP, não há qualquer perda de mensagens nesta comunicação durante a simulação. Adicionalmente, nem o *drone* simulado, nem o *MAVProxy* embutem qualquer modelo de perda, fazendo com que a comunicação ocorra de forma perfeita durante toda a simulação, independentemente de fatores como a distância entre os dois dispositivos.

Para evitar modificações no MAVProxy ou no *drone* simulado, na solução proposta (Figura 5), o *MAVProxy* e o SITL Ardupilot são conectados através um *proxy* responsável por fazer a troca de mensagens. A existência deste *proxy* é totalmente transparente ao resto do ambiente de simulação. Entretanto, ao intermediar a comunicação, o *proxy* pode inspecionar as mensagens, obtendo informações sobre o estado atual do *drone* e, eventualmente, alterar ou suprimir pacotes da comunicação.

Figura 4: Troca de Mensagens SITL Ardpupilot - Adapatado do SITL [ArduPilot 2022c]

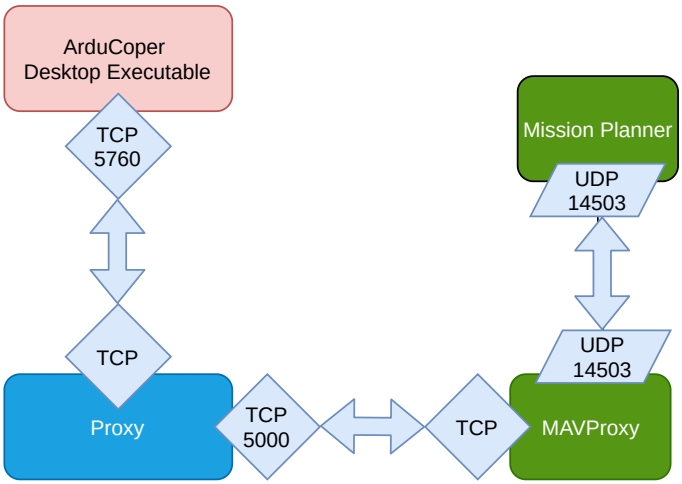


Figura 5: Solução Proposta

Com este objetivo, os modelos de propagação apresentados na Seção 2 são utilizados pelo *proxy* para calcular a potência de sinal recebido do lado do receptor (*drone*

Algoritmo 1 Fluxo de execução do *proxy*

```
1: while novas mensagens do
2:     captura a mensagem
3:     if mensagem é do tipo GLOBAL POSITION MESSAGE then
4:         Extrai coordenadas atuais do drone
5:         Calcula a distância entre o drone e a estação terrestre
6:     end if
7:     Calcula a potência recebida aplicando o modelo de propagação
8:     Calcula o SNR com base na potência recebida
9:     Calcula o BER com base no SNR
10:    Calcula o PER com base no BER
11:    Sorteia um número aleatório x ∈ [0, 1]
12:    if PER < x then
13:        Descarta o pacote
14:        Registra o erro do pacote no Log
15:    else
16:        Encaminha o pacote
17:        Registra o envio do pacote no log
18:    end if
19: end while
```

ou estação base, a depender do caso). A escolha do modelo específico para uma certa simulação pode ser feita através de um arquivo de configuração.

Mais detalhadamente, o Algoritmo 1 descreve o procedimento executado pelo *proxy* durante o processamento de cada mensagem recebida e como é feita a decisão de encaminhar ou suprimir pacotes de comunicação. Ao receber o pacote, o tipo de mensagem *MAVLink* é identificado e, se for uma mensagem do tipo `GLOBAL POSITION MESSAGE`, o *proxy* retira do *payload* da mensagem os valores de posicionamento global para calcular a distância entre o *drone* e a estação terrestre. Com a distância calculada, a potência de sinal recebido no destinatário é calculada de acordo com o modelo de propagação escolhido para a simulação. Em seguida, com base neste valor de potência, calcula-se o SNR que, por sua vez, é utilizado para calcular o BER utilizando a Equação (8). A seguir, o *proxy* calcula o PER para a mensagem com base no BER e no seu tamanho (Equação 9). Por fim, decide-se aleatoriamente se a mensagem será encaminha normalmente ao destinatário (*drone* ou estação terrestre) ou se será descartada. O *proxy* gera ainda um *log* detalhando a sua execução e as decisões tomadas a cada mensagem.

