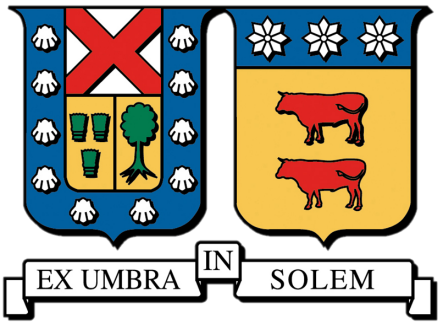
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO - CHILE



"PLATAFORMA IOT PARA MONITOREO DE FLOTA DE SCOOTERS COMPARTIDOS"

RODRIGO JOAQUIN MUÑOZ MORENO

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL TELEMÁTICO

PROFESOR GUÍA:	**MOHAMED ABDELHAMID**
PROFESOR CORREFERENTE:	**JOSÉ MANUEL MARTINEZ**

Octubre - 2026

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: Plataforma IoT para Monitoreo de Flota de Scooters Compartidos

Nombre del candidato(a): Rodrigo Joaquin Muñoz Moreno

Carrera / Grado: Ingeniería Civil Telemática

Campus: Casa Central Valparaiso **;** **Departamento**: Electrónica

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Mohamed Abdelhamid, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 13-10-2026 ; **Firma:** M. Abdelhamid

Estudiante o Candidato(a):

Fecha: 13-10-2026 ; **Firma:** _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Agradecimientos

En este punto cúlmine de mi formación académica, deseo expresar mi gratitud a todas las personas que hicieron posible la realización de esta memoria.

En primer lugar, agradezco al profesor Mohamed, por su disposición, compromiso y preocupación durante el desarrollo de esta investigación. Su guía constante enriqueció mi forma de comprender y abordar los desafíos que se presentaron.

Mi gratitud también se extiende a la Universidad Técnica Federico Santa María, por brindarme los recursos, instalaciones y apoyo necesarios para llevar a cabo este proyecto.

A mis amigos y compañeros, por su acompañamiento, sugerencias y apoyo, que me ayudaron a continuar en los momentos difíciles.

Finalmente, agradezco a mi familia por su constante apoyo y confianza, que han sido fundamentales para mi formación y desarrollo académico. Este logro también refleja su respaldo y acompañamiento a lo largo de este proceso.

Funding

This work was supported by ANID CTI 250019 Innovation Center for Sustainable Energy Transition (SET-Chile) and Proyecto Interno USM 2025 under Grant PI_LIR_25_15.

Resumen

El presente trabajo aborda el diseño e implementación de una plataforma basada en Internet de las Cosas (IoT) destinada al monitoreo de flotas de scooters eléctricos en el contexto de la micromovilidad compartida. La plataforma permite la supervisión en tiempo real del estado operativo de cada vehículo, incluyendo su ubicación, nivel de batería y disponibilidad, lo que facilita la gestión eficiente de la flota y la toma de decisiones informadas por parte de los operadores. Asimismo, el desarrollo de esta solución se enmarca dentro de la promoción del uso de la micromovilidad compartida en entornos urbanos, contribuyendo a alternativas de transporte más sostenibles y a la optimización de la movilidad dentro de la ciudad, mejorando la experiencia del usuario y fomentando un uso responsable y eficiente de los recursos de transporte compartido.

Palabras Clave: Micromovilidad compartida, monitoreo de flota, LTE/4G, LoRa, Internet de las cosas

Abstract

This work addresses the design and implementation of a platform based on the Internet of Things (IoT) for monitoring fleets of electric scooters within the context of shared micromobility. The platform enables real-time supervision of the operational status of each vehicle, including its location, battery level, and availability, thereby facilitating efficient fleet management and informed decision-making by operators. Furthermore, the development of this solution is framed within the promotion of shared micromobility in urban environments, contributing to more sustainable transportation alternatives and the optimization of urban mobility. This approach enhances the user experience while fostering responsible and efficient use of shared transportation resources.

Keywords: Shared micromobility, fleet monitoring , LTE/4G, LoRa, Internet of Things

Índice de figuras

2.1. Dispositivo IoT embebido 12
2.2. Arquitectura IoT 12
2.3. Dispositivo VL11OC 13
2.4. GPS Tracker Mlab 4G 13
2.5. Arquitectura de tres niveles para LoRaWAN 15
2.6. Prototipo implementado 15
2.7. Arquitectura de capas del sistema 16
2.8. Implementación física del prototipo 16
2.9. Arquitectura de sistema para vehículos eléctricos 17
2.10. Diagrama de comunicación 18
2.11. Diagrama de flujo del prototipo 19
2.12. Arquitectura del sistema de scooter eléctrico compartido 20
2.13. Arquitectura de red LoRaWAN implementada 21
2.14. Dispositivo IoT a bordo 21
2.15. Arquitectura de hardware 22

3.1. Esquema del sistema basado en capas funcionales 24
3.2. Mapa de cobertura 4G/LTE en la zona de interés. 31

4.1. Flujos de Node-RED de los 3 prototipos 34
4.2. Base de datos almacenando correctamente 35
4.3. Configuración de dashboards de Grafana 36

4.4. Gráficos de parámetros del scooter 37
4.5. Diagrama de comunicación del prototipo WiFi 39
4.6. Diagrama esquemático del prototipo WiFi-LTE 39
4.7. Implementación de circuito físico 40
4.8. Payload prototipo WiFi 41
4.9. Diagrama de comunicación de prototipo LoRa 42
4.10. Diagrama esquemático de prototipo LoRa 43
4.11. Circuito transmisor 44
4.12. Circuito receptor 44
4.13. Payload enviada por el microcontrolador 46
4.14. Diagrama de comunicación 4G 47
4.15. Diagrama esquemático de prototipo 4G 47
4.16. Foto de implementación de circuito 48
4.17. Payload microcontrolador 49
4.18. Gráfico de voltaje vs SOC para baterías 18650 50
4.19. Tabla de valores de voltaje vs SOC 50

5.1. Ruta de pruebas 53
5.2. Ruta coloreada por RSSI 54
5.3. Ruta coloreada por RSSNR 54
5.4. Ruta coloreada por latencia 55
5.5. Ruta coloreada por RSSI 56
5.6. Ruta coloreada por latencia 56
5.7. Ubicación del gateway LoRa en la ruta 58
5.8. Ruta coloreada por RSSI 60
5.9. Ruta coloreada por Latencia 61
5.10. Ruta coloreada por SNR 62

Índice de cuadros

2.1. Tabla comparativa de plataformas IoT de empresas de micromovilidad compartida 11
2.3. Tabla comparativa de trabajos estudiados 23
4.1. Características de los escenarios de comunicación implementados . . . 38
5.1. Resumen de los resultados obtenidos en los escenarios de comunicación 63

Índice general

1. Introducción **5**

1.1. Problema por resolver 5

1.2. Acercamiento a la solución 6

1.3. Objetivos 7

1.3.1. Objetivo general 7

1.3.2. Objetivos específicos 8

1.4. Estructura 8

2. Estado del arte de sistemas IoT en micromovilidad **10**

2.1. Estado de la micromovilidad en Chile 10

2.2. Soluciones comerciales existentes 11

2.3. Trabajos relacionados 14

3. Arquitectura propuesta **24**

3.1. Capa de percepción 24

3.2. Capa de comunicaciones 25

3.3. Capa de Middleware 27

3.4. Capa de aplicación 28

3.5. Requisitos del sistema 28

3.6. Tecnologías utilizadas 29

3.6.1. Hardware 29

3.6.2. Software 29

3.7. Análisis de Cobertura y Condiciones Locales 30

4. Prototipo Funcional **32**

4.1. Implementación de la plataforma software 33

4.1.1. Configuración de Node-RED 33

4.1.2. Configuración de Grafana 36

4.2. Implementación de prototipos 37

4.2.1. Prototipo 1: Comunicación híbrida WiFi-LTE 38

4.2.2. Prototipo 2: Comunicación LoRa 41

4.2.3. Prototipo 3: Comunicación 4G/LTE 46

5. Pruebas y Resultados **52**

5.1. Ruta escogida para las pruebas 52

5.2. Prueba de Prototipo 4G/LTE 53

5.2.1. Intensidad de señal (RSSI) 54

5.2.2. RSSNR 54

5.2.3. Latencia 55

5.2.4. Pérdida de paquetes 55

5.3. Prueba de Prototipo híbrido WiFi-LTE 55

5.3.1. Intensidad de señal (RSSI) 56

5.3.2. Latencia 56

5.3.3. Pérdida de paquetes 58

5.4. Pruebas de Prototipo LoRa 58

5.4.1. Intensidad de señal (RSSI) 59

5.4.2. Latencia 60

5.4.3. Relación Señal-Ruido (SNR) 61

5.4.4. Pérdida de paquetes 62

5.5. Comparación de los resultados obtenidos 62

6. Conclusión y trabajo futuro **64**

Apéndice **67**

Códigos 67

Datos obtenidos 134

Capítulo 1

Introducción

La creciente adopción de vehículos de micromovilidad compartida, como bicicletas y scooters eléctricos, ha transformado la movilidad urbana al ofrecer alternativas ágiles y sostenibles, pero también ha generado desafíos operacionales relacionados con el monitoreo en tiempo real. Para optimizar la distribución, reducir tiempos de inactividad y mejorar el mantenimiento, se requiere supervisar variables como ubicación, nivel de batería, condiciones de uso y detección de fallas mediante mecanismos IoT confiables.

En este contexto, el proyecto desarrolló una plataforma IoT integral dirigida a empresas de micromovilidad, combinando dispositivos embarcados, comunicación inalámbrica y servicios en la nube. Esta tesis detalla mi aporte en el diseño, implementación y validación del subsistema de hardware instalado en el scooter, incluyendo la selección de componentes, decisiones de arquitectura física, diseño del prototipo, pruebas experimentales y las justificaciones técnicas detrás de cada elección, así como su impacto en la plataforma completa.

1.1. Problema por resolver

Actualmente, las empresas de micromovilidad enfrentan dificultades para obtener visibilidad en tiempo real sobre el estado y ubicación de sus vehículos, lo que complica la planificación de operaciones, la atención de fallas técnicas y la distribución

estratégica de la flota. Esta falta de información puede generar pérdidas económicas, sobreutilización de algunos vehículos, escasez en ciertas zonas y problemas de mantenimiento que afectan la experiencia del usuario.

El sistema a desarrollar tiene como propósito resolver este problema mediante una solución IoT capaz de registrar y transmitir información en tiempo real desde cada vehículo, permitiendo una supervisión continua de su estado físico (como batería, uso o desperfectos), ubicación geográfica y disponibilidad para el usuario. Así, se busca facilitar la toma de decisiones operativas y estratégicas que maximicen la eficiencia del servicio.

1.2. Acercamiento a la solución

La solución propuesta está orientada al **monitoreo eficiente de la flota** de vehículos de micromovilidad compartida, posicionando este aspecto como el elemento central de la plataforma. Cada scooter estará equipado con sensores IoT integrados, capaces de registrar en tiempo real variables clave como la ubicación (GPS), el nivel de batería, el estado operativo y la disponibilidad de cada unidad. Esta información permitirá la visualización y gestión centralizada de toda la flota desde la plataforma.

Los dispositivos IoT transmitirán los datos de manera continua a un nodo central a través de un sistema de comunicación bidireccional, asegurando la actualización permanente de la información necesaria para el análisis y la toma de decisiones. Esta arquitectura facilita la detección temprana de fallas, la generación automática de alertas y la optimización operativa mediante herramientas de análisis y predicción de demanda.

En cuanto al software, la plataforma incluirá un sistema de gestión y optimización de flotas, diseñado para cubrir cuatro funciones principales:

- **Monitoreo de la flota**: supervisión en tiempo real del estado, ubicación y disponibilidad de los vehículos.
- **Reporte de daños**: los usuarios podrán notificar incidencias en scooters directa-

mente desde la aplicación móvil.

- **Alarmas inteligentes**: generación de alertas automáticas ante fallas, baterías críticas o incidentes de seguridad.

- **Planificación y distribución**: optimización de rutas de redistribución y mantenimiento, apoyada en análisis predictivos de patrones de uso e historial de demanda.

De este modo, se logra mejorar la cobertura del servicio, reducir los tiempos de respuesta y facilitar la toma de decisiones estratégicas para la empresa operadora. Adicionalmente, se desarrollará una aplicación móvil destinada a los usuarios finales, que integrará el **reporte de daños de scooters**, y la recepción de **notificaciones ante alertas o incidencias**.

Este enfoque integral, que combina hardware y sensores integrados, comunicación inalámbrica y software analítico, proporciona a la solución un alto grado de innovación y adaptabilidad frente a las necesidades específicas de empresas que buscan escalar y optimizar sus operaciones de micromovilidad.

Se reitera que en este informe individual, se abordará principalmente el desarrollo que se hizo del diseño e implementación del hardware.

1.3. Objetivos

1.3.1. Objetivo general

Implementar una plataforma IoT para monitorear en tiempo real la ubicación, disponibilidad y estado operativo de scooters eléctricos, con el propósito de mejorar la eficiencia en la gestión de flotas en servicios de micromovilidad compartida.

1.3.2. Objetivos específicos

1. Levantar los requerimientos técnicos, operativos y de usuario mediante revisión documental y análisis de soluciones de referencia.

2. Diseñar una arquitectura tecnológica modular y escalable que integre hardware, red de comunicación y software.

3. Integrar componentes IoT capaces de capturar variables operativas de los vehículos, tales como ubicación, nivel de batería y aceleración, utilizando sensores y microcontroladores configurados en un dispositivo de monitoreo.

4. Implementar una red de comunicación inalámbrica que permita transmitir datos entre los vehículos y el sistema central mediante distintas tecnologías de acceso, incluyendo comunicación LoRa punto a punto, conectividad WiFi asistida por pasarela móvil y comunicación celular 4G LTE.

5. Evaluar el desempeño de las tecnologías de comunicación implementadas en términos de cobertura, latencia y confiabilidad.

6. Construir una plataforma de software para almacenar, procesar y visualizar los datos recopilados mediante un backend escalable y dashboards analíticos.

7. Validar el funcionamiento del sistema IoT mediante pruebas técnicas y un despliegue de prototipos piloto.

1.4. Estructura

El presente documento está estructurado en capítulos, cada uno de los cuales aborda diferentes aspectos clave del proyecto. A continuación, se describe el contenido de cada capítulo:

Capítulo 1: Incluye el contexto, problema a resolver, acercamiento a la solución, objetivos y estructura general del documento.

Capítulo 2: Analiza el estado del arte mediante el estudio de papers, muestra soluciones disponibles en el mercado y el estado de empresas que prestan el servicio de micromovilidad compartida en Chile actualmente.

Capítulo 3: Presenta la arquitectura del sistema, incluyendo los diagramas de capas funcionales y las tecnologías empleadas en cada una de ellas, abarcando tanto los componentes de hardware como las soluciones de software.

Capítulo 4: Breve descripción de los prototipos a implementarse y de su forma de validación.

Capítulo 5: Describe en detalle el diseño de cada prototipo, muestra su implementación, pruebas realizadas en terreno y los resultados obtenidos.

Capítulo 6: Resume los hallazgos, conclusiones y proyecciones de mejora para los prototipos.

Capítulo 2

Estado del arte de sistemas IoT en micromovilidad

A continuación, se presentan diversos proyectos que abordan la aplicación de Internet de las Cosas (IoT) en el ámbito de la micromovilidad compartida. Estos trabajos, documentados en artículos científicos, proponen distintos enfoques para la resolución del problema, lo que permite observar una variedad de estrategias tecnológicas y metodológicas. Asimismo, se revisan soluciones actualmente disponibles en el mercado, junto con una descripción del estado actual de este rubro en Chile.

El análisis de estos trabajos proporciona un marco de referencia fundamental para orientar adecuadamente el desarrollo de la solución propuesta. En particular, permite comparar distintas tecnologías de comunicación, identificar qué datos resultan relevantes de censar y obtener ideas clave sobre la arquitectura del sistema a diseñar.

2.1. Estado de la micromovilidad en Chile

En los últimos años, el uso de la micromovilidad compartida ha crecido considerablemente en las ciudades, impulsando la aparición de diversas empresas que ofrecen este servicio [1]. Entre las principales se encuentran Woosh, Lime y Grin. Estas

compañías utilizan plataformas IoT para el monitoreo de sus flotas, con el objetivo de optimizar la logística y garantizar la seguridad.

Se realizó un estudio de las principales empresas que operan en Chile en el arriendo de scooters eléctricos, y se elaboró la siguiente tabla comparativa sobre las características de las plataformas IoT que emplean para el monitoreo de sus flotas.

Cuadro 2.1: Tabla comparativa de plataformas IoT de empresas de micromovilidad compartida

Empresa	Datos monitoreados	Tecnología de Comunicación
Lime [2]	GPS, Aceleración, inclinación en 3 ejes, nivel de batería, velocidad, función de bloqueo	2G/4G LTE, Bluetooth (usuario), Wi-Fi (mantenimiento)
Whoosh [3]	GPS, velocidad, nivel de batería, Sensor de bloqueo	4G LTE (principal), 2G (backup), Bluetooth, Wi-Fi
Jet [4]	GPS, nivel de batería, sistema de geofencing para limitar velocidad	LTE/4G (inferido), conectividad móvil de corto alcance con usuario
Grin [5]	GPS, nivel de batería, Sensor de bloqueo	3G/4G LTE, Bluetooth, Wi-Fi. (Inferidos)
Scoot [5]	GPS, nivel de batería, Sensor de bloqueo	2G/3G/4G LTE, Bluetooth, Wi-Fi (Inferidos)

Estos datos nos sirven como referencia para el diseño de nuestra plataforma IoT, que se expondrá más adelante. Ya que nos permite saber qué tecnologías de comunicación se están utilizando y qué datos están tomando las empresas del rubro.

2.2. Soluciones comerciales existentes

Actualmente, existen numerosas soluciones IoT en el mercado, ofrecidas por diversas empresas, para abordar el tema de la micromovilidad compartida. A continuación, se presentan algunos de estos productos:

- **XM-IoT:** Fabricado por la empresa OMNI-IoT, que vende el hardware, junto a un servicio de aplicación IoT, con servicio de monitoreo en la nube. Este producto incorpora seguimiento GPS, autenticación con código QR y sistema de bloqueo de motor del scooter. Además, cuenta con alertas de geofencing, batería baja y golpes sufridos. Utiliza conectividad BLE con el usuario y 2G/4G para el sistema de datos, además la aplicación móvil que incorpora, cuenta con sistema de pago integrado. [6]



Figura 2.1: Dispositivo IoT embebido

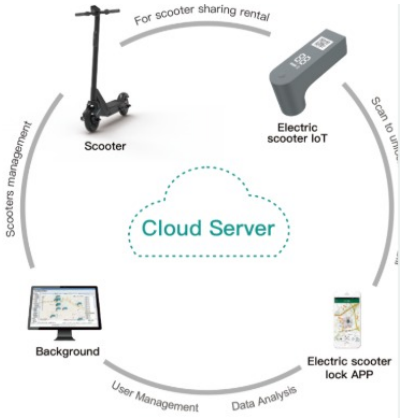


Figura 2.2: Arquitectura IoT

- **GPS VL110C 4G:** Producto ofrecido por la empresa Guardian GPS. Compuesto por módulos de hardware y software, diseñados para el monitoreo de flota, que permiten un seguimiento en tiempo real de los datos de GPS, despliegue de alarmas inteligentes, comunicación mediante red 4G y un aplicativo móvil. [7]



Figura 2.3: Dispositivo VL11OC

- **GPS Tracker 4G LTE:** Producto fabricado por la empresa Microlab, diseñado para el rastreo y monitoreo de vehículos, compatible con autos, motos, camionetas y camiones. Utiliza tecnología 4G LTE en la banda de 2600 MHz. Ofrece funciones clave como localización en tiempo real mediante mapas online o SMS, alertas de movimiento, geocercas, batería baja y fallas de energía. Además, permite detener el motor a distancia y escuchar el entorno a través de un micrófono integrado. Es resistente al agua y al polvo (IP67), incorpora batería recargable y puede ser gestionado desde la aplicación BAANOOL IOT disponible para Android e iOS. [8]



Figura 2.4: GPS Tracker Mlab 4G

2.3. Trabajos relacionados

Los autores en [9] presentan un sistema IoT para scooters basado en una arquitectura de computación en la niebla, esto permite llamarla arquitectura de tres niveles, ya que introduce una capa intermedia entre los nodos finales y la nube, permitiendo descentralizar parte del procesamiento de la información. La capa de niebla está compuesta por dispositivos tales como estaciones base, routers y gateways, ubicados físicamente en el entorno donde operan los nodos finales. Estos dispositivos cuentan, en general, con capacidades computacionales suficientes para realizar procesamiento local de datos, aunque con recursos de almacenamiento más limitados en comparación con los servidores en la nube. El uso de una arquitectura de niebla permite reducir la dependencia de la nube para determinadas tareas de procesamiento, lo que puede resultar beneficioso en escenarios donde se requiere una respuesta más rápida del sistema o una reducción en la latencia de comunicación, según lo propuesto por los autores.

En el desarrollo del trabajo se utilizó un Arduino Uno, un nodo final LoRa Dragino, un módulo GPS y una bicicleta eléctrica en la capa física del sistema. Para la implementación del gateway se utilizó una Raspberry Pi 3B+, equipada con una antena de 3 dBi y la placa concentradora LoRa IMST iC880A-SPI. Se utiliza además el servicio para comunicaciones LoRaWAN The Things Network. El modelo de referencia para micromovilidad compartida se organiza en múltiples capas funcionales, entre las que se incluyen la capa física, la capa de comunicaciones, la capa de control y gestión, y la capa de interfaz de usuario. Se adopta como referencia un modelo de micromovilidad compartida estructurado en capas: física, comunicaciones, control y gestión, e interfaz. Además, la arquitectura incorpora capas transversales de seguridad y continuidad operacional.

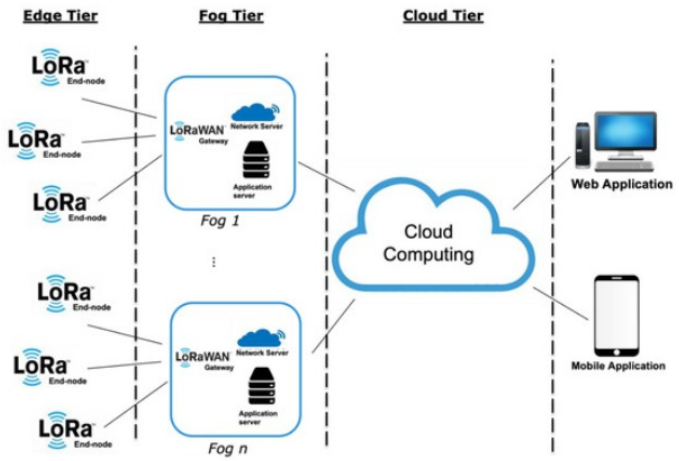


Figura 2.5: Arquitectura de tres niveles para LoRaWAN

Finalmente tenemos la siguiente representación del prototipo implementado.

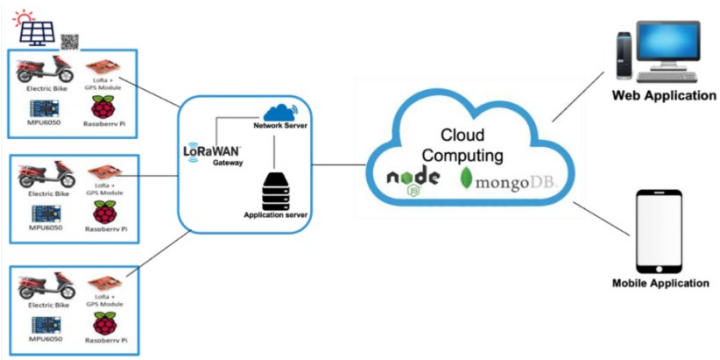


Figura 2.6: Prototipo implementado

Los autores en [10] implementan una red de monitoreo de scooters eléctricos, utilizando sensores GPS y tecnologías de comunicaciín LoRaWAN y NB-IoT. El sistema incluye computación en la nube, análisis de datos y una aplicación móvil para la interacción con el usuario. Una característica clave es la selección automática de la tecnología de red según la cobertura: LoRaWAN en zonas rurales y NB-IoT en zonas urbanas. La unidad de hardware embarcada está basada en una Raspberry Pi Zero W, que actúa como nodo a bordo para gestionar sensores y comunicaciones. Se integra un concentra-

dor de comunicaciones con tres módulos principales: un receptor GPS, un transceptor LoRaWAN y un módem NB-IoT. También se incorporan sensores ambientales y de movimiento. A continuación, se muestra la arquitectura del sistema, presentada en las Figuras 2.7 y 2.8.

