SALÃO DO CONHECIMENTO UNIJUÍ 2025
Água, ciência e sustentabilidade: desafios para o futuro
De 20 a 24 de outubro de 2025
XXXIII Seminário de Iniciação Científica
XXX Jornada de Pesquisa
XXVI Jornada de Extensão
XV Seminário de Inovação e Tecnologia
XI Mostra de Iniciação Científica Júnior
III Seminário Acadêmico da Graduação UNIJUÍ

Evento: XXVI Jornada de Extensão

ANÁLISE DE DISTÂNCIAS GEOESPACIAIS EM TESTES COM DISPOSITIVOS IOT UTILIZANDO PYTHON E GEOPY[1]

João Víctor Benetti Filipim[2], Peterson Cleyton Avi[3]

[1] Pesquisa institucional desenvolvida na bolsa de IoT da Unijuí.
[2] Bolsista do Laboratório de IoT (PIBEX), estudante do curso de Engenharia de Software.
[3] Professor coordenador das bolsas do Espaço Mais Inovação e doutor em modelagem matemática.

INTRODUÇÃO

A Internet das Coisas (IoT) tem se consolidado como uma das principais vertentes tecnológicas da atualidade, promovendo a integração entre o mundo físico e o digital por meio da comunicação entre dispositivos inteligentes. Essa convergência tem possibilitado avanços significativos em áreas como saúde, mobilidade, automação industrial e cidades inteligentes. Segundo Khan e Yuce (2019), *"IoT is expected to be the most important technology of the next decade, as it connects the digital and physical worlds"*, ressaltando a centralidade da IoT nas transformações digitais futuras.

Neste contexto, este trabalho apresenta uma aplicação prática da tecnologia IoT voltada à análise de dados geoespaciais, utilizando placas LilyGO T-Beam em rede LoRaWAN integradas ao sistema ChirpStack. Os dados de localização, capturados a partir de patinetes elétricos em deslocamento, foram processados por um software desenvolvido em Python, com uso da biblioteca Geopy, para calcular distâncias relativas entre um ponto de referência e os demais pontos enviados periodicamente. A proposta visa demonstrar a importância da análise de dados em tempo quase-real para o rastreamento e a validação de dispositivos IoT em ambientes de uma cidade real.

METODOLOGIA

A pesquisa foi conduzida em ambiente urbano real na cidade de Ijuí (RS), com o objetivo de coletar dados geoespaciais transmitidos por um dispositivo IoT em deslocamento. Para isso, foi utilizado uma placa LilyGO T-Beam, equipada com módulo GPS e rádio compatível com LoRaWAN na frequência de 915 MHz estando ligada a uma antena com rede compatível, servindo de gateway. A placa foi embarcada em um patinete elétrico que com ele

circulei por diferentes rotas da cidade. O dispositivo foi configurado para realizar transmissões periódicas de coordenadas geográficas a cada 1 minuto e 29 segundos, por meio da rede LoRaWAN.

Os dados enviados foram recebidos por um servidor ChirpStack, atuando como rede privada LoRaWAN, instalado localmente e conectado à antena de recepção. O ChirpStack permitiu o gerenciamento dos dispositivos, visualização dos pacotes LoRaWAN e exportação manual das coordenadas recebidas (latitude e longitude).

Após a captura, os dados foram processados em um software desenvolvido em Python, utilizando a biblioteca Geopy. O sistema pede ao usuário a definição de um ponto de referência inicial e, a partir disso, permitia o registro de múltiplos pontos subsequentes. A cada novo ponto inserido, o sistema calcula automaticamente a distância geodésica (em metros ou quilômetros) com base no ponto de origem.

Para escolher o método mais apropriado de cálculo de distância, foi elaborada uma tabela comparativa com base em critérios técnicos como precisão, modelo da Terra utilizado e aplicação recomendada. A tabela levou em consideração as principais fórmulas disponíveis (Haversine, Geodesic, Vincenty e Great-circle) e foi baseada na documentação oficial da biblioteca Geopy (Figura 1).

Figura 1 - Tabela de Comparação de Modelos Coerentes com o Geopy

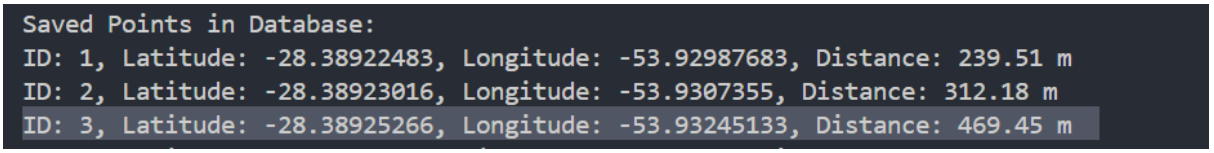
Método	**Modelo da Terra**	**Precisão**	**Ideal para**	**Suporte no Geopy**	**Função usada**
Haversine	Esférico	Média	Curta/média distância, visualizações rápidas	Indireto (via distance())	distance((lat1, lon1), (lat2, lon2))
Geodesic	Elipsóide (WGS-84)	Alta	Qualquer distância, especialmente longa	Sim	geodesic((lat1, lon1), (lat2, lon2))
Vincenty	Elipsóide (WGS-84)	Muito alta	Longas distâncias (mais preciso que Haversine)	Obsoleto no Geopy v2+	usado até v1.22