Diversos passos listados no Algoritmo 1 dependem de informações que vão além das disponíveis nas mensagens MAVLink. Tais informações são especificadas como parâmetros de execução do *proxy* através de um arquivo de configuração, ilustrado na Figura 6. Um exemplo de informação obtida através deste arquivo são os valores de latitude, longitude e altura absoluta e relativa da estação terrestre. Na arquitetura original de simulação com o SITL Ardpupilot, esses valores já são especificados atraves do arquivo *locations.txt*. Assim, basta ao usuário informá-los também no arquivo de configuração do *proxy*. Além disso, o arquivo permite também a seleção do tipo de modelo de propagação usado na simulação e a especificação de outros dados relevantes para o cálculo dos mo-

```
[parametrosConfiguracao]
GanhoAntena=3
Potencia=15
ComprimentoOnda=0.3
Ruido=-95
expoentePerda=4
distanciaRef=21
potenciaRef=37.46237
# Modelo de propagação
# 1 - Friis
# 2 - logDistance
# 3 - Los
# 4 - olos
modeloProgacao=2
alfa0 = 5.12
alfa1 = 0.3895
beta = 21.58
```

Figura 6: Exemplo de arquivo de configuração do *proxy*

delos, como ganho da antena, comprimento de onda, nível de ruído, expoente de perda, distância de referência e potencia de referência. Os campos `alfa0`, `alfa1` e `beta` são utilizados para a simulação do modelo OLoS, e seus valores correspondem a alguma das opções listadas na Tabela 1.

É importante destacar também que a modularidade da solução proposta permite implementar outros modelos de propagação, além dos modelos revisados na Seção 2 e implementados no protótipo aqui apresentado para efeito de avaliação neste artigo.

4. Validação da Arquitetura Proposta

Para validar a arquitetura proposta, foi criada uma missão utilizando o *Mission Planner*, conforme ilustrado na Figura 7, para programar e executar uma missão autônoma. A missão inicia na estação terrestre, percorre quatro *way-points* a 60 metros de altura e retorna à estação terrestre. Esta mesma missão foi repetida 10 vezes para cada um dos modelos de propagação implementados e também com a arquitetura original de simulação do SITL Ardupilot — e, portanto, sem modelo de erro de propagação.

A Figura 8a mostra a degradação do SNR ao longo do tempo de voo da missão executada de acordo com cada modelo de propagação. Não são mostrados dados da execução com a arquitetura original do SITL Ardupilot porque a mesma não computa o SNR ao longo da simulação. Em todos os modelos de propagação, o sinal sofre uma degradação à medida que a aeronave se afasta (o que ocorre durante, aproximadamente os dois primeiros minutos da missão). Analogamente, o SNR melhora gradativamente quando o *drone* começa a retornar para a estação terrestre, durante aproximadamente os dois minutos finais da missão. Como esperado, o modelo de *Friis* se mostrou o mais otimista, resultando em valores de SNR acima de 20 dBm durante toda a missão. Já os modelos LoS e OLoS se mostraram os mais pessimistas — em particular, o modelo OLoS resultou na deterioração mais rápida do sinal dentre os quatro testados.

