



Projeto Brasil 6G Fase III

Atividade 3.3 - Aplicações de IoT para Redes 6G - Parte 2



HISTÓRICO DE ATUALIZAÇÕES:

Versão	Data	Autor(es)	Notas
1	15/07/2026	Aldebaro Klautau – UFPA Daniely Gomes Silva - Inatel Diego Gabriel Soares Pivoto - Inatel Elton Vivot – UFG Cláudio Modesto – UFPA Karlla Chaves – UFG Kleber Vieira Cardoso – UFG Samuel Baraldi Mafra – Inatel Sand Luz Correa – UFG Victoria Dala Pegorara Souto – Inatel	Elaboração de conteúdo
2	31/07/2026	Luciano Leonel Mendes – Inatel Richard Demo Souza – UFSC Vanessa Mendes Rennó – Inatel Victoria Dala Pegorara Souto – Inatel	Revisão de texto

Lista de Figuras

1	Ambiente virtual remodelado do Centro de Referência em Radiocomunicações (CRR).	5
2	Ambiente virtual do laboratório <i>Wireless and Artificial Intelligence</i> (WAI). . . .	6
3	Ambiente virtual do laboratório <i>Electromagnetics and Intelligent Surfaces</i> (EMIS). .	6
4	Sala de Demonstrações desenvolvida para o Metaverso Inatel.	7
5	Ambiente da UTI Virtual desenvolvido para o Metaverso Inatel.	7
6	Arquitetura do sistema proposto.	11
7	Interface do Eclipse Ditto e estrutura JSON correspondente ao gêmeo digital. . .	12
8	Tipos de ruídos introduzidos pelo gerador sintético do Cerealia.	13
9	Acurácia do modelo de detecção de ruídos para o <i>dataset</i> do INMET.	14
10	Arquitetura do eMR-Leo e exemplos de saídas das tarefas MAR implementadas. .	16
11	Cenário de teste utilizado nos experimentos.	18
12	Métricas de Tráfego IP para cada tarefa MAR.	19
13	Distribuição empírica e ajustada do número de instruções por quadro para a tarefa de SLAM (13a); qualidade do ajuste para o SLAM (13b); distribuição empírica e ajustada do número de instruções por quadro para a tarefa de detecção de objetos (13c); e qualidade do ajuste para a detecção de objetos (13d).	20
14	Teste KS para todas as métricas e <i>streams</i> usados para análise do tráfego IP. . .	21
15	Fluxo conceitual do processamento no FLEXA.	24
16	Arquitetura interna do FLEXA integrada à camada WebRTC.	25
17	Métricas de qualidade para entradas de 64×64 ampliadas para diferentes resoluções de saída.	27
18	Eficiência entre qualidade visual e tempo de processamento para entradas de 64×64 ampliadas para diferentes resoluções de saída.	27
19	Métricas de qualidade para saída fixa em 1024×1024 a partir de diferentes resoluções de entrada.	28
20	Eficiência entre qualidade visual e tamanho do fluxo transmitido. As métricas foram normalizadas pelo tamanho dos quadros transmitidos.	28
21	Protótipo desenvolvido. À esquerda, a coleira instrumentada; à direita, a PCB. .	35
22	<i>Dashboard</i> desenvolvido em Node-RED para visualização das variáveis coletadas e dos dados recebidos via MQTT.	36
23	Imagem do Treinamento - Detecção de Incêndios	40
24	Arquitetura da solução IoT proposta para detecção de incêndios.	40
25	Evolução do treinamento do YOLOv11	41
26	<i>Dashboard</i> de detecção dos incêndios	42
27	Resultados da detecção de fumaça e fogo utilizando a solução IoT embarcada em Raspberry Pi.	43
28	<i>Framework</i> proposto que integra um simulador de RT e um simulador de rede de eventos discretos, utilizado para simular cenários de LoRaWAN em diferentes camadas de rede.	47
29	Exemplo de saída gráfica de simulações com RT (potência recebida) avaliada utilizando WI e Sionna RT.	47
30	Organização dos dispositivos finais que devem ser cobertos por um conjunto de GWs em P posições.	49

31	Solução otimizada para o posicionamento de GW utilizando diferentes tipos de canais independentes do local.	52
32	Solução otimizada para o posicionamento de GWs nos cenários Etoile e Street Canyon.	53
33	CDF da potência recebida sob diferentes modelos de canal no padrão de malha 1.	53
34	Número de GWs implantados considerando diferentes valores de ρ no padrão de malha 1.	55
35	Mapa de cobertura da configuração otimizada considerando canais de RT do Sionna RT no cenário Etoile.	56
36	Taxa média de pacotes entregues avaliado considerando canais <i>site-specific</i> e <i>site-independent</i> em diferentes cenários de rede e padrões de malha.	56
37	Tempo de simulação, em minutos, para obter métricas da camada física em diferentes modelos de canal e cenários.	58
38	<i>Trade-off</i> entre tempo computacional e EMQ, considerando a potência recebida pelos dispositivos finais do WI X3D como <i>ground truth</i>	59

Lista de Tabelas

1	Visão geral do modelo de geração de carga.	22
2	Custo estimado do protótipo desenvolvido.	36
3	Comparação do Desempenho dos modelos de Detecção de Incêndios Florestais. .	41
4	Características das malhas de dispositivos finais usados em diferentes cenários .	49
5	Parâmetros de simulação de RT utilizados nos experimentos com canais especí- ficos do local usando WI e Sionna RT	50
6	Ferramentas utilizadas no ambiente de desenvolvimento para as simulações . . .	51
7	Número de GWs sugeridos, desvio padrão e potência média recebida dos disposi- tivos finais (em dBm) para cada modelo de canal em diferentes padrões de malha.	54
8	Erro relativo em termos da potência recebida dos dispositivos finais, considerando a potência recebida do modelo WI X3D como referência (<i>ground-truth</i>)	54
9	Valores médios e de desvio padrão do tempo de simulação (minutos) e da taxa de pacotes entregues (%) considerando todos os GWs escolhidos em diferentes ambientes de rede	57

Acrônimos

3GPP	<i>3rd Generation Partnership Project</i>
5G	Quinta Geração de Rede Móvel Celular
6G	Sexta Geração de Rede Móvel Celular
AMQP	<i>Advanced Message Queuing Protocol</i>
API	<i>Application Programming Interface</i>
CDF	<i>Cumulative Distribution Function</i>
CNN	<i>Convolutional Neural Network</i>
CPU	<i>Central Processing Unit</i>
CRR	Centro de Referência em Radiocomunicações
CUDA	<i>Compute Unified Device Architecture</i>
DAE	<i>Digital Asset Exchange</i>
EMIS	<i>Electromagnetics and Intelligent Surfaces</i>
EMQ	Erro Médio Quadrático
eMR-Leo	<i>enhanced Mixed Reality Linköping Edge Offloading</i>
FFHQ	<i>Flickr-Faces-HQ</i>
FPS	<i>Frames Per Second</i>
GPS	<i>Global Positioning System</i>
GPU	<i>Graphics Processing Unit</i>
GRU	<i>Gated Recurrent Unit</i>
GW	<i>Gateway</i>
IA	Inteligência Artificial
INMET	Instituto Nacional de Meteorologia
IoT	<i>Internet of Things</i>
ITU	<i>International Telecommunication Union</i>
LLM	<i>Large Language Model</i>
LoRa	<i>Long Range</i>
LoRaWAN	<i>Long Range Wide Area Network</i>
LPIPS	<i>Learned Perceptual Image Patch Similarity</i>
LPWAN	<i>Low Power Wide Area Network</i>
MAR	<i>Mobile Augmented Reality</i>
MI	milhões de instruções
MQTT	<i>Message Queuing Telemetry Transport</i>
MR-Leo	<i>Mixed Reality Linköping Edge Offloading</i>
PCB	<i>Printed Circuit Board</i>

PSNR *Peak Signal-to-Noise Ratio*
QoS *Quality of Service*
RAG *Retrieval-Augmented Generation*
R-CNN *Region-based Convolutional Neural Network*
RIS *Reconfigurable Intelligent Surfaces*
RNN *Recurrent Neural Network*
RSSI *Received Signal Strength Indicator*
RT *Ray Tracing*
RTP *Real-time Transport Protocol*
SF *Spreading Factor*
SLAM *Simultaneous Localization and Mapping*
SoC *State of Charge*
SRGAN *Super-Resolution Generative Adversarial Network*
SSIM *Structural Similarity Index Measure*
TCP *Transmission Control Protocol*
UDP *User Datagram Protocol*
UE *User Equipment*
USB *Universal Serial Bus*
UTI *Unidade de Terapia Intensiva*
VMAF *Video Multi-Method Assessment Fusion*
VRAM *Video Random Access Memory*
WAI *Wireless and Artificial Intelligence*
WebRTC *Web Real-Time Communication*
WET *Wireless Energy Transfer*
WI *Wireless InSite[®]*
YOLO *You Only Look Once*

Sumário

1	Introdução	1
2	Evolução do Metaverso Inatel com Integração de Avatar Inteligente e Gêmeo Digital para UTI Virtual e <i>Smart Farm</i>	4
2.1	Introdução	4
2.2	Evolução dos Ambientes Virtuais	4
2.2.1	Remodelagem do Centro de Referência em Radiocomunicações	5
2.2.2	Criação do Laboratório <i>Wireless and Artificial Intelligence</i>	5
2.2.3	Criação do Laboratório <i>Electromagnetics and Intelligent Surfaces</i>	5
2.2.4	Criação da Sala de Demonstrações	5
2.2.5	Criação da UTI Virtual	6
2.2.6	Representação de uma <i>Smart Farm</i> - Fazenda Delta	8
2.3	Integração de Inteligência Artificial e Gêmeos Digitais ao Metaverso Inatel	8
2.3.1	Avatar Inteligente para Apoio às Pesquisas do Projeto Brasil 6G	8
2.3.2	Gêmeo Digital para a UTI Virtual	8
2.3.3	Gêmeo Digital para <i>Smart Farm</i>	9
2.4	Conclusões	9
3	Gêmeo Digital para Estações Meteorológicas com Detecção de Inconsistências e Suporte à Atuação em Sistemas Ciberfísicos	10
3.1	Introdução	10
3.2	Sistema proposto	10
3.3	Implementação	12
3.4	Conclusões	13
4	Caracterização de Tarefas de Realidade Aumentada Móvel Suportadas por Recursos de Computação em Borda	15
4.1	Introdução	15
4.2	<i>Enhanced</i> MR-Leo	16
4.3	Análise do tráfego IP	17
4.4	Caracterização das cargas de trabalho	18
4.5	Conclusões Parciais	21
5	Transmissão de Vídeo em Redes Sem Fio Utilizando Comunicação Semântica	23
5.1	Introdução	23
5.2	Sistema proposto	24
5.3	Avaliação e resultados	26
5.4	Conclusões e próximos passos	29
6	Sistema de IoT Baseado em LoRa para Monitoramento da Fauna Sul-Mineira	30
6.1	Introdução	30
6.2	Metodologia de Projeto	31
6.2.1	Definição das Variáveis de Monitoramento	32
6.2.2	Definição e Integração dos Componentes de <i>Hardware</i> e <i>Software</i>	32
6.2.3	Estimativa de Posição via RSSI	34
6.2.4	Nuvem e visualização	34

6.3	Protótipo Desenvolvido e Resultados Obtidos	34
6.3.1	Estimativa de custo do protótipo	35
6.4	Conclusões	36
7	Detecção Inteligente de Incêndios Florestais Usando Modelos de Aprendizado Profundo	38
7.1	Introdução	38
7.2	Solução Proposta	39
7.3	Resultados	41
7.4	Conclusão	43
8	Posicionamento de <i>Gateways</i> LoRaWAN para Planejamento de Rede Usando Modelos de Canal Baseados em Traçado de Raios	45
8.1	Introdução	45
8.2	Simulações de comunicações LoRaWAN com RT	46
8.3	RT-LoRaWAN <i>framework</i>	46
8.4	Otimização no posicionamento de <i>gateways</i>	48
8.5	Experimentos	48
8.6	Resultados	51
8.6.1	Análise de posicionamento e cobertura	51
8.6.2	Análise paramétrica com diferentes limiares	54
8.6.3	Análise de desempenho de métricas de alto nível	55
8.6.4	Análise de custo computacional	57
8.7	Conclusões	58
9	Conclusão	60

1 Introdução

Dando continuidade às atividades desenvolvidas nas Fases I e II do projeto Brasil 6G, a Atividade 3.3 da Fase III concentra-se na proposição, implementação e avaliação de soluções tecnológicas voltadas a aplicações de *Internet of Things* (IoT), sistemas ciberfísicos, comunicações sem fio inteligentes e serviços imersivos, todos componentes fundamentais para a consolidação do ecossistema da Sexta Geração de Rede Móvel Celular (6G). A crescente demanda por conectividade massiva, processamento distribuído, baixa latência, confiabilidade, eficiência energética e integração nativa entre comunicação, computação e Inteligência Artificial (IA) reforça a necessidade de arquiteturas adaptativas, escaláveis e sensíveis ao contexto. Nesse cenário, tecnologias como ambientes virtuais imersivos, gêmeos digitais, computação em borda, comunicação semântica, redes *Low Power Wide Area Network* (LPWAN), sensoramento inteligente e aprendizado profundo assumem papel estratégico para viabilizar aplicações em realidade aumentada móvel, agricultura de precisão, monitoramento ambiental, conservação da fauna, prevenção de desastres e planejamento eficiente de redes.

As contribuições apresentadas neste relatório refletem diferentes dimensões das tecnologias habilitadoras das redes 6G. Por um lado, investigam-se soluções voltadas ao suporte de aplicações computacionalmente intensivas e sensíveis à latência, como realidade aumentada móvel e transmissão de vídeo em tempo real. Por outro lado, exploram-se sistemas IoT e ciberfísicos aplicados a cenários de monitoramento ambiental, atuação inteligente e preservação de ecossistemas. Também são abordadas técnicas de planejamento e otimização de redes de comunicação sem fio, essenciais para garantir cobertura, confiabilidade e eficiência em implantações de larga escala. De forma integrada, os trabalhos desenvolvidos contribuem para o avanço de soluções alinhadas às exigências das futuras redes 6G, combinando experimentação, modelagem, prototipagem, simulação e integração de modelos de IA.

O Capítulo 2 apresenta a evolução da plataforma Metaverso Inatel, contemplando a criação de novos ambientes virtuais e a remodelagem de espaços previamente desenvolvidos, com o objetivo de ampliar sua capacidade de representação das pesquisas conduzidas no âmbito do Projeto Brasil 6G. Entre as evoluções realizadas, destacam-se a incorporação dos laboratórios *Wireless and Artificial Intelligence* (WAI) e *Electromagnetics and Intelligent Surfaces* (EMIS), da Sala de Demonstrações, da Unidade de Terapia Intensiva (UTI) Virtual e da *Smart Farm*, além da atualização de ambientes existentes. O capítulo também descreve o desenvolvimento do avatar inteligente baseado em arquitetura *Retrieval-Augmented Generation* (RAG) e do gêmeo digital da incubadora neonatal para integração à UTI Virtual, bem como apresenta a integração do gêmeo digital da *Smart Farm*, cujo desenvolvimento é detalhado no capítulo seguinte.

O Capítulo 3 aborda o desenvolvimento de um ambiente de gêmeo digital para estações meteorológicas, com foco na detecção de inconsistências, suporte à decisão e atuação em sistemas ciberfísicos. A arquitetura proposta integra uma camada física de sensoramento e atuação com uma camada digital responsável pela representação virtual da entidade monitorada, armazenamento de dados, execução de modelos de IA e simulação de cenários. A solução utiliza plataformas como Eclipse Hono e Eclipse Ditto para o gerenciamento de dispositivos, comunicação entre os componentes do sistema e representação digital da estação meteorológica, além de modelos neurais voltados à análise de séries temporais meteorológicas. O objetivo é viabilizar aplicações em agricultura de precisão e infraestruturas inteligentes, nas quais dados ambientais possam ser continuamente coletados, analisados e utilizados para apoiar decisões ou acionar mecanismos automáticos no ambiente físico.

O Capítulo 4 apresenta a caracterização de tarefas de *Mobile Augmented Reality* (MAR)

suportadas por recursos de computação em borda, com foco no protótipo *enhanced Mixed Reality Linköping Edge Offloading* (eMR-Leo). A proposta estende o protótipo *Mixed Reality Linköping Edge Offloading* (MR-Leo) original para incorporar execução heterogênea em *Central Processing Unit* (CPU) e *Graphics Processing Unit* (GPU), contemplando tarefas fundamentais como *Simultaneous Localization and Mapping* (SLAM) e detecção de objetos. São analisados aspectos relacionados à demanda computacional, uso de memória e tráfego IP gerado pelas aplicações, permitindo a construção de modelos estatísticos capazes de representar cargas de trabalho realistas em ambientes de borda. Os resultados evidenciam diferenças importantes entre tarefas predominantemente limitadas por CPU, como SLAM, e tarefas altamente paralelizáveis, como detecção de objetos, destacando a importância de infraestruturas heterogêneas para aplicações imersivas em redes futuras.

O Capítulo 5 apresenta o FLEXA, um arcabouço modular para transmissão de vídeo em redes sem fio utilizando comunicação semântica. A proposta visa desacoplar modelos de IA da infraestrutura de transporte, permitindo a integração dinâmica de diferentes técnicas de processamento semântico ao fluxo de transmissão de vídeo. O sistema é implementado sobre *Web Real-Time Communication* (WebRTC) e utiliza orquestradores orientados por políticas para gerenciar codificadores, decodificadores, modelos neurais e decisões de execução em tempo real. A avaliação experimental considera um estudo de caso com super-resolução baseada em *Super-Resolution Generative Adversarial Network* (SRGAN), analisando compromissos entre qualidade visual, custo computacional, tamanho do fluxo transmitido e políticas de operação. Os resultados demonstram o potencial da comunicação semântica para reduzir o volume de dados transmitidos e preservar a qualidade perceptual em cenários com restrições de banda e processamento.

O Capítulo 6 descreve o desenvolvimento de uma solução nacional de IoT baseada em *Long Range* (LoRa) para monitoramento da fauna sul-mineira. A proposta consiste em uma coleira instrumentada capaz de coletar variáveis ambientais, comportamentais e energéticas, transmitindo os dados por meio de comunicação de longo alcance e baixo consumo. A arquitetura utiliza múltiplos *Gateways* (GWs) para recepção dos sinais e exploração de medidas de *Received Signal Strength Indicator* (RSSI) como suporte à estimativa aproximada de localização. Além disso, os dados são publicados em nuvem por meio de *Message Queuing Telemetry Transport* (MQTT) e visualizados em um *dashboard*, permitindo o acompanhamento remoto das informações coletadas. Os resultados obtidos em ambiente de bancada demonstram a viabilidade funcional da solução proposta e seu potencial como alternativa de menor custo para aplicações de monitoramento remoto e conservação da fauna.

O Capítulo 7 apresenta uma solução inteligente para detecção e monitoramento de incêndios florestais baseada em modelos de aprendizado profundo. A abordagem utiliza modelos de visão computacional, como *You Only Look Once* (YOLO)v11, YOLOv8 e *Region-based Convolutional Neural Network* (R-CNN), para identificar focos de incêndio em imagens coletadas em diferentes condições ambientais. A solução contempla treinamento supervisionado com imagens rotuladas, avaliação comparativa entre modelos e validação experimental em uma arquitetura IoT embarcada, composta por câmera, Raspberry Pi, transmissão de dados e visualização remota em *dashboard*. Os resultados indicam desempenho superior do modelo YOLOv11 nas métricas avaliadas e demonstram a capacidade do sistema em detectar fogo e fumaça em tempo real, reforçando o potencial da IA para aplicações de prevenção, monitoramento ambiental e resposta rápida a eventos críticos.

O Capítulo 8 investiga o posicionamento de GWs *Long Range Wide Area Network* (LoRaWAN) para planejamento de rede utilizando modelos de canal baseados em traçado de raios. O tra-

balho propõe um *framework* que integra simuladores de *Ray Tracing* (RT), como Sionna e *Wireless InSite*[®] (WI), ao simulador de eventos discretos NS-3, permitindo avaliar o impacto de diferentes modelos de canal no planejamento de redes LoRaWAN. O problema de posicionamento é formulado como uma otimização inteira binária, buscando minimizar o número de GWs necessários para atender restrições de potência recebida nos dispositivos finais. Os resultados demonstram que a escolha do modelo de canal influencia significativamente tanto o número quanto a posição dos GWs, evidenciando o compromisso entre fidelidade da simulação, custo computacional e confiabilidade das soluções de planejamento.

Por fim, o Capítulo 9 apresenta as conclusões gerais do relatório, sintetizando os principais resultados alcançados, as contribuições técnicas desenvolvidas e as perspectivas de continuidade das pesquisas no âmbito da Atividade 3.3 da Fase III do projeto Brasil 6G.

2 Evolução do Metaverso Inatel com Integração de Avatar Inteligente e Gêmeo Digital para UTI Virtual e *Smart Farm*

2.1 Introdução

Este capítulo descreve os avanços realizados na plataforma do Metaverso Inatel, cujas atividades anteriores foram apresentadas na Seção 4.8 do relatório de Aplicações para Redes 6G e Integração da Fase II do Projeto Brasil 6G [1]. O trabalho desta etapa concentrou-se na expansão da plataforma por meio da criação de novos ambientes virtuais, da remodelagem de espaços previamente desenvolvidos e da incorporação de funcionalidades destinadas à demonstração das pesquisas conduzidas no âmbito do Projeto Brasil 6G. O objetivo principal foi ampliar o alcance do metaverso como ambiente de demonstração tecnológica, permitindo representar, em um único espaço virtual, diferentes laboratórios, aplicações e cenários associados às linhas de pesquisa do projeto.

Entre as realizações desta etapa destacam-se a criação dos ambientes virtuais da Sala de Demonstrações, da UTI Virtual, da Fazenda Delta e dos laboratórios WAI e EMIS, além da remodelagem do ambiente do Centro de Referência em Radiocomunicações (CRR). Complementarmente, foram implementadas melhorias no campus externo do Inatel, abrangendo atualização da iluminação, das texturas e do sombreamento, bem como otimizações gráficas voltadas ao aumento da fidelidade visual e ao desempenho da aplicação em realidade virtual.

Paralelamente ao trabalho nos ambientes virtuais, avançou-se no desenvolvimento de funcionalidades que integram IA e gêmeos digitais à plataforma. Um avatar inteligente foi projetado para acompanhar visitantes nos laboratórios virtuais e responder a perguntas técnicas sobre as pesquisas desenvolvidas no Projeto Brasil 6G. Na UTI Virtual, iniciou-se o desenvolvimento do gêmeo digital de uma incubadora neonatal, que servirá de base para futuras aplicações de comunicação entre dispositivos físicos e suas representações no metaverso. Aplicações semelhantes, voltadas ao cenário de *Smart Farm*, também integram esta fase e são descritas em capítulo próprio do relatório.

2.2 Evolução dos Ambientes Virtuais

O Metaverso Inatel cresceu de forma considerável nesta etapa, tanto pela incorporação de novos espaços quanto pela atualização dos que já existiam. O ponto de partida foi ampliar a cobertura da plataforma em relação à infraestrutura física do Inatel, criando ambientes adequados para demonstrações técnicas, visitas institucionais e apresentação das tecnologias desenvolvidas no projeto. Ao mesmo tempo, foram feitas melhorias de qualidade visual que afetaram texturas, iluminação e sombreamento de forma generalizada, com o objetivo de aproximar a experiência virtual da aparência real dos laboratórios e de melhorar o desempenho da aplicação.

A criação desses novos cenários virtuais, associada à remodelagem dos espaços já existentes, ampliou o escopo do Metaverso Inatel e estabeleceu a infraestrutura necessária para a incorporação das funcionalidades descritas na seção 2.3.

O conteúdo deste capítulo foi desenvolvido pelos pesquisadores *Daniely Gomes Silva* e *Diego Gabriel Soares Pivoto*.

2.2.1 Remodelagem do Centro de Referência em Radiocomunicações

O ambiente virtual do CRR, desenvolvido na fase anterior do projeto [1], passou por remodelagem com o objetivo de atualizar sua representação visual e adequá-la às novas funcionalidades da plataforma. As modificações abrangeram revisão de iluminação, sombreamento, texturização e otimização dos modelos tridimensionais, resultando em uma representação mais fiel ao laboratório físico e em uma melhor experiência de navegação. Essas atualizações também contribuíram para uniformizar o padrão visual adotado nos demais espaços desenvolvidos nesta etapa. O ambiente resultante pode ser observado na Figura 1.

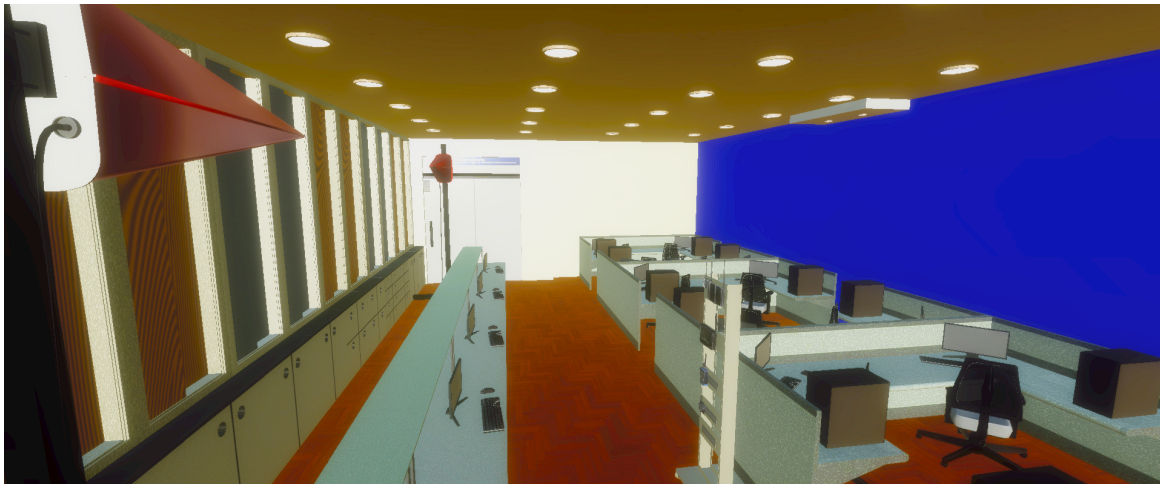


Figura 1: Ambiente virtual remodelado do Centro de Referência em Radiocomunicações (CRR).

2.2.2 Criação do Laboratório *Wireless and Artificial Intelligence*

O laboratório WAI, dedicado às pesquisas em comunicações sem fio e à aplicação de IA em redes de telecomunicações, foi incorporado ao Metaverso Inatel como um dos novos ambientes desta etapa. Sua versão virtual, apresentada na Figura 2, amplia o conjunto de laboratórios disponíveis na plataforma, permitindo que as atividades desenvolvidas nesse espaço sejam apresentadas em demonstrações realizadas no metaverso.

2.2.3 Criação do Laboratório *Electromagnetics and Intelligent Surfaces*

Outro ambiente incorporado ao Metaverso Inatel foi o laboratório EMIS, dedicado às pesquisas em *Reconfigurable Intelligent Surfaces* (RIS), à otimização de sistemas *Wireless Energy Transfer* (WET) e ao projeto e desenvolvimento de ressonadores eletromagnéticos para diversas aplicações em comunicações e sensores [2]. A representação virtual desse laboratório, mostrada na Figura 3, amplia o conjunto de cenários destinados à demonstração das pesquisas desenvolvidas no Inatel, permitindo apresentar, em um ambiente imersivo, tecnologias relacionadas ao controle inteligente da propagação eletromagnética e às aplicações de transferência de energia sem fio.

2.2.4 Criação da Sala de Demonstrações

A Sala de Demonstrações foi criada para funcionar como ponto central de apresentação das tecnologias do Projeto Brasil 6G durante visitas técnicas e institucionais. O ambiente



Figura 2: Ambiente virtual do laboratório *Wireless and Artificial Intelligence* (WAI).



Figura 3: Ambiente virtual do laboratório *Electromagnetics and Intelligent Surfaces* (EMIS).

reúne aplicações, vídeos e protótipos virtuais em um único espaço, facilitando a condução de demonstrações organizadas e acessíveis a públicos variados. Seu *layout* pode ser visto na Figura 4.

2.2.5 Criação da UTI Virtual

A UTI Virtual foi concebida para representar um cenário hospitalar voltado à demonstração de aplicações em saúde digital. O ambiente reproduz um leito de terapia intensiva e uma incubadora neonatal com equipamentos e elementos característicos desse tipo de instalação, conforme mostrado na Figura 5. A infraestrutura retratada é necessária para a integração de aplicações de monitoramento e demonstração de dispositivos médicos no Metaverso Inatel.

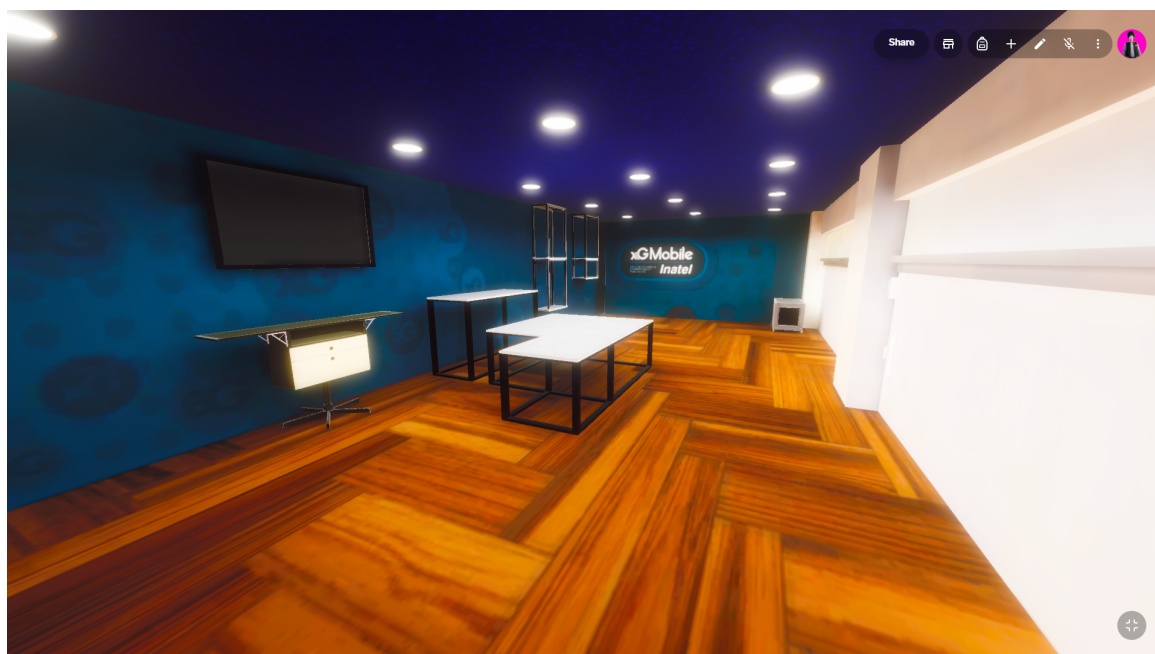


Figura 4: Sala de Demonstrações desenvolvida para o Metaverso Inatel.