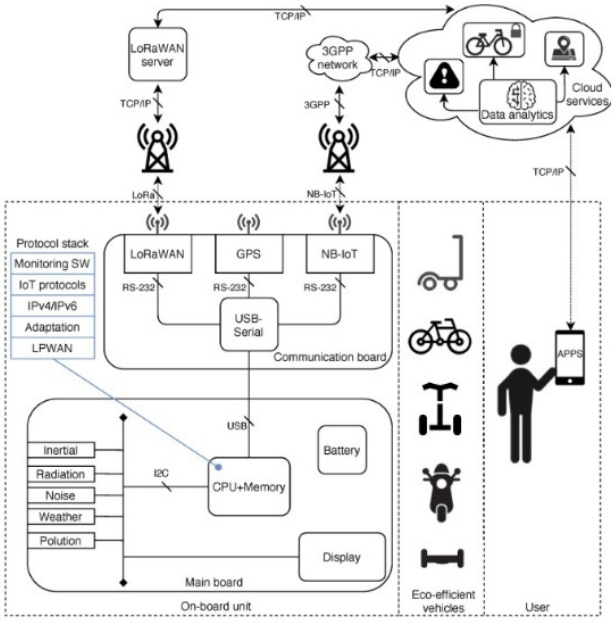


Figura 2.7: Arquitectura de capas del sistema

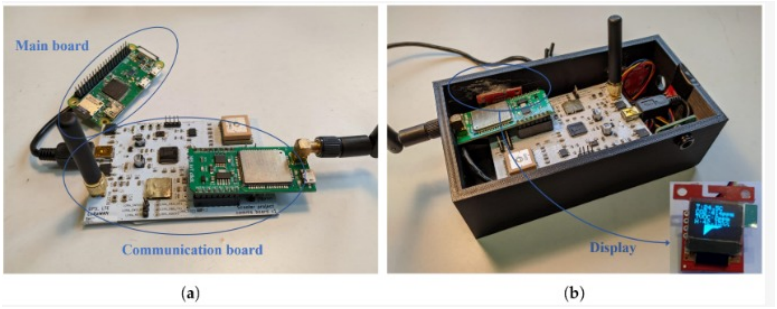


Figura 2.8: Implementación física del prototipo

Los autores en [11] abordaron el problema del monitoreo y la prevención de robos en vehículos de micromovilidad, tanto en bicicletas sin sistema eléctrico propio como en vehículos eléctricos (bicicletas eléctricas, scooters y automóviles). Se implementó una arquitectura IoT antirrobo modular y de bajo consumo, diseñada en dos variantes de hardware según la disponibilidad de energía del vehículo. En ambos casos, el sistema detecta manipulaciones mediante sensores de movimiento e inclinación, obtiene la ubicación del vehículo y la transmite al usuario a través de redes móviles o SMS, priorizando la localización en tiempo real. La arquitectura contempla redundancia energética mediante una batería secundaria oculta, comunicación bidireccional y activación automática del sistema ante eventos sospechosos.

El modelo implementado sigue una arquitectura distribuida de cuatro capas: dispositivo de rastreo, servidor central, aplicación web y terminal móvil. El dispositivo integra módulos de GPS, GSM/GPRS y Bluetooth, permitiendo configuración local, transmisión de datos a la nube y control remoto de funciones como alarmas e inmovilización del vehículo (Kill Switch), además de diagnóstico remoto vía OBD en vehículos eléctricos. La implementación utiliza componentes como el acelerómetro ADXL345, el módulo GSM Quectel M85, el Bluetooth HC-06, el microcontrolador NXP LPC1768, el GPS SiRF Star IV, sensores de inclinación, zumbador de alarma y la interfaz OBD con MCP2515, logrando una solución flexible, portátil y adaptable a distintos escenarios de conectividad y uso.

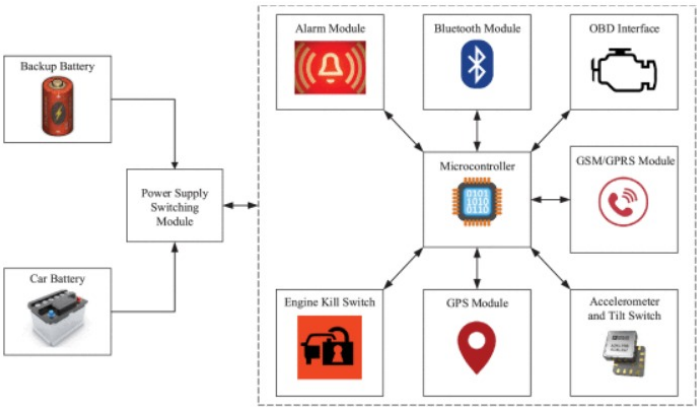


Figura 2.9: Arquitectura de sistema para vehículos eléctricos



Figura 2.10: Diagrama de comunicación

En el artículo [12] se aborda el problema de la seguridad y el monitoreo integral de bicicletas eléctricas mediante la implementación de un sistema IoT embarcado, orientado a la detección de accidentes, la prevención del robo del vehículo y de la batería, la localización en tiempo real y la gestión del estado de la batería. La arquitectura propuesta se basa en un nodo central con ESP32, que integra sensores inerciales (acelerómetro y giroscopio) para la detección de accidentes, un módulo GPS para el seguimiento de la ubicación, un sistema láser–LDR para la detección de robo de batería, relés y alarmas para respuesta ante eventos, y un BMS inteligente que proporciona parámetros como voltaje, corriente, temperatura y estado de carga (SOC). Los datos son transmitidos a una plataforma IoT en la nube mediante protocolos de comunicación eficientes, permitiendo su visualización básica y la supervisión remota del sistema, configurando una solución modular y escalable orientada a mejorar la seguridad del usuario y la protección del vehículo.

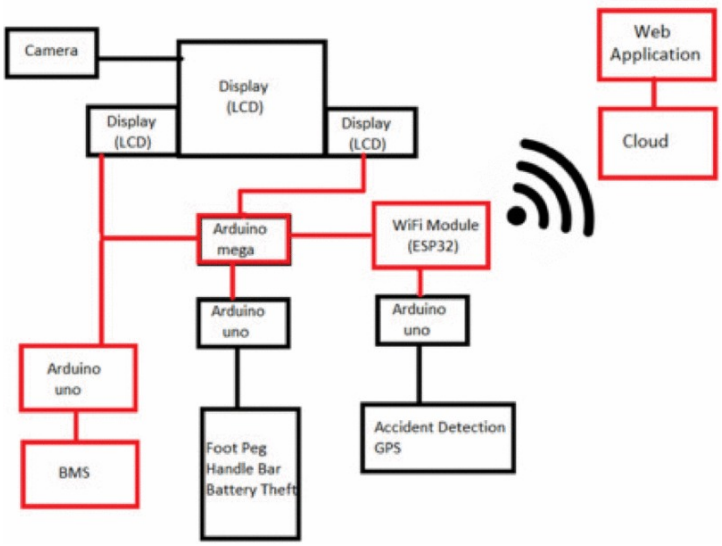


Figura 2.11: Diagrama de flujo del prototipo

En el artículo [13] se analiza la arquitectura típica de las plataformas de patinetes eléctricos compartidos desde una perspectiva de ciberseguridad y gestión de riesgos, con el objetivo de identificar vulnerabilidades que puedan comprometer su operación y sostenibilidad. El modelo se estructura en tres componentes principales: el patinete eléctrico, que integra controladores de motor y batería, módulos de comunicación y posicionamiento (GPS, BLE, CAN, UART); la aplicación de alquiler, responsable de la interacción con el usuario y la gestión de sesiones; y la plataforma en la nube, que centraliza la autenticación, el almacenamiento de datos, los pagos y el monitoreo del estado de los vehículos. Sobre esta arquitectura, el trabajo clasifica amenazas clave como violaciones de privacidad, ataques de denegación de servicio, vulnerabilidades de firmware y software, y suplantación de GPS, destacando su impacto en la seguridad física, la integridad de los datos y la disponibilidad del sistema, y proponiendo medidas de mitigación como cifrado robusto, autenticación, mecanismos anti-spoofing y actualización segura de firmware.

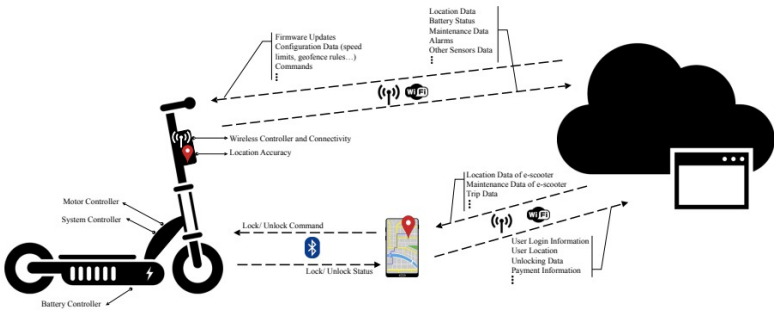


Figura 2.12: Arquitectura del sistema de scooter eléctrico compartido

En el trabajo [14] se analiza el uso de la tecnología LoRaWAN como alternativa de comunicación IoT para sistemas de bicicletas compartidas, abordando el problema de la escalabilidad, cobertura y capacidad de red en entornos urbanos frente a soluciones basadas en redes celulares. El trabajo describe la arquitectura típica de una red LoRaWAN, compuesta por dispositivos finales (bicicletas con módulos de rastreo), puertas de enlace y un servidor de red, organizada bajo una topología de "estrella de estrellas" y operando en bandas ISM con restricciones de ciclo de trabajo. Se estudia el impacto de parámetros físicos y MAC como el factor de propagación (SF), el ancho de banda, el desvanecimiento y las colisiones, evaluando su efecto sobre la tasa de recepción de paquetes (DER) mediante simulaciones de escalabilidad. Además, se implementa y valida experimentalmente un módulo rastreador LoRa basado en Arduino Yun, un transceptor Semtech SX1272 y un receptor GPS, demostrando que, si bien LoRa permite comunicaciones de largo alcance y bajo consumo, su capacidad se degrada en escenarios urbanos densos y con altos SF, lo que hace crítica la planificación y ubicación de las puertas de enlace para lograr una cobertura eficiente en aplicaciones de micromovilidad.

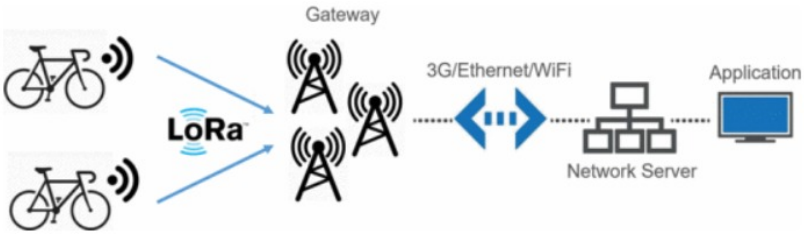


Figura 2.13: Arquitectura de red LoRaWAN implementada

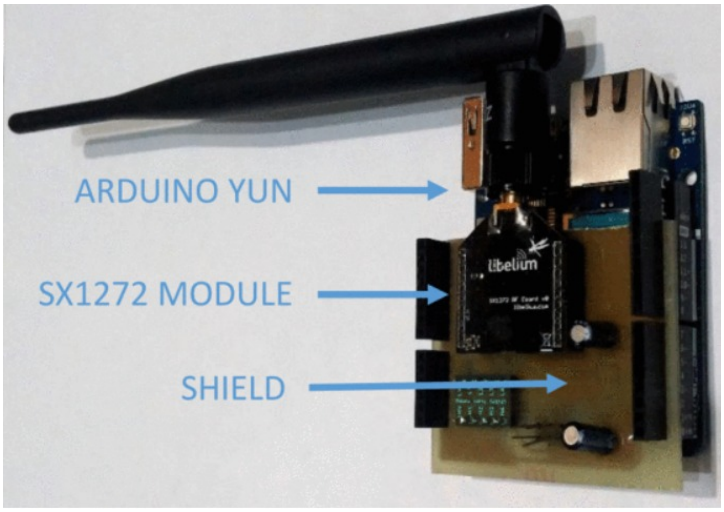


Figura 2.14: Dispositivo IoT a bordo

El estudio [15] presenta un sistema inteligente de detección de accidentes en bicicletas, basado en IoT y sensores embebidos, que combina detección de ángulos, alertas GSM y control de motor para mejorar la seguridad de los ciclistas. A diferencia de otros trabajos que solo registran caídas o envían alertas SOS, este sistema utiliza un sensor MPU6050 (acelerómetro y giroscopio) para medir cambios bruscos de aceleración e inclinación de la bicicleta, estableciendo umbrales de 45° y 60° según la gravedad del accidente. Al detectar una caída, el sistema puede enviar mensajes SMS a servicios de emergencia mediante el módulo GSM SIM900A, y simultáneamente interrumpir la alimentación del motor mediante un relé conectado a Arduino para prevenir daños

adicionales. Además, se incorpora un módulo de retroalimentación de estado de la carretera, que permite registrar y visualizar rutas con mayor incidencia de accidentes, aportando información útil a las autoridades y ciclistas. Esta es una propuesta que se destaca por ser económica y enfocada en la seguridad del conductor.

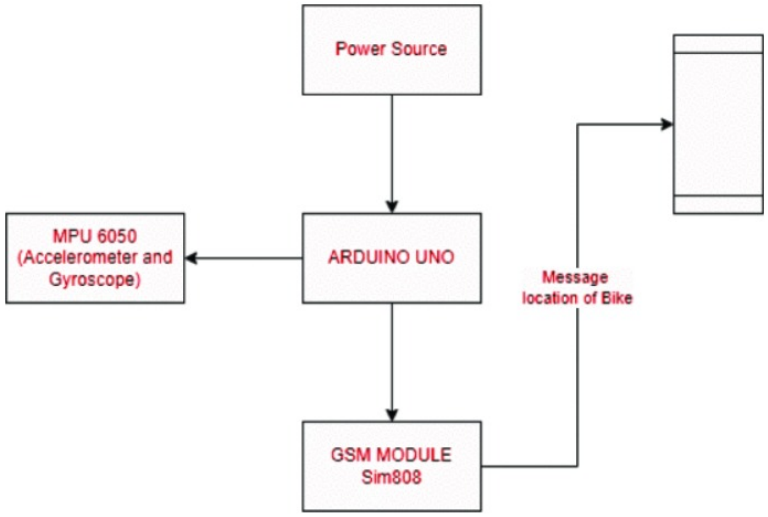


Figura 2.15: Arquitectura de hardware

Para finalizar la revisión de los trabajos relacionados, la Tabla 2.3 presenta una síntesis de los principales sistemas estudiados, considerando el tipo de vehículo, la arquitectura empleada, los dispositivos a bordo, el gateway, las tecnologías de comunicación y las variables monitoreadas.

Cuadro 2.3: Tabla comparativa de trabajos estudiados

Paper	**Vehículo**	**Arquitectura**	**OBE**	**Gateway**	**Comunicaci**	**Monitoreo**
Fog computing [9]	e-bike	Fog Computing	Arduino Uno + LoRa Dragino shield, GPS, MPU6050	Raspberry Pi 3 con concentrador LoRa IMST iC880A	LoRaWAN	GPS y aceleración
Eco-Efficient Mobility [10]	e-scooter	Cloud Computing	Raspberry Pi Zero con LoRa Murata + NB-IoT (BC95), GPS	NB-IoT (LTE del operador), LoRaWAN sin gateway especificado	LoRaWAN + NB-IoT	GPS
IoT Enabled Optimized Architectures [11]	e-bike	Cliente-servidor IoT	Mbed NXP LPC1768, ADXL345, GSM Quectel M85, Bluetooth HC-06, GPS	Red celular GSM/GPRS	GSM/GPRS	GPS
IoT Enabled Electronic Security [12]	e-bike	Cloud Computing	ESP32, GPS, acelerómetro + giroscopio, módem GSM	Infraestructura Wi-Fi hace de gateway	GSM + Wi-Fi	GPS, aceleración y batería
E-scooter Sharing Platforms [13]	e-scooter	Cloud Computing	Micro no especificado, GPS	Smartphone + WiFi	3G, LTE, 5G, WiFi	GPS y velocidad
Performance of LoRa [14]	bike	Cloud Computing	Arduino Yun + transceptor SX1272 + GPS	LoRaWAN RAK7249 Multibandwith	LoRaWAN	GPS
Bike Skid Detection [15]	e-bike	Edge Computing + GSM	Arduino UNO, GPS, MPU6050	Módulo GSM SIM900A	GSM/GPRS	Aceleración e inclinación

Capítulo 3

Arquitectura propuesta

El sistema se estructura mediante una arquitectura IoT compuesta por cuatro capas funcionales. La Figura 3.1 presenta una visión general de esta arquitectura, incluyendo las principales herramientas asociadas a cada capa. Posteriormente, se describe en detalle la función y los componentes de cada una de ellas.



Figura 3.1: Esquema del sistema basado en capas funcionales

3.1. Capa de percepción

En una arquitectura IoT estándar, la capa de percepción interactúa directamente con el entorno mediante sensores que capturan datos. Estos son procesados de manera básí-

ca por microcontroladores y enviados a las capas superiores para su almacenamiento y análisis. Algunos sistemas incluyen actuadores en esta capa, que ejecutan acciones automáticas según la información registrada. Esta capa es esencial, pues permite el monitoreo del sistema y su control automático cuando es necesario.

Componentes

- **Microcontroladores:** Son el núcleo de la capa de percepción, encargados de recolectar los datos de los sensores, realizar un procesamiento básico, tomar decisiones de acción y gestionar la comunicación con las capas superiores. En este sistema se utiliza un Heltec WiFi LoRa V2 para los prototipos WiFi y LoRa, y un ESP32 TTGO T-A7670G para el prototipo 4G.

- **Sensores:** Dispositivos encargados de medir variables físicas del entorno, como temperatura, aceleración, grado de inclinación, nivel de batería, presión o nivel de luminosidad, y de convertirlas en señales eléctricas que son enviadas al microcontrolador para su procesamiento. En este trabajo se utiliza el acelerómetro MPU9250, el medidor de voltaje INA 219 y el conversor análogo digital integrado del ESP32 TTGO para medir voltaje.

- **GPS (Sistema de Posicionamiento Global):** Dispositivo encargado de estimar la posición del sistema en tiempo real a partir de señales satelitales, proporcionando coordenadas de latitud y longitud. En este trabajo se utiliza el módulo GPS Neo M7 en el prototipo híbrido WiFi-LTE, el módulo ATGM336H en el prototipo basado en LoRa y, en el prototipo 4G, se emplea el receptor GPS integrado en el microcontrolador.

3.2. Capa de comunicaciones

Esta capa es responsable de establecer y administrar la comunicación entre el microcontrolador y la puerta de enlace correspondiente, permitiendo la transmisión de

datos hacia servidores en la nube, plataformas de aplicación u otros sistemas de control y monitoreo, e incluyendo también la comunicación entre nodos finales. Para la transmisión de datos pueden emplearse diversas tecnologías de comunicación inalámbrica, como redes celulares, LoRa, WiFi o Bluetooth. En este proyecto se desarrollaron tres prototipos, cada uno basado en una tecnología de conectividad diferente: 4G/LTE, LoRa y una comunicación híbrida WiFi-LTE.

Componentes

- **Gateway LoRa:** Dispositivo encargado de enlazar los nodos finales con la infraestructura de red, permitiendo el envío de los datos recolectados hacia servidores remotos. La tecnología LoRa se caracteriza por su capacidad de transmisión a larga distancia y su bajo consumo energético. Esta puerta de enlace recibe información desde múltiples dispositivos IoT LoRa y la retransmite utilizando tecnologías como WiFi, Ethernet o redes de telefonía celular. En este proyecto se implementó un gateway LoRa punto a punto basado en una Raspberry Pi 3B y una placa Dragino LoRa HAT Single Channel.

- **Heltec WiFi LoRa V2:** Microcontrolador que integra conectividad WiFi y LoRa, ambas utilizadas en los respectivos prototipos para la transmisión de datos hacia la infraestructura de red. En el caso de LoRa, se estableció una comunicación punto a punto entre el microcontrolador y el gateway. Para el prototipo WiFi, el dispositivo envía los datos a través de una red WiFi proporcionada por un teléfono móvil, el cual actúa como punto de acceso y reenvía la información hacia la infraestructura de red celular mediante conexión 4G.

- **Modem 4G A7670:** Modem integrado en nuestro microcontrolador ESP32 TTGO A7670. Se encarga de la comunicación entre el microcontrolador y las redes de telefonía celular, para lo que utiliza una tarjeta sim.

- **Infraestructura de red celular 4G:** Corresponde a la red pública de telefonía móvil que permite la transmisión de datos entre el dispositivo IoT y los servidores

remotos. Esta infraestructura actúa como punto de acceso a Internet, cumpliendo funcionalmente el rol de gateway, al gestionar el enrutamiento y la conectividad sin requerir una puerta de enlace dedicada, a diferencia de otras tecnologías como LoRa. En este proyecto, la comunicación se realiza a través de la red 4G del operador móvil mediante el uso de una tarjeta SIM.

- **Smartphone como Gateway WiFi:** En el prototipo híbrido WiFi-LTE, un teléfono móvil actúa como puerta de enlace al proporcionar conectividad a Internet mediante la creación de una zona WiFi. El microcontrolador ESP32 se conecta a esta red para transmitir los datos, los cuales son posteriormente reenviados por el teléfono, mediante la infraestructura de red celular, hacia los servidores remotos.

3.3. Capa de Middleware

En una arquitectura IoT estándar, la capa de middleware actúa como intermediaria entre los dispositivos y las aplicaciones, centralizando la recepción, procesamiento y gestión inicial de los datos, además de facilitar la integración de distintos protocolos y servicios.

La capa de middleware se implementó utilizando el broker público EMQX, empleando el protocolo MQTT para la publicación y suscripción de datos entre los dispositivos IoT y los servicios de procesamiento. En Node-RED se definió un flujo independiente para cada uno de los prototipos desarrollados. Para la ejecución de los servicios se utilizó una Raspberry Pi 4 Model B, donde se desplegó un entorno basado en Docker que incluye los contenedores de Node-RED, InfluxDB, Grafana y Portainer. Este último permite administrar y monitorear los servicios desplegados mediante una interfaz gráfica.

En Node-RED se reciben los datos provenientes del broker y se realiza la validación de la payload, verificando mediante identificadores que la información corresponda a alguno de los tres prototipos implementados. Asimismo, se calcula la latencia en

milisegundos, definida como la diferencia entre el tiempo de envío de los datos desde el microcontrolador y el tiempo de recepción en Node-RED. Este valor se agrega como un nuevo campo dentro de la payload antes de su almacenamiento en InfluxDB.

Finalmente, los datos son almacenados en InfluxDB, para luego ser entregados a la capa de aplicación.

3.4. Capa de aplicación

En IoT, la capa de aplicación corresponde al punto de interacción con el usuario final, proporcionando una interfaz para la visualización, supervisión y gestión de la información proporcionada por los dispositivos.

En este trabajo, se utilizó Grafana como herramienta de visualización, la cual se conecta directamente a la base de datos InfluxDB para consultar y mostrar los datos almacenados, seleccionándose la ventana de tiempo de los datos que se desea mostrar en el gráfico. Se diseñaron dashboards independientes para cada uno de los tres prototipos, los cuales filtran la información mediante el campo identificador de cada dispositivo. Cabe señalar que, en el contexto de esta tesis, la capa de aplicación tiene un propósito predominantemente funcional: verificar que los dispositivos están enviando correctamente la información, sin profundizar en el diseño de la interfaz ni en funcionalidades avanzadas.

3.5. Requisitos del sistema

- **Conectividad:** Los dispositivos finales deben tener conectividad para poder enviar los datos a la aplicación, de la misma forma que el usuario de la aplicación debe tener conexión a internet para acceder a ellos.

- **Recolección de datos:** El sistema debe recolectar los datos necesarios para el funcionamiento de la aplicación. Estos son el nivel de batería, ubicación GPS,

aceleración y disponibilidad de vehículo.

- **Interfaz de usuario:** La aplicación proporciona una interfaz básica que permite verificar la recepción de los datos enviados por los prototipos de hardware. A través de esta herramienta se pueden visualizar lecturas esenciales, como aceleración, posición y nivel de batería del scooter, asegurando que la información generada por los dispositivos llega correctamente al sistema.

- **Disponibilidad:** El sistema debe estar disponible la mayor parte del tiempo para permitir su uso continuo por parte de los usuarios.

3.6. Tecnologías utilizadas

Hacemos un breve recuento de los componentes y herramientas utilizadas.

3.6.1. Hardware

- **Heltec LoRa WiFi V2**

- **Dragino LoRa HAT Single Channel**

- **Raspberry Pi 3B para receptor LoRa**

  Raspberry Pi 4B para correr Docker

- **ESP32 LYLYGO TTGO A7670**

- **Android Smartphone**

- **GPS NEO 7M, GPS ATGM336H, MPU 9250, INA 219**

3.6.2. Software

- **Arduino IDE**

- **Lenguaje C/C++**
- **Visual Studio Code**
- **MQTT**
- **Docker**
- **Grafana**
- **Node-red**
- **InfluxDB**
- **Portainer**

3.7. Análisis de Cobertura y Condiciones Locales

Para garantizar la operación confiable de los dispositivos IoT en el contexto de la micromovilidad, es fundamental analizar la cobertura y las condiciones de conectividad locales. En este sentido, se evaluó la cobertura 4G/LTE mediante mapas disponibles para la zona de interés. En particular, la Figura 3.2 muestra la cobertura de la red móvil de Claro en Viña del Mar, según la información publicada por nPerf, una plataforma que permite consultar información sobre la cobertura y el rendimiento de redes de telecomunicaciones [16]. Por otra parte, se consideró la viabilidad de alternativas LPWAN, como LoRaWAN y LoRa punto a punto, en función de las características del entorno y de la zona objetivo.