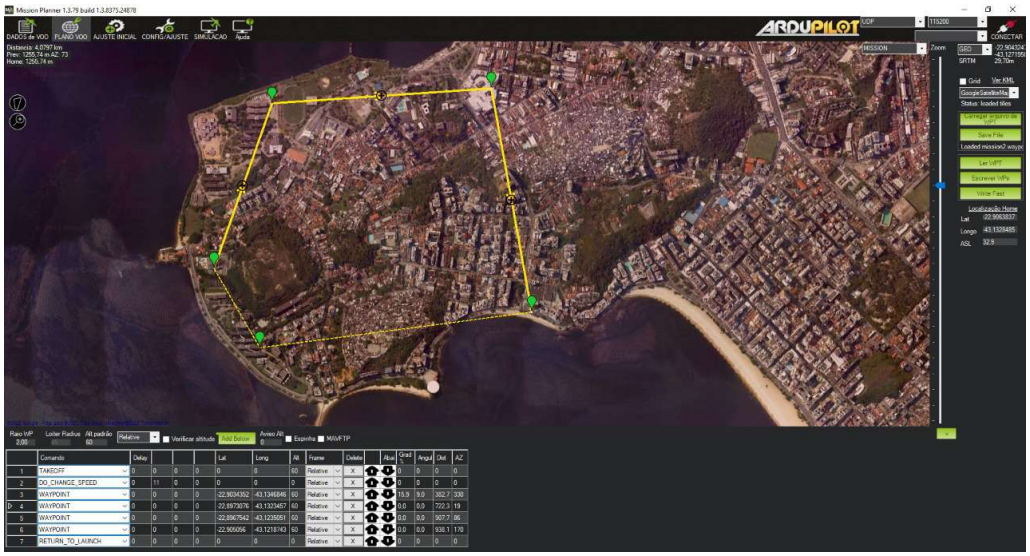
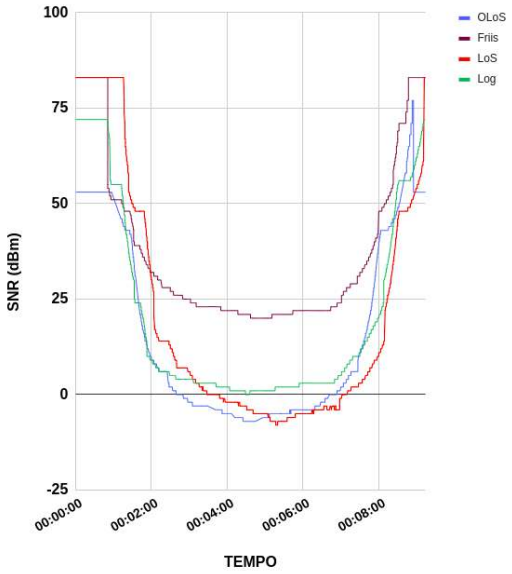


Figura 7: Missão autônoma - *Mission Planner*

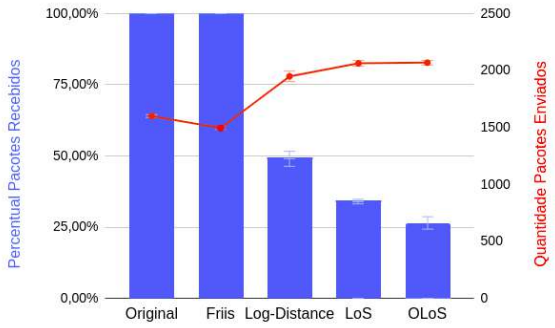
A Figura 8b mostra as médias do total de pacotes enviados e do percentual de pacotes recebidos para cada modelo. As barras de erro representam o intervalo de confiança de 95%. Com o modelo *Friis*, o *drone* conseguiu percorrer todo o trajeto sem perder a conexão nem pacotes e ficou estatisticamente empatado com o original. Já com os demais modelos, *Log-Distance*, LoS ou OLoS, a degradação é mais exacerbada chegando a causar a perda de pacotes e da conexão com a estação terrestre durante vários minutos. Os modelos LoS e OLoS ficaram estatisticamente empatados em relação ao número de pacotes enviados; estes também são os modelos que mais enviam pacotes e têm menor percentual de recebimento (*i.e.*, entrega com sucesso). De fato, há uma correlação negativa entre o percentual de pacotes entregues e o número de pacotes enviados durante a missão, sugerindo a existência de algum mecanismo de retransmissão na comunicação.

Como a missão é carregada inicialmente no *drone*, a aeronave continua percorrendo o trajeto definido pelos pontos planejados, mas não se conhece seu *status* até o momento em que a conexão entre o *drone* e a estação terrestre é restabelecida e a aeronave reaparece no *Mission Planner*. Ao executar a missão, o *Mission Planner* exibe o trajeto percorrido com uma linha roxa à medida que o *drone* avança. Na Figura 9, o traçado mostrado na linha roxa mostra que uma parte do trajeto não foi monitorado apesar de ter sido percorrido. A conexão com a aeronave é perdida durante este período. A degradação de sinal e a ”perda” da aeronave variam de acordo com o modelo de propagação utilizado na simulação, conforme mostrado na Figura 9. Nota-se, por exemplo, que o modelo OLoS é o mais pessimista.

Em um cenário como o avaliado, em que a missão é executada de forma autônoma pelo *drone* com base em pontos de controle pré-definidos, essas regiões de sombra podem fazer com que dados a serem coletados pelo *drone* sejam perdidos (por exemplo, um fluxo de vídeo para o monitoramento de certa região). Mas, para aplicações que demandam a pilotagem manual da aeronave, esta perda de conectividade pode trazer impactos severos, incluindo acidentes.