A adoção de ambientes hospitalares virtuais como plataforma de demonstração tecnológica tem ganhado relevância no contexto das comunicações de próxima geração, em especial para aplicações de telemedicina e saúde conectada [3].

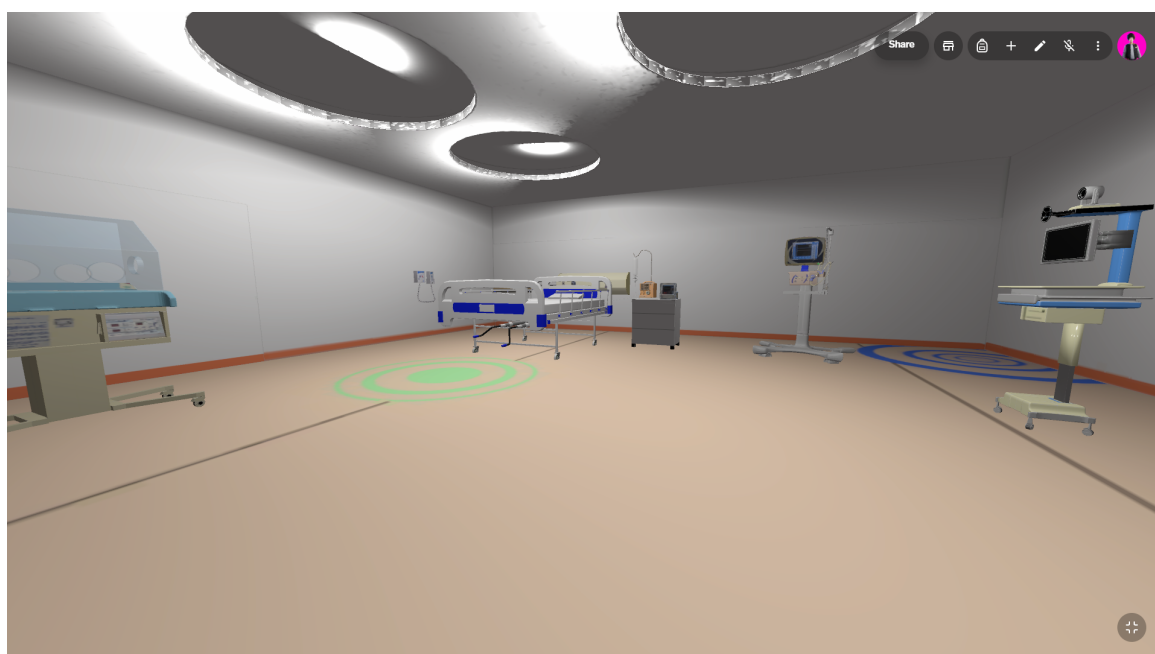


Figura 5: Ambiente da UTI Virtual desenvolvido para o Metaverso Inatel.

2.2.6 Representação de uma *Smart Farm* - Fazenda Delta

A Fazenda Delta foi digitalizada e incorporada ao metaverso para demonstrar aplicações de conectividade e transformação digital no agronegócio. O ambiente foi construído com base na estrutura física da fazenda e permite representar cenários de monitoramento remoto e uso de dispositivos IoT em área rural. Para tornar a experiência mais atrativa, foram incluídos elementos interativos, como animais animados e maquinário virtualmente operável. O espaço também foi estruturado para integrar aplicações de gêmeo digital, o que amplia as possibilidades de demonstração de soluções voltadas à agricultura inteligente e à conectividade em regiões remotas, casos de uso relevantes para redes 6G [4].

2.3 Integração de Inteligência Artificial e Gêmeos Digitais ao Metaverso Inatel

Além da evolução dos ambientes virtuais, esta etapa contemplou a incorporação de novas funcionalidades ao Metaverso Inatel, com destaque para o desenvolvimento de um avatar inteligente e a implementação de gêmeos digitais voltados à UTI Virtual e à *Smart Farm*. Essas funcionalidades ampliam as possibilidades de interação entre usuários, ambientes virtuais e aplicações baseadas em IA e sistemas ciberfísicos, aproximando o metaverso de uma plataforma de demonstração operacional das pesquisas do Projeto Brasil 6G.

2.3.1 Avatar Inteligente para Apoio às Pesquisas do Projeto Brasil 6G

O desenvolvimento do avatar inteligente teve como objetivo disponibilizar aos visitantes do Metaverso Inatel um mecanismo de consulta capaz de responder a perguntas técnicas sobre as pesquisas do Projeto Brasil 6G durante a navegação pelos laboratórios virtuais. Diferentemente de assistentes vinculados a um ambiente específico, o avatar foi concebido para atuar de forma transversal nos espaços do metaverso, acompanhando o usuário ao longo de sua experiência.

Para viabilizar essa funcionalidade, foram conduzidos estudos sobre *Large Language Models* (LLMs) e sobre a arquitetura RAG [5], que permite integrar modelos de linguagem a bases documentais externas, fundamentando as respostas geradas na documentação técnica do projeto. A arquitetura RAG tem se mostrado especialmente adequada para domínios especializados, nos quais a precisão e a rastreabilidade das informações são requisitos críticos [6]. Foram também investigadas tecnologias de integração entre aplicações, incluindo FastAPI e WebSocket, com vistas ao estabelecimento da comunicação entre o modelo de IA e o ambiente virtual desenvolvido em *Unity*.

Como resultado, foi implementado um modelo inicial baseado na arquitetura RAG, utilizando documentos técnicos produzidos ao longo do Projeto Brasil 6G como base de conhecimento. Essa implementação constitui a infraestrutura inicial para a incorporação do avatar inteligente aos laboratórios do metaverso, com previsão de evolução nas etapas subsequentes do projeto.

2.3.2 Gêmeo Digital para a UTI Virtual

Em paralelo ao desenvolvimento do avatar inteligente, foi iniciada a implementação do gêmeo digital de uma incubadora neonatal para integração ao ambiente da UTI Virtual. Essa atividade corresponde à primeira etapa do desenvolvimento da aplicação e tem como objetivo

estabelecer a infraestrutura necessária para futuras integrações entre o equipamento físico e sua representação virtual no metaverso.

Para isso, foram realizados estudos sobre plataformas embarcadas baseadas em Arduino e ESP32, responsáveis pela aquisição e processamento de dados do protótipo físico, e sobre o protocolo MQTT [7], amplamente adotado em aplicações de IoT pela sua leveza e adequação a redes com recursos limitados. Essas atividades permitiram definir a arquitetura inicial de comunicação entre os componentes físicos e o ambiente virtual.

Como parte dessa etapa, foi iniciada a montagem do protótipo físico da incubadora neonatal, estabelecendo a infraestrutura necessária para o desenvolvimento do gêmeo digital e sua futura integração ao ambiente da UTI Virtual do Metaverso Inatel.

2.3.3 Gêmeo Digital para *Smart Farm*

Além das aplicações voltadas ao ambiente hospitalar, o processo de evolução do Metaverso Inatel também contempla a incorporação de aplicações baseadas em gêmeos digitais para o setor do agronegócio. Conforme previsto na evolução da plataforma, foi realizada a digitalização da Fazenda Delta, destinada à demonstração de tecnologias relacionadas à conectividade rural, IoT e agricultura inteligente. Nesse contexto, o desenvolvimento de aplicações de gêmeo digital para *Smart Farm* integra a evolução da plataforma e complementa os cenários de demonstração disponíveis no metaverso.

A aplicação de gêmeos digitais no contexto agrícola, combinada à conectividade provida por redes de próxima geração, representa uma das vertentes mais promissoras para a transformação digital do setor rural [8]. Os detalhes dessa implementação são apresentados no capítulo 3 deste relatório.

2.4 Conclusões

As atividades desenvolvidas nesta etapa ampliaram o escopo do Metaverso Inatel e consolidaram sua utilização como plataforma de demonstração das pesquisas do Projeto Brasil 6G. A criação de novos ambientes virtuais, aliada à remodelagem dos espaços já existentes, resultou em uma representação mais abrangente da infraestrutura de pesquisa do projeto e em um ambiente mais adequado para demonstrações técnicas e visitas institucionais.

No plano das funcionalidades, o desenvolvimento do avatar inteligente e a implementação inicial do gêmeo digital da incubadora neonatal estabeleceram a base para a incorporação de IA e sistemas ciberfísicos à plataforma. Essas iniciativas, ainda em curso, indicam a trajetória de evolução do metaverso para além de um ambiente de visualização, em direção a uma plataforma interativa e operacionalmente integrada aos sistemas físicos do projeto.

A incorporação de aplicações de gêmeo digital para *Smart Farm* reforça essa direção, estendendo o alcance do metaverso a cenários de agricultura inteligente e conectividade rural, consideradas áreas centrais para os casos de uso das redes 6G. Em conjunto, as evoluções apresentadas nesta etapa consolidam o Metaverso Inatel como um ambiente integrado de demonstração tecnológica, capaz de representar de forma imersiva e interativa as pesquisas conduzidas no âmbito do Projeto Brasil 6G.

3 Gêmeo Digital para Estações Meteorológicas com Detecção de Inconsistências e Suporte à Atuação em Sistemas Ciberfísicos

3.1 Introdução

O avanço de sistemas ciberfísicos, da IoT e das redes móveis tem impulsionado a instrumentação contínua de ambientes físicos por meio de sensores distribuídos e de dispositivos inteligentes conectados. No contexto do monitoramento ambiental e da agricultura de precisão, essas tecnologias desempenham um papel fundamental ao viabilizar a coleta, transmissão e processamento, em tempo real, de variáveis como temperatura, umidade, radiação solar e velocidade do vento [9]. Neste cenário, a rede 6G oferece comunicação ubíqua, integração nativa entre IA e rede, suporte massivo a dispositivos IoT e serviços de computação distribuída na borda, criando um ecossistema propício para aplicações sensíveis à latência e orientadas a dados [10]. Entretanto, a confiabilidade das informações coletadas permanece um desafio significativo, uma vez que sensores estão sujeitos a falhas, ruídos, interferências ambientais e problemas de comunicação que afetam a qualidade dos dados utilizados [11].

Recentemente, o conceito de gêmeo digital ganhou notoriedade como uma abordagem promissora para a representação virtual de sistemas físicos, permitindo não apenas o espelhamento do estado atual do sistema, mas também a incorporação de mecanismos de análise e de sanitização de dados, simulação e suporte à decisão [12, 13, 14]. Em particular, gêmeos digitais podem integrar dados históricos e dados em tempo real para detectar inconsistências, prever comportamentos e avaliar o impacto de diferentes cenários operacionais.

No domínio da agricultura de precisão, em que aplicações como controle de irrigação, mitigação de estresse térmico e previsão de eventos climáticos adversos são altamente sensíveis a erros, o uso de gêmeos digitais torna-se especialmente relevante [15, 16, 17, 18]. Por exemplo, em [18], os autores demonstram que a utilização de gêmeos digitais pode contribuir para a detecção de anomalias e inconsistências em fluxos de dados meteorológicos, por meio da combinação de dados históricos, geração de cenários sintéticos e aplicação de modelos de aprendizado de máquina. No entanto, o trabalho não explora aspectos fundamentais para ecossistemas inteligentes de próxima geração, como o uso de plataformas padronizadas de gêmeo digital e a integração com ambientes imersivos, nos quais os usuários podem visualizar, interagir e experimentar diferentes cenários operacionais sobre a representação virtual da entidade física. Além disso, não contempla o fechamento do ciclo de controle por meio da atuação automática ou assistida sobre o ambiente real. Nesse contexto, este trabalho apresenta a evolução de um ambiente de gêmeo digital concebido com esse propósito, integrando mecanismos de monitoramento ambiental, representação virtual imersiva, suporte à tomada de decisão e interação bidirecional entre os mundos físico e digital, visando aplicações em agricultura inteligente e infraestruturas conectadas alinhadas às futuras redes 6G.

3.2 Sistema proposto

O sistema proposto neste trabalho tem como objetivo o desenvolvimento de um ambiente de gêmeo digital para monitoramento ambiental e suporte à tomada de decisão em aplicações de agricultura de precisão. A proposta é inspirada no arcabouço Cerealia [18], no qual gêmeos

O conteúdo deste capítulo foi desenvolvido pelos pesquisadores *Karlla Chaves Rodrigues* e *Sand Luz Correa*.

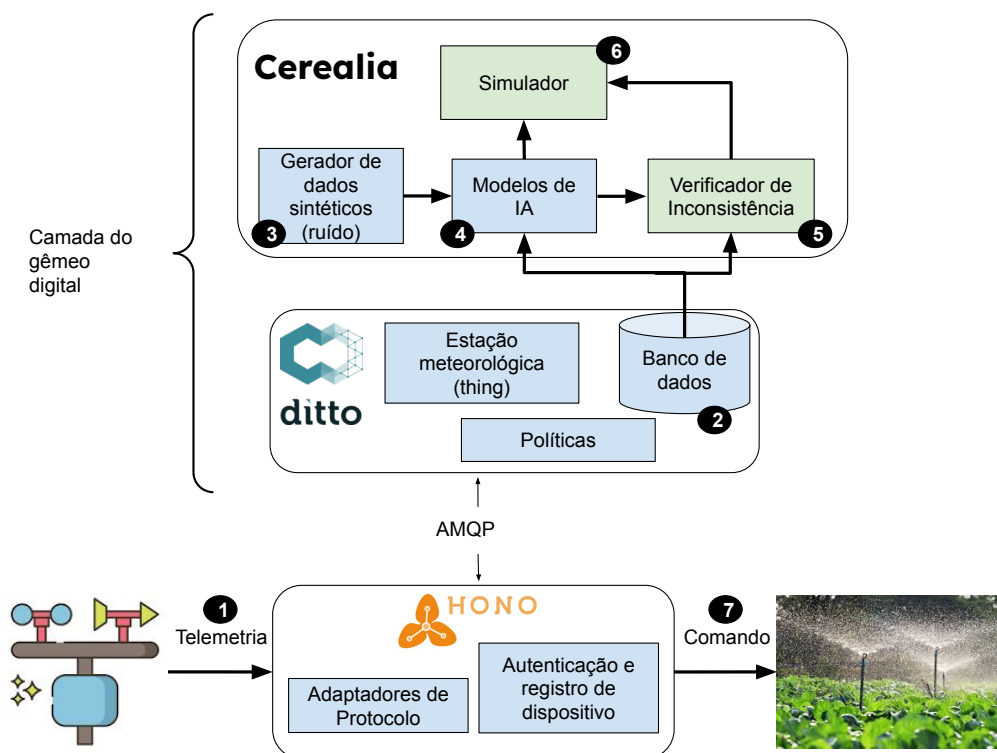


Figura 6: Arquitetura do sistema proposto.

digitais são utilizados para detectar inconsistências em dados meteorológicos e auxiliar processos de decisão agrícola baseados em informações ambientais. Especificamente, a Figura 6 apresenta uma visão geral do sistema. A arquitetura é organizada em duas camadas principais: a camada física, composta pela infraestrutura de sensoriamento e atuação em campo, e a camada do gêmeo digital, responsável pela representação virtual da entidade física, armazenamento de dados, execução de modelos de IA e mecanismos de simulação.

Na camada física, uma estação meteorológica realiza a coleta contínua de variáveis ambientais, como temperatura, umidade, velocidade do vento e radiação solar (bloco 1 da Figura 6). Esses dados são enviados em tempo real por meio de mensagens de telemetria para a plataforma de comunicação e gerenciamento de dispositivos baseada no Eclipse Hono¹. Conforme ilustrado na Figura 6, o Hono é responsável pelos mecanismos de autenticação e registro de dispositivos, além da adaptação de protocolos de comunicação utilizados pelos sensores em campo.

Após a recepção dos dados de telemetria, as informações são encaminhadas via protocolo *Advanced Message Queuing Protocol* (AMQP) para a camada do gêmeo digital, implementada sobre a plataforma Eclipse Ditto². Nessa camada, a estação meteorológica é representada virtualmente por uma entidade do tipo *thing* (conceito utilizado pelo Eclipse Ditto), responsável por manter o estado digital do dispositivo físico e suas propriedades associadas (representadas por *attributes* – informações estáticas ou semi-estáticas sobre o objeto físico – e *features* – usadas para modelar o estado operacional do objeto físico). Além disso, o Ditto disponibiliza mecanismos de políticas de acesso e controle, permitindo gerenciar permissões de leitura, escrita e atuação sobre os dispositivos monitorados. Por fim, o Ditto também disponibiliza um banco de dados usado para manter um histórico dos dados coletados na camada física (bloco 2 da

¹<https://eclipse.dev/hono/>

²<https://projects.eclipse.org/projects/iot.ditto>

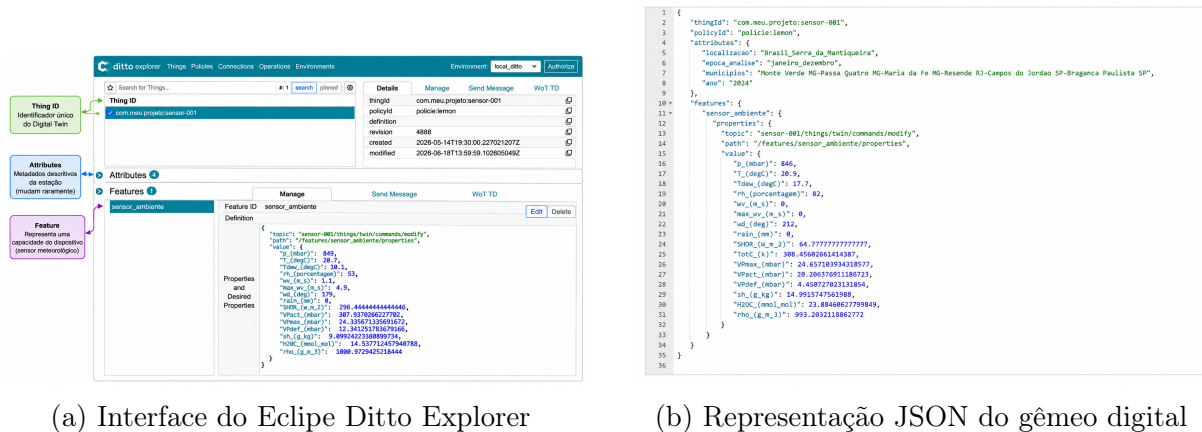


Figura 7: Interface do Eclipse Ditto e estrutura JSON correspondente ao gêmeo digital.

Figura 6).

Dados históricos e ruídos com padrões específicos são então usados para treinar modelos de IA (blocos 3 e 4 da Figura 6). Esses modelos serão posteriormente utilizados em simulações envolvendo, por exemplo, previsões de temperatura (bloco 5 da Figura 6), identificação de leituras suspeitas e dados corrompidos, ou situações anormais no comportamento do sistema físico (bloco 6 da Figura 6). Por fim, o sistema também implementa um ciclo de controle fechado entre os mundos físico e digital. Para tal, os dados coletados no ambiente físico alimentam continuamente o gêmeo digital, enquanto análises, inferências e decisões produzidas no ambiente virtual podem resultar em ações aplicadas diretamente sobre os dispositivos físicos, como sistemas de irrigação, atuadores climáticos ou mecanismos de automação agrícola (bloco 7 da Figura 6).

3.3 Implementação

A implementação do sistema proposto encontra-se em andamento. A Figura 7 ilustra a representação do gêmeo digital da estação meteorológica no Eclipse Ditto. A Figura 7a mostra a visualização do gêmeo digital por meio da interface do Ditto Explorer, incluindo os atributos e as funcionalidades dos sensores. A Figura 7b ilustra a representação em JSON do gêmeo digital contendo metadados e medições ambientais coletadas pela estação meteorológica. Esses dados foram extraídos do Banco de Dados Meteorológicos do Instituto Nacional de Meteorologia (INMET)³, considerando estações meteorológicas localizadas na Serra da Mantiqueira, mais especificamente nos municípios de Monte Verde (MG), Passa Quatro (MG), Maria da Fé (MG), Resende (RJ), Campos do Jordão (SP) e Bragança Paulista (SP).

Com relação aos mecanismos de análise inteligente, o arcabouço Cerealia encontra-se disponível publicamente⁴ e disponibiliza nove modelos de IA pertencentes a quatro classes distintas: redes convolucionais ou *Convolutional Neural Network* (CNN), redes recorrentes ou *Recurrent Neural Network* (RNN), transformadores (*Transformers*) e modelos híbridos. A escolha desses modelos busca contemplar diferentes estratégias para capturar padrões temporais, dependências de curto e longo prazo e anomalias em séries temporais provenientes de estações meteorológicas.

Além dos modelos de IA, o arcabouço Cerealia também disponibiliza um gerador sintético

³<https://bdmep.inmet.gov.br/>

⁴<https://github.com/CPS2RL/ag-dt>

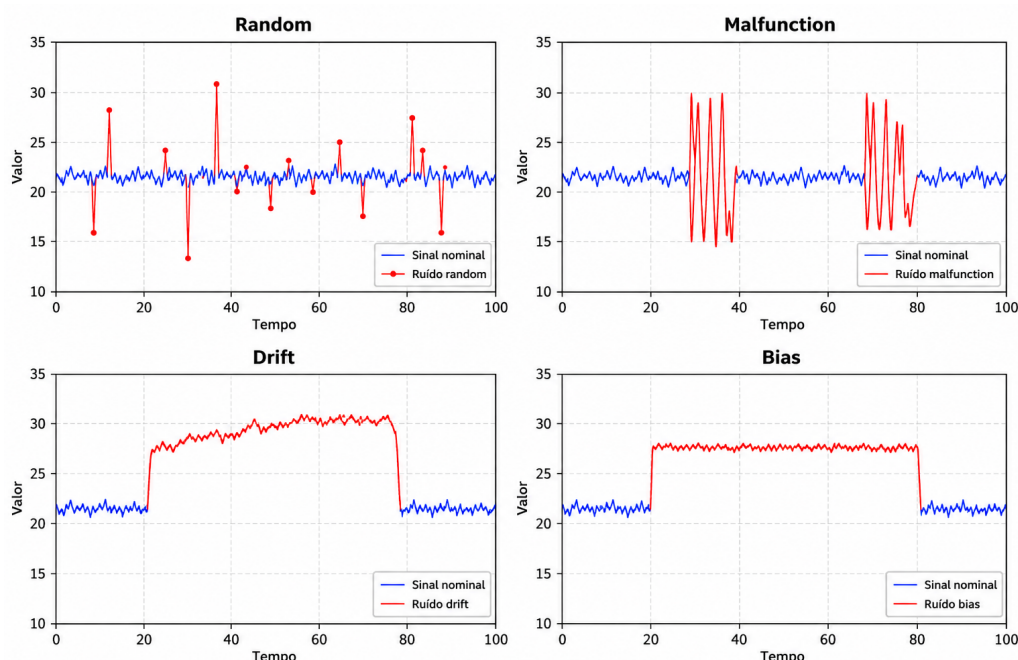


Figura 8: Tipos de ruídos introduzidos pelo gerador sintético do Cerealia.

de ruído, utilizado para inserir inconsistências artificiais nos dados meteorológicos e avaliar a robustez dos modelos sob condições imperfeitas. Atualmente, são considerados quatro tipos de ruído, como ilustrado na Figura 8: *random*, que introduz picos aleatórios nas medições; *malfunction*, que representa oscilações abruptas nos valores medidos; *drift*, semelhante ao *malfunction*, mas com deslocamento significativo dos valores em relação ao comportamento nominal; e *bias*, no qual os resultados são escalados para cima ou para baixo em relação aos valores normais.

No contexto deste projeto, esses modelos estão sendo treinados com o Banco de Dados Meteorológicos do INMET para a região da Serra da Mantiqueira. O objetivo desse treinamento é adaptar os modelos às características climáticas da região e avaliar sua capacidade de detectar inconsistências, prever comportamentos ambientais e apoiar aplicações de agricultura de precisão em cenários reais. Entretanto, como os modelos do Cerealia foram treinados com um *dataset* meteorológico europeu, foi necessário mapear e adaptar as colunas do *dataset* do INMET para o esquema usado pelo *pipeline* do arcabouço, incluindo a conversão de radiação de Kj/m^2 para W/m^2 , a derivação de variáveis meteorológicas adicionais e a divisão cronológica treino/teste. Após essa adaptação, introduzimos sinteticamente os quatro tipos de ruídos no novo *dataset* e treinamos os modelos do arcabouço. A Figura 9 mostra o resultado da fase de treinamento para o modelo *Gated Recurrent Unit* (GRU). Observamos que o modelo consegue detectar os diferentes tipos de ruído com alta acurácia.

3.4 Conclusões

Este trabalho apresentou a evolução de uma plataforma de gêmeo digital para agricultura de precisão, inspirada no arcabouço Cerealia e baseada em tecnologias abertas para IoT e gêmeos digitais. A arquitetura proposta integra uma estação meteorológica física, a plataforma Eclipse Ditto para representação digital da entidade física e um conjunto de modelos de IA destinados à detecção de inconsistências em dados meteorológicos.

Model Performance:				
Accuracy: 0.9856				
Precision: 0.9852				
Recall: 0.9856				
F1-score: 0.9851				
Classification Report:				
	precision	recall	f1-score	support
clean	0.9911	0.9958	0.9934	9354
random	0.9318	0.6833	0.7885	120
drift	0.9750	1.0000	0.9873	234
malfunction	0.8261	0.7917	0.8085	312
bias	1.0000	0.9744	0.9870	234
accuracy			0.9856	10254
macro avg	0.9448	0.8890	0.9130	10254
weighted avg	0.9852	0.9856	0.9851	10254

Figura 9: Acurácia do modelo de detecção de ruídos para o *dataset* do INMET.

Do ponto de vista da implementação, foram concluídas etapas importantes relacionadas à modelagem do gêmeo digital no Eclipse Ditto e à adaptação do *pipeline* do Cerealía para utilização de dados meteorológicos brasileiros, permitindo a criação de *datasets* compatíveis com os modelos de aprendizado de máquina do arcabouço. Os resultados preliminares obtidos com o modelo GRU indicam que a abordagem é promissora para a identificação automática de inconsistências em séries temporais meteorológicas.

Apesar dos avanços alcançados, o desenvolvimento da plataforma ainda se encontra em andamento. Como trabalhos futuros imediatos, pretende-se implementar uma *Application Programming Interface* (API) de integração que permita o consumo dos dados e resultados analíticos por aplicações externas, incluindo sua integração ao Metaverso Inatel, apresentado no Capítulo 2, para visualização e experimentação de cenários operacionais. Pretende-se também implementar um cenário operacional utilizando o gêmeo digital de uma plantação de limões da Fazenda Delta localizada na região da Serra da Mantiqueira, cuja representação virtual já integra o ambiente da *Smart Farm* no metaverso. Espera-se, com isso, demonstrar uma plataforma integrada capaz de unir monitoramento ambiental, IA, gêmeos digitais e experimentação imersiva em um ecossistema alinhado às demandas das futuras infraestruturas inteligentes conectadas e suportadas pelas redes 6G.

4 Caracterização de Tarefas de Realidade Aumentada Móvel Suportadas por Recursos de Computação em Borda

4.1 Introdução

Avanços recentes em dispositivos móveis, *software* e infraestruturas de comunicação sem fio têm possibilitado aplicações de MAR cada vez mais imersivas, favorecendo sua adoção em cenários sensíveis à latência. Apesar desse progresso, dispositivos móveis continuam limitados na capacidade de sustentar as demandas computacionais impostas pelas cargas de trabalho MAR [19], especialmente durante períodos prolongados. Uma forma de superar esse problema consiste em realizar o descarregamento (*offloading*) de tarefas computacionalmente intensivas para servidores de computação de borda [20]. Aplicações MAR assistidas por computação de borda exploram capacidade computacional adicional próxima ao usuário final, reduzindo a latência fim a fim, diminuindo o consumo de energia no *User Equipment* (UE) e mitigando restrições térmicas. Além disso, servidores de borda vêm sendo equipados com aceleradores especializados, como GPU, capazes de melhorar significativamente o desempenho de tarefas paralelizáveis e aumentar a qualidade de experiência.

Diversos trabalhos têm demonstrado a eficácia do descarregamento de tarefas para a borda da rede para aplicações MAR [21, 22, 23, 24, 25]. Entretanto, a maioria desses estudos concentra-se no projeto de sistemas e em avaliações experimentais, enquanto poucos trabalhos [26, 27] abordam explicitamente a modelagem estatística de cargas de trabalho. Tal caracterização, no entanto, é essencial para permitir avaliação sistemática de políticas de alocação de recursos e orientar o projeto de estratégias de *offloading* sob diferentes condições de implantação. Outra limitação observada em trabalhos anteriores é a suposição recorrente de infraestruturas homogêneas baseadas apenas em CPU. Na prática, diversas tarefas MAR obtêm benefícios expressivos com aceleração por GPU, e ignorar esse recurso pode levar ao uso ineficiente da infraestrutura.

No relatório anterior da atividade 3.3, apresentamos a caracterização do consumo de recursos de duas tarefas fundamentais de MAR: SLAM e detecção de objetos. Para isso, estendemos o protótipo MR-Leo [26] para: i) suportar a execução acelerada por GPU da tarefa de SLAM, usando a biblioteca ORB-SLAM2 [28]; e ii) suportar um fluxo de detecção de objetos baseado na biblioteca YOLO [29]. Utilizando esse protótipo aprimorado, denominado eMR-Leo, conduzimos uma avaliação das duas tarefas sob diferentes configurações de *hardware*, analisando o tempo de processamento, o número de instruções executadas e o consumo de memória. Neste relatório, avançamos a caracterização das tarefas, apresentando uma investigação sobre o tráfego IP gerado.

A seguir, apresentamos um resumo dos resultados obtidos, destacando o protótipo desenvolvido, a análise do tráfego IP, o modelo estatístico das cargas de trabalho e as conclusões finais.

O conteúdo deste capítulo foi desenvolvido pelos pesquisadores Karlla Chaves Rodrigues, Kleber Vieira Cardoso e Sand Luz Correa, com base no artigo submetido para o XLIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC2026) de título "*Characterization of Mobile Augmented Reality Tasks Supported by Edge Computing Resources*".