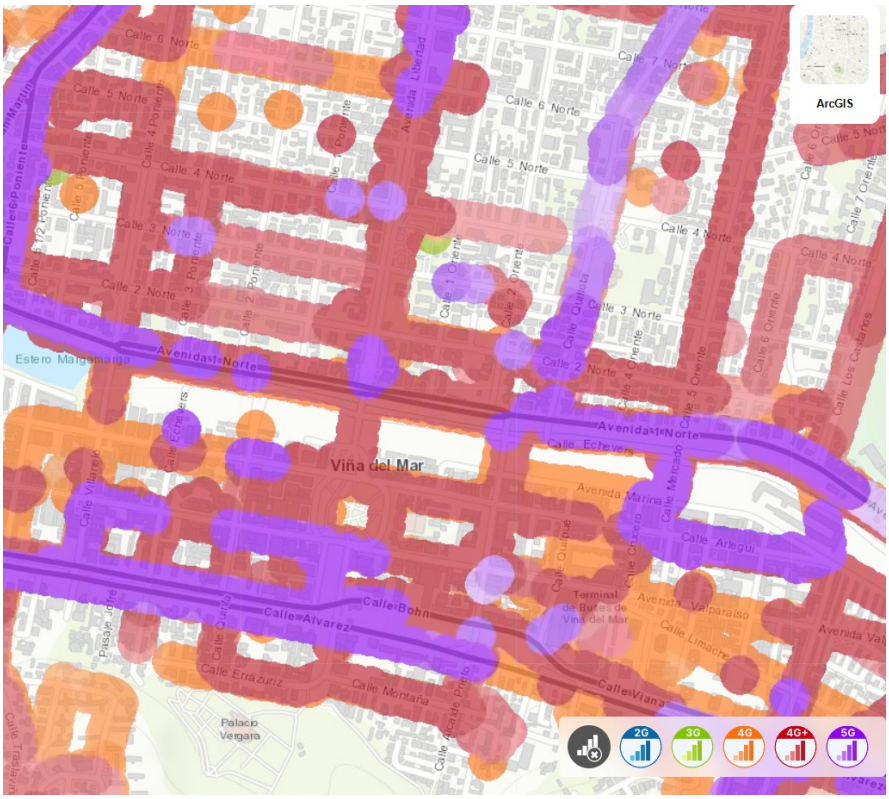


Figura 3.2: Mapa de cobertura 4G/LTE en la zona de interés.

El mapa muestra una amplia cobertura 4G en la zona de pruebas, a lo largo de avenida Libertad. Por otra parte, se prevé que la comunicación LoRa pueda verse afectada por la presencia de obstáculos en el entorno, particularmente árboles y estructuras del alumbrado público, que podrían atenuar la señal. Asimismo, la presencia de edificios a ambos lados de la ruta podría influir en la propagación de la señal; sin embargo, inicialmente no se considera un factor crítico, dado que estos no se encuentran directamente sobre el trayecto. Estas consideraciones teóricas deberán contrastarse con los resultados de las pruebas experimentales.

Capítulo 4

Prototipo Funcional

En esta sección se describe el desarrollo e implementación de la plataforma IoT para la movilidad compartida. El objetivo principal es diseñar una solución tecnológica que permita la recopilación, transmisión, procesamiento y visualización de la información generada por los vehículos, contribuyendo a impulsar la micromovilidad en las ciudades.

La plataforma se estructura siguiendo una arquitectura IoT por capas, en la cual se separan las funciones de adquisición de datos, comunicación, procesamiento y visualización. Este enfoque facilita la escalabilidad, mantenibilidad e interoperabilidad del sistema. El dispositivo de monitoreo instalado en cada vehículo está compuesto por sensores y un microcontrolador, encargados de recopilar información relevante como la ubicación, el nivel de batería y el movimiento.

Para evaluar el desempeño del sistema, se diseñaron e implementaron tres escenarios. El primero corresponde a una comunicación LoRa punto a punto; el segundo, a una conexión híbrida WiFi-LTE que utiliza la zona WiFi generada por un teléfono móvil como pasarela hacia la red celular; y el tercero, a una comunicación directa mediante redes 4G/LTE. Este enfoque permite comparar las tecnologías en términos de cobertura, confiabilidad, latencia y facilidad de despliegue.

Los datos transmitidos son recibidos por el broker EMQX en la nube y posterior-

mente redirigidos mediante Node-RED hacia la base de datos alojada en la Raspberry Pi 4 dentro de la red local, donde son almacenados para su posterior procesamiento y análisis.

A partir de estos datos, se obtienen métricas de cobertura, pérdida de paquetes y latencia, lo que permite validar y comparar experimentalmente las distintas soluciones de hardware propuestas en condiciones reales de operación.

4.1. Implementación de la plataforma software

Como base del proyecto se utiliza un repositorio [17] que despliega un entorno Docker, en el cual se ejecutan los servicios de Node-RED, Portainer, Grafana e InfluxDB. Este entorno se instala y ejecuta en una Raspberry Pi 4B conectada a la red WiFi local.

La Raspberry Pi 4B tiene como única función alojar los servicios necesarios para el funcionamiento del sistema y dar soporte a todos los prototipos. No participa como nodo transmisor, ni como nodo receptor en ninguno de ellos.

4.1.1. Configuración de Node-RED

Se diseña un flujo de datos para cada prototipo, datos capturados desde el broker EMQX.

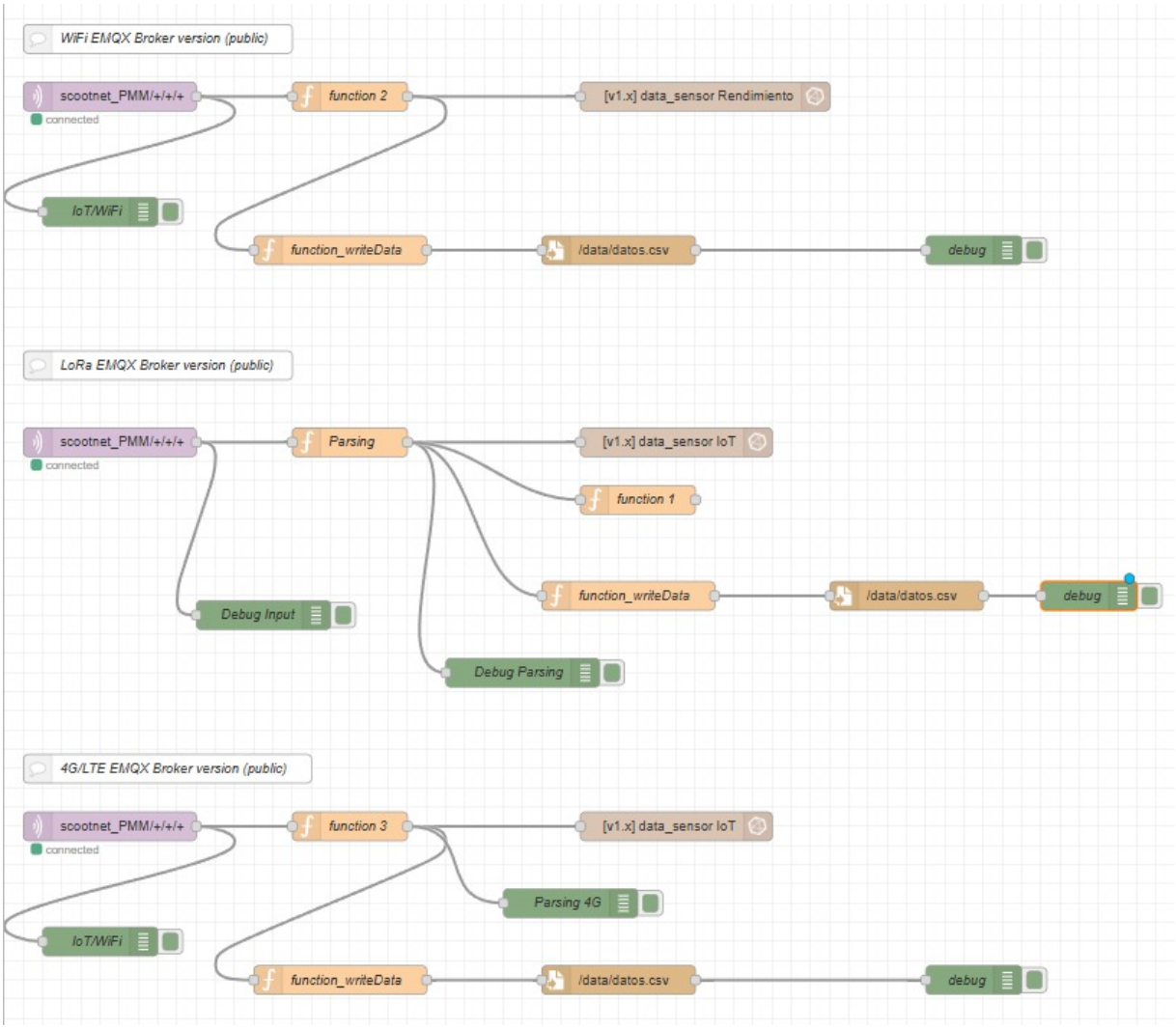


Figura 4.1: Flujos de Node-RED de los 3 prototipos

Los bloques principales son los siguientes: scootnet_PMM/+/+/+, que es la recepción de los datos que llegan al broker EMQX, suscrito al tópico con dicho nombre. Después viene un bloque de función de parsing, que se encarga de tomar la payload que llega desde el broker y darle el formato que exige la base de datos, también realiza el cálculo de la latencia con el tiempo que obtiene Node-RED con el comando Date.now() y lo incorpora en la payload de salida, que finalmente le entrega al bloque de base de datos Influx DB.

Para comprobar el correcto flujo de datos, procedemos a mostrar Influx DB alma-

cenando los datos.

```
> SELECT * FROM IoT ORDER BY time DESC LIMIT 40
name: IoT
time                ax     ay     az    gps_ok id_device latencia latitud    longitud   red rssi snr_rssnr soc time_device   time_server   tx
----                --     --     --    ------ --------- -------- -------    --------   --- ---- --------- --- -----------   -----------   --
1779313033017936600 -0.008 0.002  1.014 1      03        450      -33.021214 -71.55127  4G  -51  -2        0   1779313032561 1779313033011 303
1779313027019926590 0.068  0.154  0.988 1      03        456      -33.021168 -71.551277 4G  -51  2         0   1779313026557 1779313027013 302
1779313021009737579 -0.14  0.203  0.937 1      03        452      -33.021126 -71.551285 4G  -51  -1        0   1779313020553 1779313021005 301
1779313015306523369 -0.137 0.054  0.959 1      03        756      -33.02108  -71.55127  4G  -51  1         0   1779313014543 1779313015299 300
1779313009675014235 -0.094 0.134  1.023 1      03        1135     -33.021042 -71.551247 4G  -51  0         0   1779313008535 1779313009670 299
1779313003079108914 0.133  0.052  1.218 1      03        551      -33.021015 -71.551224 4G  -51  0         0   1779313002521 1779313003072 298
1779312997061163854 -0.102 0.083  0.879 1      03        539      -33.020985 -71.551208 4G  -51  0         0   1779312996515 1779312997054 297
1779312990966397900 -0.28  0.15   0.906 1      03        454      -33.020958 -71.551208 4G  -51  -9        0   1779312990507 1779312990961 296
1779312985062782635 -0.22  0.227  1.165 1      03        556      -33.020916 -71.551208 4G  -51  -9        0   1779312984500 1779312985056 295
```

Figura 4.2: Base de datos almacenando correctamente

En la Figura 4.2 se muestra la estructura de almacenamiento de los datos en la base de datos. El campo `time` corresponde a la marca temporal en la que el dato es registrado. Los campos `ax`, `ay` y `az` representan las aceleraciones medidas en los ejes x, y y z, respectivamente. El campo `gps_ok` indica el estado de la señal GPS, permitiendo verificar si el receptor dispone de una posición válida (*GPS fix*). El campo `id_device` identifica de manera única cada dispositivo. Los campos `latitud` y `longitud` almacenan las coordenadas geográficas obtenidas por el receptor GPS. El campo `red` especifica la tecnología de comunicación utilizada por el dispositivo.

La latencia, expresada en milisegundos, se calcula en Node-RED como la diferencia entre las variables `time_server` y `time_device`. El parámetro `RSSI` corresponde a la intensidad de la señal recibida. Por su parte, `snr_rssnr` representa la relación señal-ruido (SNR) en el prototipo LoRa, mientras que en el prototipo 4G/LTE corresponde a la Relación Señal-Ruido de la Señal de Referencia (RSSNR), cabe señalar que esta variable no se midió en el prototipo WiFi-LTE, debido a que el chip no expone este dato, quedando fuera de nuestro análisis. Finalmente, `soc` indica el porcentaje de carga de la batería y `tx` corresponde al número secuencial del paquete transmitido. Los campos `time_device` y `time_server` corresponden a los tiempos de envío desde el microcontrolador y de recepción por parte del servidor, respectivamente.

4.1.2. Configuración de Grafana

Seleccionamos crear panel, para crear los gráficos que mostraremos en los dashboard de cada prototipo. Después, procedemos a Configurar las querys que hará cada gráfico de los dashboards, haciendo consultas a la base de datos y escogiendo el campo de la payload a visualizar.

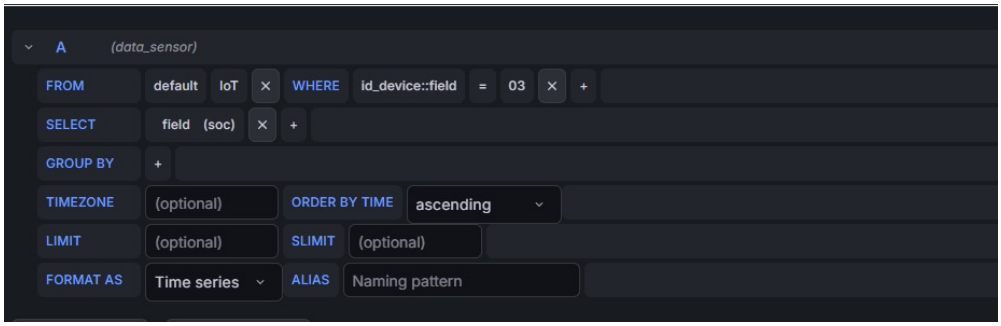


Figura 4.3: Configuración de dashboards de Grafana

En la figura 4.3 se aprecia como el parámetro id device selecciona el prototipo del cual se mostrará el parámetro a graficar, siendo 01 para el prototipo WiFi, 02 para el LoRa y 03 para el 4G. También se observa que se está graficando el nivel de batería (SOC).

A continuación, se muestran los gráficos desplegados. Cabe señalar que todos los prototipos tienen los mismos gráficos.

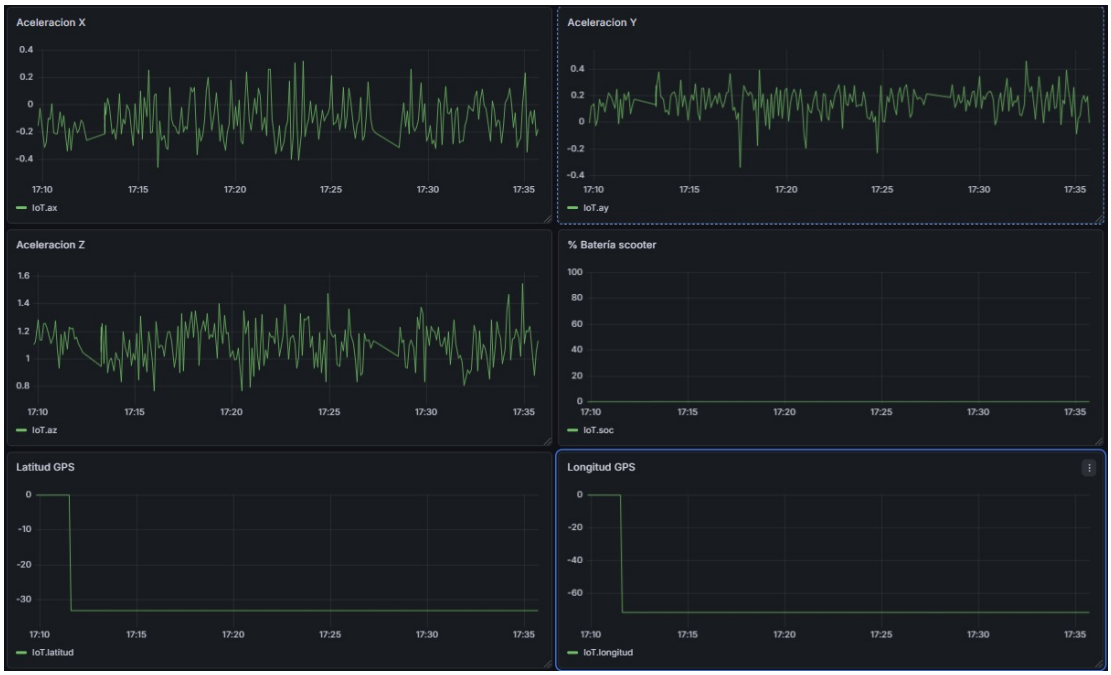


Figura 4.4: Gráficos de parámetros del scooter

Con esto comprobamos el correcto funcionamiento de la plataforma IoT para el monitoreo de los datos. Los datos de los gráficos de la figura 4.4 corresponden al porcentaje de batería, aceleración en sus 3 ejes y las coordenadas GPS (latitud y longitud).

4.2. Implementación de prototipos

Se diseñaron tres escenarios de prototipo, cada uno basado en una tecnología de conectividad distinta, con el objetivo de evaluar y comparar su desempeño. En las siguientes subsecciones se describe cada escenario en detalle.

A continuación, se muestran los tres escenarios de prototipos planteados en este trabajo, cuyas principales características se resumen en la Tabla 4.1.

Cuadro 4.1: Características de los escenarios de comunicación implementados

Escenario	Transmisor	Gateway	Hardware	Comunicaciones	Middleware	Aplicación
1	Heltec WiFi LoRa 32 V2	Smartphone Android	ESP32 + sensores	WiFi + LTE	Node-RED / MQTT	Node-RED / Grafana
2	Heltec WiFi LoRa 32 V2	Raspberry Pi 3B + Dragino LoRa HAT	ESP32 + sensores	LoRa P2P	Node-RED / MQTT	Node-RED / Grafana
3	ESP32 TTGO/-LilyGO T-A7670G	Red celular LTE/4G	ESP32 + módulo 4G/LTE + sensores	4G/LTE	Node-RED / MQTT	Node-RED / Grafana

4.2.1. Prototipo 1: Comunicación híbrida WiFi-LTE

Componentes utilizados:

- **Heltec LoRa Wi-Fi V2:** microcontrolador con módulo Wi-Fi incorporado.
- **Sensor acelerómetro:** módulo GY-91, que integra el sensor de aceleración MPU9250.
- **GPS:** módulo u-blox Neo M7
- **Medición de voltaje:** Sensor de voltaje INA 219

Diagrama de comunicación

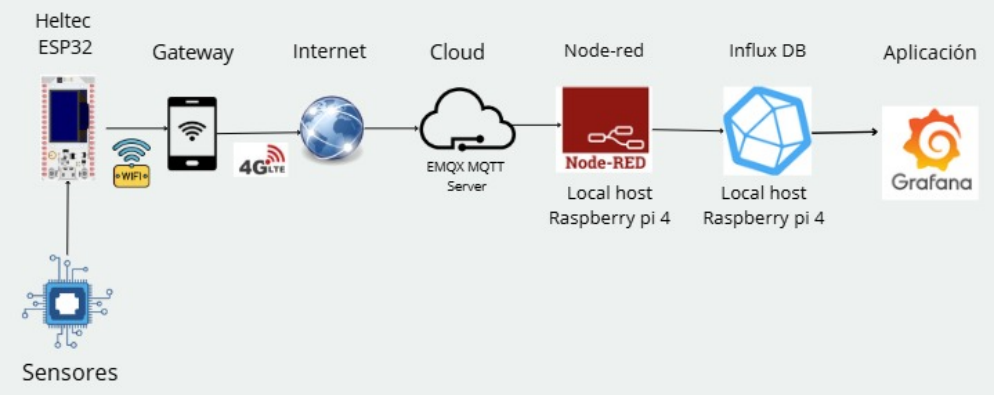


Figura 4.5: Diagrama de comunicación del prototipo WiFi

Como se aprecia en el diagrama de la figura 4.5, el microcontrolador envía los datos por medio del Wi-Fi proporcionado por el teléfono móvil, que funciona como pasarela, reenviándolos por red celular hacia el broker en la nube, donde son capturados por nuestro Node Red y traídos a nuestro servidor local, que corre en nuestra raspberry.

Diagrama de conexiones

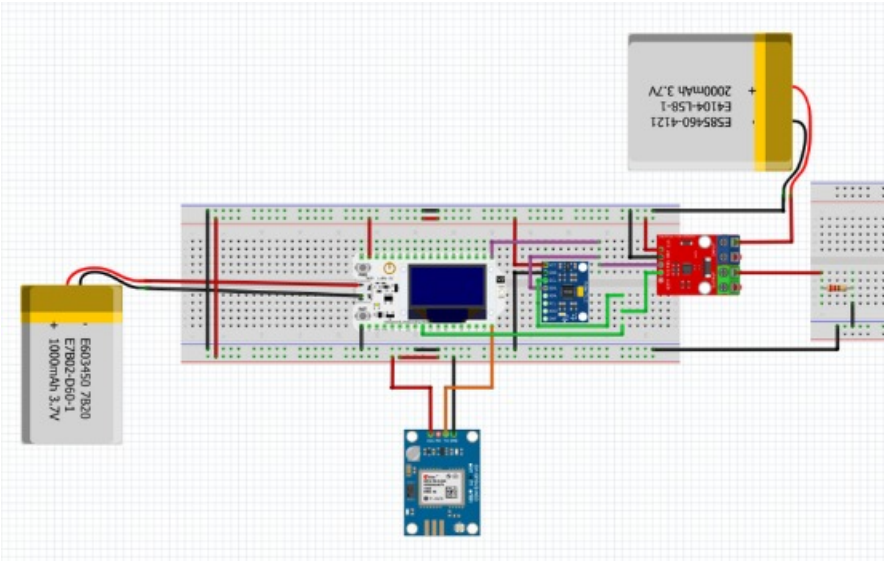


Figura 4.6: Diagrama esquemático del prototipo WiFi-LTE

En la figura 4.6, la batería de la izquierda está alimentando al microcontrolador vía usb. La batería de la derecha (arriba) está simulando la batería del scooter, conectándose al sensor de voltaje y descargándose a través de una resistencia.

Circuito físico

A continuación se muestra el circuito conectado sobre la protoboard de pruebas.

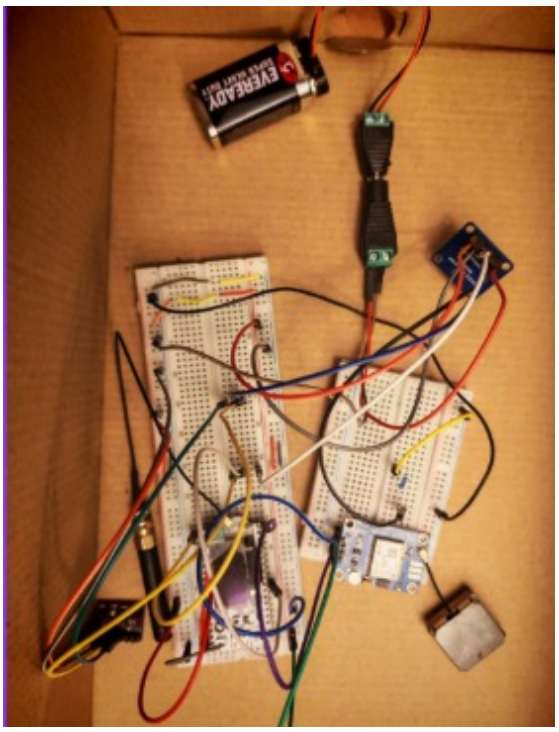


Figura 4.7: Implementación de circuito físico

Implementación real

Se desarrolla un firmware en lenguaje C que se encarga de leer los sensores y gestionar el envío de datos periódicamente, por medio del protocolo MQTT, hacia el broker público EMQX.

En esta prueba se envían datos de `id_device`, `time_device`, `gps_ok`, latitud y longitud GPS, los 3 datos del acelerómetro correspondientes a los ejes x, y y z,

porcentaje de batería (soc), número de paquete tx y el RSSI. El dato gps_ok es un verificador que nos dice si el gps está tomando señal o está colgado. El time_device, por su parte, corresponde al tiempo de envío del dato, expresado en base al tiempo universal que utilizan las máquinas, obtenido mediante protocolo NTP, con precisión de milisegundos, el cual es necesario para calcular la latencia en Node-RED, con el tiempo del servidor backend.

Cabe señalar que este prototipo no proporciona directamente la SNR, ya que el microcontrolador no expone esta métrica, por lo que no se considera en el análisis.