(a) Evolução do SNR



(b) Percentual de pacotes recebidos por pacotes enviados.

Figura 8: Evolução do SNR e perda de pacotes ao longo da duração da missão - Comparação entre os modelos de propagação implementados.

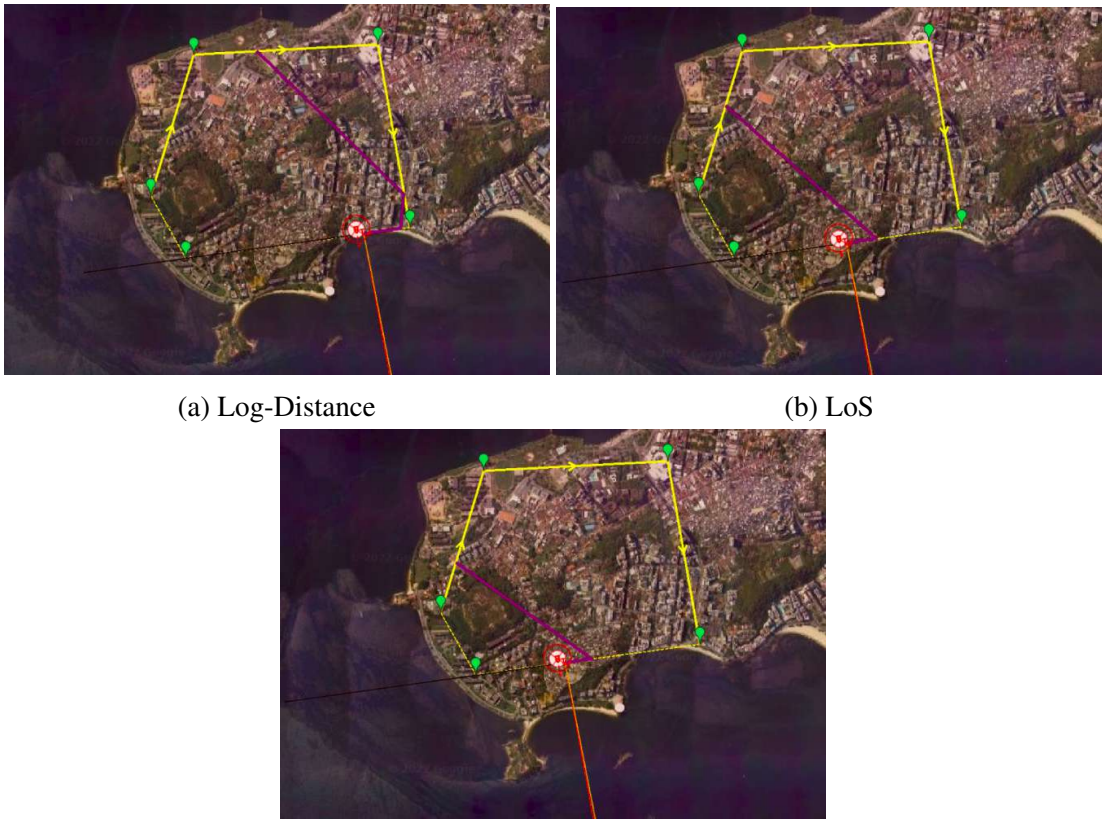
5. Conclusão

Um ambiente de simulação mais realista auxiliando no planejamento de missões é de grande valia para evitar fracassos e até mesmo a perda do equipamento devido às limitações dos *drones* como autonomia de voo e cobertura de sinal.

Nesse artigo, propusemos uma arquitetura de simulação baseada no SITL ArduPilot que permite o refinamento dos modelos de comunicação Ar-Terra permitindo simular falhas na comunicação e avaliar os impactos na missão. Através de um *proxy* que fica encarregado de encaminhar as mensagens trocadas entre o *drone* e a estação terrestre, é possível introduzir os modelos de propagação e erro encontrados na literatura tornando a simulação mais realista em relação à comunicação.

Para validar a arquitetura proposta, foram realizados experimentos utilizando o *Mission Planner* para criar e executar missões no SITL Ardupilot. Os resultados encontrados mostram que essa degradação de sinal varia com o modelo de propagação escolhido no momento da simulação e essa possibilidade de escolher ou implementar modelos mais específicos de propagação ilustra a capacidade modular e extensível da arquitetura proposta. Mesmo assim, os resultados ilustram também a inadequação da arquitetura de simulação original que ignora os aspectos de comunicação, podendo, assim, induzir o operador do *drone* a assumir a viabilidade de uma missão impossível devido a restrições de alcance de comunicação.