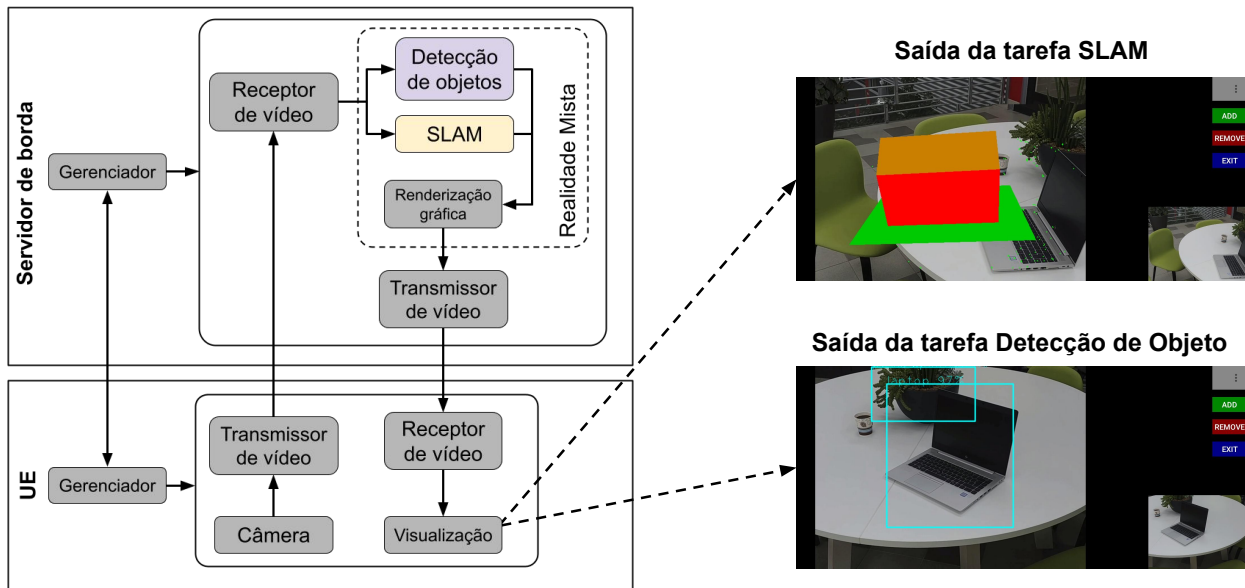


Figura 10: Arquitetura do eMR-Leo e exemplos de saídas das tarefas MAR implementadas.

4.2 Enhanced MR-Leo

Estendemos o protótipo MR-Leo original modificando sua biblioteca de SLAM e introduzindo um novo módulo de detecção de objetos. A implementação proposta preserva a arquitetura central e as funcionalidades originais do MR-Leo, restringindo as alterações aos componentes de processamento MAR. Chamamos essa nova versão de eMR-Leo. A arquitetura do eMR-Leo, ilustrada na Figura 10, segue um modelo cliente-servidor, no qual o cliente representa o UE e o servidor representa o nó (servidor) de borda. No lado do cliente, quadros de vídeo são capturados, codificados e transmitidos para a borda. O servidor recebe os quadros, os enfileira e os encaminha para processamento de uma das tarefas MAR: SLAM ou detecção de objetos. O SLAM estima a pose do dispositivo e constrói uma representação tridimensional do ambiente, enquanto a detecção de objetos identifica objetos juntamente com seus níveis de confiança. Após o processamento, os quadros são enviados para o módulo de renderização, onde ocorre o enriquecimento visual. No caso do SLAM, o enriquecimento consiste na geração e sobreposição de uma nuvem de pontos, permitindo ancoragem tridimensional de objetos. Na detecção de objetos, o enriquecimento consiste no desenho de *bounding boxes*. O lado direito da Figura 10 mostra a saída visual de cada tarefa.

O cliente não executa a análise de imagem, realizando apenas codificação e transmissão. Tanto cliente quanto servidor incluem um gerenciador responsável pela configuração dos componentes. Os serviços de transmissão e recepção de vídeo utilizam codificação H.264 e transmissão *Real-time Transport Protocol* (RTP) sobre *User Datagram Protocol* (UDP). Para codificar os fluxos e encapsular os dados em quadros RTP, utilizamos o GStreamer⁵. Embora muitas aplicações MAR utilizem *Transmission Control Protocol* (TCP) devido à sua confiabilidade, o controle de congestionamento do TCP pode introduzir oscilações de vazão indesejáveis em aplicações sensíveis à latência. Assim, optamos pelo uso de UDP, permitindo observação mais clara dos atrasos relacionados ao processamento.

Para habilitar aceleração por GPU, o eMR-Leo integra uma implementação ORB-SLAM2

⁵<https://gstreamer.freedesktop.org/>

acelerada por *Compute Unified Device Architecture* (CUDA) [30], transferindo operações intensivas de rastreamento para a GPU e executando múltiplos *kernels* CUDA de forma concorrente. A tarefa de detecção de objetos, por sua vez, foi completamente adicionada ao protótipo. Adotamos a biblioteca YOLO devido à sua ampla utilização e adequação para detecção de objetos em tempo real. Embora a YOLO seja comumente utilizada por meio de interfaces baseadas em Python, o eMR-Leo é implementado em C++, exigindo uma abordagem de integração compatível. Para esse propósito, utilizamos o *wrapper* DarkHelp⁶, que possibilita a invocação de modelos YOLO baseados em Darknet a partir de código C++.

4.3 Análise do tráfego IP

Para garantir condições consistentes e reproduzíveis entre as execuções experimentais, utilizamos um vídeo pré-gravado em vez da captura em tempo real da câmera. O vídeo possui duração de 60 segundos, resolução de 640×480 *pixels* e taxa de 30 *Frames Per Second* (FPS). Cada quadro do vídeo segue um fluxo de processamento fixo: (a) reprodução no *smartphone*, (b) codificação, (c) transmissão de *uplink* para o servidor de borda, (d) decodificação, (e) execução da tarefa e enriquecimento visual, (f) renderização, (g) codificação, (h) transmissão de *downlink* para o *smartphone*, (i) decodificação e (j) exibição ao usuário. Esse fluxo é ilustrado na Figura 11.

Utilizamos um *notebook* Lenovo Legion como servidor de borda (executando o *software* do servidor) e um *smartphone* Samsung Galaxy A21s como UE (executando o *software* cliente). O servidor de borda é equipado com um processador Intel Core i5-13420H, 16 GB de memória DDR5 e uma GPU NVIDIA GeForce RTX 3050 com 6 GB de memória, executando o Ubuntu 24.04 LTS. O *smartphone* utiliza o Android 12. A comunicação entre o *smartphone* e o servidor de borda é estabelecida por meio de *tethering Universal Serial Bus* (USB), de forma a proporcionar maior controle experimental. Essa configuração permite que o tráfego percorra toda a pilha IP, ao mesmo tempo em que evita a variabilidade introduzida por tecnologias sem fio e elementos intermediários de rede, permitindo que a análise se concentre nas características do tráfego IP gerado pela aplicação, em vez do comportamento de uma tecnologia de rede específica.

Nos resultados obtidos na análise do tráfego IP, considera-se a tarefa de SLAM executada exclusivamente na CPU (configuração básica) e a tarefa de detecção de objetos executada na GPU (configuração avançada), uma vez que essas configurações apresentaram o melhor desempenho na análise computacional realizada anteriormente.

Analizamos o tráfego IP com base em três métricas principais: i) intervalo entre pacotes (*inter-packet interval*), que corresponde ao tempo entre pacotes sucessivos dentro de um mesmo quadro; ii) intervalo entre quadros (*inter-frame interval*), que representa o tempo entre quadros individuais; e iii) tamanho do quadro (*frame size*). Essas métricas foram extraídas de capturas de pacotes coletadas utilizando o *tcpdump*⁷. O vídeo pré-gravado foi concatenado de forma a totalizar 10 minutos de duração.

Dividimos o tráfego IP gerado durante os experimentos em três fluxos distintos, cada um analisado independentemente. O *Stream 1* corresponde à transmissão em *uplink* dos quadros de vídeo codificados do *smartphone* para o servidor de borda. Esse fluxo é o mesmo para ambas as tarefas (SLAM e detecção de objetos), uma vez que o vídeo enviado é o mesmo. O *Stream 2* compreende a transmissão em *downlink* dos quadros de vídeo enriquecidos produzidos

⁶<https://github.com/stephanecharette/DarkHelp>

⁷<https://www.tcpdump.org/>

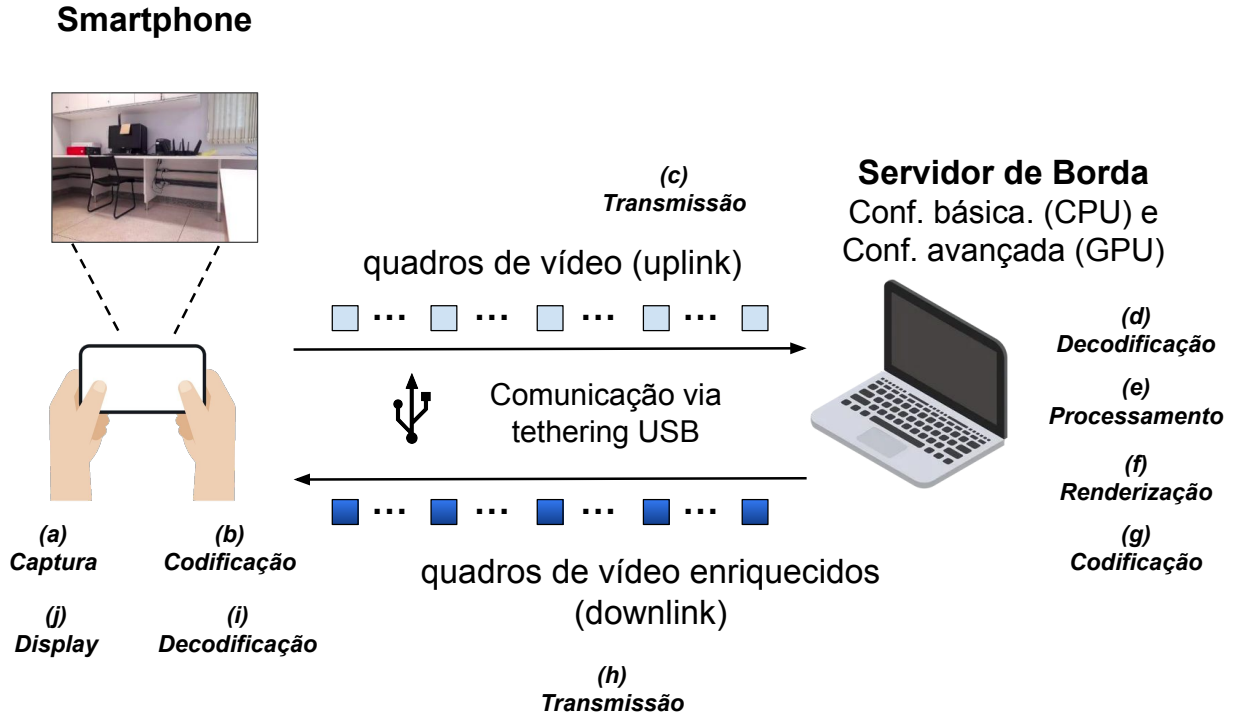


Figura 11: Cenário de teste utilizado nos experimentos.

pela tarefa de SLAM, enviados do servidor de borda para o *smartphone*. Por fim, o *Stream 3* representa a transmissão em *downlink* dos quadros de vídeo enriquecidos produzidos pelas tarefas de detecção de objetos.

Para a tarefa de SLAM, o *Stream 1*, conforme mostrado na Figura 12, apresenta intervalos entre pacotes e entre quadros regulares, alinhados à taxa de vídeo de 30 FPS, confirmando que a geração e codificação do vídeo permanecem estáveis. O *Stream 2* apresenta comportamento temporal semelhante, porém com tamanhos de quadros ligeiramente maiores e mais variáveis. Essa variabilidade reflete períodos de aumento da densidade visual do SLAM (por exemplo, renderização de nuvens de pontos mais ricas), com quadros maiores, e períodos de cenas mais estáveis com densidade visual esparsa, resultando em quadros menores. Apesar dessa variabilidade, a regularidade temporal é preservada, indicando que o enriquecimento realizado pelo SLAM aumenta o tamanho da carga útil sem comprometer o ritmo de transmissão dos quadros.

Para a tarefa de detecção de objetos, o *Stream 3*, também mostrado na Figura 12, apresenta um aumento mais pronunciado na variabilidade do tamanho dos quadros em comparação ao SLAM, refletindo a adição de *bounding boxes* e metadados. É importante destacar que o tempo entre quadros permanece estável, demonstrando que a detecção de objetos afeta principalmente a demanda de largura de banda, e não a temporização da transmissão.

4.4 Caracterização das cargas de trabalho

Nesta seção, apresentamos a caracterização da carga de trabalho das tarefas de SLAM e detecção de objetos em relação à demanda computacional e ao tráfego IP gerado. Para caracterizar a demanda computacional da tarefa de SLAM, coletamos o número de instruções executadas por quadro na CPU usando a ferramenta *perf*. Como mostrado na Figura 13a,

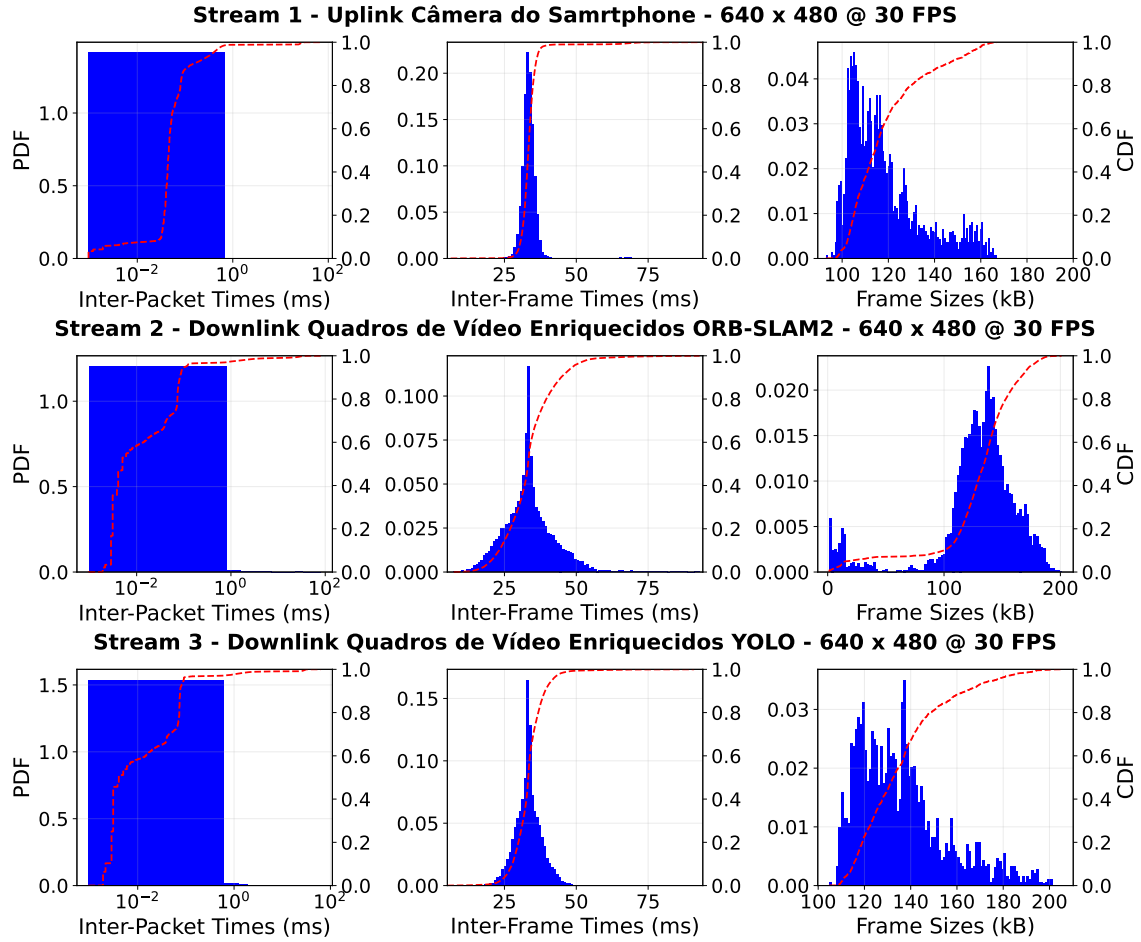
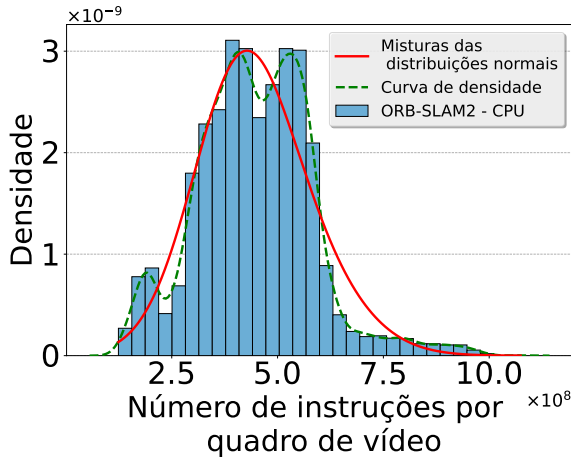


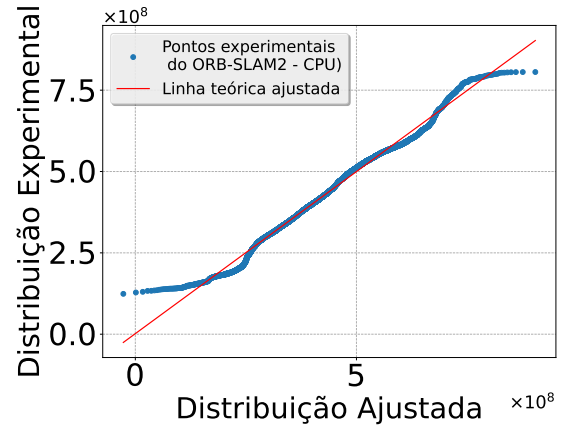
Figura 12: Métricas de Tráfego IP para cada tarefa MAR.

a maioria dos quadros requer entre 300 e 600 milhões de instruções (MI), indicando uma demanda computacional relativamente estável. A distribuição apresenta assimetria à direita, com ocorrência ocasional de quadros de maior custo computacional. O modelo ajustado de mistura de gaussianas captura esse comportamento utilizando dois componentes dominantes: um centrado em torno de 407 MI e outro próximo de 529 MI, correspondendo ao rastreamento regular e a eventos computacionalmente mais intensivos, como relocalização ou atualizações do mapa. O gráfico Q-Q da Figura 13b confirma um bom ajuste, apresentando apenas pequenos desvios nas caudas da distribuição.

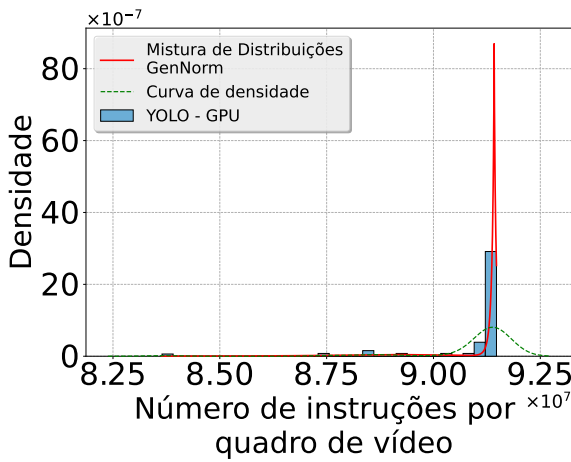
Para caracterizar a demanda computacional da tarefa de detecção de objetos, coletamos o número de instruções executadas por quadro na GPU usando a ferramenta NVIDIA Nsight Compute CLI. A Figura 13c apresenta os resultados, demonstrando uma distribuição muito mais concentrada, na qual a maioria dos quadros requer aproximadamente 91 MI. A carga de trabalho é modelada como uma mistura de duas distribuições *Generalized Normal*: um componente dominante de baixa variância (82%) representando a inferência típica e um componente secundário de maior variância (18%) capturando cenas mais complexas. Ambos os componentes exibem comportamento de cauda pesada ($\beta < 2$), refletindo desvios ocasionais na demanda de processamento. O gráfico Q-Q da Figura 13d mostra boa concordância na região central,



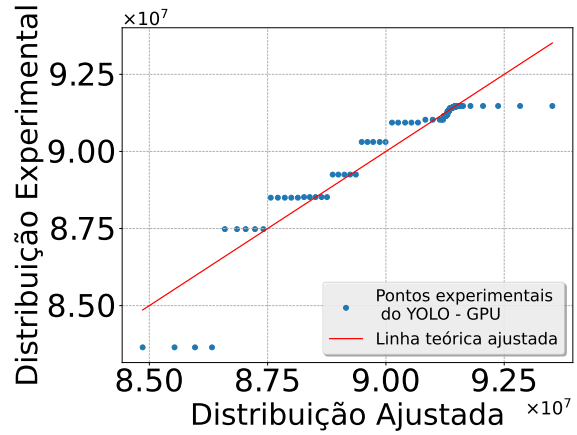
(a) SLAM



(b) Gráfico Q-Q



(c) Detecção de objetos



(d) Gráfico Q-Q

Figura 13: Distribuição empírica e ajustada do número de instruções por quadro para a tarefa de SLAM (13a); qualidade do ajuste para o SLAM (13b); distribuição empírica e ajustada do número de instruções por quadro para a tarefa de detecção de objetos (13c); e qualidade do ajuste para a detecção de objetos (13d).

com desvios esperados nas caudas devido à natureza estocástica da inferência em aprendizado profundo.

A Figura 14 apresenta os resultados do teste de Kolmogorov–Smirnov para as métricas de tráfego IP. Para o *Stream 1 (uplink)*, a distribuição *Johnson SU* foi a que melhor modelou o tamanho dos quadros, o intervalo entre quadros e o intervalo entre pacotes. Para os *Streams 2 e 3 (downlink)*, a distribuição *Johnson SU* também apresentou o melhor ajuste para o tamanho dos quadros e os intervalos entre quadros, enquanto os intervalos entre pacotes foram melhor representados por uma distribuição lognormal, indicando maior variabilidade temporal no nível dos pacotes.

A Tabela 1 resume o modelo estatístico de carga de trabalho, incluindo as distribuições ajustadas e os parâmetros relacionados à computação, uso de memória e tráfego IP das tarefas de SLAM e detecção de objetos.

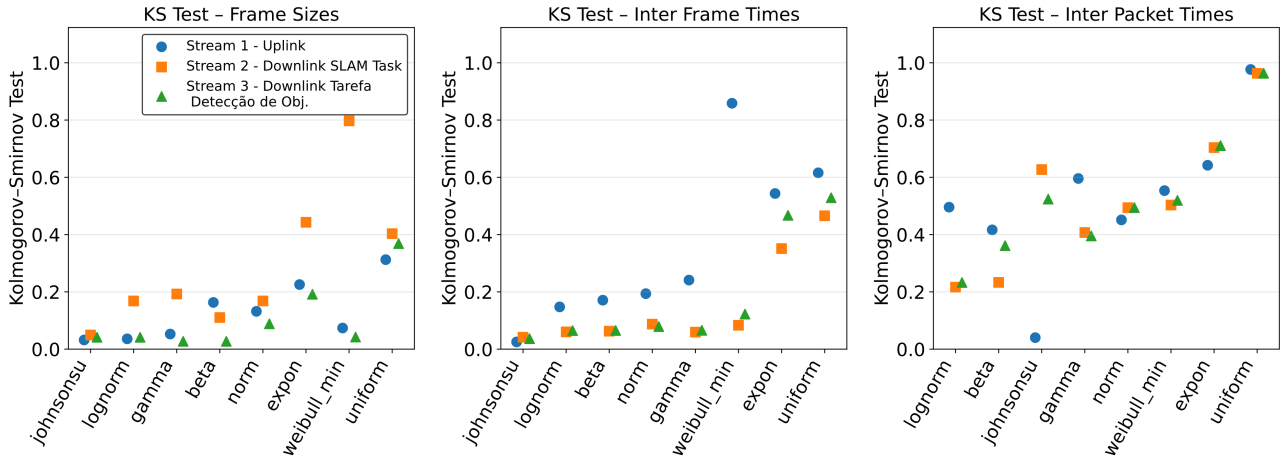


Figura 14: Teste KS para todas as métricas e *streams* usados para análise do tráfego IP.

4.5 Conclusões Parciais

Este trabalho apresentou uma caracterização da carga de trabalho de duas tarefas fundamentais de MAR, SLAM e detecção de objetos, executadas em uma arquitetura assistida por computação de borda. Ao estender o protótipo MR-Leo para suportar processamento heterogêneo CPU-GPU, avaliamos a demanda computacional, o uso de memória e os padrões de tráfego IP sob condições controladas. Os resultados revelam perfis distintos de utilização de recursos para as duas tarefas. O SLAM apresenta um comportamento predominantemente limitado por CPU, com demanda multimodal de instruções e uso de memória continuamente crescente, associado ao crescimento do mapa. A aceleração por GPU proporciona redução limitada da latência global, melhorando principalmente a renderização, em vez do processamento central do SLAM. Em contraste, a detecção de objetos demonstra uma carga de trabalho altamente paralelizável, na qual a aceleração por GPU reduz significativamente a latência de processamento, ao mesmo tempo em que estabiliza o uso de memória por meio da transferência dos *buffers* de inferência para a *Video Random Access Memory* (VRAM). A análise de tráfego também mostra que ambas as tarefas mantêm temporização estável entre quadros, alinhada a 30 FPS, enquanto o enriquecimento visual aumenta a variabilidade do tamanho dos quadros no *downlink*. Os modelos estatísticos resultantes capturam essas características distintas de computação e tráfego, permitindo a geração realista de cargas de trabalho para pesquisas em computação de borda. Embora os valores específicos da carga de trabalho dependam das bibliotecas de *software* e das implementações adotadas (por exemplo, ORB-SLAM2 e YOLO), a caracterização permanece valiosa, pois a literatura ainda carece de modelos empíricos que descrevam conjuntamente a demanda computacional, o uso de memória e o tráfego IP de tarefas MAR assistidas por computação de borda. Assim, este trabalho fornece um importante ponto de referência e uma base metodológica para estudos futuros, mesmo à medida que as implementações evoluem.

Tarefa	Hardware	Atividade	Modelo
Fluxo da câmara	UE	Chegada de tarefas	Periodicamente, a cada 33 ms
		Tráfego de <i>uplink</i>	Para tamanho do <i>frame</i> : $JohnsonSU(\gamma = -3.71, \delta = 1.54, \xi = 94.20, \lambda = 3.57)$ Para intervalo entre <i>frames</i> : $JohnsonSU(\gamma = -0.17, \delta = 1.04, \xi = 33.04, \lambda = 1.64)$ Para intervalo entre pacotes: $JohnsonSU(\gamma = -0.466, \delta = 0.416, \xi = 0.04090, \lambda = 0.00351)$
ORB-SLAM2	CPU	Chegada de tarefas	Periodicamente, a cada 33 ms
		Computação	Para 43%: $\mathcal{N}(\mu = 494 \text{ MI}, \sigma = 147.97 \text{ MI})$ Para 53%: $\mathcal{N}(\mu = 404 \text{ MI}, \sigma = 110.99 \text{ MI})$
		Tráfego de <i>downlink</i>	Para tamanho do <i>frame</i> : $JohnsonSU(\gamma = 0.46, \delta = 1.04, \xi = 144.17, \lambda = 20.98)$ Para intervalo entre <i>frames</i> : $JohnsonSU(\gamma = -0.37, \delta = 1.58, \xi = 30.48, \lambda = 10.84)$ Para intervalo entre pacotes: $Lognormal(s = 1.91, \text{loc} = 9.54 \times 10^{-4}, \text{scale} = 0.0103)$
		Memória (somente borda)	408 MB
		Memória adicional por cliente	288 MB
YOLO	GPU	Chegada de tarefas	Periodicamente, a cada 33 ms
		Computação	Para 82%: $\mathcal{GN}(\mu = 91 \text{ MI}, \alpha = 0.40 \text{ MI}, \beta = 0.80)$ Para 18%: $\mathcal{GN}(\mu = 89 \text{ MI}, \alpha = 2.05 \text{ MI}, \beta = 1.25)$
		Tráfego de <i>downlink</i>	Para tamanho do <i>frame</i> : $JohnsonSU(\gamma = 2.74, \delta = 3.56 \times 10^4, \xi = 104.56, \lambda = 4.06 \times 10^5)$ Para intervalo entre <i>frames</i> : $JohnsonSU(\gamma = -0.10, \delta = 1.24, \xi = 32.88, \lambda = 4.44)$ Para intervalo entre pacotes: $Lognormal(s = 1.97, \text{loc} = 9.53 \times 10^{-4}, \text{scale} = 0.00909)$
		Memória (somente borda)	208 MB
		Memória adicional por cliente	416 MB RAM 416 MB VRAM

Tabela 1: Visão geral do modelo de geração de carga.

5 Transmissão de Vídeo em Redes Sem Fio Utilizando Comunicação Semântica

5.1 Introdução

A crescente demanda por transmissão de vídeo em alta resolução em aplicações como telemedicina, colaboração remota e entretenimento imersivo tem intensificado a pressão sobre a infraestrutura de rede [31]. Nesse contexto, a comunicação semântica surge como uma abordagem promissora para otimizar o uso dos recursos de transmissão ao priorizar o envio de informações semanticamente relevantes [32]. Essa estratégia possibilita ganhos significativos de eficiência, como demonstrado por trabalhos recentes que utilizam reconstrução generativa para reduzir o consumo de banda mantendo a qualidade perceptual, super-resolução guiada por semântica para melhorar a fidelidade visual sob restrições de *bitrate* e técnicas neurais de ocultação de erros para preservar a continuidade do vídeo durante falhas na rede [33, 34, 35].

Diversos trabalhos recentes exploram essas vantagens em cenários específicos. O WiserVR [33] reduz o consumo de banda em transmissões de realidade virtual por meio da priorização semântica de regiões relevantes, enquanto o SAW [34] aplica super-resolução semântica baseada em modelos de difusão para melhorar a qualidade visual dos fluxos de vídeo. O Grace [36] aumenta a robustez a perdas em codificação neural de vídeo em tempo real por meio do treinamento conjunto entre codificador e decodificador. Apesar dos resultados promissores, essas soluções são fortemente acopladas às suas arquiteturas específicas, frequentemente exigindo modificações em protocolos, *codecs* ou infraestruturas de transmissão. Essa especialização dificulta a integração de novas técnicas de IA, limita a comparação entre abordagens distintas e aumenta o esforço necessário para avaliar novos modelos em ambientes de transmissão de vídeo.