A continuación mostramos la payload que se envía desde el microcontrolador al broker.

```
String payload_IoT = "{";
    //payload_IoT += "\"id\":" + String(scooterID) + ",";
    payload_IoT += "\"timestamp\":" + String(timestamp) + ",";
    payload_IoT += "\"gps_ok\":" + String(gps_ok ? 1 : 0) + ","; //
    payload_IoT += "\"lat\":" + String(lat, 6) + ",";
    payload_IoT += "\"lon\":" + String(lon, 6) + ",";
    payload_IoT += "\"ax\":" + String(aX, 3) + ",";
    payload_IoT += "\"ay\":" + String(aY, 3) + ",";
    payload_IoT += "\"az\":" + String(aZ, 3) + ",";
    payload_IoT += "\"soc\":" + String(socInt) + ",";
    payload_IoT += "\"Tx_number\":" + String(txNumber+1) + ",";
    payload_IoT += "\"rssi\":" + String(WiFi.RSSI());
    payload_IoT += "}";
```

Figura 4.8: Payload prototipo WiFi

4.2.2. Prototipo 2: Comunicación LoRa

Componentes utilizados:

Transmisor:

- **Heltec LoRa Wi-Fi V2:** Microcontrolador principal con módulo LoRa incorporado.

- **Sensor acelerómetro:** Módulo GY-91 (MPU9250 + BMP280).
- **GPS:** Módulo ATGM336H.
- **Medición de voltaje:** Sensor de voltaje INA 219

Receptor:

- **Raspberry Pi 3B:** Encargado de recibir los datos del transmisor vía LoRa.
- **Módulo Dragino LoRa Hat (Single Channel):** Interface de conectividad LoRa para la Raspberry Pi 3B.

Diagrama de comunicación

A continuación mostramos el diagrama que ilustra el flujo de la comunicación el el prototipo LoRa.

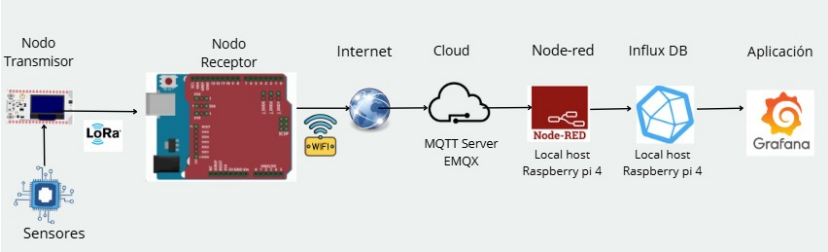


Figura 4.9: Diagrama de comunicación de prototipo LoRa

Los datos viajan por la arquitectura mostrada, para ser recibidos por el servidor en la nube que entrega los datos a la aplicación web.

Diagrama de conexiones

A continuación mostramos el diagrama de conexiones para el transmisor LoRa.

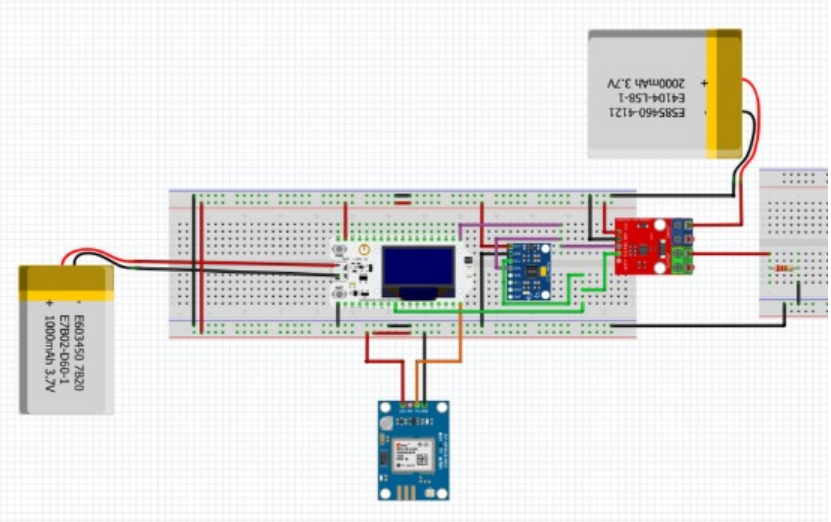


Figura 4.10: Diagrama esquemático de prototipo LoRa

En la figura 4.10, la batería de la izquierda alimenta el microcontrolador vía usb y la de la derecha se descarga a través de una resistencia, simulando ser la batería del scooter, para tomar datos con el sensor de voltaje.

El receptor por su parte es solamente la Raspberry Pi con la shield Dragino LoRa Hat montada y conectada con una fuente de corriente para raspberry.

Circuito físico

A continuación mostramos la implementación del circuito del prototipo LoRa.

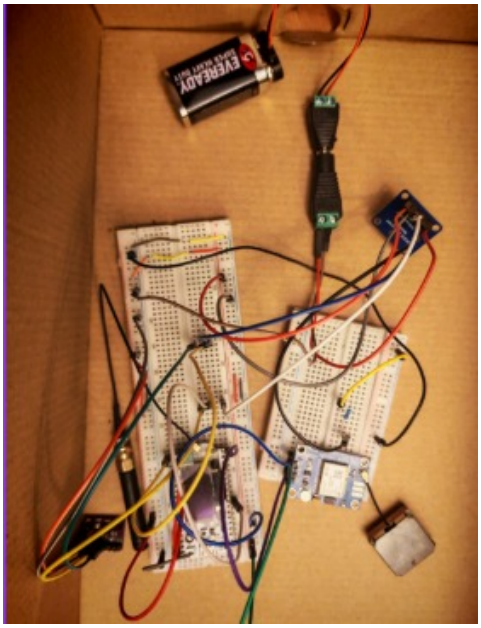


Figura 4.11: Circuito transmisor



Figura 4.12: Circuito receptor

Implementación .

Se implementó una comunicación punto a punto entre el transmisor Heltec y el receptor Raspberry + Dragino LoRa Hat, en la banda de 915 MHz. El transmisor recopila los datos y los envía por comunicación LoRa. El receptor, por su parte, recibe los datos por LoRa y los reenvía, vía WiFi, al broker EMQX en la nube, para posteriormente ser recopilados por Node-Red, para su procesamiento y almacenamiento en Influx DB. De tal forma que puedan ser utilizados por nuestra aplicación de Grafana.

Los paquetes que se envían desde el transmisor LoRa tienen la siguiente estructura: `id_scooter`, `epochTime`, `gps_ok`, `Latitud`, `Longitud`, `ax`, `ay`, `az`, el nivel de batería `(soc)` y el número de paquete `txNumber`. El peso del paquete enviado por el transmisor es de 55 bytes.

Las variables `ax`, `ay` y `az` corresponden a los valores del acelerómetro en sus 3 ejes. El `id_scooter` nos sirve para que el receptor LoRa filtre aquellos mensajes corruptos o de otras comunicaciones que pueda recibir al estar escuchando y considere solo aquellos con el identificador del prototipo LoRa. El campo `epochTime` corresponde al tiempo obtenido mediante NTP, expresado en milisegundos, y se utiliza como tiempo de envío en todos los dispositivos.

A continuación mostramos los paquetes que enviamos desde el microcontrolador.

```
String payload_IoT = "";

payload_IoT += String(scooterID);
payload_IoT += ",";
payload_IoT += String(epochTime);
payload_IoT += ",";
payload_IoT += String(gps_ok ? 1 : 0);
payload_IoT += ",";

payload_IoT += String(lat, 6);
payload_IoT += ",";
payload_IoT += String(lon, 6);
payload_IoT += ",";

payload_IoT += String(aX, 2);
payload_IoT += ",";
payload_IoT += String(aY, 2);
payload_IoT += ",";
payload_IoT += String(aZ, 2);
payload_IoT += ",";

payload_IoT += String(socInt);
payload_IoT += ",";
payload_IoT += String(txNumber + 1);
```

Figura 4.13: Payload enviada por el microcontrolador

Cuando la payload es recibida por el receptor LoRa, es decir, por la raspberry equipada con el módulo Dragino LoRa Hat, esta le agrega a la payload los valores de intensidad de señal `(RSSI)` y relación señal-ruido `(SNR)`, que el chip detecta y permite manipular. Posteriormente la reenvía por WiFi al broker EMQX, para su enrutamiento hacia nuestra plataforma.

4.2.3. Prototipo 3: Comunicación 4G/LTE

Componentes utilizados .

Este prototipo utiliza una placa ESP32 LYLYGO TTGO, que trae incorporado un modem 4G/LTE A7670G y un GPS compatible con las constelaciones GPS, GLONASS y BeiDou. Además se utiliza una tarjeta SIM de la empresa Claro, para conectarse a la red celular.

Para leer el voltaje de la batería se usó el adc interno del microcontrolador que

entrega el voltaje de entrada y mediante calculo en base al divisor resistivo se obtiene el voltaje de la batería.

Diagrama de comunicación

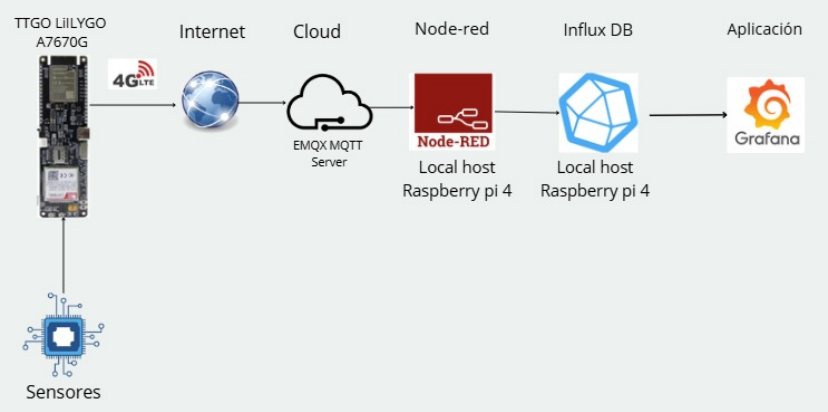


Figura 4.14: Diagrama de comunicación 4G

Se aprecia en el diagrama de la figura 4.14 que los datos son enviados desde el microcontrolador hacia el broker público EMQX, para ser enrutados a nuestra base de datos Influx db, mediante Node Red. Una vez que los datos son almacenados en la base de datos, están listos para ser utilizados por la aplicación de Grafana.

Diagrama de conexiones

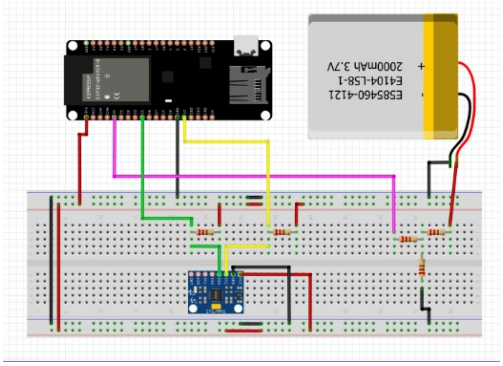


Figura 4.15: Diagrama esquemático de prototipo 4G

Para la medición del voltaje de la batería mediante el pin ADC se utilizó un divisor resistivo, el cual permite adecuar el nivel de tensión al rango de operación del microcontrolador. Las conexiones de la interfaz de comunicación I2C se implementaron utilizando resistencias pull-up de 4.7 kΩ en los buses SDA y SCL.

El módulo GPS no se muestra en el esquema, ya que se encuentra integrado de forma interna en el microcontrolador. De igual manera, la batería que alimenta el sistema es de tipo interna.

Circuito Físico

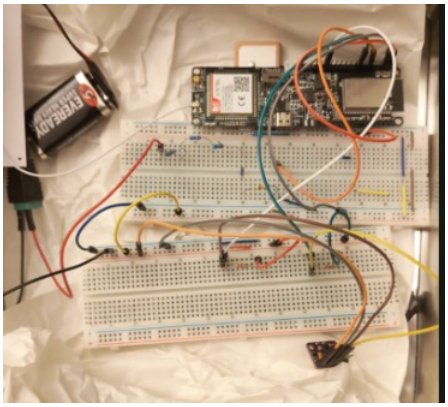


Figura 4.16: Foto de implementación de circuito

Implementación real

Se configuró el microcontrolador con el modem A7670 integrado para enviar datos periódicamente al Broker. Se utilizó una tarjeta SIM de la empresa Claro para darle conectividad celular.

Los paquetes que se envían desde el microcontrolador tienen la siguiente estructura:

`timestamp`, `gps_ok`, `Latitud`, `Longitud`, `ax`, `ay`, `az`, `soc`, `txNumber`, `RSSI` y `RSSNR`,

Cabe mencionar que el ADC del microcontrolador presentó desviaciones en cuanto

a las mediciones de voltaje respecto al valor real medido con multímetro, por lo que se implementó un proceso de calibración. Se adquirieron muestras a lo largo de todo el rango operativo, comparando las lecturas del ADC con mediciones de un multímetro de referencia. Con estos datos se generó un modelo de interpolación lineal para corregir la relación entre el valor digital y la magnitud física real. La calibración permitió reducir el error de medición a aproximadamente 0.02 V.

A continuación mostramos la payload enviada desde el microcontrolador al broker.

```
String payload_IoT = "{";

    payload_IoT += "\"timestamp\":" + String(timestamp) + ",";
    payload_IoT += "\"gps_ok\":" + String(gps_ok ? 1 : 0) + ",";
    payload_IoT += "\"lat\":" + String(lat, 6) + ",";
    payload_IoT += "\"lon\":" + String(lon, 6) + ",";
    payload_IoT += "\"ax\":" + String(aX, 3) + ",";
    payload_IoT += "\"ay\":" + String(aY, 3) + ",";
    payload_IoT += "\"az\":" + String(aZ, 3) + ",";
    payload_IoT += "\"soc\":" + String(socInt) + ",";
    payload_IoT += "\"Tx_number\":" + String(txNumber + 1) + ",";
    payload_IoT += "\"rssi\":" + String(rssi_dBm) + ",";
    payload_IoT += "\"rssnr\":" + String(rssnr);
    payload_IoT += "}";
```

Figura 4.17: Payload microcontrolador

Cálculo de porcentaje de batería .

Para finalizar con la sección de prototipos, se procede a explicar cómo se obtiene el porcentaje de batería (SOC).

Se hizo una tabla de valores (Voltaje, SOC), tanto para las baterías de litio NCR18650, como para las alcalinas de 9V, ya que con ellas se experimentó.

Para las baterías de litio NCR18650 18650, se recogió el estudio Ïnitial Conditioning Characterization Test and other preliminary testing” [18], y se recogió el siguiente gráfico.

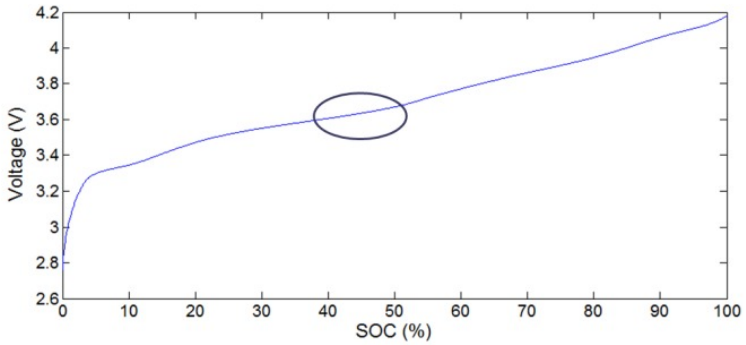


Figure 6: NCR18650B OCV=f(SOC) function.

Figura 4.18: Gráfico de voltaje vs SOC para baterías 18650

Se hizo una tabulacion de los datos del gráfico, obteniendo la siguiente tabla:

Voltaje (V)	SOC estimad
≥ 4.20	100 %
4.15	95 %
4.10	90 %
4.00	80 %
3.90	70 %
3.85	65 %
3.80	60 %
3.75	55 %
3.70	50 %
3.65	45 %
3.60	40 %
3.55	35 %
3.50	30 %
3.45	25 %
3.40	20 %
3.35	15 %
3.30	10 %
3.20	5 %

Figura 4.19: Tabla de valores de voltaje vs SOC

Posteriormente se implementó un algoritmo que asigna a cada valor de voltaje leído, un valor de SOC, en base a los valores de la tabla. El algoritmo realiza una interpolación por tramos en la tabla, para cubrir los valores medios y así abarcar todo el rango.

Lo anterior también se aplicó a las baterías alcalinas de 9 V. Sin embargo, al no

encontrarse una curva oficial que relacionara el voltaje con el estado de carga (SOC), se asumió una relación lineal entre ambas variables. De esta manera, considerando un rango de operación aproximado entre 9 V y 7 V, a modo de ejemplo, un voltaje de 8 V se traduciría en un SOC estimado del 50 %. Si bien esta aproximación simplifica el comportamiento real de la batería, se consideró adecuada para los objetivos del proyecto, ya que no se requiere una estimación precisa del estado de carga, sino un indicador representativo del nivel de batería que permita su transmisión, almacenamiento y visualización en la plataforma IoT.

Capítulo 5

Pruebas y Resultados

En este capítulo se detallan las pruebas diseñadas para cada prototipo, los escenarios planteados y los resultados obtenidos en cada caso.

Para el cálculo de métricas, todos los prototipos utilizan tiempo epoch en milisegundos. Este dato se obtiene mediante conexión WiFi, utilizando el protocolo NTP, ya que otras tecnologías de comunicación, como los comandos AT de LTE, solo entregan precisión en segundos. Debido a que el cálculo de latencia requiere una resolución en milisegundos, los tres prototipos obtienen este valor a través de una conexión WiFi proporcionada por un teléfono móvil utilizado durante las pruebas.

5.1. Ruta escogida para las pruebas

Las pruebas se realizaron en Viña del Mar, a lo largo de avenida Libertad, siguiendo un recorrido rectangular delimitado por calle 6 Norte y calle Arlegui, frente a la plaza Vergara. La Figura 5.1 muestra en verde el trayecto utilizado.

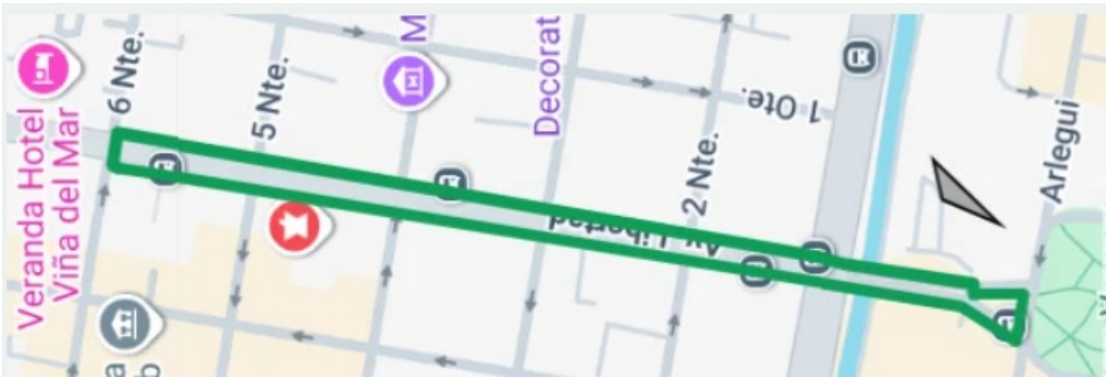


Figura 5.1: Ruta de pruebas

La ruta tiene una longitud aproximada de 730 metros y un ancho de 23,5 metros. A lo largo de su extensión, presenta una hilera de árboles altos en las veredas, postes de iluminación y diversas estructuras urbanas de menor tamaño, como paraderos de locomoción colectiva, las cuales pueden actuar como posibles obstáculos para la propagación de la señal. Además, a ambos costados de la ruta se encuentran edificios de gran altura, lo que incrementa el nivel de obstrucción y afecta potencialmente la calidad del enlace de comunicación.

Se contemplará un escenario para cada una de las tres tecnologías, los cuales se harán caminando con el dispositivo en las manos, mientras recolecta y envía datos. Asimismo, cada prototipo incorporará ciertos elementos específicos en sus pruebas, que se detallarán en las respectivas secciones, debido a características propias de las comunicaciones a evaluar. Las pruebas se centrarán en obtener métricas de cobertura, pérdidas de paquetes, latencia y calidad del enlace, con la finalidad de comparar los rendimientos de las tecnologías implementadas.

5.2. Prueba de Prototipo 4G/LTE

En esta prueba se envían datos cada 6 segundos vía comunicación 4G/LTE.

Se obtienen las siguientes métricas y mapas de ruta coloreada, que dan cuenta del rendimiento en la ruta.

5.2.1. Intensidad de señal (RSSI)

La Intensidad de Señal (RSSI) presentó un buen rendimiento, viéndose debilitado levemente en tres tramos de aproximadamente 25 metros y fuertemente en un solo tramo de largo similar. Se considera pertinente señalar que el tramo fuertemente afectado se encontraba en presencia de un paradero de micros metálico, por lo que cabe la posibilidad de que este bloqueara la señal.



Figura 5.2: Ruta coloreada por RSSI

El promedio que registró la RSSI fue de $-51{,}44\ [\mathrm{dBm}]$, con una mediana de $-51{,}00\ [\mathrm{dBm}]$.

5.2.2. RSSNR

La comunicación 4G/LTE tuvo un rendimiento ambivalente en calidad de RSSNR, puede explicarse debido a que es una zona densamente poblada, en la cual hay mucha interferencia entre las distintas antenas de redes móviles.



Figura 5.3: Ruta coloreada por RSSNR

Se registró un promedio de $2{,}66\ [\mathrm{dB}]$ y una mediana de $2{,}00\ [\mathrm{dB}]$ para la RSSNR.

5.2.3. Latencia

El prototipo 4G/LTE experimentó baja latencia, sufriendo un aumento crítico en un tramo pequeño solamente.



Figura 5.4: Ruta coloreada por latencia

Se registró una latencia promedio de $479{,}16\ [\mathrm{ms}]$, con una mediana de $437{,}50\ [\mathrm{ms}]$.

5.2.4. Pérdida de paquetes

Se enviaron 240 paquetes, experimentando una pérdida de 12, es decir, sufrió una pérdida del 5 % de los paquetes.

5.3. Prueba de Prototipo híbrido WiFi-LTE

Este prototipo es una comunicación híbrida entre el WiFi que ofrece de pasarela el teléfono a nuestro microcontrolador y la conexión que realiza el celular con internet, que es por redes móviles 4G/LTE.

En esta prueba se envía datos cada 6 segundos y se utiliza la banda de WiFi de $2{,}4\ [\mathrm{GHz}]$.

5.3.1. Intensidad de señal (RSSI)

La intensidad de señal (RSSI) presentó un buen rendimiento, viéndose debilitada en el sector entre 1 norte y la plaza Vergara.



Figura 5.5: Ruta coloreada por RSSI

El promedio que registró la RSSI fue de $-51{,}44\ [\mathrm{dBm}]$ y la mediana fue de $-50{,}00\ [\mathrm{dBm}]$.

5.3.2. Latencia

En la comunicación híbrida WiFi-LTE, la latencia tuvo el siguiente desempeño.



Figura 5.6: Ruta coloreada por latencia

Se experimentó una latencia promedio de $3311{,}82\ [\mathrm{ms}]$. y una mediana de $724\ [\mathrm{ms}]$.

Podemos apreciar una diferencia notable entre la mediana y el promedio, por lo que es necesario realizar un análisis en detalle para comprender la situación.

Hubo valores anormalmente altos que elevaron significativamente el promedio de la latencia. Estos casos se atribuyen, en principio, a problemas de conectividad entre el ESP32 y el punto de acceso móvil, y/o a interrupciones en la conexión WiFi de la red doméstica donde se encontraba la Raspberry Pi 4 que ejecuta el servidor de recepción de los datos.

Los valores anómalos ocurrieron de forma consecutiva y presentaron diferencias de aproximadamente 6 segundos entre sí, coincidiendo con el período de transmisión configurado en el sistema. Además, se observó que dichos paquetes llegaron al servidor prácticamente al mismo tiempo, formando una ráfaga de recepción.

Este comportamiento sugiere que, durante los intervalos de pérdida temporal de conectividad, los paquetes no fueron descartados inmediatamente, sino que permanecieron retenidos temporalmente en algún mecanismo de almacenamiento (buffer) de la pila de comunicaciones o de la aplicación. Una vez restablecida la conexión, los paquetes pendientes fueron transmitidos de forma consecutiva, produciendo la llegada en ráfaga observada en el servidor. Esta hipótesis es consistente con los mecanismos de reconexión y retransmisión implementados en la pila TCP/IP y en el cliente MQTT del ESP32.

Dada la magnitud en que los datos anómalos alteraron el promedio, se consideró relevante calcular los siguientes percentiles:

- P99 = 59254 ms
- 90 = 2255.40 ms
- 85 = 959.0 ms

Para efectos de este análisis, consideramos como dato evidentemente anómalo a aquel que supera a la mediana en factor de 3 o más. Por lo que el análisis por percentil arroja que un 10 % de los datos presentan anormalidad evidente y que al menos el 85 % de los datos está en un rango normal. También se evidencia que un 1 % de los datos

presentó valores de latencia extremadamente altos, llegando a los $59254\,[\mathrm{ms}]$, alterando enórmemente el promedio.