Great-circle	Esférico (R=6371km)	Média	Navegação, rotas simples	Sim	great_circle((lat1, lon1), (lat2, lon2))

Com base nessa análise, optou-se pelo método **Geodesic**, por oferecer maior precisão em distâncias urbanas reais e considerar o formato elipsoidal da Terra (modelo WGS-84).

Os dados finais, compostos por **latitude, longitude e distância ao ponto de referência**, foram armazenados em um banco de dados estruturado (Figura 2). Essa base permitiu observar tendências de afastamento ou aproximação dos dispositivos em tempo quase-real, com foco na mobilidade urbana e comportamento espacial dos equipamentos IoT em uso prático.

Figura 2 - Banco de Dados que Armazena as Informações Capturadas



RESULTADOS E DISCUSSÃO

Os dados geoespaciais coletados ao longo dos trajetos urbanos permitiram observar um comportamento de transmissão estável na maioria dos envios realizados pelo dispositivo IoT, com boa cobertura em distâncias de até aproximadamente 2 km. As falhas de envio ocorreram em casos pontuais, especialmente em regiões com presença de vegetação densa ou obstáculos físicos, o que está de acordo com as limitações conhecidas da comunicação LoRa em ambientes urbanos.

A análise das coordenadas demonstrou padrões consistentes de deslocamento, com os pontos recebidos organizando-se de forma coerente em relação ao ponto de referência. O cálculo das distâncias geográficas apresentou resultados precisos e confiáveis, o que confirmou a escolha do método geodésico da biblioteca Geopy como adequado para o contexto da pesquisa. A comparação entre diferentes métodos de cálculo (realizada previamente) contribuiu para a fundamentação técnica da abordagem adotada.

Os dados foram armazenados com suas respectivas distâncias e, posteriormente, avaliados quanto à tendência de afastamento ou aproximação do dispositivo. A estrutura do

banco de dados permitiu consultas rápidas e interpretáveis, demonstrando que o sistema desenvolvido em Python atende às necessidades de rastreamento básico em tempo quase-real. No geral, os testes demonstraram a viabilidade da solução, com desempenho satisfatório mesmo em condições reais de operação.

CONSIDERAÇÕES FINAIS

Os resultados obtidos ao longo dos testes demonstraram que a solução proposta para análise de dados geoespaciais de dispositivos IoT é viável, funcional e adequada para aplicações de rastreamento em ambientes urbanos. A combinação entre placas LilyGO T-Beam, rede LoRaWAN, plataforma ChirpStack e o software desenvolvido em Python com uso da biblioteca Geopy permitiu capturar, processar e armazenar informações espaciais de forma eficaz.

Apesar de algumas falhas pontuais de envio, atribuídas à distância máxima de cobertura e interferências causadas por vegetação ou obstáculos urbanos, a grande maioria das transmissões foi bem-sucedida, assegurando a qualidade dos dados analisados. A adoção do método geodésico se mostrou apropriada, fornecendo precisão suficiente para a finalidade da pesquisa.

Dessa forma, conclui-se que a aplicação de ferramentas livres, como Python e Geopy, aliadas a tecnologias de baixo consumo como LoRaWAN, apresenta grande potencial para projetos de monitoramento e análise espacial com dispositivos IoT. O sistema desenvolvido representa uma solução de baixo custo, replicável e de fácil adaptação para outras realidades acadêmicas ou urbanas, contribuindo para o avanço de soluções inteligentes voltadas à mobilidade, logística e monitoramento ambiental.

Palavras-chave: Python. IoT. Geopy. Análise de Dados. Geolocalização.

REFERÊNCIAS BIBLIOGRÁFICAS

KHAN, J. Y.; YUCE, M. R. ***Internet of Things (IoT): Systems and Applications*. [S.l.]**: Springer, 2019. Disponível em: https://books.google.com/books?hl=en&id=DRGwDwAAQBAJ. Acesso em: 21 jul. 2025.

GEOPY. ***geopy.distance – Distance calculations using geographic coordinates***. Documentação oficial, 2024. Disponível em: https://geopy.readthedocs.io/en/stable/#module-geopy.distance. Acesso em: 21 jul. 2025.

BENETTI, **GitHub - JaoVicy/distance-calc-python: Docker Container**. GitHub. Disponível em: https://github.com/JaoVicy/distance-calc-python. Acesso em: 23 jul. 2025.