No relatório anterior da atividade 3.3, apresentamos a proposta de um sistema modular para transmissão de vídeo com comunicação semântica, projetado para desacoplar os modelos de IA da infraestrutura de transporte e, assim, facilitar a integração e avaliação de diferentes abordagens semânticas em um ambiente padronizado baseado em WebRTC [31]. A arquitetura proposta introduz orquestradores semânticos orientados por políticas, capazes de gerenciar dinamicamente o encadeamento e a execução de diferentes modelos de IA durante a transmissão de vídeo. Para validar a proposta, concluímos a implementação do arcabouço utilizando WebRTC e a biblioteca `aiortc`, incorporando mecanismos para transmissão de *frames* em baixa resolução, integração modular de modelos semânticos e coleta de métricas de qualidade visual e desempenho da transmissão. Além disso, realizamos um estudo de caso com a integração de um modelo SRGAN [37] para super-resolução de vídeo, avaliando configurações de escalonamento sob diferentes políticas de operação, de forma a analisar os compromissos entre qualidade perceptual, custo de processamento e eficiência da transmissão em cenários de videoconferência.

A seguir, apresentamos um resumo dos resultados obtidos, destacando o arcabouço desenvolvido, sua arquitetura, a integração de modelos de IA e os principais resultados obtidos a partir da avaliação experimental realizada em cenário de videoconferência.

O conteúdo deste capítulo consiste em um resumo adaptado pelo pesquisador *Elton Vivot*, com base no artigo submetido ao 31º Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia), de autoria de Elton Vivot, Isabela Teixeira e Kleber V. Cardoso, intitulado "*FLEXA: A Modular and Flexible Framework for AI-Assisted Video Communication*".

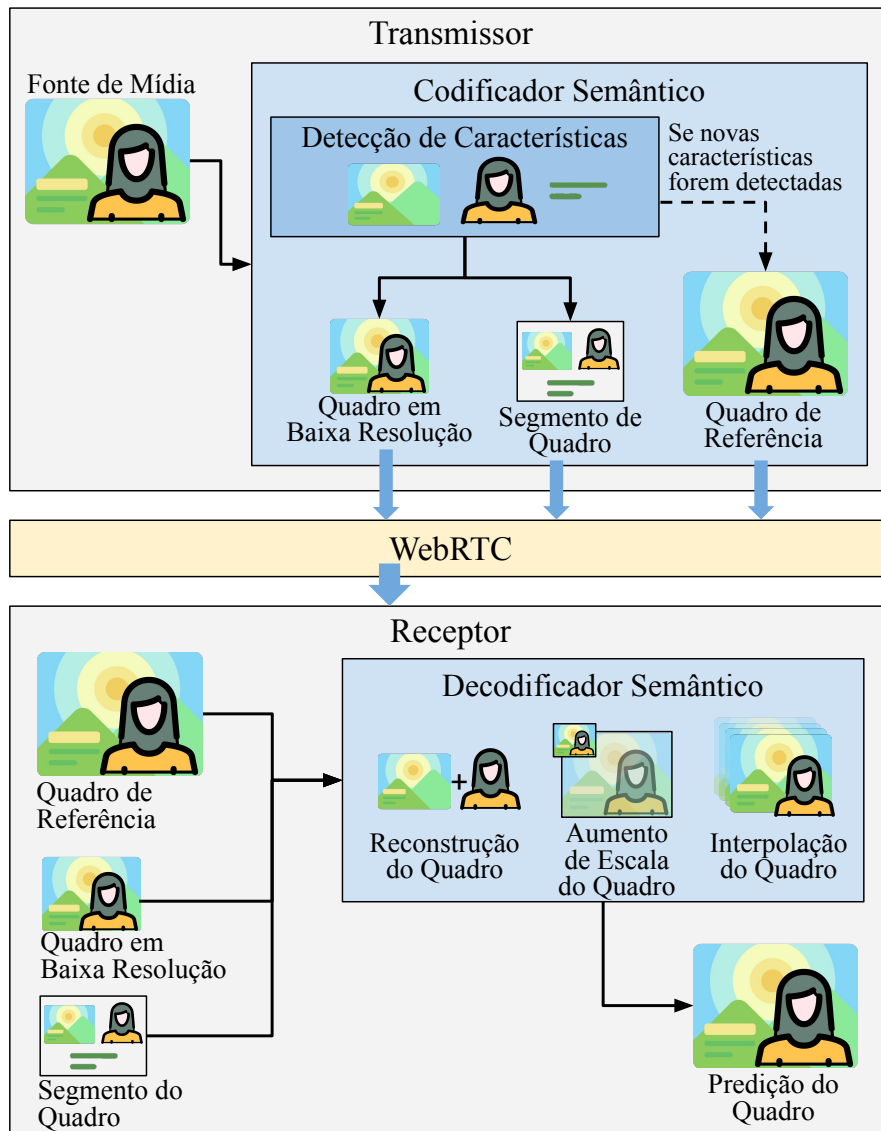


Figura 15: Fluxo conceitual do processamento no FLEXA.

5.2 Sistema proposto

O sistema proposto neste trabalho, nomeado FLEXA, é um arcabouço modular para transmissão de vídeo, desenvolvido para simplificar a integração de modelos de IA ao *pipeline* de comunicação. O objetivo é viabilizar a comunicação semântica por meio da aplicação de processamento baseado em IA em diferentes etapas da transmissão, sem a necessidade de modificar a lógica de transporte subjacente. O FLEXA oferece suporte à operação em tempo real e possibilita a integração de modelos voltados a tarefas como geração, compressão, aprimoramento e adaptação de vídeo.

A Figura 15 apresenta o funcionamento conceitual do FLEXA. No lado do transmissor, a fonte de mídia é processada pelo Codificador Semântico, que utiliza modelos de IA para identificar segmentos relevantes dos quadros ou detectar novas características visuais. Segmentos mais sensíveis à qualidade visual podem ser transmitidos em maior resolução, enquanto regiões menos relevantes são enviadas com qualidade reduzida. Quando novas características são detectadas, um quadro de referência em resolução completa pode ser transmitido. No receptor,

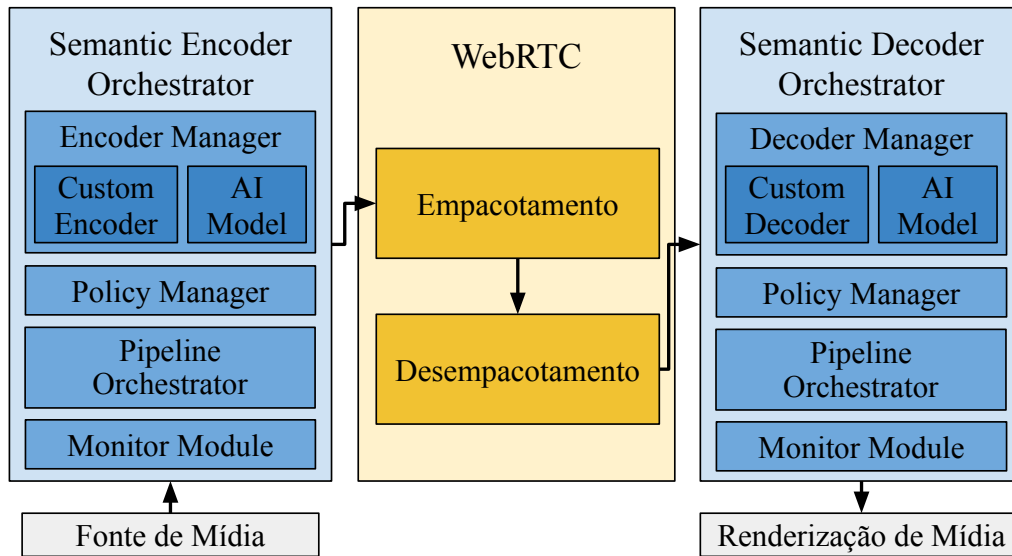


Figura 16: Arquitetura interna do FLEXA integrada à camada WebRTC.

o Decodificador Semântico reconstrói o quadro final combinando os segmentos recebidos, aplicando modelos de ampliação de resolução e, opcionalmente, utilizando modelos de interpolação para aumento da taxa de quadros.

O arcabouço foi desenvolvido sobre o WebRTC [38], uma biblioteca de código aberto voltada à transmissão de mídia em tempo real usando o protocolo RTP [39]. Para possibilitar o processamento de quadros utilizando modelos de IA e a integração com o PyTorch [40], foi adotada a biblioteca *aiortc* [41], uma implementação do WebRTC em Python.

O arcabouço implementa dois principais orquestradores. A Figura 16 apresenta a estrutura interna do FLEXA. Cada orquestrador é composto por quatro módulos principais:

- *Encoder/Decoder Manager*: Responsável pelo registro dos codificadores, decodificadores e seus respectivos modelos de IA. Múltiplos codificadores e decodificadores podem coexistir, sendo associados a tarefas específicas, como segmentação, gerenciamento de quadros de referência, aprimoramento, reconstrução ou interpolação. Esse módulo fornece os componentes necessários ao *Pipeline Orchestrator* para o encadeamento dinâmico do processamento.
- *Policy Manager*: Responsável pelo gerenciamento das políticas associadas aos codificadores e decodificadores, definindo ações a serem executadas conforme as condições de execução do sistema. As políticas são implementadas como conjuntos de regras que utilizam informações do módulo de monitoramento, como o tempo de execução do *pipeline*, para decidir as etapas de processamento. Por exemplo, ao detectar que um limite de atraso foi excedido, uma política pode ignorar determinadas etapas de processamento neural, transmitir o quadro sem aprimoramento ou descartar o quadro atual e reutilizar o último quadro processado. As políticas podem ser gerais, afetando todo o *pipeline*, ou específicas para determinados modelos. Todas as políticas são definidas antes do início da transmissão, sendo criadas programaticamente por meio de funções disponibilizadas pelo código-fonte do FLEXA, permitindo o registro de novos comportamentos de execução. Durante a transmissão, o *Policy Manager* avalia continuamente as políticas definidas

para cada quadro processado e aplica dinamicamente as ações correspondentes de acordo com as condições atuais de execução.

- *Pipeline Orchestrator*: Responsável por definir a sequência de codificadores e decodificadores aplicada ao processamento dos quadros, considerando os componentes registrados no sistema e as políticas ativas no momento da execução. No transmissor, o *pipeline* pode incluir, por exemplo, codificadores de preparação, segmentação e referência. No receptor, podem ser utilizados decodificadores de reconstrução, aprimoramento e interpolação. Esse módulo garante a interoperabilidade entre os componentes, gerenciando o fluxo de dados, ajustando a ordem de execução e ativando ou desativando módulos conforme as decisões das políticas.
- *Monitor Module*: Responsável pela coleta de estatísticas e informações de execução durante o processo de transmissão. O módulo monitora métricas como tempo de execução por etapa do *pipeline*, políticas acionadas, codificadores e decodificadores ativos, tamanho do fluxo de mídia, tempo de processamento por modelo e quantidade de quadros gerados. A coleta de métricas é configurável, permitindo registrar apenas os dados relevantes para avaliação ou adaptação do sistema.

5.3 Avaliação e resultados

A avaliação do arcabouço foi realizada em um cenário simulado de videoconferência utilizando o modelo SRGAN [37] para aprimoramento de resolução no *pipeline* de comunicação semântica. O modelo foi treinado com o conjunto de dados *Flickr-Faces-HQ* (FFHQ) [42], composto por imagens de rostos humanos. Foram consideradas sete configurações de escalonamento de resolução, variando entre entradas de 64×64 e saídas de 1024×1024 . No transmissor, o *framework* utilizou um codificador responsável pela redução da resolução dos quadros antes da transmissão via WebRTC, enquanto o receptor empregou um decodificador baseado em SRGAN para reconstrução e ampliação da resolução dos quadros recebidos.

A integração semântica foi controlada por políticas de execução responsáveis por definir a resolução de saída e o comportamento do *pipeline* em situações de atraso de processamento. Foram avaliadas duas políticas principais: *Skipping*, que ignora o processamento do SRGAN quando o atraso excede 300 ms, e *Freezing*, que reutiliza o último quadro aprimorado processado. Também foi considerado um cenário *Unconstrained*, no qual o modelo sempre possui tempo suficiente para processar todos os quadros. Os experimentos foram executados em uma máquina equipada com processador AMD Ryzen 7 5700X3D, GPU NVIDIA RTX 4070 Super e 32 GB de memória RAM. A qualidade visual foi avaliada utilizando as métricas *Peak Signal-to-Noise Ratio* (PSNR), *Structural Similarity Index Measure* (SSIM) e *Learned Perceptual Image Patch Similarity* (LPIPS), além de métricas de desempenho relacionadas ao tamanho do fluxo transmitido e ao tempo de processamento dos quadros.

Os resultados foram gerados a partir da comparação entre diferentes configurações de entrada e saída de resolução sob os três modos de operação avaliados. A análise foi dividida em dois cenários principais: i) ampliação de quadros fixos de 64×64 para diferentes resoluções de saída e ii) ampliação de diferentes resoluções de entrada para uma saída fixa de 1024×1024 .

A Figura 17 apresenta os resultados de qualidade visual obtidos para diferentes resoluções de saída a partir de quadros de entrada com resolução fixa de 64×64 . Observa-se uma degradação gradual das métricas conforme o fator de ampliação aumenta até 1024×1024 . O modo

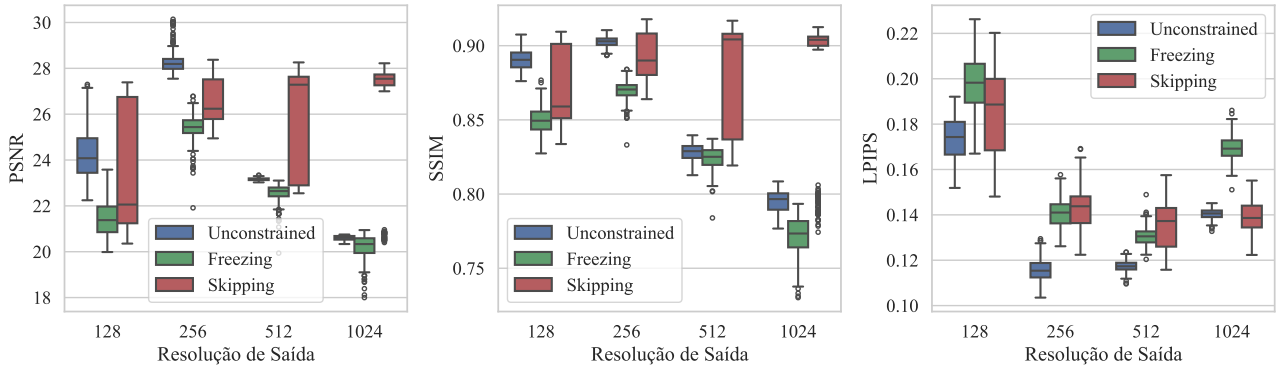


Figura 17: Métricas de qualidade para entradas de 64×64 ampliadas para diferentes resoluções de saída.

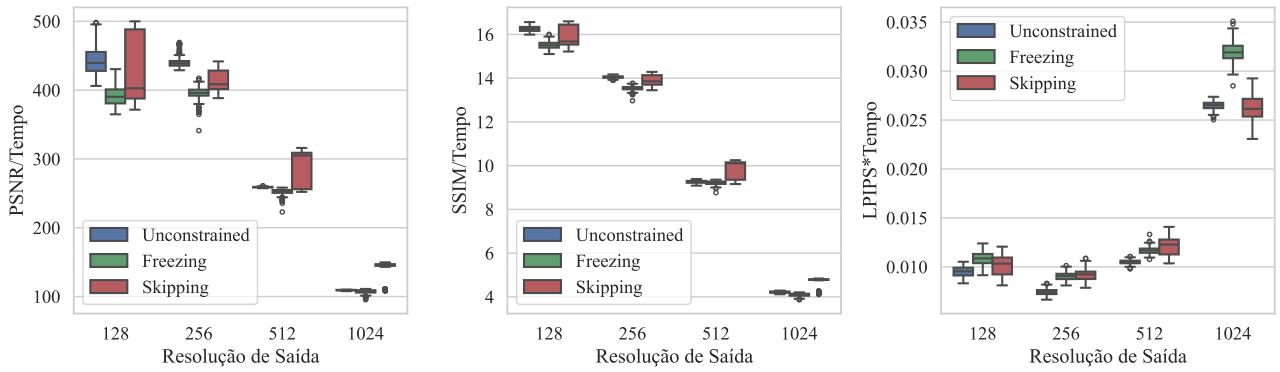


Figura 18: Eficiência entre qualidade visual e tempo de processamento para entradas de 64×64 ampliadas para diferentes resoluções de saída.

Unconstrained apresentou os melhores resultados de qualidade em todas as resoluções avaliadas, enquanto a política *Skipping* obteve desempenho superior ao *Freezing* em resoluções mais elevadas, especialmente a partir de 512×512 . Esse comportamento ocorre porque o *Skipping* preserva melhor a consistência temporal ao exibir quadros atualizados, ainda que em menor resolução, enquanto o *Freezing* reutiliza quadros anteriores durante períodos de sobrecarga, introduzindo artefatos perceptíveis de movimento. Além disso, a métrica LPIPS apresentou maior variabilidade em ampliações extremas, evidenciando a dificuldade do SRGAN em reconstruir detalhes de alta frequência a partir de entradas muito reduzidas.

A Figura 18 relaciona a qualidade visual obtida com o tempo de processamento necessário para cada configuração. Os resultados indicam que a configuração $64 \times 64 \rightarrow 256 \times 256$ apresentou o melhor equilíbrio entre qualidade e custo computacional, alcançando a maior razão entre PSNR e tempo de processamento, o que sugere um limite mais adequado para aplicações em tempo real. Em resoluções mais altas, a política *Skipping* apresentou melhor eficiência perceptual em relação ao *Freezing*, principalmente em 1024×1024 , por evitar o acúmulo adicional de latência causado pelo processamento contínuo do modelo de super-resolução. A partir de 512×512 , observou-se uma tendência de estabilização dos ganhos de qualidade, enquanto o tempo de processamento continuou aumentando, indicando resultados piores para ampliações mais agressivas.

A Figura 19 apresenta os resultados obtidos para diferentes resoluções de entrada considerando uma resolução fixa de saída em 1024×1024 . Em alguns cenários, a política *Skipping*

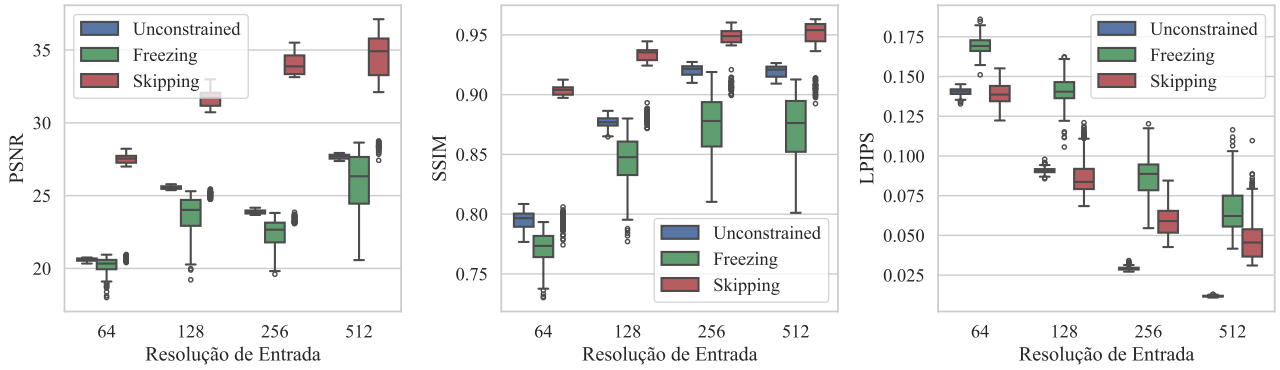


Figura 19: Métricas de qualidade para saída fixa em 1024×1024 a partir de diferentes resoluções de entrada.

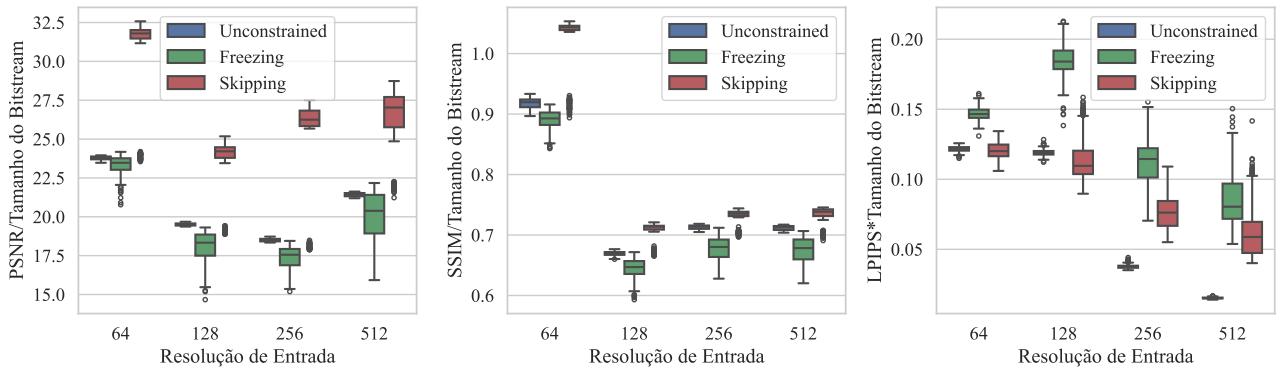


Figura 20: Eficiência entre qualidade visual e tamanho do fluxo transmitido. As métricas foram normalizadas pelo tamanho dos quadros transmitidos.

alcançou valores superiores de PSNR e SSIM em relação ao modo *Unconstrained*. Na configuração $512 \times 512 \rightarrow 1024 \times 1024$, por exemplo, a mediana do PSNR foi de 34,2 para *Skipping*, enquanto o modo *Unconstrained* apresentou 32,5. Esse comportamento ocorre porque o componente generativo do SRGAN introduz pequenas diferenças em nível de pixel que podem reduzir métricas de referência completa, mesmo quando a qualidade perceptual do resultado é visualmente superior. Essa característica é refletida pelo LPIPS, no qual o modo *Unconstrained* apresentou menor valor (0,042) em comparação ao *Skipping* (0,068), indicando melhor similaridade perceptual apesar de métricas tradicionais inferiores. Observou-se ainda que o aumento da resolução de entrada reduz a diferença entre os modos avaliados, evidenciando que o SRGAN se beneficia de quadros de entrada com maior quantidade de informação visual.

Por fim, a Figura 20 relaciona a qualidade visual ao tamanho do fluxo transmitido. As resoluções de entrada 256×256 e 512×512 apresentaram valores semelhantes de LPIPS normalizado pelo tamanho do fluxo, resultado associado à eficiência semelhante de compressão do codec VP8 nesses dois cenários. A configuração $128 \times 128 \rightarrow 1024 \times 1024$ apresentou o melhor compromisso entre SSIM e tamanho transmitido, alcançando aproximadamente 0,87 de SSIM por MB transmitido, superando em cerca de 23% a configuração $512 \times 512 \rightarrow 1024 \times 1024$, mesmo com menor qualidade absoluta. Em resoluções de entrada muito baixas, como $64 \times 64 \rightarrow 1024 \times 1024$, a política *Skipping* obteve eficiência significativamente superior ao *Freezing*, alcançando aproximadamente 2,1 vezes mais PSNR por tamanho transmitido, tornando-se mais adequada para cenários com restrição de largura de banda.

5.4 Conclusões e próximos passos

Este trabalho investigou a aplicação da comunicação semântica em sistemas de transmissão de vídeo por meio da proposta de um arcabouço modular baseado em WebRTC. O sistema implementa orquestradores orientados por políticas capazes de integrar dinamicamente diferentes modelos de IA sem modificar a infraestrutura de transmissão. Além disso, foram realizadas implementações e avaliações experimentais que evidenciaram os compromissos entre qualidade visual, desempenho e eficiência de transmissão em cenários de comunicação de vídeo com restrições de recursos.

Como trabalhos futuros, pretende-se evoluir o arcabouço proposto de modo a simplificar ainda mais o processo de integração de novos modelos de IA e de diferentes protocolos de transmissão, ampliando sua flexibilidade e reutilização em distintos cenários de comunicação semântica. Além disso, será incorporado um mecanismo de controle de parâmetros de rede, permitindo a simulação de condições adversas, como perda de pacotes, variações de latência e limitações de largura de banda. Pretende-se também substituir o modelo SRGAN por abordagens mais adequadas ao contexto da transmissão de vídeo em tempo real, visando melhorar a qualidade perceptual e reduzir o custo computacional do processamento. Por fim, serão incorporadas novas métricas de avaliação da qualidade visual, incluindo o *Video Multi-Method Assessment Fusion* (VMAF), com o objetivo de ampliar a análise da qualidade percebida dos vídeos transmitidos.

6 Sistema de IoT Baseado em LoRa para Monitoramento da Fauna Sul-Mineira

6.1 Introdução

Os animais desempenham um papel fundamental na manutenção do equilíbrio dos ecossistemas e sua extinção ou redução populacional pode provocar impactos ambientais severos. Nesse contexto, a proteção e a preservação das espécies tornaram-se desafios globais prioritários. A IoT tem se destacado como uma ferramenta promissora no apoio às ações de conservação, ao possibilitar o monitoramento contínuo e em tempo real de animais e de seus *habitats* [43, 44]. Sensores conectados à rede permitem a coleta e a transmissão de dados sobre localização, comportamento, saúde e variáveis ambientais, contribuindo para a identificação de padrões ecológicos e de ameaças potenciais [45]. Além disso, soluções baseadas em IoT podem ser integradas a estratégias como *geofencing* (cercas virtuais) e sistemas de alerta para prevenção de caça ilegal, aumentando a eficiência e a segurança das ações de preservação [46, 47].

No sul de Minas Gerais e no Pantanal, a conservação da fauna é especialmente relevante devido à presença de grandes carnívoros e espécies consideradas topo de cadeia ou espécies guarda-chuva, como a onça-parda (*Puma concolor*), a onça-pintada (*Panthera onca*) e o lobo-guará (*Chrysocyon brachyurus*) [48]. O acompanhamento dessas espécies é essencial para compreender sua distribuição espacial, padrões de movimentação, uso do *habitat* e dinâmica populacional, fornecendo subsídios para a elaboração de planos de manejo, estratégias de conservação e a definição de corredores ecológicos que conectam áreas fragmentadas [49, 50]. A relevância desse tipo de monitoramento é evidenciada por iniciativas como o Projeto Onçaria, no qual o uso de colares de rastreamento possibilitou a descoberta de tocas com filhotes de onça-pintada em vida livre, trazendo informações importantes sobre a ecologia da espécie [51].

Apesar da importância do rastreamento por meio de dispositivos IoT, atualmente não há soluções nacionais eficientes disponíveis para esse fim. Os principais equipamentos comerciais são fornecidos por empresas internacionais, como Lotek⁸, Telonics⁹ e Vectronics¹⁰, sendo importados e cotados em moeda estrangeira, o que eleva significativamente os custos de aquisição e inviabiliza sua utilização em larga escala. Além disso, tais dispositivos geralmente fornecem apenas um a dois pontos de localização *Global Positioning System* (GPS) por dia, resultando em baixa resolução espacial, vida útil limitada da bateria, em torno de um ano, e ausência de monitoramento do estado energético do dispositivo. Também são restritos quanto às variáveis ambientais coletadas, como umidade e luminosidade, informações relevantes para a compreensão da preferência de *habitat* de diferentes espécies [52].

Além disso, embora os módulos de GPS sejam amplamente utilizados nas coleiras comerciais para estimativa de localização, seu uso implica maior consumo energético, aumento do custo do dispositivo e dependência de cobertura adequada de satélites [53]. Em cenários regionais de monitoramento, alternativas baseadas em comunicação de longo alcance e baixo consumo, como a tecnologia LoRa, tornam-se atrativas [54]. Além de possibilitar transmissão de dados

O conteúdo deste capítulo consiste em um resumo adaptado pela pesquisadora Victoria D. P. Souto, com base no artigo submetido para o XLIV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT2026) por Murilo L. Massini, Túlio A. de A. Rezende, Samuel B. Mafra e Victoria D. P. Souto. "Sistema de IoT Baseado em LoRa para Monitoramento da Fauna Sul-Mineira".

⁸<https://www.lotek.com/>

⁹<https://www.telonics.com/>

¹⁰<https://www.vectronic-aerospace.com/>

a longas distâncias com reduzido gasto energético, o uso de múltiplos GWs permite explorar métricas como o RSSI para estimativas aproximadas de localização, adequadas para validação arquitetural e análise preliminar de movimentação [54].

Diante desse cenário, este trabalho tem como objetivo propor e validar uma arquitetura IoT nacional, de baixo custo e baixo consumo energético, voltada ao monitoramento de fauna silvestre. As contribuições do artigo concentram-se em três aspectos principais: i) o desenvolvimento de uma solução integrada para aquisição e transmissão remota de dados ambientais e de movimento; ii) a investigação do uso da tecnologia LoRa como alternativa viável para cenários de monitoramento em regiões extensas e com restrições energéticas; e iii) a avaliação do emprego de múltiplos GWs e medidas de RSSI como mecanismo de suporte à estimativa aproximada de localização. De forma complementar, o trabalho apresenta uma infraestrutura de visualização e acompanhamento remoto dos dados, evidenciando a viabilidade de uma abordagem ponta a ponta adaptada ao contexto brasileiro. Os resultados obtidos em ambiente de bancada/laboratório demonstram o potencial da arquitetura proposta como base para futuras etapas de validação em campo e sua aplicação em estratégias de conservação da fauna.

6.2 Metodologia de Projeto

O objetivo deste projeto é desenvolver uma coleira de monitoramento baseada em tecnologias de IoT, capaz de rastrear a localização de animais silvestres, coletar variáveis ambientais e comportamentais e monitorar a vida útil da bateria. Esse monitoramento é essencial para compreender padrões de deslocamento e uso do *habitat*, avaliar o estado dos indivíduos, identificar preferências ambientais e antecipar riscos associados a mudanças climáticas e ações antrópicas. Os dados obtidos podem subsidiar planos de manejo, estratégias de conservação de espécies ameaçadas e a definição de corredores ecológicos, contribuindo para a preservação da fauna brasileira.