5.3.3. Pérdida de paquetes

Se enviaron y recibieron exitosamente 289 paquetes, experimentando packet loss de $0\,\%$

5.4. Pruebas de Prototipo LoRa

Para estas pruebas, el receptor, conformado por la Raspberry Pi 3B con el Dragino LoRa HAT se instaló en un departamento del noveno piso de un edificio situado en la intersección de calle 2 Norte con avenida Libertad. El equipo se ubicó junto a una ventana orientada en dirección a avenida Libertad. La ubicación del edificio donde se instaló el gateway se indica mediante un círculo rojo en la Figura 5.7.

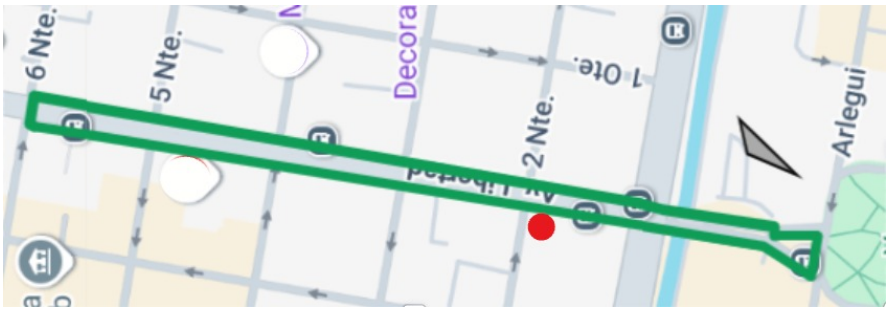


Figura 5.7: Ubicación del gateway LoRa en la ruta

Dado que no se dispone de un despliegue profesional de antenas LoRa, el equipo se instaló aproximadamente a 20 cm al interior del edificio, detrás de la ventana. Esta ubicación podría introducir pérdidas adicionales en el enlace por la atenuación de la señal al atravesar los materiales de la fachada y por posibles obstrucciones de la línea de vista (LoS) a lo largo del trayecto.

Las pruebas se realizaron utilizando un factor de propagación fijo de SF = 7, sin variaciones durante la campaña de medición. Aunque se recorrió la totalidad de la ruta,

la comunicación efectiva se mantuvo únicamente entre el sector de la plaza y calle 3 Norte, por lo que el análisis se limita a ese segmento.

En los mapas de ruta coloreada que se presentan en las siguientes subsecciones, se observa un tramo en el que las mediciones provienen únicamente de la vereda opuesta al edificio donde se ubica el receptor. Esto se debe a que la estructura del edificio bloquea la línea de vista (LoS) directa entre el transmisor y el receptor en la vereda del edificio, generando una mayor atenuación de la señal en dicha zona, lo que se traduce en peor calidad del enlace y mayor pérdida de paquetes.

5.4.1. Intensidad de señal (RSSI)

La RSSI presentó, en general, valores bajos, debido a la distancia entre el receptor (ubicado en el noveno piso) y el nodo transmisor, al bajo *spread factor* utilizado (SF = 7) y a la limitada línea de vista disponible, ya que el receptor se encontraba aproximadamente a 20 cm de la ventana. Se observó que el enlace presentó un buen desempeño únicamente en la vereda opuesta a aquella en la que se encontraba el gateway, en la cuadra comprendida entre las calles 1 Norte y 2 Norte. Este resultado podría explicarse por la existencia de línea de vista (LoS) entre el transmisor y el receptor en dicho tramo. Sin embargo, al continuar el recorrido por avenida Libertad, tanto en dirección a la plaza como en sentido contrario, la intensidad de la señal disminuyó considerablemente, comportamiento que es consistente con la propagación por multicamino y el aumento de las obstrucciones entre el transmisor y el receptor.



Figura 5.8: Ruta coloreada por RSSI

La intensidad de señal registrada es de un promedio de $-87{,}93\,[\mathrm{dBm}]$. y una mediana de $-90{,}00\,[\mathrm{dBm}]$.

5.4.2. Latencia

La latencia presentó un promedio de $552{,}67\,[\mathrm{ms}]$ y una mediana: $526{,}00\,[\mathrm{ms}]$. También se obtuvo una desviación estándar de $147{,}25\,[\mathrm{ms}]$. No hubo fuerte presencia de valores anómalos que alteraran las métricas.



Figura 5.9: Ruta coloreada por Latencia

5.4.3. Relación Señal-Ruido (SNR)

Se experimentó una SNR promedio de $3{,}85\,[\mathrm{dB}]$ y una mediana de $5\,[\mathrm{dB}]$. Asimismo, la desviación estándar fue de $4{,}90\,[\mathrm{dB}]$. Los valores máximos y mínimos registrados fueron de $10{,}5\,[\mathrm{dB}]$ y $-8{,}5\,[\mathrm{dB}]$, respectivamente.



Figura 5.10: Ruta coloreada por SNR

5.4.4. Pérdida de paquetes

De 131 mensajes que se enviaron en dentro de la zona de utilidad de nuestra comunicación LoRa, se recibieron 91 y se extraviaron 40, experimentándose un Packet Loss de 30.53 %.

5.5. Comparación de los resultados obtenidos

Con el objetivo de facilitar la comparación entre los escenarios implementados, la Tabla 5.1 resume los valores promedio de las métricas obtenidas para cada prototipo.

Cuadro 5.1: Resumen de los resultados obtenidos en los escenarios de comunicación

Escenario	RSSI [dBm]	Latencia [ms]	Pérdida de paquetes	SNR [dB]	RSSNR [dB]
WiFi-LTE	$-51{,}44$	$3311{,}82$	$0\,\%$	N/A	N/A
LoRa P2P	$-87{,}93$	$552{,}67$	$30{,}53\,\%$	$3{,}85$	N/A
4G/LTE	$-51{,}44$	$479{,}16$	$5\,\%$	N/A	$2{,}66$

Capítulo 6

Conclusión y trabajo futuro

En este trabajo se desarrolló una plataforma IoT para el monitoreo de flotas de scooters compartidos, estructurada en cuatro capas: adquisición, comunicación, procesamiento y visualización. La solución integra dispositivos IoT para la captura de variables del vehículo, tecnologías de comunicación inalámbrica y un sistema de procesamiento, almacenamiento y visualización de los datos. La evaluación experimental permitió comparar el desempeño de distintas tecnologías de comunicación en condiciones reales de prueba e identificar sus ventajas y limitaciones para el escenario de monitoreo propuesto.

A partir de los resultados obtenidos con equipos reales en la ruta de pruebas, resumidos en la Tabla 5.1, se observa que los prototipos 4G/LTE y el híbrido WiFi-LTE presentaron los mejores desempeños globales. El prototipo WiFi-LTE registró la menor pérdida de paquetes, con un 0 %, mientras que el 4G/LTE presentó una pérdida del 5 %. En cuanto a la intensidad de señal (RSSI), ambos prototipos registraron un promedio de $-51{,}44\,\mathrm{dBm}$, lo que refleja niveles de señal similares durante el trayecto evaluado. Por otra parte, el prototipo 4G/LTE presentó el menor promedio de latencia, con $479{,}16\,\mathrm{ms}$ y una mediana de $437{,}50\,\mathrm{ms}$. En contraste, el prototipo híbrido WiFi-LTE registró un promedio de $3311{,}82\,\mathrm{ms}$ y una mediana de $724\,\mathrm{ms}$, lo que evidencia una diferencia considerable entre ambas medidas.

Por su parte, LoRa presentó la mayor pérdida de paquetes, alcanzando un 30,53 %, junto con los niveles de RSSI más bajos, con un promedio de $-87{,}93\,\mathrm{dBm}$. Si bien obtuvo la segunda menor latencia promedio, con $552{,}67\,\mathrm{ms}$ y una mediana de $526{,}00\,\mathrm{ms}$, esta ventaja no compensa la elevada pérdida de paquetes y los bajos niveles de señal registrados. En consecuencia, la implementación LoRa punto a punto presentó el peor desempeño global entre los prototipos evaluados. Estos resultados indican que, bajo las condiciones específicas de despliegue del gateway y de configuración utilizadas, esta implementación no resulta adecuada para aplicaciones que requieran una transmisión de telemetría confiable y con baja tolerancia a la pérdida de paquetes.

En conjunto, los resultados indican que las comunicaciones WiFi-LTE y 4G/LTE constituyen las alternativas más favorables para la transmisión de telemetría en tiempo casi real dentro de las condiciones evaluadas. Ambos prototipos presentan ventajas que podrían aprovecharse en una implementación real, dependiendo de los requisitos operativos del sistema. Por un lado, la solución WiFi-LTE permite aprovechar la conexión a Internet del teléfono del usuario como gateway, evitando la necesidad de incorporar una conexión móvil independiente en el dispositivo IoT. Por otro lado, la solución 4G/LTE, aunque requiere un plan de datos móviles para el dispositivo, ofrece una mayor independencia operativa al no depender de un teléfono externo para transmitir los datos.

En contraste, los resultados obtenidos con LoRa muestran que su implementación requiere un diseño de red más cuidadoso, particularmente en aspectos como la ubicación de los gateways y la configuración de los parámetros de radio. Por tanto, resulta necesario evaluar otras configuraciones y arquitecturas antes de determinar su idoneidad para los requisitos de desempeño del sistema propuesto.

Como trabajo futuro, se proponen cuatro líneas de desarrollo. En primer lugar, implementar LoRaWAN en reemplazo del esquema LoRa punto a punto, con el propósito de evaluar una arquitectura de red que incorpore uno o más gateways y analizar su desempeño bajo distintas condiciones de despliegue. En segundo lugar, realizar pruebas con distintos *Spread Factors* para determinar su efecto sobre el alcance, la latencia y

la pérdida de paquetes. En tercer lugar, integrar el dispositivo desarrollado al circuito eléctrico del scooter, etapa que no pudo abordarse en este trabajo debido a la disponibilidad limitada del vehículo y a la falta de acceso a su batería. Finalmente, comparar el consumo energético de las distintas tecnologías de comunicación implementadas, con el fin de evaluar su impacto en la autonomía del sistema y en los requerimientos de mantenimiento.

Apéndice

Códigos

A continuación se adjunta el código fuente utilizado en este proyecto:

Firmware del transmisor WiFi

```
#include <WiFi.h>
#include Arduino.h
#include <HardwareSerial.h>
#include <TinyGPS++.h>
#include <Wire.h>
#include <PubSubClient.h>
#include <Adafruit_INA219.h>
#include <MPU9250.h>

#include <time.h>
#include <sys/time.h>


// ===== WiFi =====

const char* ssid = HUAWEI_Y8s;
const char* password = jubiloso123;

```

```
19
20 // ===== MQTT =====
21 const char *mqtt_broker = broker.emqx.io;
22 const int port = 1883;
23
24
25 const char *publish_topic_1 = scootnet_PMM/01/IoT/WiFi;
26
27
28 // ===== Objetos =====
29 WiFiClient espClient;
30 PubSubClient client(espClient);
31
32 // ===== Variables =====
33 // I2C pins
34 const int I2C_SDA = 21;
35 const int I2C_SCL = 22;
36
37 // INA219
38 Adafruit_INA219 ina219;
39 float busvoltage;
40
41 // Variables acelermetro
42 MPU9250 mpu;
43 // Variables para guardar offset (calibracin)
44 float accelBiasX = 0, accelBiasY = 0, accelBiasZ = 0;
45 float aX, aY, aZ;
46
47 // Variables GPS
48 HardwareSerial gpsSerial(2);
49 TinyGPSPlus gps;
50 //Polyline mypolyline;
51
52
```

```
// ID del scooter
uint8_t scooterID = 1; // prototipo funcional, SC_01. Para otros:
    2, 3, ..., 100
String clientID = ESP32_SC_ + String(scooterID);
// Contador de paquetes
int txNumber = 0;

//Variable para TimeStamp
unsigned long currentTime;

//Variables y funciones para calcular SOC de bateras de 4.2V (Litio
    ) y 9V (alcalina)
typedef struct {
    float voltage;   // Voltaje en V
    float soc;       // SOC en %
} BatteryPoint;

// ==== NTP ====
const char* ntpServer = pool.ntp.org;
const long gmtOffset_sec = 0;
const int daylightOffset_sec = 0;


// Tabla de referencia NCR18650B 4.2V
#define NUM_POINTS_4V 19
BatteryPoint batteryTable4V[NUM_POINTS_4V] = {
    {4.20, 100}, {4.15, 95}, {4.10, 90}, {4.00, 80}, {3.90, 70},
    {3.85, 65}, {3.80, 60}, {3.75, 55}, {3.70, 50}, {3.65, 45},
    {3.60, 40}, {3.55, 35}, {3.50, 30}, {3.45, 25}, {3.40, 20},
    {3.35, 15}, {3.30, 10}, {3.20, 5}, {3.00, 0}
};

// Tabla de referencia para alcalinas 9V
#define NUM_POINTS_9V 11
```

```
BatteryPoint batteryTable9V[NUM_POINTS_9V] = {
    {9.00, 100}, {8.80, 90}, {8.60, 80}, {8.40, 70}, {8.20, 60},
    {8.00, 50},  {7.80, 40}, {7.60, 30}, {7.40, 20}, {7.20, 10},
    {7.00, 0}    //  7.00  0 %
};
//LA BATERIA A USAR SE SELECCIONA EN EL LOOP, AL LLAMAR LA FUNCION




// Funcin calculo de SOC que recibe cualquier tabla de batera
float getBatterySOC(float voltage, BatteryPoint batteryTable[], int
    numPoints) {
    // Ajustar voltaje fuera de rango
    if (voltage >= batteryTable[0].voltage) return 100.0;
    if (voltage <= batteryTable[numPoints - 1].voltage) return 0.0;

    // Buscar el segmento donde est el voltaje
    for (int i = 0; i < numPoints - 1; i++) {
        float v1 = batteryTable[i].voltage;
        float v2 = batteryTable[i+1].voltage;
        float soc1 = batteryTable[i].soc;
        float soc2 = batteryTable[i+1].soc;

        if (voltage <= v1 && voltage >= v2) {
            // Interpolacin lineal
            float ratio = (voltage - v2) / (v1 - v2);
            return soc2 + ratio * (soc1 - soc2);
        }
    }

    return 0.0; // fallback
}

```

```
118 //******** FUNCION PARA CALIBRAR ACELEROMETRO (CALCULO DE OFFSETS
        PARA EL REPOSO)********
119 void calibrarAcelerometro() {
120   const int N = 200;  // nmero de muestras para promediar
121   float sumX = 0, sumY = 0, sumZ = 0;
122
123   Serial.println(Por favor, deja el sensor quieto en una superficie
        plana...);
124
125   for (int i = 0; i < N; i++) {
126     mpu.update();
127     sumX += mpu.getAccX();
128     sumY += mpu.getAccY();
129     sumZ += mpu.getAccZ();
130     delay(20);
131   }
132
133   accelBiasX = sumX / N;
134   accelBiasY = sumY / N;
135   // En reposo el eje Z mide ~1g hacia arriba
136   accelBiasZ = (sumZ / N) - 1.0;
137
138   Serial.println(Offsets calculados:);
139   Serial.print(X bias: ); Serial.println(accelBiasX, 3);
140   Serial.print(Y bias: ); Serial.println(accelBiasY, 3);
141   Serial.print(Z bias: ); Serial.println(accelBiasZ, 3);
142 }
143
144
145
146
147
148 // ===== FUNCIONES =====
149 void startWifi() {
```

```
150     Serial.println(Conectando a WiFi...);
151     WiFi.begin(ssid, password);
152 
153     while (WiFi.status() != WL_CONNECTED) {
154         delay(500);
155         Serial.print(.);
156     }
157 
158     Serial.println(\nWiFi conectado);
159     Serial.print(IP: );
160     Serial.println(WiFi.localIP());
161     Serial.print(RSSI: );
162     Serial.println(WiFi.RSSI());
163 }
164 
165 void reconnectMQTT() {
166     while (!client.connected()) {
167         Serial.println(\nIntentando conectar a MQTT...);
168 
169         if (client.connect(clientID.c_str())) {
170             Serial.println( MQTT conectado correctamente);
171         } else {
172             Serial.print( Error MQTT, rc=);
173             Serial.println(client.state());
174 
175             Serial.println(Posibles errores:);
176             Serial.println(-2 = Fallo conexin TCP);
177             Serial.println(-1 = Desconectado);
178             Serial.println(1 = Protocolo incorrecto);
179             Serial.println(2 = ID rechazado);
180             Serial.println(3 = Broker no disponible);
181             Serial.println(4 = Usuario/contrasea incorrectos);
182             Serial.println(5 = No autorizado);
183 
```

```
            delay(2000);
        }
    }
}

void syncTimeWithNTP() {
    Serial.print(Sincronizando NTP...);
    configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);

    struct tm timeinfo;
    while (!getLocalTime(&timeinfo)) {
        Serial.print(.);
        delay(500);
    }

    Serial.println( OK);
}

long long getEpochTimeMs() {
    struct timeval tv;
    gettimeofday(&tv, NULL);
    return ((long long)tv.tv_sec * 1000LL) + (tv.tv_usec / 1000);
}



void setup() {
    Serial.begin(115200);
    delay(1000);
    Serial.println(=== INICIANDO SISTEMA ===);


    //Iniciamos GPS
    gpsSerial.begin(9600, SERIAL_8N1, 16, 17);
```

```
218     Serial.println(Esperando fix de GPS...);
219 
220     //Iniciamos bus I2C
221     Wire.begin(I2C_SDA, I2C_SCL);
222 
223     //Inicializar MPU 9250
224     Serial.println(Inicializando MPU9250...);
225     if (!mpu.setup(0x68)) {  // Direccin I2C por defecto
226         Serial.println(Error: no se detecta MPU9250.);
227         Wire.end();// Si no se reconoce reinicia el bus I2C
228         delay(10);//
229         Wire.begin(21, 22);//
230         //while (1);
231     }
232     delay(1000);
233     Serial.println(Calibrando acelermetro...);
234     calibrarAcelerometro();
235     Serial.println(Calibracin completada.);
236 
237     // Inicializar INA219
238     if (!ina219.begin()) {
239         Serial.println(No se detect INA219, revisa conexiones!);
240         // Si no se reconoce reinicia el bus I2C
241         Wire.end();
242         delay(10);
243         Wire.begin(21, 22);
244     }
245 
246     startWifi();          //Iniciamos WiFi
247     syncTimeWithNTP();    //Sincronizamos tiempo NTP
248     client.setServer(mqtt_broker, port);
249     reconnectMQTT();
250 }
251 
```

```
252 void loop() {
253
254     if (!client.connected()) {
255         reconnectMQTT();
256     }
257     client.loop(); // <-- Siempre ejecuta esto, aunque no ests
    enviando datos
258     // Leer GPS
259     while (gpsSerial.available() > 0) {   //Actualizamos el GPS
    fuera del timer, para evitar que se congele.
260         char c = gpsSerial.read();
261         gps.encode(c);
262         Serial.write(c);   // opcional, solo para depurar
263     }
264
265     // Temporizador de envio
266     static unsigned long lastSend = 0;
267     if (millis() - lastSend > 6000) {
268         lastSend = millis();
269
270         // Leer Acelermetro
271         if (mpu.update()) {
272         aX = mpu.getAccX() - accelBiasX;
273         aY = mpu.getAccY() - accelBiasY;
274         aZ = mpu.getAccZ() - accelBiasZ;
275         }
276
277         // Leemos INA 219 y calculamos SOC
278         busvoltage = ina219.getBusVoltage_V();
279         float soc = getBatterySOC(busvoltage, batteryTable9V,
    NUM_POINTS_9V); // Para litio solo cambiar 9V por 4V en los
    argumentos que salen
280         int socInt = (int)soc;
281
```

```
282
283         // ===== Estado GPS =====
284         bool gps_ok = gps.location.isValid();
285
286         // Actualizar TimeStamp (usamos ms)
287         long long timestamp = getEpochTimeMs();
288
289         // ===== Debug  GPS=====
290         Serial.print(Chars recibidos: );
291         Serial.println(gps.charsProcessed());
292
293         Serial.print(Satelites: );
294         Serial.println(gps.satellites.value());
295
296         Serial.print(HDOP: );
297         Serial.println(gps.hdop.value());
298
299         Serial.print(Location valid: );
300         Serial.println(gps.location.isValid());
301
302         Serial.print(Location updated: );
303         Serial.println(gps.location.isUpdated());
304         if (gps_ok) {
305             Serial.print(FIX OBTENIDO! Latitud: );
306             Serial.print(gps.location.lat(), 5);
307             Serial.print( | Longitud: );
308             Serial.println(gps.location.lng(), 5);
309         } else {
310             Serial.println(Sin fix todava...);
311         }
312         // ===== Variables seguras =====
313         float lat = gps_ok ? gps.location.lat() : 0.0;
314         float lon = gps_ok ? gps.location.lng() : 0.0;
315
```

```
316
317         //  FORMATO CORRECTO PARA THINGSPEAK MQTT **
318         // Para IoT Channel (3334965)
319         String payload_IoT = {;
320             //payload_IoT += \id\: + String(scooterID) + ,;
321             payload_IoT += \timestamp\: + String(timestamp) + ,;
322             payload_IoT += \gps_ok\: + String(gps_ok ? 1 : 0) + ,;
    // gps ok es booleano
323             payload_IoT += \lat\: + String(lat, 6) + ,;
324             payload_IoT += \lon\: + String(lon, 6) + ,;
325             payload_IoT += \ax\: + String(aX, 3) + ,;
326             payload_IoT += \ay\: + String(aY, 3) + ,;
327             payload_IoT += \az\: + String(aZ, 3) + ,;
328             payload_IoT += \soc\: + String(socInt) + ,;
329             payload_IoT += \Tx_number\: + String(txNumber+1) + ,;
330             payload_IoT += \rssi\: + String(WiFi.RSSI());
331             payload_IoT += };
332
333
334
335
336
337         // Calcular tamao del payload
338         int payloadSize = payload_IoT.length();
339
340         Serial.print(Tamao del payload: );
341         Serial.print(payloadSize);
342         Serial.println( bytes);
343
344         if (client.publish(publish_topic_1, payload_IoT.c_str())) {
345             Serial.println( Datos enviados al broker publico MQTT);
346             Serial.println(Topic:  + String(publish_topic_1));
347             Serial.println(Payload:  + payload_IoT);
348
```

```
            Serial.print(Bus voltage raw: );
            Serial.println(ina219.getBusVoltage_V(), 3);
            Serial.print(Shunt voltage: );
            Serial.println(ina219.getShuntVoltage_mV(), 3);
            Serial.print(socInt: );
            Serial.println(socInt);
        } else {
            Serial.println( Error enviando datos payolad_IoT MQTT);
        }

        txNumber++;
    }

    delay(100);
}
```

Firmware del transmisor LoRa Heltec

```
%codigo
/* Heltec P2P transmitter
   Enva:
   I,scooterID,timestamp,lat,lon,ax,ay,az,soc
   R,timestamp,lat,lon,txNumber
*/

#include <SPI.h>
#include <LoRa.h>
#include <HardwareSerial.h>
#include <TinyGPS++.h>
#include <Wire.h>
#include <MPU9250_asukiaaa.h>
#include <time.h>
#include <sys/time.h>
#include <WiFi.h>
#include <Arduino.h>
#include <Adafruit_INA219.h>

// ================ LoRa ================
#define LORA_SS    18
#define LORA_RST   14
#define LORA_DIO0  26

// ================ GPS ================
#define TX_GPS_PIN 16
#define RX_GPS_PIN 17
#define BAUD_RATE 9600

// ================ WiFi ================
/*
const char* ssid = VTR-6216549;
```

```
const char* password = n4rdVvqptnkf;
*/

const char* ssid = HUAWEI_Y8s;           //
const char* password = jubiloso123;

// ================ NTP ================
const long gmtOffset_sec = 0;
const int daylightOffset_sec = 0;

// ================ I2C ================
const int I2C_SDA = 21;
const int I2C_SCL = 22;

// ================ Sensores ================
MPU9250_asukiaaa mySensor;

float aX, aY, aZ, aSqrt;

// GPS
HardwareSerial gpsSerial(2);
TinyGPSPlus gps;

// INA219
Adafruit_INA219 ina219;

float shuntvoltage;
float busvoltage;
float current_mA;
float loadvoltage;
float power_mW;

// ================ Datos ================
int txNumber = 0;
```

```
67
68  bool ntpSynced = false;
69  unsigned long lastNtpRetry = 0;
70
71  // ================ Scooter ================
72  uint8_t scooterID = 2;
73
74  String clientID = ESP32_SC_ + String(scooterID);
75
76  // =======================================================
77  // ==================== BATERIAS =========================
78  // =======================================================
79
80  typedef struct {
81      float voltage;
82      float soc;
83  } BatteryPoint;
84
85  // 4.2V Litio
86  #define NUM_POINTS_4V 19
87
88  BatteryPoint batteryTable4V[NUM_POINTS_4V] = {
89      {4.20, 100}, {4.15, 95}, {4.10, 90}, {4.00, 80},
90      {3.90, 70}, {3.85, 65}, {3.80, 60}, {3.75, 55},
91      {3.70, 50}, {3.65, 45}, {3.60, 40}, {3.55, 35},
92      {3.50, 30}, {3.45, 25}, {3.40, 20}, {3.35, 15},
93      {3.30, 10}, {3.20, 5}, {3.00, 0}
94  };
95
96  // 9V Alcalina
97  #define NUM_POINTS_9V 11
98
99  BatteryPoint batteryTable9V[NUM_POINTS_9V] = {
100     {9.00, 100}, {8.80, 90}, {8.60, 80}, {8.40, 70},
```

```
    {8.20, 60}, {8.00, 50}, {7.80, 40}, {7.60, 30},
    {7.40, 20}, {7.20, 10}, {7.00, 0}
};

// ======================================================
// ==================== FUNCIONES =======================
// ======================================================

float getBatterySOC(float voltage, BatteryPoint batteryTable[], int
    numPoints) {

    if (voltage >= batteryTable[0].voltage) return 100.0;

    if (voltage <= batteryTable[numPoints - 1].voltage) return 0.0;

    for (int i = 0; i < numPoints - 1; i++) {

        float v1 = batteryTable[i].voltage;
        float v2 = batteryTable[i + 1].voltage;

        float soc1 = batteryTable[i].soc;
        float soc2 = batteryTable[i + 1].soc;

        if (voltage <= v1 && voltage >= v2) {

            float ratio = (voltage - v2) / (v1 - v2);

            return soc2 + ratio * (soc1 - soc2);
        }
    }

    return 0.0;
}