As ideias para trabalhos futuros incluem a comparação da degradação do sinal no ambiente simulado com voos utilizando *drones* reais para testar e avaliar a acurácia dos modelos de propagação atualmente implementados. Além disso, uma proposta de trabalho futuro seria a criação de um *framework* baseado na arquitetura proposta que simplifique a implementação de outros modelos de propagação e erro, de acordo com



(a) Log-Distance (b) LoS

(c) OLoS

Figura 9: Efeitos do modelos de propagação na observação da missão

as necessidades de cada usuário em particular. Também planeja-se a introdução de um modelo de atraso de comunicação, de forma a permitir avaliar o impacto deste aspecto na viabilidade das missões. Por fim, planeja-se fazer experimentos com missões pilotadas manualmente (ao invés de voos autônomos) para avaliar o quanto problemas de comunicação prejudicam a capacidade de controle da aeronave.

Referências

ArduPilot (2022a). Mavproxy. Available at `https://ardupilot.org/mavproxy/`.

ArduPilot (2022b). Mission planner. Available at `https://ardupilot.org/planner/`.

ArduPilot (2022c). Sitl simulator(software in the loop. Available at `https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html`.

Bastos, C. A. M., Passos, D., Barbosa, W. M., dos Santos Felipe, Y. S., Loureiro, T. B., dos Santos Dias, G., and Passos, F. G. O. (2022). Drones for civil defense: a case study in the city of niterói. In *14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*.

Coopmans, C., Podhradský, M., and Hoffer, N. V. (2015). Software- and hardware-in-the-loop verification of flight dynamics model and flight control simulation of a fixed-wing

unmanned aerial vehicle. In *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*.

DJI (2022). Dji flight simulator. Available at `https://www.dji.com/br/simulator`.

Dronecode, P. (2022). Mavlink. Available at `https://mavlink.io/en/`.

Feng, Q., McGeehan, J., Tameh, E., and Nix, A. (2006). Path loss models for air-to-ground radio channels in urban environments. In *2006 IEEE 63rd Vehicular Technology Conference*, volume 6, pages 2901–2905.

Gazebo (2022). Gazebo. Available at `http://https://gazebosim.org/home`.

Koubâa, A., Allouch, A., Alajlan, M., Javed, Y., Abdelfettah, and Khalgui, M. (2019). Micro air vehicle link (mavlink) in a nutshell: A survey. *IEEE Access, vol. 7, pp. 87658 - 87680, 2019*.

Lacage, M. and Henderson, T. R. (2006). Yet another network simulator. *Proceedings of the 2006 Workshop on ns-3*.

Lassabe, F., Canalda, P., Chatonnay, P., Spies, F., and Baala, O. (2005). A friis-based calibrated model for wifi terminals positioning. In *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, pages 382–387.

Lee, B. j., hwan Lee, S., and Rhee, S. h. (2010). Rate-adaptive mac protocol for efficient use of channel resource in wireless multi-hop networks. In *2010 Second International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 115–120.

Motlagh, N. H., Taleb, T., and Arouk, O. (2016). Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives. *IEEE Internet of Thigs Journal*.

Nekrasov, M., Allen, R., Artamonova, I., and Belding, E. (2019). Optimizing 802.15.4 outdoor iot sensor networks for aerial data collection. *Sensors*.

Research, M. (2022). Airsim. Available at `https://microsoft.github.io/AirSim/`.

Shakhatreh, H., Sawalmeh, A. H., Al-Fuqaha, A., (Senior Member, I., Dou, Z., Almaita, E., Khalil, I., Othman, N. S., Khreishah, A., Guizani, M., and (Fellow, I. (2019). Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access, vol. 7, pp. 48572 - 48634, 2019*.

Siddiqui, S. A., Fatima, N., and Ahmad, A. (2019). Comparative analysis of propagation path loss models in lte networks. In *2019 International Conference on Power Electronics, Control and Automation (ICPECA)*, pages 1–3.

Software, K. E. (2022). Realflight simulator. Available at `https://www.realflight.com/`.

Vergouw, B., Nagel, H., Bondt, G., and Curters, B. (2016). *Drone Technology: Types, Payloads, Applications, Frequency Spectrum Issues and Future Developments*, pages 21–45. T.M.C. Asser Press, The Hague.