Diante disso, a metodologia do projeto foi estruturada de forma sistemática e progressiva, abrangendo desde a definição dos requisitos até a validação do protótipo final. Buscou-se estabelecer um fluxo de desenvolvimento capaz de contemplar, de maneira integrada, a coleta, a transmissão, o armazenamento e a visualização dos dados. Para tanto, o projeto foi organizado em etapas sequenciais e complementares. Inicialmente, definiram-se as variáveis fundamentais a serem monitoradas, de modo a assegurar a relevância das informações geradas para o acompanhamento dos animais e para o suporte a ações de conservação. Em seguida, realizou-se a seleção dos componentes necessários à implementação do protótipo, considerando os requisitos de sensoriamento, comunicação e alimentação energética.

Na etapa seguinte, foi realizada a integração dos elementos de aquisição de dados e comunicação. Paralelamente, definiu-se a infraestrutura de recepção, composta pelos GWs responsáveis por captar os sinais transmitidos e encaminhá-los à camada de processamento. Posteriormente, estabeleceu-se a arquitetura de comunicação do sistema, viabilizando o envio das informações para a nuvem e sua disponibilização em plataformas de visualização. Por fim, foram implementados os mecanismos de armazenamento, monitoramento e apresentação dos dados, possibilitando o acompanhamento remoto das informações geradas. A metodologia foi concluída com a integração da solução proposta e a realização de testes de funcionamento, a fim de verificar a consistência da coleta, da transmissão e da exibição dos dados.

As subseções seguintes apresentam em detalhes cada uma dessas etapas, descrevendo sua função no desenvolvimento da solução proposta.

6.2.1 Definição das Variáveis de Monitoramento

A definição das variáveis de monitoramento constitui uma etapa fundamental do projeto, pois estabelece quais informações deverão ser coletadas pela coleira para atender aos objetivos propostos. Nesse contexto, foram definidas três categorias principais de variáveis a serem monitoradas: localização, variáveis ambientais e comportamentais e energia. Em conjunto, essas informações permitem uma análise mais abrangente da interação dos indivíduos com o ambiente, de seus padrões de deslocamento e atividade, bem como da confiabilidade de operação do sistema ao longo do tempo.

- **Localização:** A obtenção da posição geográfica dos animais é essencial para identificar padrões de movimentação, áreas de permanência, rotas de deslocamento e uso do *habitat*.
- **Variáveis ambientais:** O monitoramento de temperatura, umidade e luminosidade permite caracterizar as condições ambientais associadas à presença e ao deslocamento dos indivíduos. Essas informações contribuem para a identificação de preferências de *habitat*, para a análise da influência de fatores climáticos e sazonais sobre o comportamento animal e para a avaliação de possíveis impactos decorrentes de mudanças ambientais.
- **Comportamento:** Dados relacionados à aceleração e ao movimento possibilitam inferir padrões comportamentais, como períodos de atividade e repouso, deslocamentos mais intensos e possíveis alterações no comportamento habitual dos indivíduos.
- **Energia:** O acompanhamento do nível de bateria é indispensável para garantir a confiabilidade do sistema de monitoramento. Essa variável permite avaliar a autonomia do dispositivo, prever necessidades de manutenção e reduzir o risco de interrupções na coleta de dados causadas por falhas energéticas, assegurando maior continuidade e robustez à operação em campo.

A seleção dessas variáveis busca, portanto, atender de forma integrada às demandas de monitoramento espacial, ambiental e comportamental dos animais, sem negligenciar os requisitos operacionais do próprio sistema. Dessa forma, os dados coletados não se limitam ao acompanhamento individual dos espécimes, mas também fornecem subsídios relevantes para análises ecológicas mais amplas, planejamento territorial, definição de estratégias de conservação e apoio à tomada de decisão em ações voltadas à preservação da fauna.

6.2.2 Definição e Integração dos Componentes de *Hardware* e *Software*

A definição dos componentes de *hardware* e *software* do protótipo foi realizada com base nos requisitos de sensoramento, comunicação sem fio, autonomia energética e visualização remota dos dados. O objetivo foi selecionar elementos capazes de viabilizar uma solução integrada, de baixo custo e adequada ao monitoramento contínuo de animais silvestres em ambientes externos.

O protótipo da coleira foi desenvolvido com o microcontrolador Heltec ESP32 LoRa V2, escolhido por integrar capacidade de processamento, conectividade Wi-Fi/Bluetooth e comunicação LoRa em uma única plataforma, o que simplifica a implementação do sistema e reduz sua complexidade. A esse microcontrolador foram incorporados sensores destinados à aquisição

de variáveis ambientais, comportamentais e energéticas, bem como uma fonte de alimentação compatível com os requisitos de autonomia do dispositivo, conforme descrito a seguir:

- **DHT11:** sensor digital utilizado para medição de temperatura e umidade relativa do ar. Opera na faixa típica de 0°C a 50°C para temperatura e de 20% a 90% para umidade relativa, sendo uma solução de baixo custo adequada para monitoramento ambiental básico [55].
- **BH1750:** sensor digital de luminosidade, com medições expressas em lux, empregado para quantificar a intensidade luminosa do ambiente. Apresenta comunicação via interface I²C e faixa de medição típica de aproximadamente 1 a 65535 lux, sendo útil para a caracterização das condições de iluminação associadas ao *habitat* [56].
- **MPU6050:** módulo composto por acelerômetro e giroscópio triaxiais, utilizado para monitoramento de movimento e inferência comportamental. O acelerômetro opera com faixas configuráveis de $\pm 2g$, $\pm 4g$, $\pm 8g$ e $\pm 16g$, enquanto o giroscópio pode operar em ± 250 , ± 500 , ± 1000 e ± 2000 graus por segundo, permitindo capturar diferentes níveis de atividade e deslocamento [57].
- **MAX17043:** circuito integrado destinado ao monitoramento do nível de carga da bateria. Baseia-se em algoritmo interno de *fuel gauge* para estimativa do estado de carga (*State of Charge* (SoC)), fornecendo informações importantes para o acompanhamento da autonomia energética e da confiabilidade operacional do sistema [58].
- **Bateria LS20:** responsável pela alimentação do dispositivo, selecionada com foco em autonomia prolongada e compatibilidade com aplicações de monitoramento remoto. Sua utilização visa garantir maior tempo de operação do protótipo em cenários com restrição de manutenção frequente.

No que se refere ao *software*, o sistema foi estruturado em camadas funcionais responsáveis pela aquisição, transmissão, armazenamento, processamento e visualização dos dados, conforme descrito a seguir:

- **Firmware:** Desenvolvido e validado na Arduino IDE, sendo responsável pela leitura periódica dos sensores, organização dos dados em pacotes e transmissão das informações via LoRa.
- **Aplicativo de monitoramento:** Destinado à recepção, análise e acompanhamento remoto dos dados coletados, possibilitando a geração de relatórios, alertas e visualizações para apoio ao monitoramento.
- **Banco de dados:** Encarregado do armazenamento estruturado das informações coletadas, permitindo consultas, análises históricas e suporte à tomada de decisão.
- **Dashboard:** Interface de visualização empregada para acompanhamento integrado das variáveis ambientais, comportamentais e energéticas monitoradas pelo sistema.
- **Estimativa de localização baseada em RSSI:** Realizada a partir das medições de intensidade de sinal recebidas por múltiplos GWs, permitindo inferir de forma aproximada a posição relativa da coleira no ambiente monitorado.

Dessa forma, a combinação dos componentes de *hardware* e *software* definidos permite a implementação de uma arquitetura funcional e escalável, capaz de realizar, de maneira integrada, a coleta, a transmissão, o armazenamento e a visualização dos dados.

6.2.3 Estimativa de Posição via RSSI

A infraestrutura de recepção foi composta por três GWs baseados na plataforma Heltec ESP32 LoRa V2, distribuídos de forma a ampliar a cobertura de recepção dos pacotes transmitidos pela coleira. Cada GW recebe as transmissões LoRa e extrai, para cada pacote recebido, o valor de RSSI, utilizado como métrica de intensidade do sinal. Os GWs secundários encaminham seus valores de RSSI ao GW principal, responsável por concentrar as informações provenientes dos diferentes pontos de recepção. A partir da agregação dessas medidas, torna-se possível realizar uma estimativa aproximada da origem do sinal, com base na comparação dos níveis de potência recebida nos diferentes GWs. Embora essa abordagem não forneça posicionamento de alta precisão, ela constitui uma alternativa de baixo custo e baixo consumo energético para inferência espacial preliminar em cenários de monitoramento remoto.

6.2.4 Nuvem e visualização

O GW principal atua como nó concentrador da arquitetura, sendo responsável por publicar os dados agregados na nuvem por meio do protocolo MQTT, adotado em função de sua simplicidade e adequação a aplicações IoT. As informações transmitidas incluem tanto as variáveis monitoradas pelos sensores quanto os parâmetros associados à recepção dos sinais. Para a visualização e o acompanhamento remoto do sistema, foi implementado um *dashboard* na plataforma Node-RED, capaz de exibir, em tempo quase real, as variáveis coletadas e o estado operacional do protótipo.

6.3 Protótipo Desenvolvido e Resultados Obtidos

O protótipo desenvolvido consiste em uma coleira IoT baseada na plataforma Heltec ESP32 LoRa V2, integrada a sensores destinados ao monitoramento de variáveis ambientais, comportamentais e energéticas. Foram utilizados o sensor DHT11 para medição de temperatura e umidade, o BH1750 para monitoramento da luminosidade, o MPU6050 para aquisição de dados de aceleração e movimento, e o MAX17043 para monitoramento do nível de carga da bateria. A alimentação do sistema é realizada por uma bateria LS20. O *firmware*, validado na plataforma Arduino IDE, executa a leitura periódica dos sensores, organiza os dados em pacotes e realiza sua transmissão via LoRa. A Figura 21 apresenta o protótipo implementado.

Para o envio dos dados à infraestrutura de monitoramento, foi adotado o protocolo MQTT, com utilização do *broker* público `test.mosquitto.org`, acessado pela porta padrão 1883. Cada variável monitorada foi associada a um tópico específico, permitindo a organização do fluxo de dados e sua posterior integração com ferramentas de visualização e análise. Foram definidos tópicos para temperatura, umidade, luminosidade, aceleração, giroscópio e também para as estimativas de distância associadas aos diferentes pontos de recepção.

A estimativa de posição relativa da coleira foi realizada a partir dos valores de RSSI obtidos pelos GWs. Com base nesses valores, foi implementada uma função de conversão de RSSI em distância fundamentada em um modelo logarítmico de perda de percurso, expressa por

$$d = 10^{\frac{RSSI_0 - RSSI}{10n}}, \quad (1)$$

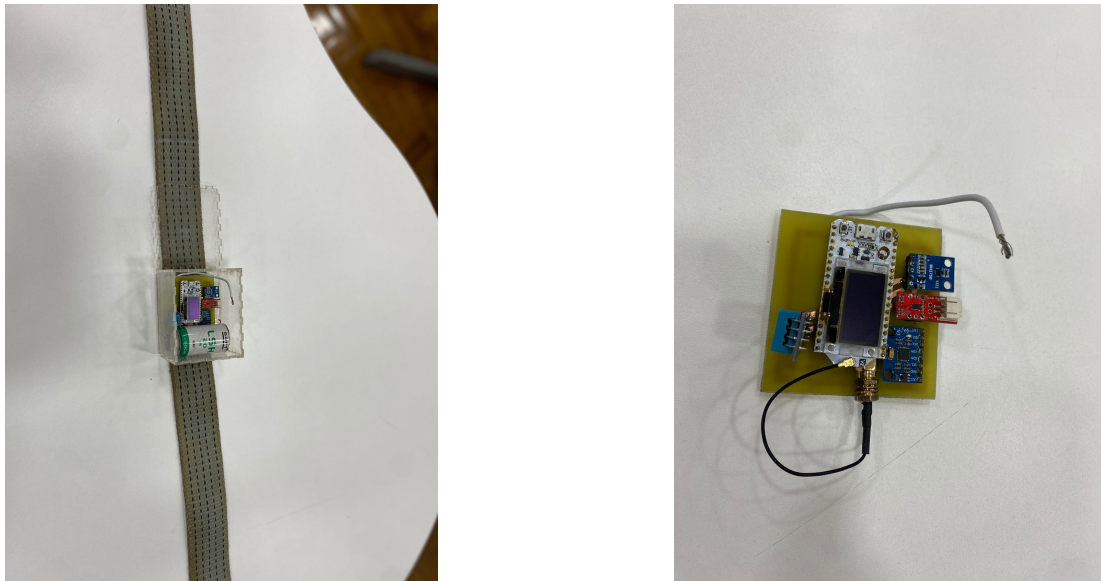


Figura 21: Protótipo desenvolvido. À esquerda, a coleira instrumentada; à direita, a PCB.

em que d representa a distância estimada, $RSSI_0$ corresponde ao valor de referência medido a 1 metro de distância, $RSSI$ é a intensidade do sinal recebido e n é o coeficiente de perda de percurso, adotado como $n = 4$ neste trabalho. Embora influenciada por fatores ambientais, essa abordagem permite obter uma estimativa espacial preliminar adequada ao contexto de validação do protótipo.

Após o processamento local, os dados são publicados periodicamente nos tópicos MQTT correspondentes. O *firmware* incorpora ainda mecanismos de verificação de validade, por meio da função `isnan()`, a fim de evitar a transmissão de valores inconsistentes e aumentar a confiabilidade do sistema. Essa lógica permite que o protótipo realize, de forma integrada, a aquisição, o processamento e a disponibilização dos dados em tempo quase real para acompanhamento remoto.

Os testes foram conduzidos em ambiente de bancada/laboratório com o objetivo de validar o funcionamento ponta a ponta da arquitetura proposta. Nessa etapa, foram verificados: (i) a aquisição das variáveis monitoradas pelos sensores; (ii) a transmissão dos dados via LoRa entre a coleira e os GWs; (iii) a obtenção e o encaminhamento dos valores de RSSI pelos pontos de recepção; e (iv) a publicação e a visualização das informações por meio da infraestrutura MQTT/Node-RED. A Figura 22 apresenta o *dashboard* implementado em Node-RED, utilizado para o acompanhamento, em tempo quase real, das leituras dos sensores e do estado da comunicação do sistema. Os resultados obtidos evidenciaram a integração adequada entre os módulos de sensoriamento, comunicação, processamento e visualização, demonstrando a viabilidade funcional da arquitetura desenvolvida.

6.3.1 Estimativa de custo do protótipo

Com o objetivo de avaliar a viabilidade econômica da solução proposta, foi realizado o levantamento dos custos dos principais componentes eletrônicos empregados na construção do protótipo da coleira de monitoramento. A Tabela 2 apresenta os valores estimados de cada componente utilizado.

Considerando os dispositivos empregados na implementação, o custo total estimado do pro-

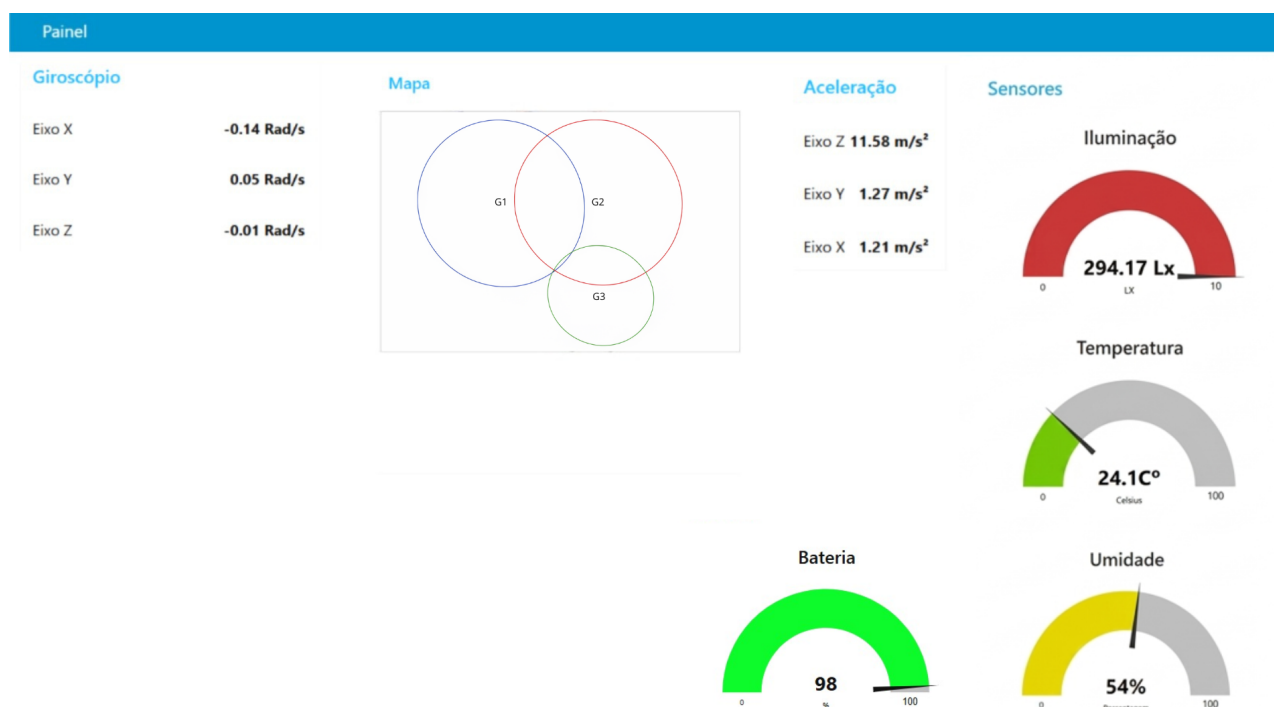


Figura 22: *Dashboard* desenvolvido em Node-RED para visualização das variáveis coletadas e dos dados recebidos via MQTT.

tótipo foi de aproximadamente R\$ 923,42. Observa-se que a bateria representa a maior parcela desse valor, enquanto os demais componentes apresentam custo relativamente reduzido. Esses resultados indicam que a arquitetura proposta possui potencial para implementação de uma solução de monitoramento remoto com custo inferior ao de alternativas comerciais importadas (USD 2.000,00), mantendo a capacidade de aquisição, transmissão e visualização das variáveis de interesse.

Tabela 2: Custo estimado do protótipo desenvolvido.

Componente	Quantidade	Custo (R\$)
Heltec ESP32 LoRa V2	1	235,00
Sensor DHT11	1	20,00
Sensor BH1750	1	18,52
Sensor MPU6050	1	20,80
MAX17043	1	49,50
Bateria LS20	1	579,60
Total	—	923,42

6.4 Conclusões

Este trabalho apresentou o desenvolvimento e a validação, em bancada/laboratório, de um protótipo nacional de coleira IoT para monitoramento de fauna, baseado em comunicação LoRa, múltiplos GWs e uso de RSSI para estimativa aproximada de posição. A arquitetura proposta

integrou sensoriamento ambiental, comportamental e energético, além de transmissão, agregação, publicação em nuvem e visualização remota dos dados. Os resultados demonstraram a viabilidade funcional da solução, confirmando a operação ponta a ponta do sistema, desde a aquisição das variáveis até sua transmissão e visualização em *dashboard*. Adicionalmente, a estimativa de custo indica potencial para o desenvolvimento de uma alternativa nacional de menor custo em relação a soluções comerciais importadas. Como trabalhos futuros, destacam-se a miniaturização do dispositivo por meio de *Printed Circuit Board* (PCB) dedicada, o aprimoramento da eficiência energética, o desenvolvimento de encapsulamento apropriado para uso externo, a evolução do aplicativo de monitoramento e a investigação de soluções LoRa via satélite para ampliação da cobertura. Esses avanços podem contribuir para tornar a solução mais robusta, escalável e adequada a aplicações de monitoramento remoto e conservação da fauna.

7 Detecção Inteligente de Incêndios Florestais Usando Modelos de Aprendizado Profundo

7.1 Introdução

Incêndios florestais caracterizam-se pela combustão rápida e descontrolada de biomassa vegetal, incluindo folhas, galhos e árvores, resultando na degradação extensiva de áreas naturais e na degradação dos ecossistemas. Esses eventos representam uma ameaça crítica à biodiversidade e promovem alterações microclimáticas e atmosféricas. Além disso, comprometem a saúde pública devido à emissão de poluentes atmosféricos, como monóxido de carbono, material particulado e compostos orgânicos voláteis. As causas podem ser de origem natural, como descargas atmosféricas, ou antrópicas, como práticas agropecuárias irregulares, queimadas intencionais e expansão desordenada da fronteira agrícola. A detecção precoce é estratégica para a mitigação de danos ambientais e humanos, demandando sistemas integrados de monitoramento e resposta, com a participação coordenada de órgãos governamentais, comunidades locais e tecnologias emergentes [59, 60, 61].

No Brasil, a incidência de incêndios florestais tem apresentado uma tendência crescente e preocupante. Entre os anos de 1985 e 2023, estima-se que mais de 10 milhões de hectares de vegetação nativa foram consumidos por queimadas. O bioma Caatinga, exclusivamente brasileiro e caracterizado por uma vegetação xerofítica, apresenta, em média, 0,56% de sua extensão comprometida anualmente por incêndios. Em 2022, foram registrados aproximadamente 85 mil focos ativos de incêndio no território nacional. Esse número aumentou para 102 mil em 2023, e em 2024 o país alcançou um novo recorde, com mais de 110 mil focos registrados, correspondendo à devastação de uma área estimada em 114 mil km², um crescimento de 116% em relação ao ano anterior. A intensificação desses eventos, fortemente associada a períodos prolongados de estiagem e à ocorrência do fenômeno climático El Niño, reforça a necessidade urgente de desenvolver e implementar soluções tecnológicas robustas para o monitoramento, a prevenção e o combate a incêndios florestais [62, 63].

Considerando o agravamento do cenário atual, a detecção precoce de incêndios florestais torna-se um elemento crítico para a mitigação dos impactos ambientais, sociais e econômicos. A identificação de focos de calor ainda em estágios iniciais viabiliza ações de resposta mais rápidas e eficazes, contribuindo para a contenção da propagação do fogo e a preservação de ecossistemas, bem como para a redução dos riscos à saúde pública. As estratégias tradicionais de monitoramento incluem torres de vigilância, estruturas posicionadas em pontos estratégicos de áreas florestais, com alturas variando entre 15 e 50 metros, dependendo das características topográficas, além de patrulhamento aéreo e análise de séries temporais de dados históricos. No entanto, essas abordagens convencionais apresentam limitações quanto à cobertura espacial e à acurácia em tempo real, fatores essenciais para decisões operacionais em contextos de emergência. Frente às crescentes ameaças provocadas pelas mudanças climáticas e à complexidade intrínseca dos ecossistemas florestais, torna-se imperativo adotar soluções baseadas em

O conteúdo deste capítulo consiste em um resumo adaptado pelo pesquisador *Samuel B. Mafra*, com base no artigo apresentado no XLIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT2025) por Tarciso G. B. de Bello e Samuel B. Mafra. "Detecção Inteligente de Incêndios Florestais Usando Modelos de Aprendizado Profundo".

tecnologias emergentes, capazes de fornecer monitoramento contínuo, detecção automatizada e resposta integrada.

Neste trabalho, propõe-se uma abordagem avançada para a detecção e análise de incêndios florestais com base em algoritmos de aprendizado profundo. O principal objetivo é monitorar a dinâmica dos focos de incêndio, identificando padrões de variação temporal que indiquem se o fogo está em fase de expansão ou contenção. Essa análise permite não apenas a detecção precoce, mas também o acompanhamento contínuo da evolução do incêndio, subsidiando ações mais precisas de resposta e alocação de recursos.

7.2 Solução Proposta

Neste trabalho, propõe-se uma solução baseada em IoT para o monitoramento em tempo real e a detecção de focos de incêndio, utilizando algoritmos de aprendizado profundo. Foram selecionados os modelos R-CNN, YOLOv8 e YOLOv11, considerando seu elevado desempenho, ampla adoção em estudos correlatos e capacidades computacionais. A metodologia adotada consiste em uma análise comparativa entre esses modelos sob condições experimentais similares, com o objetivo de avaliar suas respectivas métricas de desempenho, como acurácia, precisão, revocação e tempo de inferência. O modelo R-CNN foi incluído por sua abordagem baseada em múltiplas passagens e capacidade de análise detalhada de regiões específicas da imagem. Por outro lado, os modelos YOLOv8 e YOLOv11 foram escolhidos por sua arquitetura de detecção de passagem única (*single-shot*), que permite processamento mais rápido e eficiente, tornando-os particularmente adequados para aplicações em tempo real.

Neste contexto, são empregados diferentes modelos de aprendizado profundo com o objetivo de realizar a detecção precisa de focos de incêndio em ambientes florestais. Essa tarefa apresenta elevada complexidade devido à variabilidade temporal e espacial das condições ambientais durante os incêndios, como alterações climáticas, presença de fumaça, variações de luminosidade e heterogeneidade da cobertura vegetal. Os modelos analisados foram aplicados tanto à identificação de focos de pequena quanto de grande extensão, abrangendo diferentes tipos de vegetação e cenários ambientais. A Figura 23 exemplifica uma dessas situações, onde é possível identificar focos de incêndio em uma área de vegetação.

Tendo como objetivo a detecção do aumento ou redução dos focos de incêndios, é proposto um sistema como ilustrado na Figura 24, onde uma câmera é responsável por coletar as imagens dos focos de incêndio e um dispositivo computacional como a Raspberry Pi pode ser utilizada para processamento das imagens e envio dos dados coletados, como em um cenário de IoT.

A coleta e o treinamento de dados constituem etapas fundamentais para assegurar que os modelos de detecção sejam expostos a um conjunto de treinamento diversificado e representativo das condições reais associadas a incêndios florestais. A construção do banco de imagens, ou *dataset*, foi orientada pela necessidade de contemplar tanto cenários com presença de fogo quanto situações de normalidade, abrangendo diferentes condições ambientais, tipos de vegetação e variações de iluminação. As imagens utilizadas foram obtidas durante treinamentos operacionais realizados pelo Corpo de Bombeiros de Pouso Alegre, Minas Gerais, Brasil. As coletas foram realizadas por meio de drones e câmeras posicionadas em múltiplos pontos da área queimada, permitindo a obtenção de um conjunto de dados heterogêneo e adequado ao treinamento robusto dos modelos propostos.

Após a etapa de seleção das imagens, procede-se à fase de rotulagem, a qual desempenha um papel fundamental na qualidade do treinamento dos modelos de detecção, como R-CNN, YOLOv8 e YOLOv11. A acurácia do processo de anotação impacta diretamente a capacidade



Figura 23: Imagem do Treinamento - Detecção de Incêndios

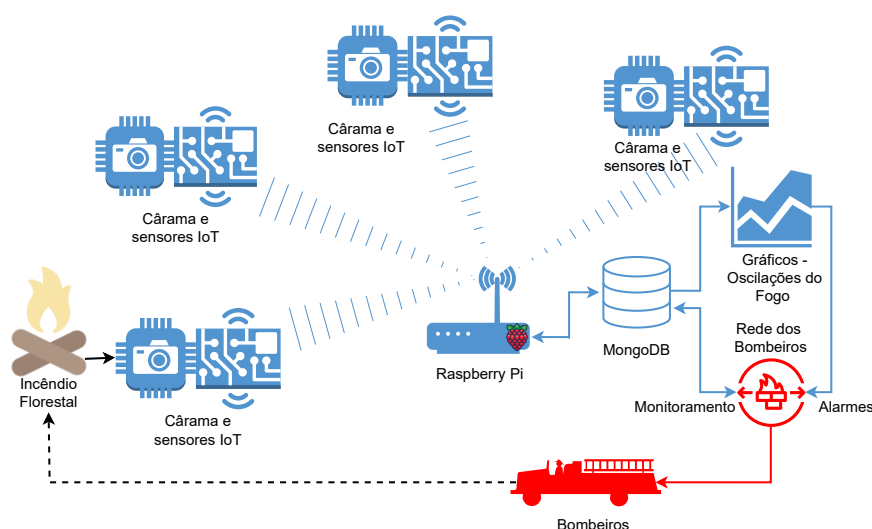


Figura 24: Arquitetura da solução IoT proposta para detecção de incêndios.

dos modelos em aprender a identificar, de forma precisa, os objetos de interesse — neste caso, os focos de incêndio. A rotulagem foi realizada manualmente, por meio da inserção de caixas delimitadoras (*bounding boxes*) sobre as regiões afetadas pelo fogo, permitindo a marcação exata das áreas relevantes nas imagens. Para essa tarefa, utilizou-se a plataforma Roboflow, que oferece um ambiente integrado para anotação, gerenciamento e exportação de *datasets*, otimizando o fluxo de trabalho e garantindo maior consistência nos dados utilizados para treinamento e validação dos modelos.

Após a etapa de rotulagem, as 80 imagens selecionadas foram particionadas em três conjuntos: 70% destinadas ao treinamento, 15% à validação e 15% ao teste. Essa divisão visa maximizar o aprendizado do modelo a partir do maior volume possível de dados, ao mesmo tempo em que permite monitorar o desempenho durante o processo de treinamento (validação) e avaliar sua capacidade de generalização em dados não vistos (teste). Em seguida, os con-

juntos de imagens foram integrados aos modelos R-CNN, YOLOv8 e YOLOv11 para a fase de treinamento supervisionado.

7.3 Resultados

Com o objetivo de avaliar a eficácia da metodologia proposta, os experimentos foram conduzidos utilizando o banco de dados de imagens de incêndios florestais descrito na seção anterior. Para os testes, todos os modelos de aprendizado profundo, R-CNN, YOLOv8 e YOLOv11, foram treinados por 2000 épocas, com imagens redimensionadas para 640×640 pixels e tamanho de lote (*batch size*) igual a 16. Os resultados obtidos indicam que o modelo YOLOv11 apresentou desempenho superior em comparação aos demais, conforme evidenciado na Tabela 3. A evolução do treinamento do YOLOv11 é ilustrada na Figura 25, a qual mostra a redução progressiva das funções de perda (*loss functions*) ao longo das iterações, tanto para o conjunto de treinamento quanto para o de validação, sinalizando uma convergência estável do modelo. Tal comportamento é indicativo de um processo de aprendizado eficiente, com adequada adaptação aos dados e ausência de indícios de *overfitting*. A trajetória semelhante observada na curva de perdas do conjunto de validação reforça a capacidade de generalização do modelo frente a dados não previamente vistos.