```

```
// ======================================================

bool syncTimeWithNTP() {

    Serial.println(Iniciando sincronizacion NTP...);

    WiFi.setSleep(false);

    configTime(
        gmtOffset_sec,
        daylightOffset_sec,
        time.google.com,
        pool.ntp.org,
        time.windows.com
    );

    struct tm timeinfo;

    int retries = 0;
    const int maxRetries = 30;

    while (!getLocalTime(&timeinfo) && retries < maxRetries) {

        Serial.print(.);

        delay(1000);

        retries++;
    }

    if (retries >= maxRetries) {

        Serial.println(\nERROR: no se pudo sincronizar NTP);

```

```
168         return false;
169     }
170 
171     Serial.println(\nTiempo sincronizado correctamente);
172 
173     Serial.print(Hora UTC: );
174 
175     Serial.println(&timeinfo, %Y-%m-%d %H:%M:%S);
176 
177     return true;
178 }
179 
180 // ======================================================
181 
182 long long getEpochTimeMs() {
183 
184     struct timeval tv;
185 
186     gettimeofday(&tv, NULL);
187 
188     return ((long long)tv.tv_sec * 1000LL) + (tv.tv_usec / 1000);
189 }
190 
191 // ======================================================
192 
193 void startWifi() {
194 
195     Serial.print(Conectando a WiFi);
196 
197     WiFi.mode(WIFI_STA);
198 
199     WiFi.begin(ssid, password);
200 
201     unsigned long startTime = millis();
```

```
202
203     while (WiFi.status() != WL_CONNECTED &&
204            millis() - startTime < 30000) {
205
206         delay(500);
207
208         Serial.print(.);
209     }
210
211     if (WiFi.status() == WL_CONNECTED) {
212
213         Serial.println(\nWiFi conectado);
214
215         Serial.print(IP: );
216         Serial.println(WiFi.localIP());
217
218         Serial.print(Gateway: );
219         Serial.println(WiFi.gatewayIP());
220
221         Serial.print(DNS: );
222         Serial.println(WiFi.dnsIP());
223
224         Serial.print(RSSI: );
225         Serial.println(WiFi.RSSI());
226
227     } else {
228
229         Serial.println(\nERROR WiFi);
230     }
231 }
232
233 // ======================================================
234 // ====================== SETUP =========================
235 // ======================================================
```

```
236
237 void setup() {
238
239     Serial.begin(115200);
240
241     delay(1000);
242
243     // ================= LoRa =================
244
245     LoRa.setPins(LORA_SS, LORA_RST, LORA_DIO0);
246
247     int retry = 0;
248
249     const int MAX_RETRY = 5;
250
251     while (!LoRa.begin(915E6) && retry < MAX_RETRY) {
252
253         Serial.println(Fallo LoRa, reintentando...);
254
255         retry++;
256
257         delay(2000);
258     }
259
260     if (retry == MAX_RETRY) {
261
262         Serial.println(ERROR LoRa);
263
264     } else {
265
266         Serial.println(LoRa OK);
267
268         LoRa.setTxPower(17);
269
```

```
270         LoRa.setSpreadingFactor(7);
271
272         LoRa.setSignalBandwidth(125E3);
273
274         LoRa.enableCrc();
275     }
276
277     // ================= I2C =================
278
279     Wire.begin(I2C_SDA, I2C_SCL);
280
281     // ================= MPU9250 =================
282
283     mySensor.setWire(&Wire);
284
285     mySensor.beginAccel();
286
287     // ================= INA219 =================
288
289     if (!ina219.begin()) {
290
291         Serial.println(INA219 NO detectado);
292
293     } else {
294
295         Serial.println(INA219 OK);
296     }
297
298     // ================= GPS =================
299     // OJO: begin(rx, tx)
300
301     gpsSerial.begin(
302         BAUD_RATE,
303         SERIAL_8N1,
```

```
304         RX_GPS_PIN,
305         TX_GPS_PIN
306     );
307 
308     Serial.println(GPS iniciado);
309 
310     // ================= WiFi =================
311 
312     startWifi();
313 
314     delay(3000);
315 
316     // ================= NTP =================
317 
318     ntpSynced = syncTimeWithNTP();
319 }
320 
321 // ======================================================
322 // ====================== LOOP ======================
323 // ======================================================
324 
325 void loop() {
326 
327     // ================================================
328     // Reintento NTP
329     // ================================================
330 
331     if (!ntpSynced &&
332         millis() - lastNtpRetry > 30000) {
333 
334         lastNtpRetry = millis();
335 
336         Serial.println(Reintentando NTP...);
337 
```

```
338         ntpSynced = syncTimeWithNTP();
339     }
340 
341     // ================================================
342     // Alimentar parser GPS SIEMPRE
343     // ================================================
344 
345     while (gpsSerial.available()) {
346 
347         gps.encode(gpsSerial.read());
348     }
349 
350     // ================================================
351     // Envio cada 15 segundos
352     // ================================================
353 
354     static unsigned long lastSend = 0;
355 
356     if (millis() - lastSend >= 15000) {
357 
358         lastSend = millis();
359 
360         // ================ MPU9250 ================
361 
362         mySensor.accelUpdate();
363 
364         aX = mySensor.accelX();
365 
366         aY = mySensor.accelY();
367 
368         aZ = mySensor.accelZ();
369 
370         aSqrt = mySensor.accelSqrt();
371 
```

```
372         // ================= INA219 =================
373
374         shuntvoltage = ina219.getShuntVoltage_mV();
375
376         busvoltage = ina219.getBusVoltage_V();
377
378         current_mA = ina219.getCurrent_mA();
379
380         power_mW = ina219.getPower_mW();
381
382         loadvoltage = busvoltage + (shuntvoltage / 1000);
383
384         // ================= SOC =================
385
386         float soc = getBatterySOC(
387             busvoltage,
388             batteryTable9V,
389             NUM_POINTS_9V
390         );
391
392         int socInt = (int)soc;
393
394         // ================= Tiempo =================
395
396         long long epochTime = getEpochTimeMs();
397
398         // ================= GPS =================
399
400         bool gps_ok = gps.location.isValid();
401
402         // ===== Debug  GPS=====
403         if (gps_ok) {
404             Serial.print(FIX OBTENIDO! Latitud: );
405             Serial.print(gps.location.lat(), 5);
```

```
406             Serial.print( | Longitud: );
407             Serial.println(gps.location.lng(), 5);
408         } else {
409             Serial.println(Sin fix todava...);
410         }
411         // ===== Variables seguras =====
412         float lat = gps_ok ? gps.location.lat() : 0.0;
413         float lon = gps_ok ? gps.location.lng() : 0.0;
414 
415         // ================================================
416         // PAYLOAD IOT
417         // ================================================
418 
419         String payload_IoT = ;
420 
421         payload_IoT += String(scooterID);
422         payload_IoT += ,;
423         payload_IoT += String(epochTime);
424         payload_IoT += ,;
425         payload_IoT += String(gps_ok ? 1 : 0);
426         payload_IoT += ,;
427 
428         payload_IoT += String(lat, 6);
429         payload_IoT += ,;
430         payload_IoT += String(lon, 6);
431         payload_IoT += ,;
432 
433         payload_IoT += String(aX, 2);
434         payload_IoT += ,;
435         payload_IoT += String(aY, 2);
436         payload_IoT += ,;
437         payload_IoT += String(aZ, 2);
438         payload_IoT += ,;
439 
```

```
        payload_IoT += String(socInt);
        payload_IoT += ,;
        payload_IoT += String(txNumber + 1);

        // ================ Envio LoRa ================

        LoRa.beginPacket();
        LoRa.print(payload_IoT);
        LoRa.endPacket();
        delay(50);

        Serial.println(payload_IoT);
        Serial.print(Bytes payload: );
        Serial.println(payload_IoT.length());

        // Aumentar contador de paquetes
        txNumber++;
    }
}
```

Firmware de receptor LoRa Raspberry

```
%codigo raspi
/*
    ***************************************************************************

 *
 * Copyright (c) 2018 Dragino
 *
 * http://www.dragino.com
 *

```

```
    ********************************************************************************
    */

#include <string.h> //Ver si se usa esto o se hace con char y
    punteros
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <sys/time.h>
#include <signal.h>
#include <stdlib.h>
#include <unistd.h>       // Para sleep()
#include <stdbool.h>      // Para bool, true, false

#include <sys/ioctl.h>

#include <wiringPi.h>
#include <wiringPiSPI.h>


#include MQTTClient.h
#define ADDRESS     broker.emqx.io
#define PORT 1883


#define CLIENTID    ProtoLoRa_pi3_01k
#define TOPIC      scootnet_PMM/02/IoT/LoRa
#define QOS         1
#define TIMEOUT     10000L
```

```
// =========Variables de conexion MQTT
MQTTClient client;
MQTTClient_connectOptions conn_opts =
    MQTTClient_connectOptions_initializer;

//=====Variables auxiliares para los datos
uint64_t Last_Time_Stamp = 0; //Buffer de 1 dato para el ultimo
    timestamp recibido
int rssi_lora;
char payloadWithRSSI[256];  // Buffer global para concatenar el
    RSSI


/* ################# CONFIGURACION DEL LORA ################ */
#define REG_FIFO                    0x00
#define REG_OPMODE                  0x01
#define REG_FIFO_ADDR_PTR           0x0D
#define REG_FIFO_TX_BASE_AD         0x0E
#define REG_FIFO_RX_BASE_AD         0x0F
#define REG_RX_NB_BYTES             0x13
#define REG_FIFO_RX_CURRENT_ADDR    0x10
#define REG_IRQ_FLAGS               0x12
#define REG_DIO_MAPPING_1           0x40
#define REG_DIO_MAPPING_2           0x41
#define REG_MODEM_CONFIG            0x1D
#define REG_MODEM_CONFIG2           0x1E
#define REG_MODEM_CONFIG3           0x26
#define REG_SYMB_TIMEOUT_LSB      0x1F
#define REG_PKT_SNR_VALUE     0x19
#define REG_PAYLOAD_LENGTH          0x22
#define REG_IRQ_FLAGS_MASK          0x11
#define REG_MAX_PAYLOAD_LENGTH    0x23
#define REG_HOP_PERIOD              0x24
#define REG_SYNC_WORD       0x39
```

```
#define REG_VERSION           0x42

#define PAYLOAD_LENGTH              0x40

#define REG_LNA                     0x0C
#define LNA_MAX_GAIN                0x23
#define LNA_OFF_GAIN                0x00
#define LNA_LOW_GAIN          0x20

#define RegDioMapping1               0x40
#define RegDioMapping2               0x41
#define RegPaConfig                  0x09
#define RegPaRamp                    0x0A
#define RegPaDac                     0x5A

#define SX72_MC2_FSK                0x00
#define SX72_MC2_SF7                0x70
#define SX72_MC2_SF8                0x80
#define SX72_MC2_SF9                0x90
#define SX72_MC2_SF10               0xA0
#define SX72_MC2_SF11               0xB0
#define SX72_MC2_SF12               0xC0

#define SX72_MC1_LOW_DATA_RATE_OPTIMIZE  0x01

#define SX1276_MC1_BW_125                0x70
#define SX1276_MC1_BW_250                0x80
#define SX1276_MC1_BW_500                0x90
#define SX1276_MC1_CR_4_5            0x02
#define SX1276_MC1_CR_4_6            0x04
#define SX1276_MC1_CR_4_7            0x06
#define SX1276_MC1_CR_4_8            0x08
#define SX1276_MC1_IMPLICIT_HEADER_MODE_ON    0x01
#define SX1276_MC2_RX_PAYLOAD_CRCON        0x04
```

```
105 #define SX1276_MC3_LOW_DATA_RATE_OPTIMIZE  0x08
106 #define SX1276_MC3_AGCAUTO                 0x04
107
108 #define LORA_MAC_PREAMBLE                  0x34
109
110 #define RXLORA_RXMODE_RSSI_REG_MODEM_CONFIG1 0x0A
111 #ifdef LMIC_SX1276
112 #define RXLORA_RXMODE_RSSI_REG_MODEM_CONFIG2 0x70
113 #elif LMIC_SX1272
114 #define RXLORA_RXMODE_RSSI_REG_MODEM_CONFIG2 0x74
115 #endif
116
117 #define REG_FRF_MSB                 0x06
118 #define REG_FRF_MID                 0x07
119 #define REG_FRF_LSB                 0x08
120 #define FRF_MSB                      0xD9
121 #define FRF_MID                      0x06
122 #define FRF_LSB                      0x66
123
124 #define OPMODE_LORA      0x80
125 #define OPMODE_MASK      0x07
126 #define OPMODE_SLEEP     0x00
127 #define OPMODE_STANDBY   0x01
128 #define OPMODE_FSTX      0x02
129 #define OPMODE_TX        0x03
130 #define OPMODE_FSRX      0x04
131 #define OPMODE_RX        0x05
132 #define OPMODE_RX_SINGLE 0x06
133 #define OPMODE_CAD       0x07
134
135 #define IRQ_LORA_RXTOUT_MASK 0x80
136 #define IRQ_LORA_RXDONE_MASK 0x40
137 #define IRQ_LORA_CRCERR_MASK 0x20
138 #define IRQ_LORA_HEADER_MASK 0x10
```

```
#define IRQ_LORA_TXDONE_MASK 0x08
#define IRQ_LORA_CDDONE_MASK 0x04
#define IRQ_LORA_FHSSCH_MASK 0x02
#define IRQ_LORA_CDDETD_MASK 0x01

#define MAP_DIO0_LORA_RXDONE   0x00
#define MAP_DIO0_LORA_TXDONE   0x40
#define MAP_DIO1_LORA_RXTOUT   0x00
#define MAP_DIO1_LORA_NOP      0x30
#define MAP_DIO2_LORA_NOP      0xC0

typedef unsigned char byte;
float SNR_LoRa;

static const int CHANNEL = 0;

char message[128];  //Lo aumentamos de 50 a 128
bool sx1272 = true;
byte receivedbytes;

enum sf_t { SF7=7, SF8=8, SF9, SF10, SF11, SF12 };

int ssPin = 6;
int dio0  = 7;
int RST   = 0;
sf_t sf = SF7;
uint32_t  freq = 915E6;
byte hello[32] = HELLO;




/* ################# FUNCIONES ################# */
```

```
173
174 void die(const char *s) {
175     perror(s);
176     exit(1);
177 }
178
179 void selectreceiver() { digitalWrite(ssPin, LOW); }
180 void unselectreceiver() { digitalWrite(ssPin, HIGH); }
181
182 byte readReg(byte addr) {
183     unsigned char spibuf[2];
184     selectreceiver();
185     spibuf[0] = addr & 0x7F;
186     spibuf[1] = 0x00;
187     wiringPiSPIDataRW(CHANNEL, spibuf, 2);
188     unselectreceiver();
189     return spibuf[1];
190 }
191
192 void writeReg(byte addr, byte value) {
193     unsigned char spibuf[2];
194     spibuf[0] = addr | 0x80;
195     spibuf[1] = value;
196     selectreceiver();
197     wiringPiSPIDataRW(CHANNEL, spibuf, 2);
198     unselectreceiver();
199 }
200
201 static void opmode (uint8_t mode) {
202     writeReg(REG_OPMODE, (readReg(REG_OPMODE) & ~OPMODE_MASK) |
    mode);
203 }
204
205 static void opmodeLora() {
```

```
206     uint8_t u = OPMODE_LORA;
207     if (!sx1272) u |= 0x8;
208     writeReg(REG_OPMODE, u);
209 }
210 
211 void SetupLoRa() {
212     digitalWrite(RST, HIGH);
213     delay(100);
214     digitalWrite(RST, LOW);
215     delay(100);
216 
217     byte version = readReg(REG_VERSION);
218     if (version == 0x22) {
219         printf(SX1272 detected, starting.\n);
220         sx1272 = true;
221     } else {
222         digitalWrite(RST, LOW);
223         delay(100);
224         digitalWrite(RST, HIGH);
225         delay(100);
226         version = readReg(REG_VERSION);
227         if (version == 0x12) {
228             printf(SX1276 detected, starting.\n);
229             sx1272 = false;
230         } else {
231             printf(Unrecognized transceiver.\n);
232             exit(1);
233         }
234     }
235 
236     opmode(OPMODE_SLEEP);
237     uint64_t frf = ((uint64_t)freq << 19) / 32000000;
238     writeReg(REG_FRF_MSB, (uint8_t)(frf>>16));
239     writeReg(REG_FRF_MID, (uint8_t)(frf>>8));
```

```
240     writeReg(REG_FRF_LSB, (uint8_t)(frf>>0));
241     writeReg(REG_SYNC_WORD, 0x34);
242 
243     if (sx1272) {
244         if (sf == SF11 || sf == SF12) writeReg(REG_MODEM_CONFIG,0
    x0B);
245         else writeReg(REG_MODEM_CONFIG,0x0A);
246         writeReg(REG_MODEM_CONFIG2,(sf<<4) | 0x04);
247     } else {
248         if (sf == SF11 || sf == SF12) writeReg(REG_MODEM_CONFIG3,0
    x0C);
249         else writeReg(REG_MODEM_CONFIG3,0x04);
250         writeReg(REG_MODEM_CONFIG,0x72);
251         writeReg(REG_MODEM_CONFIG2,(sf<<4) | 0x04);
252     }
253 
254     if (sf == SF10 || sf == SF11 || sf == SF12) writeReg(
    REG_SYMB_TIMEOUT_LSB,0x05);
255     else writeReg(REG_SYMB_TIMEOUT_LSB,0x08);
256 
257     writeReg(REG_MAX_PAYLOAD_LENGTH,0x80);
258     writeReg(REG_PAYLOAD_LENGTH,PAYLOAD_LENGTH);
259     writeReg(REG_HOP_PERIOD,0xFF);
260     writeReg(REG_FIFO_ADDR_PTR, readReg(REG_FIFO_RX_BASE_AD));
261     writeReg(REG_LNA, LNA_MAX_GAIN);
262 }
263 
264 /* ############ MQTT ############ */
265 void setupMQTT() {
266     int rc;
267     MQTTClient_create(&client, ADDRESS, CLIENTID,
    MQTTCLIENT_PERSISTENCE_NONE, NULL);
268     conn_opts.keepAliveInterval = 20;
269     conn_opts.cleansession = 1;
```

```
270     //Agregamos las credenciales del broker
271     //conn_opts.username = USERNAME;
272     //conn_opts.password = PASSWORD;
273 
274     while ((rc = MQTTClient_connect(client, &conn_opts)) !=
    MQTTCLIENT_SUCCESS) {   //En mqttpaho MQTTCLIENT_SUCCESS  es 0 (
    funcion se ejecuta sin error) (MQTTClient.h)
275         printf(Dentro de la funcion Setup_MQTT);
276         printf(Failed to connect, return code %d. Reconnecting in 5
     seconds...\n, rc);
277         delay(5000);
278     }
279     printf(MQTT connected successfully!\n);
280 }
281 
282 void sendToMQTT(char* payload) {
283     // Hacer una copia local del payload para no modificar el
    original
284     char localPayload[128];
285     strncpy(localPayload, payload, sizeof(localPayload) - 1);
286     localPayload[sizeof(localPayload) - 1] = '\0';
287 
288     // Limpiar caracteres no deseados
289     for (size_t i = 0; i < strlen(localPayload); i++) {
290         if (localPayload[i] == '\r' || localPayload[i] == '\n' ||
    localPayload[i] == '\0') {
291             localPayload[i] = '\0';
292             break; // Cortar en el primer caracter no deseado
293         }
294     }
295 
296     MQTTClient_message pubmsg = MQTTClient_message_initializer;
297     MQTTClient_deliveryToken token;
298     int rc;
```

```
299     char primerbyte = localPayload[0];
300     char aux[256];
301
302
303     if(primerbyte == '2'){
304         // Usar snprintf para evitar desbordamientos. Combina
    localpayload y rssi_lora y
305         //guarda el resultado en payloadWithRSSI
306         snprintf(payloadWithRSSI, sizeof(payloadWithRSSI), %s,%d
    ,%.2f,
307                  localPayload, rssi_lora, SNR_LoRa);
308
309         pubmsg.payload = payloadWithRSSI;
310         pubmsg.payloadlen = (int)strlen(payloadWithRSSI);
311         pubmsg.qos = QOS;
312         pubmsg.retained = 0;
313
314         while ((rc = MQTTClient_publishMessage(client, TOPIC, &
    pubmsg, &token)) != MQTTCLIENT_SUCCESS) {
315             printf(Failed to publish message, return code %d.
    Trying to reconnect...\n, rc);
316             while ((rc = MQTTClient_connect(client, &conn_opts)) !=
     MQTTCLIENT_SUCCESS) {
317                 printf(Reconnect failed, return code %d. Retrying
    in 5 seconds...\n, rc);
318                 delay(500);
319             }
320             printf(Reconnected to MQTT broker Rendimiento.\n);
321         }
322     }
323
324     else {
325         printf(Topico desconocido: %c\n, primerbyte);
326         return;
```

```
327     }
328
329     MQTTClient_waitForCompletion(client, token, TIMEOUT);
330
331     // DEBUG: Verificar que se esta enviando
332     printf(Enviado - Tipo: %c, Payload: %s\n, primerbyte,
333            primerbyte == '2' ? payloadWithRSSI : localPayload);
334 }
335
336 /* ############ FUNCIONES DE RECEPCION ############ */
337 bool receive(char *payload) {
338
339     /*
340     //Debugeo
341     printf(\n Entrando a funcion receive() \n)
342     printf(payload)
343     */
344
345     writeReg(REG_IRQ_FLAGS, 0x40);
346     int irqflags = readReg(REG_IRQ_FLAGS);
347     if((irqflags & 0x20) == 0x20) {
348         printf(CRC error\n);
349         writeReg(REG_IRQ_FLAGS, 0x20);
350         return false;
351     } else {
352         byte currentAddr = readReg(REG_FIFO_RX_CURRENT_ADDR);
353         byte receivedCount = readReg(REG_RX_NB_BYTES);
354         receivedbytes = receivedCount;
355         writeReg(REG_FIFO_ADDR_PTR, currentAddr);
356         for(int i = 0; i < receivedCount; i++) //VER QUITAR ESTE
    FOR
357             payload[i] = (char)readReg(REG_FIFO);
358
359         // TERMINADOR STRING
```

```
360         payload[receivedCount] = '\0'; //Agregado al final
361     }
362 
363     return true;
364 }
365 
366 bool receivepacket() {
367 
368     int rssicorr;
369 
370     if(digitalRead(dio0) == 1) {
371 
372         if(receive(message)) {
373 
374             byte value = readReg(REG_PKT_SNR_VALUE);
375 
376             /*
377             if(value & 0x80)
378                 value = ((~value + 1) & 0xFF) >> 2, SNR = -value;
379             else
380                 SNR = (value & 0xFF) >> 2;
381             */
382 
383             SNR_LoRa = ((int8_t)readReg(REG_PKT_SNR_VALUE)) * 0.25;
    //Agregado
384 
385             rssicorr = sx1272 ? 139 : 157;
386 
387             printf(Packet RSSI: %d, RSSI: %d, SNR: %f, Length: %i\
    n,
388                    readReg(0x1A)-rssicorr,
389                    readReg(0x1B)-rssicorr,
390                    SNR_LoRa,
391                    (int)receivedbytes);
```

```
392
393             printf(Payload: %s\n, message);
394
395             int rssiReal = readReg(0x1A) - rssicorr;
396             rssi_lora = rssiReal;
397
398             printf(Aca imprimimos el rssi que sacamos: %d\n,
    rssi_lora);
399
400             // Limpiamos IRQ
401             writeReg(REG_IRQ_FLAGS, IRQ_LORA_RXDONE_MASK);
402
403             // Volver a RX
404             opmode(OPMODE_RX);
405
406             return true;
407         }
408         else {
409             return false;
410         }
411     }
412
413     return false;
414 }
415
416 /* CONFIGURAR POTENCIA */
417 static void configPower (int8_t pw) {
418     if (!sx1272) {
419         if(pw >= 17) pw = 15;
420         else if(pw < 2) pw = 2;
421         writeReg(RegPaConfig, (uint8_t)(0x80|(pw&0xf)));
422         writeReg(RegPaDac, readReg(RegPaDac)|0x4);
423     } else {
424         if(pw > 17) pw = 17;
```

```
425         else if(pw < 2) pw = 2;
426         writeReg(RegPaConfig, (uint8_t)(0x80|(pw-2)));
427     }
428 }
429
430 static void writeBuf(byte addr, byte *value, byte len) {
431     unsigned char spibuf[256];
432     spibuf[0] = addr | 0x80;
433     for (int i = 0; i < len; i++) spibuf[i + 1] = value[i];
434     selectreceiver();
435     wiringPiSPIDataRW(CHANNEL, spibuf, len + 1);
436     unselectreceiver();
437 }
438
439 void txlora(byte *frame, byte datalen) {
440     writeReg(RegDioMapping1, MAP_DIO0_LORA_TXDONE|MAP_DIO1_LORA_NOP
    |MAP_DIO2_LORA_NOP);
441     writeReg(REG_IRQ_FLAGS, 0xFF);
442     writeReg(REG_IRQ_FLAGS_MASK, ~IRQ_LORA_TXDONE_MASK);
443     writeReg(REG_FIFO_TX_BASE_AD, 0x00);
444     writeReg(REG_FIFO_ADDR_PTR, 0x00);
445     writeReg(REG_PAYLOAD_LENGTH, datalen);
446     writeBuf(REG_FIFO, frame, datalen);
447     opmode(OPMODE_TX);
448     printf(send: %s\n, frame);
449 }
450
451 /* ################# MAIN ################# */
452 int main (int argc, char *argv[]) {
453     setvbuf(stdout, NULL, _IONBF, 0);  // <-- Desactiva el buffer
    del printf
454     printf(Debug activado (stdout sin buffer)\n);
455     if (argc < 2) {
456         printf (Usage: argv[0] sender|rec [message]\n);
```

```
457         exit(1);
458     }
459 
460     wiringPiSetup();
461     pinMode(ssPin, OUTPUT);
462     pinMode(dio0, INPUT);
463     pinMode(RST, OUTPUT);
464     wiringPiSPISetup(CHANNEL, 500000);
465 
466     SetupLoRa();
467     setupMQTT();
468 
469     if (!strcmp(sender, argv[1])) {
470         opmodeLora();
471         opmode(OPMODE_STANDBY);
472         writeReg(RegPaRamp, (readReg(RegPaRamp) & 0xF0) | 0x08);
473         configPower(23);
474         printf(Send packets at SF%i on %.6lf Mhz.\n, sf,(double)
    freq/1000000);
475         printf(------------------\n);
476         if (argc > 2) strncpy((char *)hello, argv[2], sizeof(hello)
    );
477         while(1) {
478             txlora(hello, strlen((char *)hello));
479             delay(5000);
480         }
481     } else {
482         opmodeLora();
483         opmode(OPMODE_STANDBY);
484         opmode(OPMODE_RX);
485         printf(Listening at SF%i on %.6lf Mhz.\n, sf,(double)freq
    /1000000);
486         printf(SF=%d\n, sf);
487         printf(MODEM_CONFIG2=0x%02X\n, (sf<<4)|0x04);
```

```
488         fflush(stdout);
489         printf(debuggeando en else if not sender\n);
490         fflush(stdout);
491         printf(------------------\n);
492         while(1) {
493     if (receivepacket()) {
494         // Limpiar el buffer temporal completamente
495         char temp[128] = {0};
496         strncpy(temp, message, sizeof(temp) - 1);
497 
498         char tipo = temp[0];   //Extraer primer byte identificador
    (2)
499         uint64_t current_timestamp = 0;
500         uint32_t packet_id = 0;
501 
502         // Parsear segun tipo con mejor manejo de errores
503         char *token = strtok(temp, ,);
504 
505         if (token == NULL) {
506             printf(Mensaje vacio o mal formado\n);
507             continue;
508         }
509 
510         if (tipo == '2') {
511 
512             // timestamp Unix
513             token = strtok(NULL, ,);
514 
515             if (token != NULL) {
516                 current_timestamp = strtoull(token, NULL, 10); //
    cambie 11 por 10
517             }
518 
519             // opcional: usar txNumber como packet_id
```

```
520             // avanzar hasta ultimo campo
521             int campo = 2;
522 
523             while ((token = strtok(NULL, ,)) != NULL) {
524 
525                 campo++;
526 
527                 // ultimo campo = txNumber
528                 if (campo == 9) { //cambie 11 por 9
529                     packet_id = strtoul(token, NULL, 10); //cambie
    11 por 10
530 
531                 }
532             }
533         }
534 
535         else {
536             printf(Tipo de mensaje desconocido: %c\n, tipo);
537             continue;
538         }
539 
540         // Evitar duplicados
541         uint8_t enviar = 0;
542 
543         if (tipo == '2') {
544             if (current_timestamp != Last_Time_Stamp) {
545                 enviar = 1;
546                 Last_Time_Stamp = current_timestamp;
547                 printf(Nuevo mensaje - ID: %u, TS: %llu\n,
    packet_id, current_timestamp);
548             } else {
549                 printf(Mensaje duplicado - ID: %u, TS: %llu\n,
    packet_id, current_timestamp);
550             }
```

```
        }

        // Enviar si corresponde
        if (enviar) {
            sendToMQTT(message);
        }

        // Limpiar el buffer message para el proximo ciclo
        memset(message, 0, sizeof(message));
    }
    delay(100);
}
}
}
//Recuerda: Tenemos que asignar el valor del actual timestamp a la
   variable auxiliar Last_Time_Stamp
```