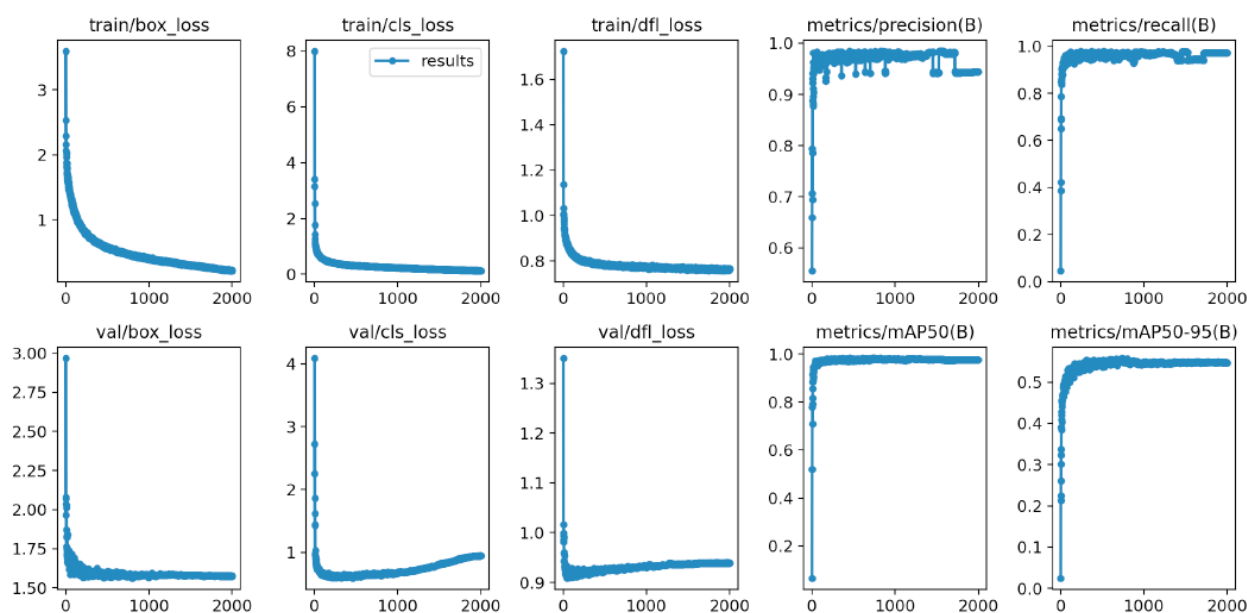


Figura 25: Evolução do treinamento do YOLOv11

Tabela 3: Comparação do Desempenho dos modelos de Detecção de Incêndios Florestais.

Modelos	Acurácia	Precisão	Revocação	F1-Score
YOLOv11	98,60%	98,97%	98,90%	98,97%
YOLOv8	98,10%	98,10%	98,30%	98,35%
R-CNN	94,65%	95,80%	96,85%	96,87%

As subfiguras correspondentes às métricas de *Precision*, *Recall* e *mean Average Precision* fornecem uma análise detalhada da evolução do desempenho do sistema ao longo das épocas

de treinamento. Essas métricas são fundamentais para avaliar a capacidade dos modelos em detectar focos de incêndio com precisão, mantendo um equilíbrio entre sensibilidade e especificidade, sem comprometer a eficiência em tempo real. As variações observadas nas curvas de validação podem ser interpretadas como reflexos da adaptação do modelo às variações nos dados, enquanto a estabilidade das métricas durante as iterações indica a robustez e a consistência do YOLOv11 frente ao processo de generalização. O acompanhamento contínuo desses indicadores ao longo das fases de treinamento e validação é essencial para garantir que o modelo alcance o melhor desempenho possível na tarefa de detecção de incêndios florestais.

Foi realizado um teste experimental de validação da solução IoT para detecção de incêndios utilizando uma câmera acoplada a uma plataforma Raspberry Pi. O experimento empregou uma pequena churrasqueira como fonte controlada de fogo, permitindo avaliar a capacidade de captura, processamento e transmissão dos dados pela solução proposta.

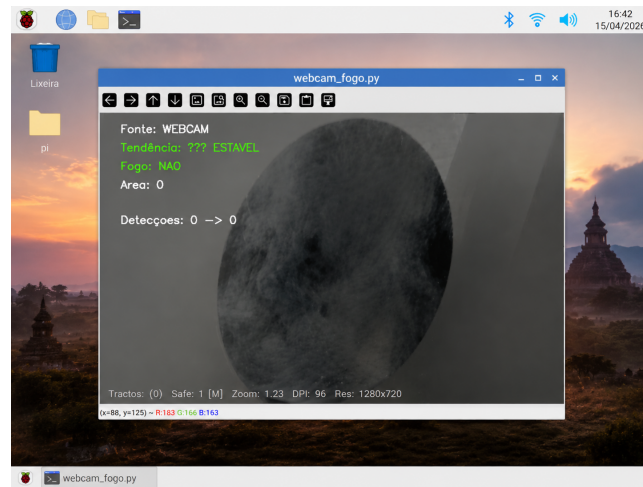


Figura 26: *Dashboard* de detecção dos incêndios

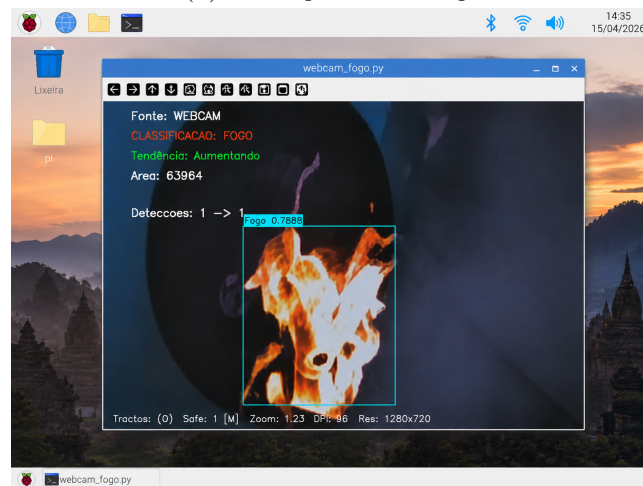
A Figura 26 mostra o funcionamento do sistema de monitoramento em tempo real, incluindo a identificação automática de focos de incêndio, o cálculo da probabilidade de ocorrência de fogo e o armazenamento do histórico de detecções. Os dados foram transmitidos e visualizados em um *dashboard* desenvolvido na plataforma Grafana, possibilitando o acompanhamento gráfico dos eventos detectados, bem como a análise temporal das ocorrências.

As Figuras 27a e 27b, referentes à detecção de fumaça e fogo, evidenciam a capacidade do sistema em identificar diferentes estágios de ocorrência de incêndio, realizando a detecção automática dos eventos e registrando as informações em tempo real para monitoramento e análise no *dashboard* IoT.

Os resultados indicam o correto funcionamento da arquitetura IoT desenvolvida, validando a integração entre câmera, processamento embarcado na Raspberry Pi, transmissão de dados e visualização remota das informações de detecção de incêndio.



(a) Detecção de fumaça



(b) Detecção de fogo

Figura 27: Resultados da detecção de fumaça e fogo utilizando a solução IoT embarcada em Raspberry Pi.

7.4 Conclusão

A proposta apresentada neste trabalho evidencia o potencial dos modelos de aprendizado profundo na tarefa de detecção automatizada de incêndios florestais. Conforme demonstrado nos experimentos, o modelo YOLOv11 destacou-se em relação ao YOLOv8 e ao R-CNN, apresentando melhor desempenho nas métricas avaliadas. Apesar dos resultados promissores, ainda persistem desafios relevantes, como a necessidade de um conjunto de dados mais abrangente e representativo, que contemple diferentes condições ambientais, tipos de vegetação e variações sazonais. Além disso, a inclusão de novas classes, como a detecção de fumaça e outros indícios precoces de incêndio, pode contribuir significativamente para o aumento da acurácia e da capacidade preditiva dos modelos.

A aplicação de modelos de visão computacional e aprendizado profundo, como YOLOv11, YOLOv8 e R-CNN, na detecção de incêndios florestais representa uma abordagem inovadora e altamente eficaz, com grande potencial para a proteção ambiental e a mitigação de riscos à vida humana. Em contraste com as técnicas convencionais de monitoramento, que geralmente dependem de processos manuais e apresentam limitações em termos de tempo de resposta e

cobertura espacial, as soluções baseadas em IA viabilizam a detecção em tempo real de focos de incêndio. Essa capacidade de resposta rápida e precisa é especialmente relevante em regiões de difícil acesso, onde os métodos tradicionais frequentemente enfrentam limitações operacionais. Dessa forma, o uso de tecnologias inteligentes se configura como uma ferramenta estratégica para aprimorar os sistemas de vigilância ambiental e gestão de emergências florestais.

8 Posicionamento de *Gateways* LoRaWAN para Planejamento de Rede Usando Modelos de Canal Baseados em Traçado de Raios

8.1 Introdução

O planejamento de rede é uma tarefa fundamental em sistemas de comunicação sem fio, particularmente em implantações de LPWAN, devido à alta variabilidade e aos desafios inerentes a essas redes, incluindo taxas de transmissão muito baixas, vazão limitada e restrições rigorosas de energia [64, 65, 66, 67]. Além disso, o planejamento de rede é uma tarefa crucial no que diz respeito à viabilidade econômica do sistema, pois permite um controle rigoroso sobre os investimentos em *hardware* e os custos contínuos de manutenção [64, 68, 69]. Dessa forma, garante que a infraestrutura atenda a requisitos como *Quality of Service* (QoS) e confiabilidade para cenários de aplicação específicos, evitando falhas de conectividade e assegurando a entrega estável de dados [70]. Portanto, esse tipo de tarefa também é essencial para a escalabilidade, pois prepara a rede para suportar o crescimento futuro da carga de tráfego e da densidade de dispositivos sem degradação de desempenho [71]. Em relação aos tipos de LPWANs, um dos mais populares é o LoRaWAN, que é particularmente adequado para cenários de IoT que exigem comunicação de longo alcance, baixo consumo de energia e baixo custo.

LoRaWAN é um protocolo baseado no esquema de modulação LoRa [72, 73] e opera utilizando três componentes principais: dispositivo final, GW e um servidor de rede. Em relação a esses componentes, um dos aspectos mais importantes do planejamento de redes LoRaWAN é a posição e a quantidade de GWs [65], uma vez que esses fatores afetam diretamente a área de cobertura alcançável e a capacidade dos dispositivos finais de atender aos seus requisitos de comunicação. Um planejamento inadequado da rede pode resultar em dispositivos finais apresentando problemas de conectividade [74].

Para determinar a configuração ótima de GWs para uma determinada topologia de rede, abordagens baseadas em medições [64, 75] e métodos baseados em simulação podem ser empregados para planejar as melhores posições e a quantidade de GWs destinadas a atender aos requisitos dos dispositivos finais. Neste trabalho, assumimos uma abordagem baseada em simulação, considerando diferentes modelos de canal. Nesse sentido, modelos comuns incluem canais estocásticos, como aqueles definidos pelo *3rd Generation Partnership Project* (3GPP) [76], e abordagens empíricas, como modelos *log-distance* [77, 70] e Okumura-Hata [68, 66], que podem oferecer flexibilidade em termos de parâmetros e baixo custo computacional. Para realizar simulações independentes de um cenário 3D específico, é necessário aceitar um compromisso entre a fidelidade dos resultados em termos de parâmetros de larga escala, como o padrão de cobertura de cada transmissor. Assim, para alcançar um desempenho confiável nessa tarefa, simular com precisão o canal sem fio é indispensável para garantir a qualidade das simulações sem fio e, conseqüentemente, a confiabilidade dos resultados do planejamento de rede [78]. Tal precisão pode ser alcançada utilizando ferramentas especializadas, como simuladores de RT [79, 80].

Considerando esse contexto, nas próximas seções, são apresentados nossos esforços no desenvolvimento de um *framework* que integra simuladores de RT a um simulador de rede de eventos

O conteúdo deste capítulo consiste em um resumo adaptado pelo pesquisador *Cláudio Modesto*, com base no artigo submetido para o *IEEE Open Journal of the Communications Society* por Cláudio Modesto, Lucas Mozart, Glauco Gonçalves, Cleverson Nahum, Bruno Castro, e Aldebaro Klautau. “LoRaWAN Gateway Placement for Network Planning Using Ray Tracing-Based Channel Models”.

discretos. Utilizando esse arcabouço para gerar métricas de dados sem fio em LoRaWAN, empregamos um modelo de otimização que determina o posicionamento ótimo dos GWs sob diferentes modelos de canal e restrições de potência recebida. Nossos resultados mostram que a solução otimizada é altamente sensível ao modelo de canal escolhido, mesmo ao considerar os mesmos cenários com diferentes simuladores de RT, revelando um claro *trade-off* entre custo computacional e fidelidade da solução às condições do mundo real.

8.2 Simulações de comunicações LoRaWAN com RT

A integração entre RT e simuladores de rede de eventos discretos, como o NS-3, depende fortemente do sistema de comunicação alvo e das métricas de desempenho de alto nível de interesse. Um exemplo recente desse tipo de integração é o uso de canais gerados por RT para possibilitar simulações mais realistas de Quinta Geração de Rede Móvel Celular (5G), por exemplo, ao acoplar o módulo 5G-LENA [81] com respostas ao impulso do canal produzidas pelo Sionna RT [82]. Uma abordagem semelhante pode ser adotada para sistemas Wi-Fi 802.11, utilizando o mesmo mecanismo de RT para fornecer modelos de canal detalhados ao simulador de rede [83, 84]. No entanto, para simulações de LoRaWAN, pelo nosso conhecimento, esse tipo de integração ainda não foi explorado na literatura. Como é essencial simular e visualizar um sistema de comunicação em diferentes níveis de abstração, abordamos essa lacuna com um *framework* que integra simuladores de RT (tanto de código aberto quanto comerciais) com um simulador de rede discreto (NS-3), com o objetivo de fornecer simulações de LoRaWAN com diferentes níveis de fidelidade em relação aos canais sem fio.

8.3 RT-LoRaWAN *framework*

O fluxo operacional do *framework* proposto é definido pela Figura 28. Esta figura demonstra como o *pipeline* do nosso *framework* opera. Em particular, ela mostra que o *framework* pode funcionar tanto em conjunto com simuladores RT e NS-3-LoRaWAN, quanto como um sistema independente (por exemplo, utilizando apenas o NS-3 LoRaWAN). Caso seja executado de forma conjunta utilizando o canal realista proveniente de RT, o *pipeline* inicia com (1) a definição dos cenários 3D que devem ser utilizados e dos parâmetros de configuração do RT. Para cenários 3D, o formato de entrada requerido depende do simulador de RT selecionado. Se o Sionna RT for utilizado, o cenário deve ser fornecido no formato Mitsuba [85]. Caso o WI seja utilizado, o cenário 3D deve seguir o formato *Digital Asset Exchange* (DAE) (COLLADA). Por fim, para obter esses cenários 3D, é possível importar a partir do OpenStreetMap, conforme proposto em [82], observando que, após a importação, é necessário converter para os formatos de arquivo mencionados anteriormente, utilizando *softwares* como o Blender [86].

Em relação à configuração dos parâmetros, o *framework* inclui a posição do transmissor, a frequência de portadora utilizada, o padrão de radiação, a polarização, os algoritmos de RT e outros parâmetros específicos da simulação de RT. É importante destacar que, nesse nível de simulação, abstraímos o conceito de GW ou dispositivo final; na camada física, consideramos apenas transmissores e receptores [78].

Após a conclusão de uma simulação de RT na etapa (2), podemos extrair diversas características da camada física. Uma dessas características é o ganho de percurso $\mathcal{G}$ em dB para cada dispositivo final, que é armazenado em arquivos separados por posição de GW. Nesse sentido, essa métrica é extraída de uma matriz que representa o ganho de percurso em uma porção específica da área, cujo tamanho é definido na etapa anterior da simulação de RT. Essas

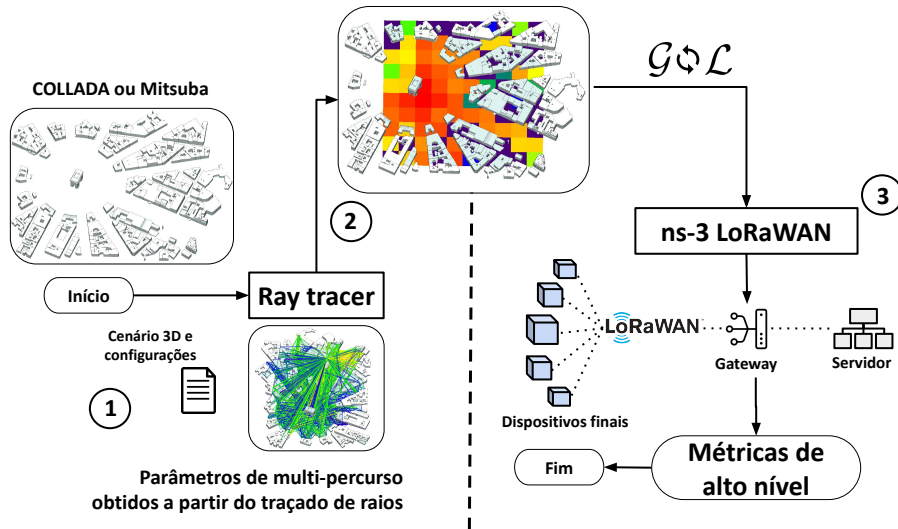
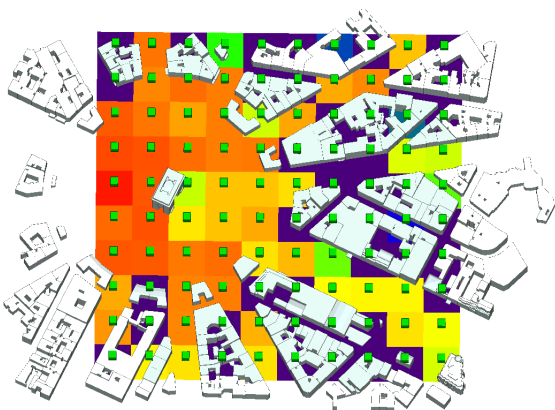


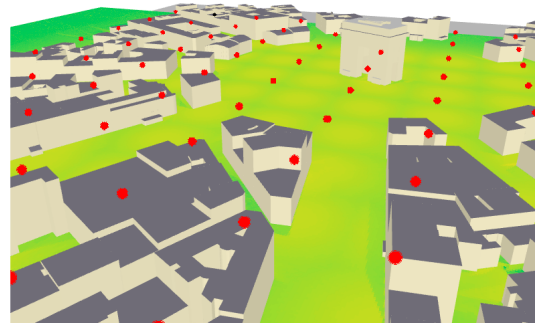
Figura 28: *Framework* proposto que integra um simulador de RT e um simulador de rede de eventos discretos, utilizado para simular cenários de LoRaWAN em diferentes camadas de rede.

informações servirão como interface entre ambos os simuladores. As Figuras 29b e 29a mostram o ganho de percurso calculado utilizando Sionna RT e WI, respectivamente. De posse dessas informações, nós as utilizamos no módulo NS-3 LoRaWAN para configurar os canais LoRaWAN. No entanto, como o NS-3 requer a perda de percurso $\mathcal{L}$ em dB, convertemos o ganho de percurso em perda de percurso, observando que $\mathcal{L} = -\mathcal{G}$.

Por fim, com as informações de perda de percurso em mãos, estas podem ser utilizadas na etapa (3) para simular cenários de rede LoRaWAN sob uma perspectiva de alto nível, no sentido de avaliar e obter métricas em nível de pacotes, como a taxa de pacotes entregues, que é a razão entre pacotes recebidos e pacotes enviados. Além disso, nesse nível de simulação, temos maior flexibilidade para considerar modelos de configuração de energia e diferentes modelos de canal para fins de comparação. Por exemplo, nesse nível de simulação, podemos definir parâmetros como o tempo de simulação e o *Spreading Factor* (SF).



(a) Mapa de cobertura da potência recebida utilizando o WI com o algoritmo X3D.



(b) Mapa de cobertura da potência recebida utilizando Sionna.

Figura 29: Exemplo de saída gráfica de simulações com RT (potência recebida) avaliada utilizando WI e Sionna RT.

8.4 Otimização no posicionamento de *gateways*

Podemos classificar formalmente esse problema como um problema de programação linear inteira binária, com uma variável de decisão binária x^p que indica se existe um GW em uma determinada posição. Nesse caso, a posição p pertence a um conjunto de posições possíveis $\mathcal{P} = \{1, 2, \dots, p, \dots, P\}$, com um total de P posições provenientes de uma malha. Nesse conjunto, cada índice representa uma coordenada cartesiana no espaço. Cada GW posicionado deve oferecer cobertura aos dispositivos finais na rede. Nesse sentido, assumimos que um dispositivo final d pertence a um conjunto $\mathcal{D} = \{1, 2, \dots, d, \dots, D\}$, composto por um total de D dispositivos. Assim, nosso objetivo é minimizar o número de GW implantados, levando à seguinte função objetivo:

$$\min_{x^p} \sum_{p \in \mathcal{P}} x^p. \quad (2)$$

Quanto às restrições, a função objetivo definida na Equação (2) está sujeita a uma restrição de potência recebida no dispositivo final. Neste trabalho, assumimos que a potência recebida α é igual ao ganho de percurso ($\alpha = \mathcal{G}$), uma vez que a potência de transmissão é definida como 0 dBm nos dados fornecidos pelo nosso *framework*. Dessa forma, essa restrição garante que a comunicação entre um dispositivo final e um GW ocorra apenas quando a potência recebida α no dispositivo final excede um determinado limiar ρ em dBm. Assumimos que já pré-computamos se o dispositivo final d -ésimo está coberto por algum GW.

Nesse sentido, esse cálculo pressupõe que podemos determinar a potência recebida para cada dispositivo final uma vez que a posição de um GW está definida, utilizando o simulador RT ou por meio da simulação com canais independentes do local usando o NS-3. Assim, para cada possível localização de GW, podemos determinar previamente se um determinado dispositivo final está coberto. Com essa informação, definimos a variável β_d^p , que indica a cobertura do d -ésimo dispositivo final pelo p -ésimo GW, onde $\beta_d^p = 0$ indica que o d -ésimo dispositivo final não está coberto ($\alpha < \rho$) pelo GW na posição p , e $\beta_d^p = 1$ indica que o dispositivo final está coberto ($\alpha \geq \rho$) pelo GW na posição p . Com essa informação de cobertura dos dispositivos finais pré-computada, definimos, então, um conjunto de restrições de cobertura múltipla da seguinte forma:

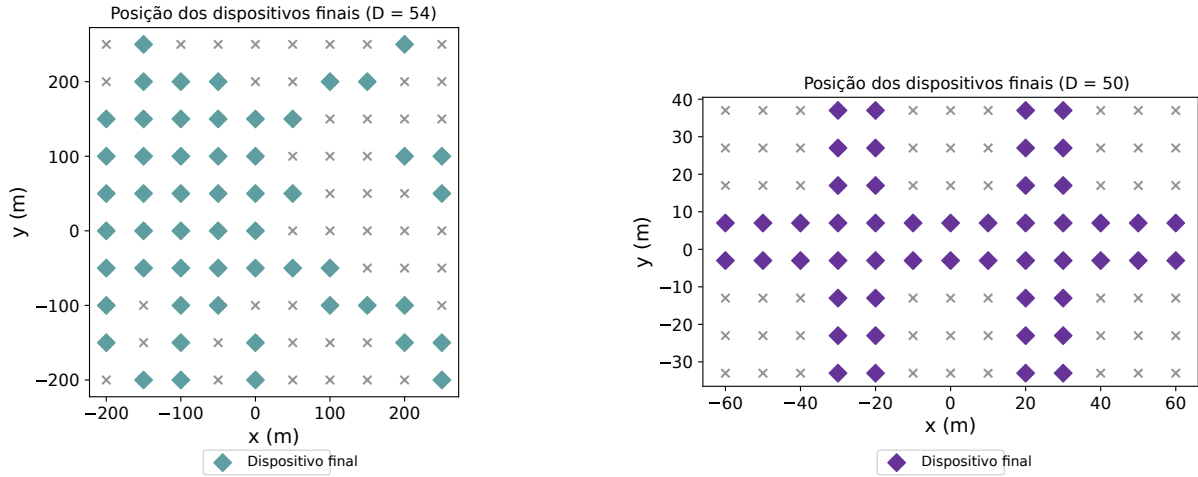
$$\sum_{p \in \mathcal{P}} \beta_d^p x^p \geq 1, \quad \forall d \in \mathcal{D}. \quad (3)$$

Com esse modelo padrão de otimização binária, buscamos destacar o impacto do modelo de canal nos resultados do planejamento de rede sob condições controladas e reproduzíveis, com o objetivo de demonstrar, na próxima seção dedicada aos experimentos propostos, que mesmo com um conjunto reduzido de restrições, a solução ótima pode variar abruptamente em termos do número de GWs, da posição de cada GW e, conseqüentemente, superestimar negativamente o desempenho de métricas de alto nível, como a taxa de pacotes entregues.

8.5 Experimentos

Para avaliar como diferentes modelos de canal afetam as soluções para o problema formulado de posicionamento de GW, tanto em termos do número de GW sugeridos quanto do custo computacional necessário para obtê-los, conduzimos uma série de avaliações utilizando o *framework* proposto. Para essas avaliações, consideramos duas topologias de rede baseadas em *layouts* de malha uniforme. Nas configurações de malha 1 e 2, as posições candidatas adjacentes estão espaçadas em 50 m e 10 m, respectivamente. Seguindo uma metodologia semelhante à de [70],

essas malhas fornecem $P = 100$ e $P = 104$ posições candidatas para GW. Para cada posição candidata, um GW é implantado a uma altura de 30 m na configuração de malha 1 e 60 m na configuração de malha 2.



(a) Configuração espacial dos dispositivos finais usando a malha 1

(b) Configuração espacial dos dispositivos finais usando a malha 2

Figura 30: Organização dos dispositivos finais que devem ser cobertos por um conjunto de GWs em P posições.

Quanto à organização dos dispositivos finais, assumimos que cada GW implantado atende a $D = 54$ dispositivos na configuração de malha 1 e $D = 50$ dispositivos na configuração de malha 2. Cada dispositivo é posicionado a uma altura de 1,4 m e é distribuído espacialmente de acordo com as malhas mostradas nas Figuras 30a e 30b. Cada posição válida é definida de modo a evitar interseções com quaisquer edificações quando utilizada em um cenário 3D. Com essa configuração, para cada posição na malha, posicionamos um GW para avaliar a potência recebida no conjunto de dispositivos finais. Como consideramos cada GW de forma isolada, não modelamos interferência entre GWs. Após calcular essas potências recebidas para todas as posições possíveis de GW, utilizamo-las para resolver o problema de otimização. Em resumo, as características topológicas utilizadas em cada cenário são definidas na Tabela 4.

Tabela 4: Características das malhas de dispositivos finais usados em diferentes cenários

Malha	P	D	Altura do GW (m)	Altura do dispositivo final (m)
1	100	54	30	1,4
2	104	50	60	1,4

Os dados coletados a partir dessa topologia proposta foram, então, utilizados em experimentos envolvendo dois grupos de modelos de canal. O primeiro grupo contém todos os modelos independentes do local, representados por Okumura-Hata, COST-231 [87], *log-distance* e 3GPP-UMa. Quanto aos parâmetros dos modelos, alguns requerem configurações específicas. O modelo *log-distance* foi configurado com um expoente de perda de percurso de 3,76 e uma distância de referência de 32 m. Para os modelos dependentes da frequência (excluindo o modelo *log-distance*), utilizamos uma frequência de portadora de 1 GHz. Por fim, para os modelos

que consideram as alturas do GW e dos dispositivos finais, utilizamos os valores previamente definidos de 30 e 60 m para o GW (para cada cenário distinto) e 1,4 m para os dispositivos finais. Com todos esses parâmetros configurados, a potência recebida é calculada considerando o uso apenas do simulador NS-3 na etapa (3), conforme ilustrado na Figura 28.

O segundo grupo de canais é representado pelos modelos de canal provenientes do Sionna RT e do WI. Para esse segundo grupo, também avaliamos a potência recebida utilizando o *framework* proposto. Dessa forma, seguindo o *pipeline* apresentado na Figura 28, iniciamos o processo de coleta do ganho de percurso na etapa (1), utilizando cenários 3D com diferentes níveis de detalhe do Sionna, denominados Etoile e Street Canyon [88], com totais de 13058 e 37 faces, respectivamente. A escolha de ambos os cenários se deve à necessidade de avaliar o impacto de objetos 3D, como edifícios, tanto em ambientes com menor quantidade desses elementos (cenário Street Canyon) quanto em áreas mais densas, com maior número de construções (cenário Etoile). Utilizamos o *software* Blender para converter o formato dos arquivos para aquele requerido pelo nosso *framework*. Nesses cenários 3D, empregamos as mesmas topologias de malha de rede utilizadas na avaliação com canais independentes do local para alocar transmissores e receptores (que serão posteriormente interpretados pelo NS-3 como dispositivo final e GW, respectivamente). Nesse sentido, para os cenários Etoile e Street Canyon, utilizamos os padrões de malha dos dispositivos finais 1 e 2, respectivamente.

Para obter as métricas da camada física, especificamente a potência recebida, realizamos seu cálculo para cada posição possível de GW. Os parâmetros utilizados em ambos os simuladores de RT para calcular o mapa de cobertura estão descritos na Tabela 5. Para a frequência de portadora, assumimos um valor *proxy* de 1 GHz, pois os modelos de materiais de rádio nos simuladores de RT não suportam propriedades como condutividade e permissividade para frequências abaixo de 1 GHz. Esses simuladores utilizam os valores de propriedades fornecidos pela recomendação *International Telecommunication Union* (ITU)-R P.2040-3 [89] para materiais como concreto, tijolo, vidro, entre outros. Além disso, como as mesmas propriedades de materiais podem ser aplicadas também a frequências abaixo de 1 GHz [89], essa escolha é aceitável.

Tabela 5: Parâmetros de simulação de RT utilizados nos experimentos com canais específicos do local usando WI e Sionna RT

Parâmetro de simulação	Valor
Frequência de portadora (GHz)	1
Padrão de radiação	Isotrópico
Polarização do Tx	Vertical
Número máximo de interações	6
Forma de onda do Tx	Senoidal
Interações	Difração e reflexão

Realizamos essas simulações considerando o *framework* RT-LoRaWAN em um ambiente configurado, conforme definido na Tabela 6. Nesse ambiente, utilizamos principalmente ferramentas como Sionna RT, WI e NS-3 [90]. Para as simulações com Sionna RT e NS-3, utilizamos um servidor equipado com um Intel® Xeon Silver 4514Y, uma GPU NVIDIA® A2 e 512 GB de RAM. Para as simulações de RT com WI, utilizamos um servidor equipado com um Intel® Core™ i7-10700F @ 2,90 GHz, uma GPU NVIDIA® GeForce 3060 e 64 GB de RAM.

Com essa configuração, obtivemos as métricas de potência recebida para os dois grupos de

canais, que foram então utilizadas como entrada para o modelo de otimização implementado em Pyomo [91] e resolvido com GLPK [92], utilizando o algoritmo de *branch-and-bound*.

Por fim, utilizando os dados coletados de ambos os grupos de canais, propomos quatro tipos de avaliações nos dois cenários. A primeira avalia o impacto dos modelos de canal no posicionamento de GW com um limiar mínimo fixo de potência recebida de $\rho = -90$ dBm. A segunda utiliza a mesma configuração, mas estuda esse impacto ao longo de uma faixa de limiares de potência recebida, variando de -120 a -80 dBm. A terceira avaliação examina a influência do posicionamento de GW sob diferentes modelos de canal em métricas de desempenho de alto nível, como a taxa de pacotes entregues. Por fim, o quarto conjunto de avaliações analisa o custo computacional, sob diferentes modelos de canal, necessário para obter as métricas do problema de posicionamento de GWs.

Tabela 6: Ferramentas utilizadas no ambiente de desenvolvimento para as simulações

Ferramenta	Versão
Sistema operacional	Ubuntu 20.04.6 e Windows 10
Pyomo	6.9.5
NS-3	3.45
Sionna RT	1.2.1
GLPK	4.65
Wireless InSite	3.4.5
Blender	3.6.23

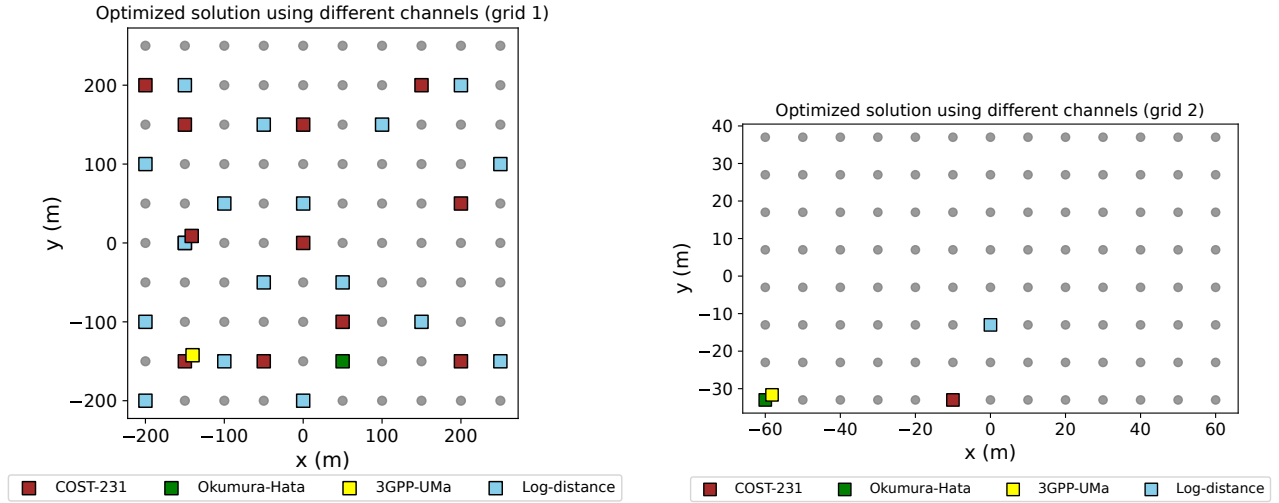
8.6 Resultados

8.6.1 Análise de posicionamento e cobertura

No primeiro conjunto de avaliações, consideramos a análise espacial dos GWs posicionados para os diferentes canais utilizados. Para representar graficamente esse tipo de resultado experimental, focamos em apresentar os resultados gráficos para as configurações com as malhas 1 e 2. Utilizando esses padrões, os GWs sugeridos nas malhas 1 e 2 são apresentados nas Figuras 31a e 31b, respectivamente, em termos da posição e da quantidade de GWs alocados. Nessas figuras, consideramos apenas o modelo de canal independente do local.

Na organização de rede da malha 1, as soluções otimizadas são fortemente influenciadas pelo modelo de canal selecionado. Utilizando o modelo Okumura-Hata, a otimização resulta em uma implantação com 1 GWs, enquanto os modelos 3GPP-UMa e COST-231 resultam em soluções que requerem 1 e 11 GWs, respectivamente. Em contraste com esses modelos, o modelo *log-distance* apresenta um comportamento significativamente diferente, sugerindo 17 GWs. Por fim, como o modelo *log-distance* resulta em um maior número de GWs posicionados, algumas localizações de GW se sobrepõem e são compartilhadas pelas soluções obtidas com os modelos de canal COST-231 e 3GPP-UMa. Para a malha 2, por representar o cenário mais simples, a variabilidade posicional de cada GW é menor do que na malha 1. Nesse caso, as simulações para cada modelo de canal indicam que um único GW é suficiente para atender aos requisitos.

Para os canais *site-specific*, obtivemos as soluções apresentadas nas Figuras 32a e 32b, que mostram a posição de cada GW alocado para os cenários Etoile e Street Canyon, respectivamente, considerando os canais criados com Sionna RT e WI. Os resultados obtidos em Etoile,



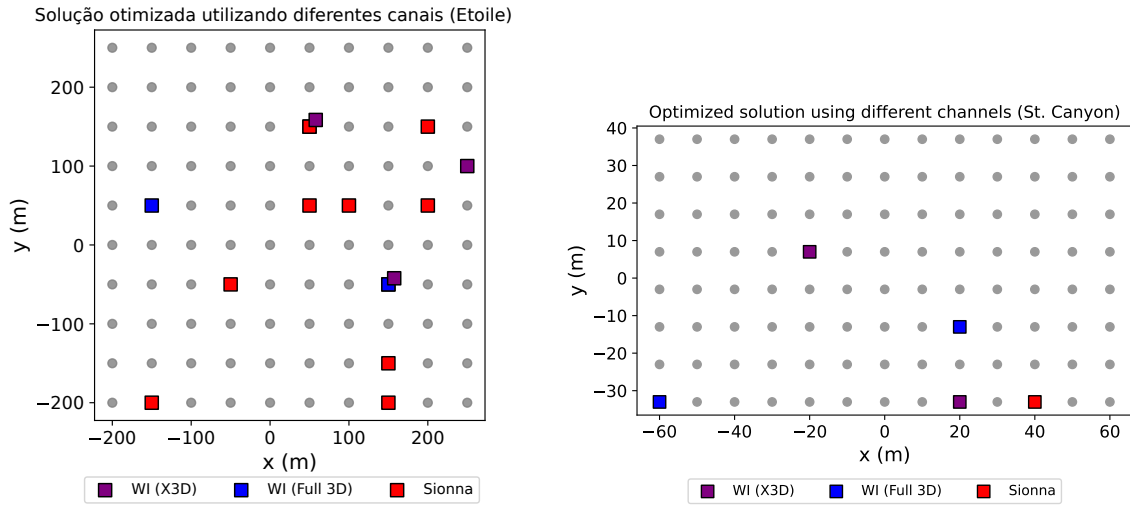
(a) GWs selecionados usando o padrão de malha 1. (b) GWs selecionados usando o padrão de malha 2.

Figura 31: Solução otimizada para o posicionamento de GW utilizando diferentes tipos de canais independentes do local, neste caso: COST-231, Okumura-Hata, 3GPP-UMa e *log-distance*.

utilizando os canais Sionna RT e WI, também produzem uma solução otimizada diferente daquela observada nos demais canais *site-independent*. Utilizando o algoritmo X3D, foram implantados 3 GWs, sendo um GW posicionado na mesma localização sugerida ao utilizar o canal Sionna RT. Em contraste, ao utilizar o algoritmo Full 3D, apenas 2 GWs são sugeridos, e uma de suas posições coincide com aquelas propostas pelo modelo de otimização tanto para os canais X3D Sionna RT quanto WI. Por fim, ao utilizar os canais Sionna, é sugerido um número maior de GWs, evidenciando uma diferença significativa entre os algoritmos empregados nos dois simuladores de RT. Para o cenário Street Canyon, a variabilidade é maior em comparação com os resultados correspondentes dos canais *site-independent*, com as simulações sob canais WI sugerindo 2 GWs para atender aos requisitos.

Em termos gerais, a diferença entre essa quantidade de GWs e as coordenadas de posicionamento, considerando todos os modelos de canal, deve-se à distribuição dos dados de potência recebida coletados. Essa diferença pode ser observada na Figura 33, que apresenta a *Cumulative Distribution Function* (CDF) da potência recebida de cada modelo de canal, considerando as soluções com um limiar fixo de $\rho = -90$ dBm. A partir dessa figura, pode-se observar que, mesmo com 9 GWs implantados sob o canal Sionna RT, a faixa de valores de potência recebida para cada dispositivo final permanece inferior à potência recebida obtida ao utilizar um canal proveniente de WI, com menos GWs implantados no cenário Etoile. Por fim, os resultados numéricos para ambas as malhas de rede, o número de GWs sugeridos, bem como a média e o desvio padrão da potência recebida dos dispositivos finais para cada modelo de canal, são descritos na Tabela 7, também considerando um limiar fixo de potência recebida de $\rho = -90$ dBm.

Uma análise adicional que pode ser realizada com as posições dos GWs determinadas após o processo de otimização é comparar a diferença na potência recebida para cada dispositivo final sob diferentes modelos de canal. Em particular, consideramos a potência recebida sob o modelo WI (X3D) como referência (*ground-truth*) e avaliamos as diferenças de potência recebida para o mesmo dispositivo final quando associado a outros modelos de canal. Assim, a Tabela 8 descreve o Erro Médio Quadrático (EMQ) para cada comparação. A partir desses resultados,



(a) GWs escolhidos no cenário de Etoile. (b) GWs escolhidos no cenário de Street Canyon.

Figura 32: Solução otimizada para o posicionamento de GWs nos cenários Etoile (utilizando o padrão de malha 1) e Street Canyon (utilizando o padrão de malha 2), considerando canais provenientes do Sionna RT e do WI utilizando os algoritmos X3D e Full 3D.

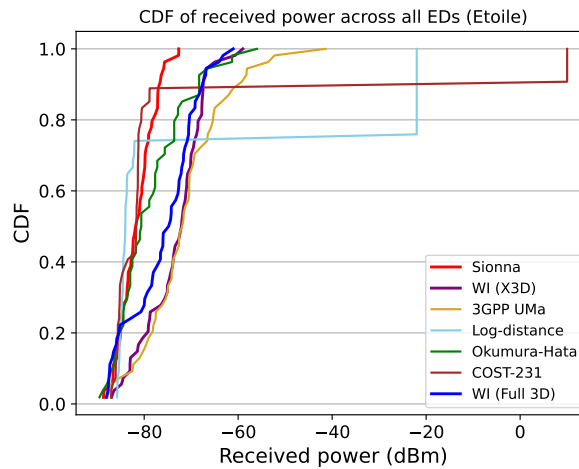


Figura 33: CDF da potência recebida sob diferentes modelos de canal no padrão de malha 1. É possível observar a diferença entre a distribuição de potência recebida proveniente dos canais de RT em comparação com outros modelos *site-independent*.

é possível observar que, em uma área mais densa (cenário Etoile com malha 1), a potência recebida dos dispositivos finais sob o modelo 3GPP-UMa é o canal *site-independent* que mais se aproxima da potência recebida obtida pelo modelo WI X3D. Em contraste, no cenário Street Canyon com malha 2, o modelo *log-distance* é o canal *site-independent* que mais se aproxima da potência recebida dos dispositivos finais proveniente do WI X3D. Dessa forma, ambos os comportamentos são esperados, uma vez que, em cenários de baixa densidade, a potência recebida é majoritariamente influenciada pela distância, o que explica o bom ajuste do modelo *log-distance*. Por outro lado, em ambientes urbanos densos, nos quais bloqueios, sombreamento e transições entre linha de visada e não linha de visada predominam, o modelo 3GPP-UMa fornece uma melhor aproximação devido à sua representação mais detalhada desses efeitos.

Tabela 7: Número de GWs sugeridos, desvio padrão e potência média recebida dos dispositivos finais (em dBm) para cada modelo de canal em diferentes padrões de malha.

Modelo de Canal	Malha 1			Malha 2		
	# GW	Pot. média - disp. final	desvio padrão - disp. final	# GW	Pot. média - disp. final	desvio padrão - disp. final
Okumura-Hata	1	-78.65	7.47	1	-53.62	6.07
COST-231	11	-83.11	2.39	1	-78.14	7.03
<i>log-distance</i>	17	-68.15	27.32	1	-79.85	6.96
3GPP-UMa	1	-71.31	8.94	1	-60.29	6.16
Sionna	9	-81.47	3.76	1	-83.16	1.89
WI (X3D)	3	-73.39	6.23	2	-72.73	6.07
WI (Full 3D)	2	-75.94	7.20	2	-68.93	1.52

Tabela 8: Erro relativo em termos da potência recebida dos dispositivos finais, considerando a potência recebida do modelo WI X3D como referência (*ground-truth*)

Modelo de canal	EMQ (Malha 1)	EMQ (Malha 2)
Okumura-Hata	139.99	477.84
COST-231	888.31	142.41
<i>log-distance</i>	825.51	131.72
3GPP-UMa	113.25	244.30
Sionna	118.12	137.85
WI (Full 3D)	52.99	54.93
WI (X3D)	—	—

8.6.2 Análise paramétrica com diferentes limiares

No segundo conjunto de avaliações, variamos o valor de ρ , sendo possível verificar o modelo de otimização sob uma perspectiva geral em termos do número de GWs sugeridos em todas as organizações de malha. No padrão de malha 2, observamos menor variabilidade em relação ao número de GWs necessários para atender às restrições sob diferentes valores de ρ . Nesse caso, 1 GW é suficiente ao utilizar todas as opções de canais *site-independent*, e 1 ou 2 são suficientes quando um canal *site-specific* é utilizado. Entretanto, ao considerar um cenário mais complexo fornecido pelo padrão de malha 1, obtemos algumas percepções importantes no contexto de planejamento de rede. Nesse sentido, a Figura 34 apresenta o número de GWs sugeridos para cada modelo ao longo de diferentes valores de ρ , considerando um intervalo que varia de -120 a -80 dBm. Nessa figura, também é possível observar o comportamento atípico do modelo de otimização sob os modelos de canal COST-231 e *log-distance*. Esse comportamento inicia em $\rho = -85$ para o modelo *log-distance* e em $\rho = -80$ para o modelo COST-231, com o modelo de otimização propondo pelo menos 17 vezes mais GWs do que modelos como Okumura-Hata, resultando em um total de 54 GWs. Isso indica que, no contexto de planejamento de rede, algumas das soluções sugeridas sob determinados modelos de canal podem ser altamente imprecisas, mesmo apresentando certa similaridade em termos da CDF da potência recebida, como observado na Figura 33. No caso do modelo COST-231, isso resulta em uma solução não realista, o que é explicado pelo fato de que o modelo COST-231 foi originalmente projetado para cenários urbanos de macrocélulas em frequências mais elevadas. Assim, a perda de percurso prevista torna-se excessivamente pessimista para bandas e ambientes LoRaWAN. Combinado com limiares rígidos de sensibilidade do receptor, isso leva a uma fragmentação artificial da cobertura, na qual a otimização degenera para o posicionamento

de um GW por dispositivo final.

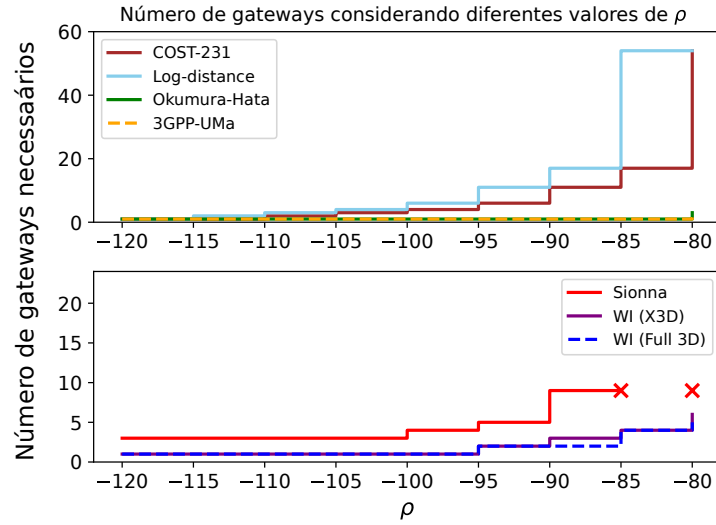


Figura 34: Número de GWs implantados considerando diferentes valores de ρ no padrão de malha 1. Neste gráfico, comparamos o número de GWs sugeridos considerando canais *site-independent* (superior) e canais *site-specific* (inferior). A marca “x” representa uma solução inviável para $\rho = -85$ e $\rho = -80$ dBm.

Além disso, para os canais do Sionna RT, a inviabilidade de encontrar uma solução otimizada para determinados valores de ρ também pode ser explicada pela geometria do cenário no qual a potência recebida é coletada e pelo algoritmo de RT utilizado pelo simulador [93]. Podemos observar que, mesmo próximo ao GW, a cobertura do dispositivo final pode ser atenuada devido ao espalhamento. Isso se torna mais evidente quando visualizamos uma representação 3D do cenário, considerando as 9 posições de GW escolhidas. Em termos gerais, a diferença entre esse número de GWs e as coordenadas de posição, considerando todos os modelos de canal, deve-se à distribuição dos dados de potência recebida coletados. Para os canais obtidos com o Sionna RT, podemos observar que, mesmo em regiões muito próximas ao GW, a cobertura do dispositivo final pode ser significativamente atenuada devido ao espalhamento, como pode ser visualizado na representação 3D do cenário com o mapa de cobertura, conforme ilustrado na Figura 35.

8.6.3 Análise de desempenho de métricas de alto nível

No terceiro conjunto de avaliações, para analisar o impacto do posicionamento dos GWs sob a perspectiva de desempenho em alto nível, reutilizamos o ambiente LoRaWAN do NS-3 introduzido na etapa (3) da Figura 28 e avaliamos a taxa de pacotes entregues com as posições otimizadas dos GWs para um limiar de sensibilidade de $\rho = -90$ dBm. Para os ambientes com malhas 1 e 2, assumimos que cada dispositivo final transmite 1000 e 1040 pacotes, respectivamente, para os GWs, com um SF igual a 7 e um tempo de simulação de 600 segundos para cada canal. Assim, para cada modelo de canal (em ambos os padrões de malha), obtemos a taxa média de pacotes entregues considerando todos os GWs, apresentada na Figura 36, e o desvio padrão descrito na Tabela 9.

A taxa média de pacotes entregues apresentada na Figura 36 destaca o impacto da modelagem de canal sobre métricas de alto nível no planejamento de redes. Para os modelos de

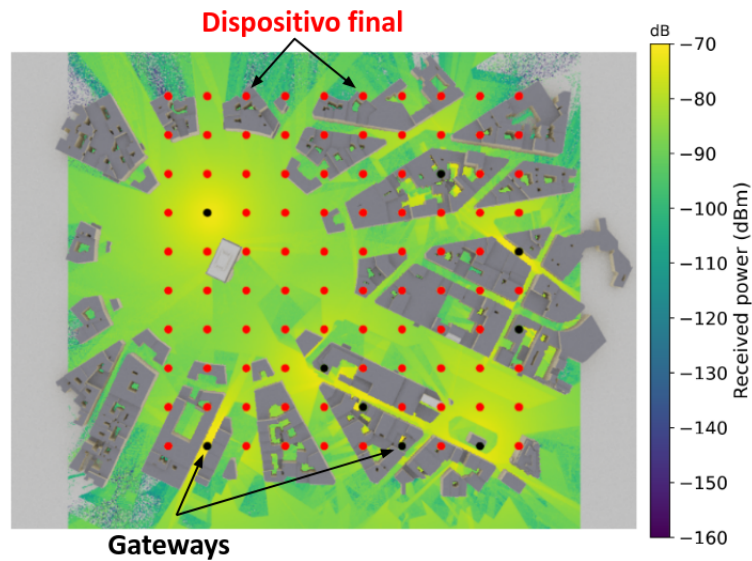


Figura 35: Mapa de cobertura da configuração otimizada considerando canais de RT do Sionna RT no cenário Etoile. Entre todas as posições possíveis representadas por pontos vermelhos, os pontos pretos representam as posições de GW escolhidas.

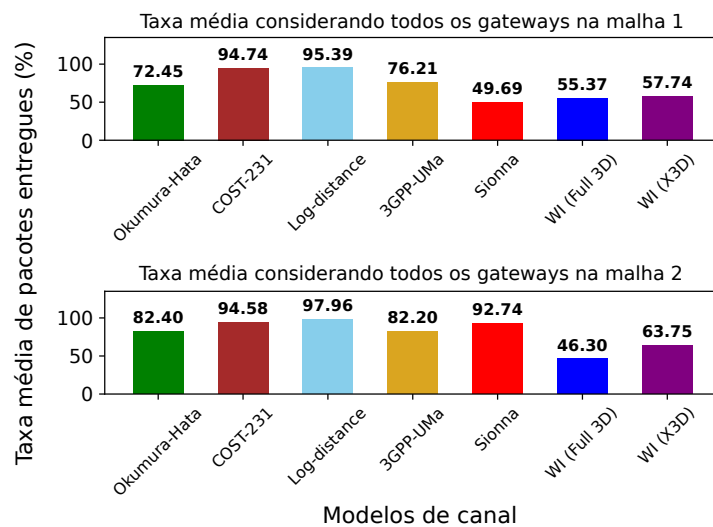


Figura 36: Taxa média de pacotes entregues avaliado considerando canais *site-specific* e *site-independent* em diferentes cenários de rede e padrões de malha.

canal *site-specific* no cenário Etoile, os resultados são pessimistas, uma vez que esses modelos consideram explicitamente os efeitos de espalhamento nas simulações. Isso leva a uma maior atenuação do sinal transmitido do dispositivo final para o GW, resultando em uma taxa média de pacotes entregues inferior a 58%. Em contraste, os modelos de canal *site-independent* produzem resultados mais otimistas para o planejamento de rede, com cada modelo alcançando uma taxa média de pacotes entregues superior a 72%. Para os experimentos utilizando o padrão de malha 2, os resultados otimistas também permanecem quando utilizamos canais *site-independent*. Entretanto, a primeira diferença está relacionada ao uso de canais *site-specific* do Sionna, que, neste caso, apresentou uma taxa média de pacotes entregues semelhante à dos canais *site-independent*. Essa similaridade pode ser atribuída ao nível de detalhamento im-

Tabela 9: Valores médios e de desvio padrão do tempo de simulação (minutos) e da taxa de pacotes entregues (%) considerando todos os GWs escolhidos em diferentes ambientes de rede

Modelo de canal	Malha 1				Malha 2			
	PDR médio	Desv. PDR	Tempo médio de simul.	Desv. tempo de simul.	PDR médio	Desv. PDR	Tempo médio de simul.	Desv. tempo de simul.
Okumura-Hata	72.45	0	0.01	3×10^{-5}	82.40	0	0.01	4×10^{-5}
COST-231	94.74	1.96	0.01	2×10^{-5}	94.58	0	0.01	5×10^{-5}
<i>log-distance</i>	95.39	1.44	0.01	3×10^{-5}	97.96	0	0.01	3×10^{-5}
3GPP-UMa	76.21	0	0.01	4×10^{-5}	82.20	0	0.01	5×10^{-5}
Sionna	49.69	6.25	0.27	0.02	92.74	0	0.17	0.01
WI (Full 3D)	57.74	0.87	65.61	0.06	46.30	13.25	2.69	0.04
WI (X3D)	55.37	0.57	24.95	0.08	63.75	0.24	4.37	0.02

plementado no simulador de RT. Como consequência, em cenários relativamente simples, com menos edifícios e eventos de espalhamento, como no caso Street Canyon, os resultados de ganho de percurso do modelo RT do Sionna se aproximam daqueles obtidos pelos modelos de canal *site-independent*.

Para a taxa média de pacotes entregues em um ambiente com canais advindos do WI, o desempenho permanece semelhante. De forma geral, esses resultados sugerem imediatamente possíveis problemas ocultos no planejamento de redes, uma vez que a escolha de utilizar modelos de canal *site-independent* pode mascarar problemas sérios em ambientes de rede próximos ao mundo real. Dessa forma, o uso de canais mais realistas baseados em RT pode demonstrar que uma rede LoRaWAN pode perder até metade de seus pacotes durante a comunicação. Essa observação reforça a necessidade de realizar simulações de rede que considerem o ambiente específico, a fim de melhorar a fidelidade dos resultados de desempenho nas fases iniciais de implantação da rede, visando adotar otimizações multiobjetivo complementares considerando outros parâmetros da rede, como diferentes SFs, métricas de energia e potência de transmissão.

8.6.4 Análise de custo computacional

Por fim, entre as soluções anteriormente mencionadas, é possível observar um claro *trade-off* entre a fidelidade dos resultados e o custo computacional necessário para obter as métricas utilizadas no modelo de otimização. Nesse sentido, a Figura 37 e a Tabela 9 apresentam o tempo médio de simulação ao longo de 10 realizações, em minutos, necessário para calcular as métricas de nível físico utilizadas no modelo de otimização, empregando ambos os padrões de malha e cenários 3D quando um canal *site-specific* é utilizado. Essa figura revela duas discrepâncias principais. Primeiro, existe uma diferença muito maior no tempo de simulação entre os modelos *site-independent* e *site-specific*, sendo que os primeiros requerem menos de um segundo para obter as métricas de nível físico. A segunda discrepância é a diferença marcante entre os dois simuladores de RT: o tempo de simulação utilizando o cenário Etoile no WI é pelo menos 87 vezes maior do que o do RT do Sionna. Essa diferença pode se tornar ainda maior caso o algoritmo Full 3D seja utilizado no WI. Nesse contexto, todas essas simulações *site-specific* baseadas em RT ocorreram em cenários 3D com menos de 20000 faces. Mesmo com essa quantidade de polígonos representando os objetos do cenário, simuladores de RT de alto detalhamento, como o WI, ainda apresentam um elevado custo computacional para produzir métricas precisas, como ganho de percurso. Esse *trade-off* torna-se mais evidente ao analisar um gráfico de dispersão, com o tempo de simulação no eixo y e os valores de EMQ da Tabela 8 no eixo x , conforme ilustrado na Figura 38.

A partir desses resultados, podemos observar que alcançar o menor EMQ com canais WI

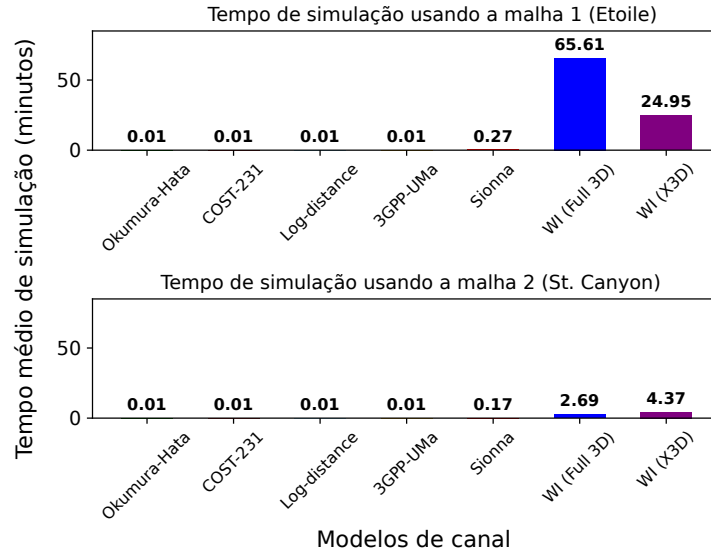


Figura 37: Tempo de simulação, em minutos, para obter métricas da camada física em diferentes modelos de canal e cenários.

avaliados utilizando o algoritmo Full 3D requer um elevado tempo de simulação. Além disso, para alguns canais *site-specific* e *site-independent*, o compromisso entre tempo de simulação e EMQ é comparável. Por exemplo, em ambientes com malha 1 e Etoile, o *trade-off* para Okumura-Hata, 3GPP-UMa e Sionna é semelhante. Da mesma forma, em ambientes com malha 2 e Street Canyon, essa similaridade é observada para *log-distance*, COST-231 e Sionna. Portanto, este resultado final reforça a sensibilidade do planejamento de rede nesse contexto. Na primeira avaliação, observamos que, mesmo com CDF de potência recebida pelos dispositivos finais semelhantes, elas se originavam de diferentes organizações espaciais dos GWs. Agora, vemos que os diferentes resultados das taxas médias de pacotes entregues analisados na subseção anterior decorrem de modelos de canal com *trade-offs* semelhantes entre tempo de simulação e EMQ.

8.7 Conclusões

Neste trabalho, desenvolvemos um *framework* que integra ferramentas de RT, como Sionna e WI, ao simulador de rede de eventos discretos NS-3 para apoiar o planejamento de rede no problema de posicionamento de GW em diferentes cenários LoRaWAN. Utilizando esse *framework* integrado, avaliamos como diferentes modelos de canal influenciam os resultados do modelo de otimização. Nesse sentido, mostramos que a qualidade da solução de planejamento de rede depende fortemente do tipo de canal quando uma abordagem independente do cenário é utilizada, podendo também variar com a escolha do simulador de RT quando uma abordagem específica ao cenário é adotada. Essa escolha influencia o número de GWs que a solução sugere implantar e a posição de cada um deles. Numericamente, para modelos independentes do cenário, como o *log-distance*, essa discrepância pode ser altamente imprecisa, alcançando valores até 17 vezes maiores do que aqueles sugeridos por modelos como Okumura-Hata e 3GPP-UMa. Para canais específicos ao cenário, certas restrições de potência recebida relacionadas à cobertura não admitem uma solução viável (por exemplo, ao utilizar Sionna RT). No entanto, para os mesmos valores de parâmetros, soluções viáveis existem quando se utilizam canais

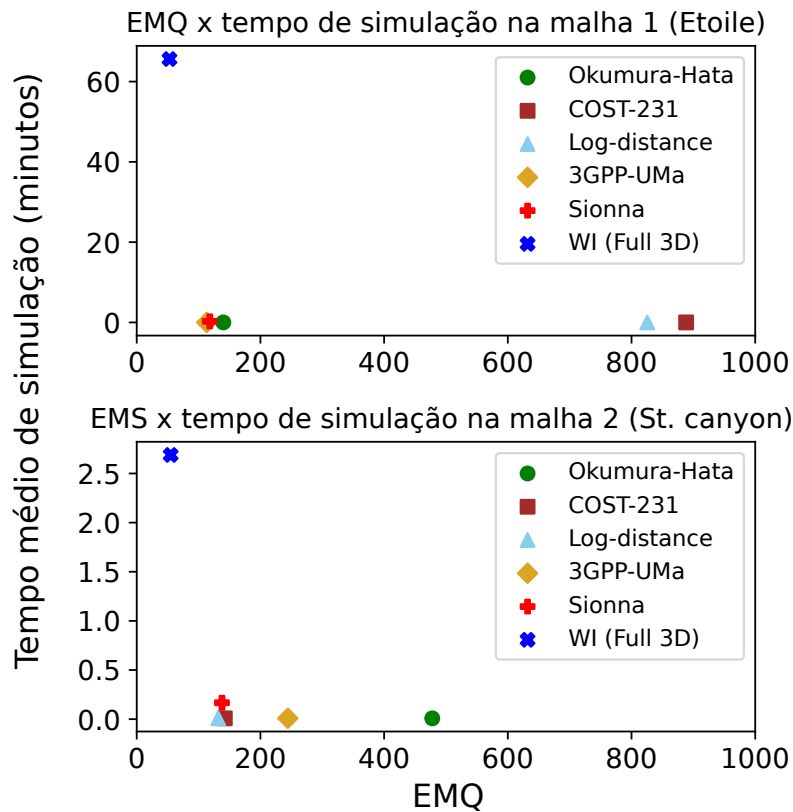


Figura 38: *Trade-off* entre tempo computacional e EMQ, considerando a potência recebida pelos dispositivos finais do WI X3D como *ground truth*.