Firmware transmisor 4G

```
#define TINY_GSM_MODEM_A7670
#define SerialMon Serial
#define SerialAT Serial1
#define SerialGPS Serial2
#define TINY_GSM_DEBUG SerialMon
#define GSM_PIN

// Libreras
#include <PubSubClient.h>
#include <TinyGsmClient.h>
#include <HardwareSerial.h>
#include <TinyGPS++.h>
#include <Wire.h>
#include <MPU9250.h>
```

```
#include <WiFi.h>
#include <time.h>
#include <sys/time.h>

// ================ Configuracin WiFi ================

// Celular viejo Huawei
const char* ssid = HUAWEI_Y8s;
const char* password = jubiloso123;

// ================ NTP ================

// Chile UTC-4
const long gmtOffset_sec = -14400;
const int daylightOffset_sec = 0;

bool ntpSynced = false;
unsigned long lastNtpRetry = 0;

// Variables para mantener tiempo aunque falle NTP
long long lastValidEpochMs = 0;
unsigned long millisAtLastNtp = 0;

// ================ Configuracin MQTT ================

const char* mqtt_broker = broker.emqx.io;
const int port = 1883;

const char* publish_topic = scootnet_PMM/03/IoT/4G;
const char* mqtt_client_id = GsmClientN_;

// Variables MQTT
unsigned long lastMqttConnectionAttempt = 0;
unsigned long lastMqttKeepAlive = 0;
```

```
int mqttReconnectAttempts = 0;
bool mqttManuallyDisconnected = false;
unsigned long lastSuccessfulPublish = 0;
unsigned long lastMqttLoop = 0;

// ================ Pines y variables ================

#define ANALOG_PIN 34

const float VREF = 3.3;
const int ADC_MAX = 4095;

#define NUM_SAMPLES 25

float R_DIV_1 = 106400.0;
float R_DIV_2 = 50400.0;

// ================ Estructuras para calcular SOC
    ================

struct CalibPoint {
  float adc_volt;
  float volt_real;
};

typedef struct {
    float voltage;
    float soc;
} BatteryPoint;

// ================ Tabla de calibracin ADC ================

const CalibPoint calibTable[] = {
  {1.452f, 1.627f},
```

```
  {1.746f, 1.933f},
  {1.945f, 2.135f},
  {2.041f, 2.231f},
  {2.226f, 2.415f},
  {2.377f, 2.566f},
  {2.485f, 2.654f},
  {2.535f, 2.691f},
  {2.592f, 2.739f},
  {2.720f, 2.830f},
  {2.818f, 2.913f},
  {2.856f, 2.922f},
  {2.868f, 2.931f},
  {2.982f, 2.998f},
  {3.042f, 3.034f},
  {3.085f, 3.053f},
  {3.105f, 3.061f},
  {3.134f, 3.081f},
  {3.177f, 3.105f},
  {3.264f, 3.157f},
  {3.290f, 3.165f}
};

const int NUM_CALIB =
sizeof(calibTable) / sizeof(calibTable[0]);

// ================ Comunicacin I2C ================

#define I2C_SDA 25
#define I2C_SCL 13

// ================ Acelermetro ================

MPU9250 mpu;

```

```
116 float accelBiasX = 0;
117 float accelBiasY = 0;
118 float accelBiasZ = 0;
119
120 float aX = 0;
121 float aY = 0;
122 float aZ = 0;
123
124 // ================= GPS =================
125
126 TinyGPSPlus gps;
127
128 #define BOARD_GPS_RX_PIN 22  //Los damos vuelta
129 #define BOARD_GPS_TX_PIN 21
130
131 // ================= Tabla batera =================
132
133 #define NUM_POINTS_9V 11
134
135 BatteryPoint batteryTable9V[NUM_POINTS_9V] = {
136     {9.00, 100},
137     {8.80, 90},
138     {8.60, 80},
139     {8.40, 70},
140     {8.20, 60},
141     {8.00, 50},
142     {7.80, 40},
143     {7.60, 30},
144     {7.40, 20},
145     {7.20, 10},
146     {7.00, 0}
147 };
148
149 // ================= Variables globales =================
```

```
150
151 int txNumber = 0;
152
153 int16_t SignalQuality = 0;
154 int rssi_dBm = -999;
155
156 unsigned long lastDataSent = 0;
157 unsigned long lastModemCheck = 0;
158
159 // ================ APN ================
160
161 const char apn[] = bam.clarochile.cl;
162 const char gprsUser[] = ;
163 const char gprsPass[] = ;
164
165 // ================ Pines modem ================
166
167 #define MODEM_RESET_PIN 5
168 #define MODEM_PWKEY 4
169 #define MODEM_POWER_ON 12
170
171 #define MODEM_TX 26
172 #define MODEM_RX 27
173
174 #define MODEM_RESET_LEVEL HIGH
175
176 #define BUILTIN_LED 2
177
178 // ================ Instancias ================
179
180 #ifdef DUMP_AT_COMMANDS
181 #include <StreamDebugger.h>
182 StreamDebugger debugger(SerialAT, SerialMon);
183 TinyGsm modem(debugger);
```

```
#else
TinyGsm modem(SerialAT);
#endif

TinyGsmClient client(modem);
PubSubClient mqtt(client);

// ======================================================
// ================= FUNCIONES ==========================
// ======================================================

int readADCRaw() {

    long sum = 0;

    for (int i = 0; i < NUM_SAMPLES; i++) {
        sum += analogRead(ANALOG_PIN);
        delay(5);
    }

    return sum / NUM_SAMPLES;
}

float getRealADCVoltageFromRaw(int rawADC) {

    float volt_adc = (rawADC * VREF) / ADC_MAX;

    if (volt_adc <= calibTable[0].adc_volt)
        return calibTable[0].volt_real;

    if (volt_adc >= calibTable[NUM_CALIB - 1].adc_volt)
        return calibTable[NUM_CALIB - 1].volt_real;

    for (int i = 0; i < NUM_CALIB - 1; i++) {
```

```
218
219         float a1 = calibTable[i].adc_volt;
220         float a2 = calibTable[i + 1].adc_volt;
221
222         if (volt_adc >= a1 && volt_adc <= a2) {
223
224             float v1 = calibTable[i].volt_real;
225             float v2 = calibTable[i + 1].volt_real;
226
227             float t = (volt_adc - a1) / (a2 - a1);
228
229             return v1 + t * (v2 - v1);
230         }
231     }
232
233     return volt_adc;
234 }
235
236 float getBatteryVoltageFromADC(float volt_est_adc) {
237
238     return volt_est_adc *
239            (R_DIV_1 + R_DIV_2) /
240            R_DIV_2;
241 }
242
243 float getBatterySOC(
244     float voltage,
245     BatteryPoint batteryTable[],
246     int numPoints
247 ) {
248
249     if (voltage >= batteryTable[0].voltage)
250         return 100.0;
251
```

```
    if (voltage <= batteryTable[numPoints - 1].voltage)
        return 0.0;

    for (int i = 0; i < numPoints - 1; i++) {

        float v1 = batteryTable[i].voltage;
        float v2 = batteryTable[i + 1].voltage;

        float soc1 = batteryTable[i].soc;
        float soc2 = batteryTable[i + 1].soc;

        if (voltage <= v1 && voltage >= v2) {

            float ratio =
                (voltage - v2) /
                (v1 - v2);

            return soc2 +
                   ratio * (soc1 - soc2);
        }
    }

    return 0.0;
}

void calibrarAcelerometro() {

    const int N = 200;

    float sumX = 0;
    float sumY = 0;
    float sumZ = 0;

    Serial.println(Calibrando acelermetro...);
```

```
286     //Para calcular offset de calibracion se promedian 200 muestras
287     for (int i = 0; i < N; i++) {
288 
289         if (mpu.update()) {
290 
291             sumX += mpu.getAccX();
292             sumY += mpu.getAccY();
293             sumZ += mpu.getAccZ();
294         }
295 
296         delay(20);
297     }
298 
299     accelBiasX = sumX / N;
300     accelBiasY = sumY / N;
301     accelBiasZ = (sumZ / N) - 1.0;
302 
303     Serial.println(Calibracin completada);
304 }
305 
306 // ================= MQTT =================
307 
308 boolean mqttConnect() {
309 
310     if (mqttManuallyDisconnected)
311         return false;
312 
313     SerialMon.print(Conectando MQTT... );
314 
315     String clientId = String(mqtt_client_id) + String(random(0xffff
    ), HEX);
316 
317     mqtt.setKeepAlive(60);
318 
```

```
319     boolean status = mqtt.connect(clientId.c_str());
320 
321     if (!status) {
322 
323         SerialMon.print(Fallo MQTT rc=);
324         SerialMon.println(mqtt.state());
325         mqttReconnectAttempts++;
326 
327         return false;
328     }
329 
330     SerialMon.println(OK);
331 
332     mqttReconnectAttempts = 0;
333 
334     lastMqttKeepAlive = millis();
335     lastSuccessfulPublish = millis();
336 
337     return true;
338     }
339 
340 void maintainMQTTConnection() {
341 
342     if (mqttManuallyDisconnected)
343         return;
344 
345     unsigned long now = millis();
346     mqtt.loop();
347     lastMqttLoop = now;
348 
349     if (!mqtt.connected()) {
350 
351         if (now - lastMqttConnectionAttempt > 30000) {
352 
```

```
353             lastMqttConnectionAttempt = now;
354             SerialMon.println(Reconectando MQTT...);
355             mqttConnect();
356         }
357     }
358 }
359
360 // ================= NTP =================
361
362 bool syncTimeWithNTP() {
363
364     Serial.println(Sincronizando NTP...);
365
366     if (WiFi.status() != WL_CONNECTED) {
367
368         Serial.println(WiFi no conectado);
369         return false;
370     }
371
372     configTime(
373         gmtOffset_sec,
374         daylightOffset_sec,
375         time.google.com,
376         pool.ntp.org,
377         time.cloudflare.com
378     );
379
380     struct tm timeinfo;
381
382     for (int i = 0; i < 15; i++) {
383
384         if (getLocalTime(&timeinfo, 500)) {
385
386             if (timeinfo.tm_year > 100) {
```

```
387
388                 struct timeval tv;
389                 gettimeofday(&tv, NULL);
390                 lastValidEpochMs = ((long long)tv.tv_sec * 1000LL)
    + (tv.tv_usec / 1000);
391                 millisAtLastNtp = millis();
392                 Serial.println(NTP sincronizado!);
393                 return true;
394             }
395         }
396
397         delay(500);
398     }
399
400     Serial.println(Error NTP);
401
402     return false;
403 }
404
405 long long getEstimatedEpochTimeMs() {
406
407     // Si el ultimo tiempo NTP fue valido, devuelve ese tiempo +
    tiempo hace cuanto se obtuvo
408     // Asi calculamos tiempo actual combinando ultimo NTP y millis
    () del microcontrolador
409
410     if (lastValidEpochMs > 0) {
411
412         unsigned long elapsedMillis = millis() - millisAtLastNtp;
413
414         return lastValidEpochMs + elapsedMillis;
415     }
416
417     return millis();
```

```
418 }
419
420 // ================= 4G =================
421
422 bool isModemConnected() {  //Funcion para debug del estado del
        modem
423
424     if (!modem.isNetworkConnected()) {
425         SerialMon.println(Red 4G desconectada);
426         return false;
427     }
428
429     if (!modem.isGprsConnected()) {
430         SerialMon.println(GPRS desconectado);
431         return false;
432     }
433
434     return true;
435 }
436
437 void reconnectModem() {
438
439     SerialMon.println(Reconectando modem...);
440     digitalWrite(BUILTIN_LED, HIGH);
441     modem.gprsDisconnect();
442     delay(2000);
443
444   if (!modem.gprsConnect(apn, gprsUser, gprsPass)) {
445
446     SerialMon.println(Error reconexin GPRS);
447   } else {
448
449     SerialMon.println(GPRS reconectado);
450   }
```

```
451
452   digitalWrite(BUILTIN_LED, LOW);
453 }
454
455 //=== Funciones para obtener y parsear Comandos AT
456 String read_AT_Command(const String &cmd, uint32_t timeout = 3000)
    {
457
458     // limpiar basura previa
459     while (SerialAT.available()) SerialAT.read();
460
461     SerialAT.println(cmd);
462
463     String response = ;
464     uint32_t start = millis();
465
466     while (millis() - start < timeout) {
467
468         while (SerialAT.available()) {
469             char c = SerialAT.read();
470             response += c;
471
472             // cortar cuando termina la respuesta AT
473             if (response.indexOf(OK) != -1 || response.indexOf(
    ERROR) != -1) {
474                 return response;
475             }
476         }
477     }
478
479     return response;
480 }
481
482 int getSNRfromCPSI(String cpsi) {
```

```
483
484     // Limpiar string
485     cpsi.trim();
486
487     // Buscar inicio de +CPSI:
488     int start = cpsi.indexOf(:);
489     if (start == -1) {
490         return -999;
491     }
492
493     // Avanzar despus de :
494     start += 2;
495
496     // Buscar primera coma -> fin del RAT
497     int firstComma = cpsi.indexOf(,, start);
498     if (firstComma == -1) {
499         return -999;
500     }
501
502     // Extraer RAT (LTE, GSM, WCDMA, etc.)
503     String rat = cpsi.substring(start, firstComma);
504     rat.trim();
505
506     // Verificar si es LTE
507     if (rat != LTE) {
508         return -999;
509     }
510
511     // Buscar ltima coma -> RSSNR
512     int lastComma = cpsi.lastIndexOf(,);
513     if (lastComma == -1) {
514         return -999;
515     }
516
```

```
517     // Extraer RSSNR
518     String snrStr = cpsi.substring(lastComma + 1);
519 
520     // Limpiar posibles OK o saltos de lnea
521     snrStr.replace(OK, );
522     snrStr.trim();
523 
524     return snrStr.toInt();
525 }
526 
527 
528 
529 // ================= ENVO DATOS =================
530 
531 void send_data(long long timestamp) {
532 
533     if (!mqtt.connected()) {
534         SerialMon.println(MQTT desconectado);
535         return;
536     }
537 
538     // ===== Batera =====
539 
540     int lectura_adc = readADCRaw();
541     float volt_calib_adc = getRealADCVoltageFromRaw(lectura_adc);
542     float volt_bat = getBatteryVoltageFromADC(volt_calib_adc);
543     float SOC = getBatterySOC(volt_bat, batteryTable9V,
    NUM_POINTS_9V);
544     int socInt = (int)SOC;
545 
546     // ===== MPU =====
547 
548     if (mpu.update()) {
549 
```

```
550         aX = mpu.getAccX() - accelBiasX;
551         aY = mpu.getAccY() - accelBiasY;
552         aZ = mpu.getAccZ() - accelBiasZ;
553     }
554
555     // ===== GPS =====
556
557     while (SerialGPS.available() > 0) {
558         gps.encode(SerialGPS.read());
559     }
560
561     bool gps_ok = gps.location.isValid();
562
563     float lat = gps_ok ? gps.location.lat() : 0.0;
564     float lon = gps_ok ? gps.location.lng() : 0.0;
565
566     // ===== RSSI =====
567
568     if (SignalQuality >= 0 &&
569         SignalQuality <= 31) {
570
571         rssi_dBm =
572             -113 + 2 * SignalQuality;
573
574     } else {
575
576         rssi_dBm = -999;
577     }
578
579     //===== RSSNR (SNR) =====
580     String cpsi = read_AT_Command(AT+CPSI?);
581     SerialMon.println(AT+CENG: );
582     SerialMon.println(cpsi);
583     int rssnr = getSNRfromCPSI(cpsi);
```

```
584     SerialMon.println(rssnr: );
585     SerialMon.println(rssnr);
586
587
588
589
590     // ===== Payload =====
591
592     String payload_IoT = {;
593
594         payload_IoT += \timestamp\: + String(timestamp) + ,;
595         payload_IoT += \gps_ok\: + String(gps_ok ? 1 : 0) + ,;
596         payload_IoT += \lat\: + String(lat, 6) + ,;
597         payload_IoT += \lon\: + String(lon, 6) + ,;
598         payload_IoT += \ax\: + String(aX, 3) + ,;
599         payload_IoT += \ay\: + String(aY, 3) + ,;
600         payload_IoT += \az\: + String(aZ, 3) + ,;
601         payload_IoT += \soc\: + String(socInt) + ,;
602         payload_IoT += \Tx_number\: + String(txNumber + 1) + ,;
603         payload_IoT += \rssi\: + String(rssi_dBm) + ,;
604         payload_IoT += \rssnr\: + String(rssnr);
605         payload_IoT += };
606
607     // ===== Publicar =====
608
609     if (mqtt.publish(publish_topic, payload_IoT.c_str())) {
610
611         txNumber++;
612         lastSuccessfulPublish = millis();
613         Serial.println(Datos enviados);
614     } else {
615         Serial.println(Error publish MQTT);
616     }
617 }
```

```
618
619 // ================= WIFI =================
620
621 void startWifi() {
622
623     Serial.print(Conectando WiFi);
624     WiFi.mode(WIFI_STA);
625     WiFi.begin(ssid, password);
626
627     unsigned long startTime = millis();
628
629     while (WiFi.status() != WL_CONNECTED && millis() - startTime <
    30000) {
630
631         delay(500);
632         Serial.print(.);
633     }
634
635     if (WiFi.status() == WL_CONNECTED) {
636         Serial.println(\nWiFi conectado!);
637     } else {
638         Serial.println(\nError WiFi);
639     }
640 }
641
642 // ================= SETUP =================
643
644 void setup() {
645
646     SerialMon.begin(115200);
647     delay(2000);
648     SerialMon.println(INICIANDO SISTEMA );
649
650     // ===== MPU =====
```

```
651
652     Wire.begin(I2C_SDA, I2C_SCL);
653
654     if (mpu.setup(0x68)) {
655
656         delay(1000);
657         calibrarAcelerometro();
658     } else {
659         Serial.println(MPU9250 no detectado);
660     }
661
662     // ===== ADC =====
663
664     pinMode(ANALOG_PIN, INPUT);
665
666     // ===== GPS =====
667
668     SerialGPS.begin(9600, SERIAL_8N1, BOARD_GPS_RX_PIN,
    BOARD_GPS_TX_PIN);
669     SerialMon.println(GPS inicializado);
670
671     // ===== MODEM LTE =====
672
673     pinMode(MODEM_POWER_ON, OUTPUT);
674     pinMode(BUILTIN_LED, OUTPUT);
675     digitalWrite(BUILTIN_LED, LOW);
676     digitalWrite(MODEM_POWER_ON, HIGH);
677     pinMode(MODEM_RESET_PIN, OUTPUT);
678     digitalWrite(MODEM_RESET_PIN, !MODEM_RESET_LEVEL);
679     delay(100);
680     digitalWrite(MODEM_RESET_PIN, MODEM_RESET_LEVEL);
681     delay(2600);
682     digitalWrite(MODEM_RESET_PIN, !MODEM_RESET_LEVEL);
683     pinMode(MODEM_PWKEY, OUTPUT);
```

```
684     digitalWrite(MODEM_PWKEY, LOW);
685     delay(100);
686     digitalWrite(MODEM_PWKEY, HIGH);
687     delay(1000);
688     digitalWrite(MODEM_PWKEY, LOW);
689 
690     SerialAT.begin(115200, SERIAL_8N1, MODEM_RX, MODEM_TX);
691 
692     delay(3000);
693 
694     SerialMon.println(Inicializando modem...);
695 
696     if (!modem.init()) {
697 
698         SerialMon.println(Error modem);
699         return;
700     }
701 
702     SerialMon.print(Esperando red...);
703 
704     if (!modem.waitForNetwork()) {
705 
706         SerialMon.println( fail);
707         return;
708     }
709 
710     SerialMon.println( OK);
711 
712     if (!modem.gprsConnect(apn, gprsUser, gprsPass)) {
713 
714         SerialMon.println(Error GPRS);
715         return;
716     }
717 
```

```
718     SerialMon.println(GPRS OK);
719
720     delay(5000);
721
722     mqtt.setServer(mqtt_broker, port);
723     mqtt.setKeepAlive(60);
724
725     // Desconectamos WiFi antes de iniciarlo para evitar que haya
        quedado colgado
726
727     WiFi.disconnect(true);
728     delay(1000);
729     WiFi.mode(WIFI_OFF);
730     delay(1000);
731     WiFi.mode(WIFI_STA);
732     delay(1000);
733     startWifi();
734
735     ntpSynced = syncTimeWithNTP();   //Sincronizamos el tiempo NTP
736
737     // ===== MQTT =====
738
739     mqttConnect();
740 }
741
742 // ================= LOOP =================
743
744 void loop() {
745
746     unsigned long now = millis();
747
748     // ===== GPS continuo =====
749
750     while (SerialGPS.available() > 0) {
```

```
751         gps.encode(SerialGPS.read());
752     }
753 
754     // ===== Revisar modem =====
755 
756     if (now - lastModemCheck > 30000) {
757 
758         lastModemCheck = now;
759 
760         if (!isModemConnected()) {
761             reconnectModem();
762         }
763 
764         SignalQuality = modem.getSignalQuality();
765     }
766 
767     // ===== Mantener MQTT =====
768 
769     maintainMQTTConnection();
770 
771     // ===== Reintento NTP =====
772 
773     if (!ntpSynced && now - lastNtpRetry > 30000) {
774 
775         lastNtpRetry = now;
776 
777         if (WiFi.status() == WL_CONNECTED) {
778 
779             ntpSynced = syncTimeWithNTP();
780         }
781     }
782 
783     // ===== Enviar datos =====
784 
```

```
785     if (now - lastDataSent > 6000) {
786 
787         lastDataSent = now;
788 
789         if (mqtt.connected()) {
790 
791             long long estimatedTime = getEstimatedEpochTimeMs();
792             send_data(estimatedTime);
793         }
794     }
795 
796     delay(10);
797 }
```

Datos obtenidos

A continuación se adjuntan los datos obtenidos durante las pruebas.