WI, também variando os algoritmos de RT. Esse comportamento destaca a importância e a diferença entre dois simuladores de RT e seu impacto no planejamento de rede.

Além disso, mostramos o impacto do posicionamento dos GWs, considerando métricas de alto nível, como a taxa de pacotes entregues em diferentes modelos de canal. Nesse sentido, os resultados apontam na mesma direção quando consideramos apenas o impacto na potência recebida: um impacto decisivo ao adotar um modelo de canal específico ao cenário em vez de um independente, sendo que os primeiros produzem uma taxa de pacotes entregues mais pessimista em comparação com os últimos. Por fim, mostramos que todos esses resultados envolvem um *trade-off* entre o custo computacional e a precisão da solução necessária para obtê-los.

9 Conclusão

Este relatório consolidou os principais avanços obtidos na Atividade 3.3 da Fase III do projeto Brasil 6G, com foco no desenvolvimento, implementação e avaliação de soluções voltadas a aplicações IoT, sistemas ciberfísicos, computação em borda, comunicação semântica, monitoramento inteligente e planejamento de redes sem fio. Os trabalhos apresentados refletem a natureza multidisciplinar do ecossistema 6G, no qual conectividade, processamento distribuído, IA, eficiência energética, confiabilidade e sensibilidade ao contexto tornam-se elementos fundamentais para viabilizar aplicações críticas, imersivas e de larga escala.

No contexto de aplicações imersivas, destaca-se a evolução do Metaverso Inatel, com a ampliação de ambientes virtuais e a integração de gêmeos digitais, consolidando-o como uma plataforma para demonstração e integração das pesquisas desenvolvidas no projeto. Complementarmente, foram desenvolvidos ambientes de gêmeos digitais para uma incubadora neonatal virtual e para estações meteorológicas, integrando sensoriamento, detecção de inconsistências e mecanismos de atuação automatizada. Além disso, a extensão eMR-Leo permitiu caracterizar tarefas de MAR em arquiteturas de borda, revelando perfis distintos de demanda computacional e de tráfego que reforçam a necessidade de modelos empíricos realistas. Por fim, o arcabouço FLEXA demonstrou o potencial da comunicação semântica para a transmissão de vídeo via WebRTC, evidenciando os compromissos entre qualidade visual, custo computacional e eficiência da transmissão.

No âmbito das aplicações de monitoramento ambiental e de redes de sensoriamento, foi desenvolvida e validada, em ambiente de laboratório, uma coleira IoT nacional baseada em LoRa para o rastreamento de fauna silvestre. Além disso, modelos de visão computacional foram avaliados para a detecção de incêndios florestais em tempo real. Por fim, avanços no planejamento de redes LoRaWAN, por meio da integração entre modelos de traçado de raios e técnicas de otimização para o posicionamento de GWs, demonstraram o impacto significativo da escolha do modelo de canal sobre a cobertura da rede e o custo de implantação.

De modo geral, os resultados apresentados evidenciam a viabilidade de soluções orientadas à IoT para as futuras redes 6G, integrando comunicação sem fio, computação distribuída, IA, sensoriamento, simulação e prototipagem experimental. As contribuições obtidas nesta etapa estabelecem bases metodológicas e tecnológicas para a continuidade do projeto, abrangendo desde a caracterização de cargas de trabalho e a avaliação de arquiteturas até o desenvolvimento de protótipos e ferramentas de planejamento. Como próximos passos, destacam-se a ampliação das validações em campo e em cenários reais, a integração entre plataformas físicas e digitais e o aprimoramento dos mecanismos de interoperabilidade, segurança e escalabilidade. Adicionalmente, pretende-se evoluir as soluções propostas para ambientes cada vez mais próximos das condições operacionais esperadas nas futuras redes 6G.

Referências

- [1] P. B. 6G, “Projeto Brasil 6G - Fase II - Relatório Técnico: Aplicações para Redes 6G e Integração (Atividades 6.1 e 6.2),” RNP, Tech. Rep., 2024.
- [2] R. Chen, M. Liu, Y. Hui, N. Cheng, and J. Li, “Reconfigurable intelligent surfaces for 6g iot wireless positioning: A contemporary survey,” *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23 570–23 582, 2022.
- [3] M. Rahim, W. Lalouani, E. Toubal, and L. Emokpae, “A digital twin-based platform for medical cyber-physical systems,” *IEEE Access*, vol. 12, pp. 174 591–174 607, 2024.
- [4] H. Kaschel, S. Cordero, P. Adasme, and C. Ahumada, “Study on the challenges and future trends of iot and 5g/6g networks in smart agriculture 5.0,” in *2025 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 2025, pp. 1–6.
- [5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf
- [6] F. Rahat, M. S. Hossain, A. Ramanathan, S. Jha, H. Zheng, and R. Ewetz, “Attr-rag: Attribution-guided retrieval-augmented generation for scientific experiment design,” in *2025 International Conference on Machine Learning and Applications (ICMLA)*, 2025, pp. 424–431.
- [7] OASIS, “MQTT Version 5.0,” OASIS Standard, Tech. Rep., 2019.
- [8] M. Escribà-Gelonch, S. Liang, P. van Schalkwyk, I. Fisk, N. V. D. Long, and V. Hessel, “Digital twins in agriculture: orchestration and applications,” *Journal of agricultural and food chemistry*, vol. 72, no. 19, pp. 10 737–10 752, 2024.
- [9] J. P. Boomgard-Zagrodnik and D. J. Brown, “Machine learning imputation of missing Mesonet temperature observations,” *Computers and Electronics in Agriculture*, vol. 192, p. 106580, 2022, doi: <https://doi.org/10.1016/j.compag.2021.106580>.
- [10] N. P. Kuruvatti, M. A. Habibi, S. Partani, B. Han, A. Fellan, and H. D. Schotten, “Empowering 6G Communication Systems With Digital Twin Technology: A Comprehensive Survey,” *IEEE Access*, vol. 10, pp. 112 158–112 186, 2022, doi: <https://doi.org/10.1109/ACCESS.2022.3215493>.
- [11] L. Weiwei, L. Xiaohua, T. Zuoyang, X. Xiaohua, and X. Yong, “Fault Analysis and Maintenance of DZZ Series of Automatic Weather Stations,” *Meteorological and Environmental Research; Cranston*, vol. 9, no. 3, pp. 35–37, 2018, doi: <https://doi.org/10.19547/j.issn2152-3940.2018.03.008>.
- [12] M. Grieves and J. Vickers, *Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems*. Springer International Publishing, 2017, pp. 85–113".

- [13] B. R. Barricelli, E. Casiraghi, and D. Fogli, “A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications,” *IEEE Access*, vol. 7, pp. 167 653–167 671, 2019, doi: <https://doi.org/10.1109/ACCESS.2019.2953499>.
- [14] S. Mihai, M. Yaqoob, D. V. Hung, W. Davis, P. Towakel, M. Raza, M. Karamanoglu, B. Barn, D. Shetve, R. V. Prasad, H. Venkataraman, R. Trestian, and H. X. Nguyen, “Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects,” *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2255–2291, 2022.
- [15] J. Cho, C. Kim, K. J. Lim, J. Kim, B. Ji, and J. Yeon, “Web-based agricultural infrastructure digital twin system integrated with GIS and BIM concepts,” *Computers and Electronics in Agriculture*, vol. 215, p. 108441, 2023.
- [16] T. Jin and X. Han, “Robotic arms in precision agriculture: A comprehensive review of the technologies, applications, challenges, and future prospects,” *Computers and Electronics in Agriculture*, vol. 221, p. 108938, 2024.
- [17] T. Yang, M. R. Palattella, C. Chitu, P. Claudon, and M. Boukhebouze, “Digital Twin as a Service for Sustainable Agriculture in Africa: A case study from the ESA DT4CMI project,” in *2024 IEEE International Humanitarian Technologies Conference (IHTC)*, 2024, pp. 1–6.
- [18] T. Ahmed and M. Hasan, “Weather-Driven Agricultural Decision-Making Under Imperfect Conditions,” in *Proceedings of the 33rd ACM International Conference on Advances in Geographic Information Systems*, ser. SIGSPATIAL ’25. ACM, 2025, doi: <https://doi.org/10.1145/3748636.3764158>.
- [19] X. Wang *et al.*, “Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models,” *ACM Comput. Surv.*, vol. 57, no. 9, Apr. 2025, doi: <https://doi.org/10.1145/3724420>.
- [20] Y. Siriwardhana *et al.*, “A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects,” *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1160–1192, 2021, doi: <https://doi.org/10.1109/COMST.2021.3061981>.
- [21] Z. Chen *et al.*, “An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance,” in *Proc. of the Second ACM/IEEE Symposium on Edge Computing*. Association for Computing Machinery, 2017, doi: <https://doi.org/10.1145/3132211.3134458>.
- [22] N. Pereira *et al.*, “ARENA: The Augmented Reality Edge Networking Architecture,” in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2021, pp. 479–488, doi: <https://doi.org/10.1109/ISMAR52148.2021.00065>.
- [23] W. Zhang *et al.*, “EdgeXAR: A 6-DoF Camera Multi-Target Interaction Framework for MAR with User-Friendly Latency Compensation,” *Proc. ACM Hum.-Comput. Interact.*, vol. 6, no. EICS, 2022, doi: <https://doi.org/10.1145/3532202>.

- [24] N. Hammad *et al.*, “V-Light: Leveraging Edge Computing For The Design of Mobile Augmented Reality Games,” in *Proc. of the 18th International Conference on the Foundations of Digital Games*. Association for Computing Machinery, 2023, doi: <https://doi.org/10.1145/3582437.3582456>.
- [25] G. Espindola *et al.*, “Demonstrating the Advantages of Computational Offloading of XR Services via WebAssembly,” in *2025 IEEE Network Operations and Management Symposium*, 2025, pp. 1–3, doi: <https://doi.org/10.1109/NOMS57970.2025.11073626>.
- [26] K. Toczé *et al.*, “Characterization and modeling of an edge computing mixed reality workload,” *Journal of Cloud Computing*, vol. 9, no. 46, 2020, doi: <https://doi.org/10.1186/s13677-020-00190-x>.
- [27] D. G. Morín *et al.*, “An eXtended Reality Offloading IP Traffic Dataset and Models,” *IEEE Transactions on Mobile Computing*, vol. 23, no. 6, pp. 6820–6834, 2024, doi: <https://doi.org/10.1109/TMC.2023.3326893>.
- [28] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017, doi: <https://doi.org/10.1109/TRO.2017.2705103>.
- [29] J. Redmon *et al.*, “You only look once: Unified, real-time object detection,” in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788, doi: <https://doi.org/10.1109/CVPR.2016.91>.
- [30] F. Muzzini *et al.*, “Brief Announcement: Optimized GPU-accelerated Feature Extraction for ORB-SLAM Systems,” in *Proc. of the 35th ACM Symposium on Parallelism in Algorithms and Architectures*. Association for Computing Machinery, 2023, doi: <https://doi.org/10.1145/3558481.3591310>.
- [31] N. Blum, S. Lachapelle, and H. Alvestrand, “WebRTC: Real-time communication for the open web platform,” *Communications of the ACM*, vol. 64, no. 8, pp. 50–54, 2021, doi: <https://doi.org/10.1145/3454122.3457587>.
- [32] X. Luo, H.-H. Chen, and Q. Guo, “Semantic communications: Overview, open issues, and future research directions,” *IEEE Wireless Communications*, vol. 29, no. 1, pp. 210–219, 2022, doi: <https://doi.org/10.1109/MWC.101.2100269>.
- [33] L. Xia, Y. Sun, C. Liang, D. Feng, R. Cheng, Y. Yang, and M. A. Imran, “WiserVR: Semantic communication enabled wireless virtual reality delivery,” *IEEE Wireless Communications*, vol. 30, no. 2, pp. 32–39, 2023, doi: <https://doi.org/10.1109/MWC.004.2200393>.
- [34] Y. Wen, Z. Zhang, J. Sun, J. Li, C. S. Chen, and G. Niu, “SAW: Semantic-Aware WebRTC Transmission Using Diffusion-Based Scalable Video Coding,” *IEEE Internet of Things Journal*, vol. 12, no. 5, pp. 5346–5359, 2025, doi: <https://doi.org/10.1109/JIOT.2024.3486725>.
- [35] T. Li, V. Sivaraman, P. Karimi, L. Fan, M. Alizadeh, and D. Katabi, “Reparo: Loss-Resilient Generative Codec for Video Conferencing,” 2024. [Online]. Available: <https://arxiv.org/abs/2305.14135>

- [36] Y. Cheng, Z. Zhang, H. Li, A. Arapin, Y. Zhang, Q. Zhang, Y. Liu, K. Du, X. Zhang, F. Y. Yan, A. Mazumdar, N. Feamster, and J. Jiang, “GRACE: loss-resilient real-time video through neural codecs,” in *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI’24. USA: USENIX Association, 2024, pp. 509–531.
- [37] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*. USA: IEEE, 2017, pp. 4681–4690, doi: <https://doi.org/10.1109/CVPR.2017.19>.
- [38] WebRTC, “Webrtc,” <https://webrtc.org/>, 2025, accessed: 2025-08-11.
- [39] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RFC3550: RTP: A Transport Protocol for Real-Time Applications,” Blue Coat Systems Inc., USA, Tech. Rep., 2003, doi: <https://doi.org/10.17487/RFC3550>.
- [40] S. Imambi, K. B. Prakash, and G. R. Kanagachidambaresan, *PyTorch*. Cham: Springer International Publishing, 2021, pp. 87–104, doi: https://doi.org/10.1007/978-3-030-57077-4_10. [Online]. Available: https://doi.org/10.1007/978-3-030-57077-4_10
- [41] aiortc, “aiortc,” <https://github.com/aiortc/aiortc>, 2018, accessed: 2025-08-11.
- [42] T. Karras, S. Laine, and T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. USA: IEEE, 2019, pp. 4401–4410, doi: <https://doi.org/10.1109/CVPR.2019.00453>.
- [43] R. Kays and M. Wikelski, “The Internet of Animals: what it is, what it could be,” *Trends in Ecology & Evolution*, vol. 38, no. 9, pp. 859–869, Sep. 2023, doi: <https://doi.org/10.1016/j.tree.2023.04.007>.
- [44] A. M. Davison, K. de Koning, F. Taubert, and J.-K. Schakel, “Automated near real-time monitoring in ecology: Status quo and ways forward,” *Ecological Informatics*, vol. 89, p. 103157, 2025, doi: <https://doi.org/10.1016/j.ecoinf.2025.103157>.
- [45] D. Zeuss *et al.*, “Nature 4.0: A networked sensor system for integrated biodiversity monitoring,” *Global Change Biology*, vol. 30, no. 1, p. e17056, Jan. 2024, doi: <https://doi.org/10.1111/gcb.17056>.
- [46] J. Kamminga, E. Ayele, N. Meratnia, and P. Havinga, “Poaching Detection Technologies—A Survey,” *Sensors*, vol. 18, no. 5, p. 1474, 2018, doi: <https://doi.org/10.3390/s18051474>.
- [47] Max Planck Society, “GPS transmitters can protect animals from poaching,” <https://www.mpg.de/9356217/gps-animal-tracking-species-protection>, Aug. 2015, accessed: 2026-02-24.
- [48] D. Thornton, L. C. Zeller, E. Rondinini, R. Boitani, K. A. Crooks, A. Burdett, A. Rabinowitz, and M. Quigley, “Assessing the umbrella value of a range-wide conservation network for jaguars (*Panthera onca*),” *Ecological Applications*, vol. 26, no. 4, pp. 1112–1124, 2016, doi: <https://doi.org/10.1890/15-0602>.

- [49] L. Cullen Jr., J. C. Stanton, F. S. Lima, A. Uezu, R. Akçakaya, and D. S. Maehr, “Implications of Fine-Grained Habitat Fragmentation and Road Mortality for Jaguar Conservation in the Atlantic Forest, Brazil,” *PLOS ONE*, vol. 11, no. 12, p. e0167372, Dec. 2016, doi: <https://doi.org/10.1371/journal.pone.0167372>.
- [50] C. Rodríguez-Soto, P. Monroy-Vilchis, M. Maiorano, R. Boitani, L. Faller Menéndez, M. A. Briones, A. Núñez, G. Rosas-Rosas, M. Castañeda, J. Falcucci, and G. Ceballos, “Corridors for jaguar (*Panthera onca*) in Mexico: Conservation strategies,” *Journal for Nature Conservation*, vol. 21, no. 6, pp. 438–443, Dec. 2013, doi: <https://doi.org/10.1016/j.jnc.2013.07.002>.
- [51] Onçafari, “Onçafari Yearly Report 2019,” 2020, accessed: 2026-02-25. [Online]. Available: https://oncafari.org/wp-content/uploads/2020/10/Oncafari_Yearly_Report_2019_us.pdf
- [52] A. d. N. Carvalho, Ø. Wiig, G. L. d. Oliveira, H. C. Oliveira, L. Sena, R. G. Morato, K. M. P. M. B. d. Ferraz, and A. C. Mendes-Oliveira, “How large felids of South America are influenced by environmental and anthropogenic variables in the most degraded portion of the Amazon,” *Austral Ecology*, vol. 49, no. 10, 2024, doi: <https://doi.org/10.1111/aec.70002>.
- [53] H.-I. Kim and K.-D. Park, “Satellite Positioning Accuracy Improvement in Urban Canyons Through a New Weight Model Utilizing GPS Signal Strength Variability,” *Sensors*, vol. 25, p. 4678, 2025, doi: <https://doi.org/10.3390/s25154678>.
- [54] H. Ruan, P. Sun, Y. Dong, H. Tahaei, and Z. Fang, “An overview of LoRa localization technologies,” *Computers, Materials and Continua*, vol. 82, no. 2, pp. 1645–1680, 2025, doi: <https://doi.org/10.32604/cmc.2024.059746>.
- [55] ASAIR, “DHT11 Temperature and Humidity Sensor Module,” 2026, página oficial do produto. Acesso em: 9 abr. 2026. [Online]. Available: <https://www.aosong.com/en/Products/info.aspx?itemid=2257&lcid=139>
- [56] ROHM Semiconductor, “Digital 16-bit Serial Output Type Ambient Light Sensor ICs,” 2026, página oficial da família de sensores de luminosidade. Acesso em: 9 abr. 2026. [Online]. Available: <https://www.rohm.com/products/sensors-mems/ambient-light-sensor-ics/digital-16bit-serial-output>
- [57] TDK InvenSense, “MPU-6000/MPU-6050 Product Specification,” 2026, datasheet oficial. Acesso em: 9 abr. 2026. [Online]. Available: https://product.tdk.com/system/files/dam/doc/product/sensor/motion-inertial/imu/data_sheet/mpu-6000-datasheet1.pdf
- [58] Analog Devices, “MAX17043/MAX17044 Fuel Gauge with ModelGauge,” 2026, página oficial do produto e datasheet. Acesso em: 9 abr. 2026. [Online]. Available: <https://www.analog.com/en/products/max17043.html>
- [59] J. Verde and J. L. Zêzere, “Assessment and validation of wildfire susceptibility and hazard in Portugal,” *Natural Hazards and Earth System Sciences*, vol. 10, no. 3, pp. 485–497, 2010, doi: <https://doi.org/10.5194/nhess-10-485-2010>.

- [60] M. Flannigan, A. S. Cantin, W. J. De Groot, M. Wotton, A. Newbery, and L. M. Gowman, "Global wildland fire season severity in the 21st century," *Forest Ecology and Management*, vol. 294, pp. 54–61, 2013, doi: <https://doi.org/10.1016/j.foreco.2012.10.022>.
- [61] J. F. Santos, R. V. Soares, and A. C. Batista, "Perfil dos incêndios florestais no brasil em áreas protegidas no período de 1998 a 2002," *Floresta*, vol. 36, no. 1, pp. 93–100, 2006.
- [62] W. J. Franca Rocha, R. N. Vasconcelos, S. G. Duverger, D. P. Costa, N. A. Santos, R. O. Franca Rocha, M. M. de Santana, A. A. Alencar, V. L. Arruda, W. V. d. Silva *et al.*, "Mapping Burned Area in the Caatinga Biome: Employing Deep Learning Techniques," *Fire*, vol. 7, no. 12, p. 437, 2024, doi: <https://doi.org/10.3390/fire7120437>.
- [63] M. M. Veras, "Mudanças climáticas e incêndios florestais: implicações sobre a saúde," *Ciência e Cultura*, vol. 76, no. 3, pp. 01–07, 2024.
- [64] W. G. Lima, A. V. Lopes, C. M. Cardoso, J. P. Araújo, M. C. Neto, M. E. Tostes, A. A. Nascimento, M. Rodriguez, and F. J. Barros, "LoRa Technology Propagation Models for IoT Network Planning in the Amazon Regions," *Sensors*, vol. 24, no. 5, p. 1621, 2024.
- [65] A. Pagano, D. Garlisi, F. Giuliano, T. Cattai, and F. Cuomo, "Application-Aware LoRaWAN Gateway Placement in Massive IoT Water Distribution Networks," in *2024 IEEE 10th World Forum on Internet of Things (WF-IoT)*. IEEE, 2024, pp. 475–480.
- [66] F. Loh, N. Mehling, S. Geißler, and T. Hoßfeld, "Graph-Based Gateway Placement for Better Performance in LoRaWAN Deployments," in *2022 20th Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2022, pp. 190–199, doi: <https://doi.org/10.1109/MedComNet55087.2022.9810404>.
- [67] M. Said, M. Jabberi, A. Hakiri, H. K. Ben Ayed, and N. Mellouli, "Optimal LoRaWAN Gateway Placement for Efficient Large-Scale IoT Deployments," in *2025 International Wireless Communications and Mobile Computing (IWCMC)*, 2025, pp. 1210–1215, doi: <https://doi.org/10.1109/IWCMC65282.2025.11059464>.
- [68] N. Matni, J. Moraes, H. Oliveira, D. Rosário, and E. Cerqueira, "LoRaWAN Gateway Placement Model for Dynamic Internet of Things Scenarios," *Sensors*, vol. 20, no. 15, 2020, doi: <https://doi.org/10.3390/s20154336>.
- [69] W. S. Costa, W. G. dos Santos, H. A. Camporez, M. J. Faber, J. A. Silva, M. E. Segatto, and H. R. Rocha, "Planning and Resource Allocation of a Hybrid IoT Network Using Artificial Intelligence," *Internet of Things*, vol. 26, p. 101225, 2024, doi: <https://doi.org/10.1016/j.iot.2024.101225>.
- [70] W. T. Pires-Jr, D. C. da Silva, R. S. Silva, L. d. L. Pinto, A. Oliveira-Jr, M. J. Abdel-Rahman, and K. V. Cardoso, "QoS-aware Optimal Deployment of LoRa Gateways in UAV-enabled LoRaWANs," in *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*. SBC, 2024, pp. 574–587.
- [71] F. Loh, D. Bau, J. Zink, A. Wolff, and T. Hoßfeld, "Robust Gateway Placement for Scalable LoRaWAN," in *2021 13th IFIP Wireless and Mobile Networking Conference (WMNC)*, 2021, pp. 71–78, doi: <https://doi.org/10.23919/WMNC53478.2021.9618988>.

- [72] Lora Alliance, “LoRaWAN™ 1.1 Specification,” *LoRa Alliance*, vol. 11, pp. 2018–04, 2017.
- [73] J. C. da Silva, D. d. L. Flor, V. A. de Sousa Junior, N. S. Bezerra, and A. A. de Medeiros, “A Survey of LoRaWAN Simulation Tools in ns-3,” *Journal of Communication and Information Systems*, vol. 36, no. 1, pp. 17–30, 2021.
- [74] F. Loh, C. Baur, S. Geißler, H. ElBakoury, and T. Hoßfeld, “Collision and Energy Efficiency Assessment of LoRaWANs with Cluster-Based Gateway Placement,” in *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2023, pp. 391–396, doi: <https://doi.org/10.1109/ICCWorkshops57953.2023.10283556>.
- [75] H. A. O. Cruz, S. C. B. Ferreira, J. P. L. Araújo, F. J. B. Barros, F. S. Farias, M. C. A. Neto, M. E. L. Tostes, A. A. Nascimento, and G. P. S. Cavalcante, “Methodology for LoRa Gateway Placement Based on Bio-Inspired Algorithms for a Smart Campus in Wooded Area,” *Sensors*, vol. 22, no. 17, 2022, doi: <https://doi.org/10.3390/s22176492>.
- [76] 3rd Generation Partnership Project (3GPP), “Study on Channel Model for Frequencies from 0.5 to 100 GHz,” 3GPP, Technical Report TR 38.901, 2022, version 17.0.0, Section 7.6, pp. 55–60.
- [77] F. P. Correia, S. R. d. Silva, F. B. S. d. Carvalho, M. S. d. Alencar, K. D. R. Assis, and R. M. Bacurau, “LoRaWAN Gateway Placement in Smart Agriculture: An Analysis of Clustering Algorithms and Performance Metrics,” *Energies*, vol. 16, no. 5, 2023, doi: <https://doi.org/10.3390/en16052356>.
- [78] A. Ruz-Nieto, E. Egea-Lopez, J.-M. Molina-Garcia-Pardo, and J. Santa, “A 3D Simulation Framework With Ray-tracing Propagation for LoRaWAN Communication,” *Internet of Things*, vol. 24, p. 100964, 2023.
- [79] J. Borges, F. Bastos, I. Correa, P. Batista, and A. Klautau, “CAVIAR: Co-Simulation of 6G Communications, 3-D Scenarios, and AI for Digital Twins,” *IEEE Internet of Things Journal*, vol. 11, no. 19, pp. 31 287–31 300, 2024, doi: <https://doi.org/10.1109/JIOT.2024.3418675>.
- [80] C. Modesto, L. Mozart, P. Batista, A. Cavalcante, and A. Klautau, “Accelerating Ray Tracing-Based Wireless Channels Generation for Real-Time Network Digital Twins,” *IEEE Open Journal of the Communications Society*, vol. 6, pp. 5464–5478, 2025, doi: <https://doi.org/10.1109/OJCOMS.2025.3583202>.
- [81] 5G-LENA, “5G-LENA: The 5G NR Module for the ns-3 Simulator,” <https://5g-lena.cttc.es/>, [Accessed 19-03-2026].
- [82] J. Albuquerque, G. Gonçalves, and A. Klautau, “Uso de Traçado de Raio com Nvidia Sionna para Simulação Fim-a-Fim via NS-3,” in *XLII Brazilian Symposium on Telecommunications and Signal Processing*, 2024, pp. 1–2, doi: <https://doi.org/10.14209/sbrt.2024.1571036838>. [Online]. Available: <http://dx.doi.org/10.14209/sbrt.2024.1571036838>
- [83] A. Zubow, S. Rösler, and F. Dressler, “Ns3Sionna: Realistic Wireless Network Simulation with Ray Tracing in ns-3,” in *2026 International Conference on Computing, Networking and Communications (ICNC)*, 2026, pp. 1–7, doi: <https://doi.org/10.1109/ICNC68183.2026.11416882>.

- [84] R. Pegurri, F. Linsalata, E. Moro, J. Hoydis, and U. Spagnolini, “Toward Digital Network Twins: Integrating Sionna RT in ns-3 for 6G Multi-Rat Networks Simulations,” in *IEEE INFOCOM 2025 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2025, pp. 1–6, doi: <https://doi.org/10.1109/INFOCOMWKSHPS65812.2025.11152946>.
- [85] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang, “Mitsuba 3 renderer,” 2022, <https://mitsuba-renderer.org>.
- [86] B. Foundation, “Blender - The Free and Open Source 3D Creation Software — blender.org — blender.org,” <https://www.blender.org/>, 2026, [Accessed 30-03-2026].
- [87] R. W. Heath, Jr and A. Lozano, *Foundations of MIMO Communication*. Cambridge, England: Cambridge University Press, Dec. 2018, pp. 139–140.
- [88] NVIDIA corporation, “Ray Tracing - Sionna 1.2.1 documentation — nvlabs.github.io,” <https://nvlabs.github.io/sionna/>, 2025, [Accessed 01-28-2026].
- [89] International Telecommunication Union (ITU), “P.2040-3: Effects of Building Materials and Structures on Radiowave Propagation Above About 100 MHz ,” International Telecommunication Union (ITU), Tech. Rep., 2023.
- [90] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, “Network Simulations With the ns-3 Simulator,” *SIGCOMM demonstration*, vol. 14, no. 14, p. 527, 2008.
- [91] W. E. Hart, C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeitl, B. L. Nicholson, J. D. Sirola *et al.*, *Pyomo-Optimization Modeling in Python*. Springer, 2017, vol. 67.
- [92] GNU Project, “GLPK - GNU Project - Free Software Foundation (FSF) — gnu.org,” <https://www.gnu.org/software/glpk/>, 2026, [Accessed 21-03-2026].
- [93] M. Zhu, L. Cazzella, F. Linsalata, M. Magarini, M. Matteucci, and U. Spagnolini, “Toward Real-Time Digital Twins of EM Environments: Computational Benchmark for Ray Launching Software,” *IEEE Open Journal of the Communications Society*, vol. 5, pp. 6291–6302, 2024, doi: <https://doi.org/10.1109/OJCOMS.2024.3463963>.