Datos de prototipo WiFi-LTE

Si bien los registros del prototipo WiFi-LTE incluyen un valor de SNR, este corresponde a una estimación realizada en el backend mediante una fórmula que finalmente no se consideró en el análisis de resultados. Por este motivo, dicho valor debe ignorarse.

name,time,ax,ay,az,gps_ok,id_device,latencia,latitud,longitud,red,rssi,snr_rssnr,soc,time_device,time_server,tx
IoT,1780161948120251751,0.227,-0.109,1.135,1,01,765,-33.022175,-71.551369,WiFi,-66,24,0,1780161947347,1780161948112,12
IoT,1780161954154455722,0.26,-0.112,0.835,1,01,754,-33.022198,-71.551422,WiFi,-64,26,0,1780161953395,1780161954149,13
IoT,1780161960145812817,0.048,-0.113,1.527,1,01,696,-33.022232,-71.551445,WiFi,-75,15,0,1780161959444,1780161960140,14
IoT,1780161966188559231,0.196,0.011,0.846,1,01,685,-33.022228,-71.551506,WiFi,-56,34,0,1780161965494,1780161966179,15
IoT,1780161972270912112,0.194,-0.215,1.28,1,01,720,-33.022221,-71.551575,WiFi,-55,35,0,1780161971541,1780161972261,16
IoT,1780161978277539387,0.254,-0.001,0.873,1,01,680,-33.022266,-71.551582,WiFi,-53,37,0,1780161977588,1780161978268,17
IoT,1780161984367411628,0.075,-0.268,1.371,1,01,728,-33.022327,-71.55159,WiFi,-59,31,0,1780161983634,1780161984362,18
IoT,1780161990405139341,0.149,-0.195,1.08,1,01,712,-33.022392,-71.551598,WiFi,-55,35,0,1780161989684,1780161990396,19
IoT,1780161996431126389,0.169,-0.078,1.018,1,01,693,-33.022438,-71.551605,WiFi,-56,34,0,1780161995732,1780161996425,20
IoT,1780162002460566834,0.173,-0.131,0.975,1,01,673,-33.022449,-71.551598,WiFi,-58,32,0,1780162001778,1780162002451,21
IoT,1780162008516261870,0.184,-0.134,0.984,1,01,684,-33.02248,-71.551575,WiFi,-56,34,0,1780162007823,1780162008507,22
IoT,1780162014668623007,0.104,-0.042,1.027,1,01,792,-33.022484,-71.551567,WiFi,-53,37,0,1780162013871,1780162014663,23
IoT,1780162020600653763,0.189,-0.131,0.999,1,01,671,-33.022495,-71.551559,WiFi,-60,30,0,1780162019920,1780162020591,24
IoT,1780162026635210694,0.137,-0.123,0.988,1,01,658,-33.022526,-71.551537,WiFi,-58,32,0,1780162025968,1780162026626,25
IoT,1780162051173203683,0.076,-0.056,1.02,1,01,19147,-33.022533,-71.551529,WiFi,-64,26,0,1780162032016,1780162051163,26
IoT,1780162051406633072,0.009,0.052,1.169,1,01,13325,-33.022545,-71.551529,WiFi,-49,41,0,1780162038065,1780162051390,27
IoT,1780162051419164714,0.127,0.107,1.067,1,01,7279,-33.022491,-71.551567,WiFi,-48,42,66,1780162044116,1780162051395,28
IoT,1780162051420372313,0.247,0.016,1.146,1,01,1235,-33.022488,-71.551575,WiFi,-59,31,59,1780162050164,1780162051399,29
IoT,1780162123722007125,0.298,-0.065,1.034,1,01,67488,-33.022484,-71.551575,WiFi,-62,28,64,1780162056212,1780162123700,30
IoT,1780162123722484972,0.322,-0.086,0.998,1,01,61445,-33.022488,-71.551575,WiFi,-55,35,64,1780162062259,1780162123704,31
IoT,1780162123914673668,0.333,-0.066,1.026,1,01,55574,-33.022488,-71.551575,WiFi,-55,35,64,1780162068308,1780162123882,32
IoT,1780162123917277884,0.32,-0.03,1.034,1,01,49529,-33.022488,-71.55159,WiFi,-60,30,65,1780162074356,1780162123885,33
IoT,1780162123954480683,0.305,-0.047,1.038,1,01,43482,-33.022491,-71.551575,WiFi,-64,26,64,1780162080406,1780162123888,34
IoT,1780162123955584117,0.293,0.063,0.955,1,01,25338,-33.022541,-71.551628,WiFi,-66,24,62,1780162098559,1780162123897,37
IoT,1780162123955807930,0.338,0.094,0.716,1,01,31389,-33.022518,-71.551582,WiFi,-61,29,62,1780162092505,1780162123894,36
IoT,1780162123956304759,0.308,-0.035,1.046,1,01,37437,-33.022491,-71.551575,WiFi,-60,30,63,1780162086454,1780162123891,35
IoT,1780162123956939827,0.405,-0.236,1.195,1,01,13239,-33.022636,-71.551628,WiFi,-63,27,63,1780162110665,1780162123904,39
IoT,1780162123957052955,0.574,0.099,0.877,1,01,19288,-33.022587,-71.551628,WiFi,-61,29,63,1780162104612,1780162123900,38
IoT,1780162123959443322,0.564,0.044,0.972,1,01,7223,-33.022678,-71.551643,WiFi,-58,32,63,1780162116721,1780162123944,40
IoT,1780162123963094362,0.401,-0.063,0.764,1,01,1172,-33.022736,-71.551659,WiFi,-54,36,62,1780162122776,1780162123948,41
IoT,1780162129519567412,0.336,0.029,0.881,1,01,679,-33.022793,-71.551674,WiFi,-63,27,62,1780162128830,1780162129509,42
IoT,1780162136316005288,0.588,0.153,1.011,1,01,1428,-33.022846,-71.551689,WiFi,-62,28,62,1780162134883,1780162136311,43

IoT,1780162141653271006,0.312,-0.065,1.07,1,01,704,-33.022903,-71.551697,WiFi,-64,26,62,1780162140937,1780162141641,44
IoT,1780162147672131644,0.302,0.158,1.255,1,01,667,-33.022964,-71.551712,WiFi,-48,42,61,1780162146995,1780162147662,45
IoT,1780162155075343094,0.4,-0.079,1.075,1,01,2011,-33.023026,-71.55172,WiFi,-63,27,62,1780162153054,1780162155065,46
IoT,1780162192585317636,0.354,0.053,0.913,1,01,33453,-33.023079,-71.551735,WiFi,-64,26,61,1780162159113,1780162192566,47
IoT,1780162192585817465,0.327,-0.054,1.002,1,01,27400,-33.023132,-71.55175,WiFi,-67,23,60,1780162165170,1780162192570,48
IoT,1780162192765596255,0.55,-0.023,0.935,1,01,21524,-33.023186,-71.551765,WiFi,-65,25,60,1780162171225,1780162192749,49
IoT,1780162192769259461,0.352,0.026,0.971,1,01,15473,-33.023239,-71.551781,WiFi,-67,23,60,1780162177280,1780162192753,50
IoT,1780162192784532040,0.655,0.132,1.018,1,01,9422,-33.0233,-71.551788,WiFi,-58,32,60,1780162183334,1780162192756,51
IoT,1780162192784912777,0.409,0.05,1.276,1,01,3367,-33.023361,-71.551796,WiFi,-64,26,59,1780162189392,1780162192759,52
IoT,1780162196620334353,0.494,0.029,1.368,1,01,1164,-33.023418,-71.551804,WiFi,-56,34,58,1780162195453,1780162196617,53
IoT,1780162246927091736,0.386,-0.049,0.834,1,01,45400,-33.023464,-71.551842,WiFi,-67,23,56,1780162201509,1780162246909,54
IoT,1780162246927945228,0.38,-0.071,0.765,1,01,39343,-33.02351,-71.551865,WiFi,-63,27,62,1780162207570,1780162246913,55
IoT,1780162247167580766,0.416,-0.105,0.85,1,01,33504,-33.023552,-71.551895,WiFi,-59,31,62,1780162213633,1780162247137,56
IoT,1780162247171345842,0.369,0.122,1.047,1,01,27446,-33.023594,-71.551933,WiFi,-58,32,62,1780162219695,1780162247141,57
IoT,1780162247189803909,0.439,-0.028,1.007,1,01,21391,-33.023647,-71.551964,WiFi,-58,32,62,1780162225754,1780162247145,58
IoT,1780162247190050573,0.434,-0.062,1.141,1,01,15338,-33.023689,-71.552002,WiFi,-59,31,62,1780162231812,1780162247150,59
IoT,1780162247190479625,0.542,0.04,0.937,1,01,9284,-33.023731,-71.552032,WiFi,-51,39,62,1780162237870,1780162247154,60
IoT,1780162247190913935,0.371,-0.102,0.851,1,01,3233,-33.023777,-71.552063,WiFi,-56,34,62,1780162243928,1780162247161,61
IoT,1780162250668884825,0.418,0.075,0.643,1,01,677,-33.023842,-71.552063,WiFi,-50,40,62,1780162249985,1780162250662,62
IoT,1780162256730150787,0.466,-0.156,1.13,1,01,683,-33.023876,-71.55201,WiFi,-55,35,60,1780162256042,1780162256725,63
IoT,1780162262919676920,0.447,-0.317,1.152,1,01,807,-33.023884,-71.551941,WiFi,-55,35,60,1780162262099,1780162262906,64
IoT,1780162268938577523,0.365,0.104,0.911,1,01,769,-33.023899,-71.551872,WiFi,-55,35,60,1780162268159,1780162268928,65
IoT,1780162274915238386,0.573,0.175,1.432,1,01,679,-33.023911,-71.551796,WiFi,-51,39,56,1780162274225,1780162274904,66
IoT,1780162280980139055,0.193,0.065,1.312,1,01,681,-33.023918,-71.551727,WiFi,-49,41,61,1780162280290,1780162280971,67
IoT,1780162287188754991,0.329,0.099,0.981,1,01,828,-33.023861,-71.551712,WiFi,-60,30,61,1780162286355,1780162287183,68
IoT,1780162293099479505,0.615,-0.285,1.206,1,01,673,-33.023819,-71.551712,WiFi,-53,37,61,1780162292419,1780162293092,69
IoT,1780162299156022942,0.607,0.085,0.855,1,01,660,-33.023777,-71.55172,WiFi,-51,39,63,1780162298484,1780162299144,70
IoT,1780162305306175118,0.411,-0.091,0.903,1,01,739,-33.023724,-71.55172,WiFi,-52,38,63,1780162304548,1780162305287,71
IoT,1780162311333490610,0.428,-0.204,1.338,1,01,712,-33.02367,-71.551712,WiFi,-51,39,62,1780162310612,1780162311324,72
IoT,1780162317387386005,0.403,0.069,0.702,1,01,701,-33.023621,-71.551689,WiFi,-52,38,62,1780162316681,1780162317382,73
IoT,1780162323507995978,0.432,-0.058,0.84,1,01,754,-33.023575,-71.551628,WiFi,-52,38,62,1780162322748,1780162323502,74
IoT,1780162329508624719,0.502,0.096,0.647,1,01,688,-33.023521,-71.551613,WiFi,-51,39,62,1780162328810,1780162329498,75
IoT,1780162335704332862,0.342,0.157,0.968,1,01,816,-33.023468,-71.551605,WiFi,-51,39,56,1780162334878,1780162335694,76

IoT,1780162341638115447,0.371,0.142,0.905,1,01,681,-33.023426,-71.55159,WiFi,-51,39,58,1780162340947,1780162341628,77
IoT,1780162347729192942,0.467,-0.112,0.981,1,01,705,-33.02338,-71.551575,WiFi,-52,38,59,1780162347019,1780162347724,78
IoT,1780162353776116705,0.635,-0.245,1.336,1,01,674,-33.023327,-71.551559,WiFi,-51,39,59,1780162353091,1780162353765,79
IoT,1780162359867204902,0.408,-0.163,0.866,1,01,695,-33.023266,-71.551544,WiFi,-51,39,59,1780162359161,1780162359856,80
IoT,1780162366011458854,0.412,-0.037,0.907,1,01,767,-33.023205,-71.551537,WiFi,-54,36,58,1780162365234,1780162366001,81
IoT,1780162372055277479,0.483,-0.1,0.84,1,01,737,-33.023151,-71.551529,WiFi,-52,38,58,1780162371305,1780162372042,82
IoT,1780162378054068609,0.51,-0.007,1.014,1,01,674,-33.023106,-71.551514,WiFi,-53,37,58,1780162377375,1780162378049,83
IoT,1780162387517955543,0.511,-0.137,0.978,1,01,4064,-33.023056,-71.551498,WiFi,-49,41,58,1780162383444,1780162387508,84
IoT,1780162390200560515,0.729,0.025,0.9,1,01,681,-33.023014,-71.551498,WiFi,-50,40,58,1780162389513,1780162390194,85
IoT,1780162396340752958,0.6,-0.288,1.274,1,01,745,-33.022949,-71.551483,WiFi,-51,39,57,1780162395584,1780162396329,86
IoT,1780162402423327566,0.5,-0.043,0.945,1,01,756,-33.022892,-71.55146,WiFi,-50,40,57,1780162401657,1780162402413,87
IoT,1780162408420158001,0.612,-0.204,0.967,1,01,684,-33.022835,-71.551453,WiFi,-51,39,57,1780162407731,1780162408415,88
IoT,1780162414490745115,0.633,-0.242,1.223,1,01,678,-33.022762,-71.55143,WiFi,-49,41,56,1780162413803,1780162414481,89
IoT,1780162420688882982,0.593,-0.293,1.19,1,01,809,-33.022705,-71.551422,WiFi,-48,42,56,1780162419871,1780162420680,90
IoT,1780162426676654840,0.46,-0.031,0.763,1,01,725,-33.022652,-71.551407,WiFi,-52,38,56,1780162425942,1780162426667,91
IoT,1780162433182955336,0.508,0.147,0.607,1,01,1145,-33.022591,-71.551376,WiFi,-49,41,56,1780162432026,1780162433171,92
IoT,1780162438830368111,0.472,0.111,1.132,1,01,717,-33.022526,-71.551361,WiFi,-50,40,55,1780162438108,1780162438825,93
IoT,1780162444954506626,0.566,-0.144,1.15,1,01,766,-33.022465,-71.551346,WiFi,-50,40,54,1780162444183,1780162444949,94
IoT,1780162450969614825,0.685,-0.227,1.254,1,01,705,-33.022427,-71.551331,WiFi,-49,41,54,1780162450254,1780162450959,95
IoT,1780162457034855108,0.719,0.06,0.802,1,01,700,-33.022396,-71.551315,WiFi,-49,41,53,1780162456325,1780162457025,96
IoT,1780162463184466672,0.695,0.11,1.159,1,01,784,-33.022343,-71.551292,WiFi,-49,41,55,1780162462391,1780162463175,97
IoT,1780162469163352870,0.59,-0.072,1.339,1,01,699,-33.022297,-71.551285,WiFi,-48,42,56,1780162468459,1780162469158,98
IoT,1780162475371617483,0.675,0.081,0.861,1,01,829,-33.022228,-71.5513,WiFi,-48,42,55,1780162474533,1780162475362,99
IoT,1780162481300211156,0.547,0.081,0.874,1,01,687,-33.022179,-71.551285,WiFi,-51,39,55,1780162480603,1780162481290,100
IoT,1780162487406493738,0.563,-0.082,0.867,1,01,719,-33.022125,-71.551277,WiFi,-51,39,55,1780162486677,1780162487396,101
IoT,1780162493500037847,0.637,-0.122,1.368,1,01,736,-33.022076,-71.55127,WiFi,-51,39,55,1780162492750,1780162493486,102
IoT,1780162499519057128,0.423,0.051,0.709,1,01,693,-33.022026,-71.55127,WiFi,-49,41,54,1780162498820,1780162499513,103
IoT,1780162505687407861,0.659,0.061,0.871,1,01,785,-33.021969,-71.551254,WiFi,-49,41,55,1780162504893,1780162505678,104
IoT,1780162511668463748,0.43,-0.263,1.18,1,01,694,-33.021912,-71.551254,WiFi,-47,43,55,1780162510963,1780162511657,105
IoT,1780162517763472803,0.513,-0.12,0.929,1,01,721,-33.02187,-71.551254,WiFi,-49,41,54,1780162517032,1780162517753,106
IoT,1780162523920637136,0.43,-0.287,1.436,1,01,810,-33.021812,-71.551247,WiFi,-47,43,52,1780162523101,1780162523911,107
IoT,1780162529856740167,0.533,-0.122,1.068,1,01,680,-33.021755,-71.551239,WiFi,-44,46,54,1780162529171,1780162529851,108
IoT,1780162535951133665,0.519,-0.093,0.962,1,01,697,-33.02169,-71.551216,WiFi,-48,42,55,1780162535244,1780162535941,109

IoT,1780162542040269520,0.766,0.03,0.808,1,01,719,-33.021637,-71.551208,WiFi,-47,43,55,1780162541312,1780162542031,110
IoT,1780162548176895723,0.616,-0.317,0.925,1,01,786,-33.021584,-71.551186,WiFi,-50,40,58,1780162547382,1780162548168,111
IoT,1780162554184584858,0.472,-0.069,0.758,1,01,725,-33.021538,-71.551178,WiFi,-50,40,55,1780162553450,1780162554175,112
IoT,1780162560254889604,0.497,0.084,1.081,1,01,731,-33.021484,-71.551163,WiFi,-50,40,55,1780162559518,1780162560249,113
IoT,1780162566318808989,0.817,-0.145,1.307,1,01,718,-33.021431,-71.551147,WiFi,-47,43,56,1780162565589,1780162566307,114
IoT,1780162572353483985,0.689,-0.148,1.165,1,01,687,-33.021378,-71.551155,WiFi,-47,43,55,1780162571658,1780162572345,115
IoT,1780162578487282816,0.537,0.099,0.751,1,01,743,-33.02132,-71.551132,WiFi,-47,43,56,1780162577734,1780162578477,116
IoT,1780162584535130723,0.712,0.15,0.92,1,01,717,-33.021252,-71.551117,WiFi,-47,43,56,1780162583808,1780162584525,117
IoT,1780162590666983070,0.667,-0.111,1.105,1,01,780,-33.021198,-71.551102,WiFi,-49,41,56,1780162589881,1780162590661,118
IoT,1780162596711957299,0.544,0.041,1.192,1,01,749,-33.021145,-71.551086,WiFi,-49,41,56,1780162595957,1780162596706,119
IoT,1780162603011984137,0.481,-0.078,0.926,1,01,969,-33.021084,-71.551079,WiFi,-49,41,55,1780162602031,1780162603000,120
IoT,1780162608910619282,0.664,0.031,0.948,1,01,792,-33.021034,-71.551064,WiFi,-49,41,56,1780162608108,1780162608900,121
IoT,1780162614886924651,0.762,-0.196,1.157,1,01,686,-33.020981,-71.551056,WiFi,-49,41,56,1780162614190,1780162614876,122
IoT,1780162620977163775,0.475,0.188,0.882,1,01,708,-33.020927,-71.551048,WiFi,-50,40,56,1780162620264,1780162620972,123
IoT,1780162627130800667,0.787,0.142,0.923,1,01,784,-33.020878,-71.551041,WiFi,-49,41,54,1780162626337,1780162627121,124
IoT,1780162633100833871,0.465,0.058,0.999,1,01,683,-33.020813,-71.551018,WiFi,-47,43,54,1780162632408,1780162633091,125
IoT,1780162639209352118,0.696,-0.127,1.229,1,01,719,-33.02076,-71.551018,WiFi,-47,43,58,1780162638480,1780162639199,126
IoT,1780162645752154218,0.565,-0.074,0.749,1,01,1179,-33.020699,-71.55101,WiFi,-47,43,58,1780162644561,1780162645740,127
IoT,1780162651364071587,0.673,-0.225,1.169,1,01,723,-33.020641,-71.551003,WiFi,-48,42,58,1780162650632,1780162651355,128
IoT,1780162657422409281,0.483,0.028,1.078,1,01,702,-33.020584,-71.550972,WiFi,-49,41,58,1780162656706,1780162657408,129
IoT,1780162663460142155,0.582,0,0.808,1,01,672,-33.020519,-71.550957,WiFi,-49,41,57,1780162662779,1780162663451,130
IoT,1780162669621931960,0.635,-0.27,1.121,1,01,763,-33.02047,-71.550941,WiFi,-49,41,56,1780162668849,1780162669612,131
IoT,1780162675617041322,0.542,-0.019,0.679,1,01,685,-33.020412,-71.550941,WiFi,-49,41,55,1780162674923,1780162675608,132
IoT,1780162681815123575,0.459,0.06,1.098,1,01,814,-33.020348,-71.550926,WiFi,-49,41,58,1780162680991,1780162681805,133
IoT,1780162687765922384,0.795,0.177,1.331,1,01,689,-33.02029,-71.550903,WiFi,-48,42,59,1780162687066,1780162687755,134
IoT,1780162693871794297,0.754,-0.193,1.065,1,01,724,-33.020222,-71.550896,WiFi,-50,40,59,1780162693138,1780162693862,135
IoT,1780162699934994481,0.527,-0.018,0.978,1,01,718,-33.020164,-71.55088,WiFi,-48,42,58,1780162699207,1780162699925,136
IoT,1780162706033213401,0.771,-0.009,0.991,1,01,748,-33.020111,-71.550865,WiFi,-49,41,57,1780162705276,1780162706024,137
IoT,1780162712064882011,0.588,-0.103,0.79,1,01,701,-33.020054,-71.55085,WiFi,-46,44,53,1780162711348,1780162712049,138
IoT,1780162718163075625,0.755,-0.334,1.123,1,01,730,-33.020008,-71.550835,WiFi,-49,41,53,1780162717420,1780162718150,139
IoT,1780162724227656145,0.569,0.034,0.879,1,01,728,-33.019958,-71.550812,WiFi,-47,43,51,1780162723489,1780162724217,140
IoT,1780162730352028528,0.704,-0.183,1.012,1,01,785,-33.019909,-71.550812,WiFi,-47,43,51,1780162729556,1780162730341,141
IoT,1780162736333366442,0.622,-0.015,0.667,1,01,697,-33.019852,-71.550827,WiFi,-47,43,56,1780162735626,1780162736323,142

IoT,1780162742411410731,0.611,-0.115,1.066,1,01,695,-33.019802,-71.550789,WiFi,-49,41,56,1780162741706,1780162742401,143
IoT,1780162748492385642,0.713,-0.078,0.824,1,01,697,-33.019726,-71.550781,WiFi,-48,42,56,1780162747783,1780162748480,144
IoT,1780162754570842261,0.701,-0.192,1.209,1,01,705,-33.019695,-71.550789,WiFi,-50,40,55,1780162753856,1780162754561,145
IoT,1780162760762878695,0.532,0.068,0.831,1,01,823,-33.019661,-71.550774,WiFi,-51,39,56,1780162759929,1780162760752,146
IoT,1780162766703861778,0.774,-0.277,1.01,1,01,693,-33.019608,-71.550751,WiFi,-49,41,55,1780162766001,1780162766694,147
IoT,1780162772784613173,0.87,-0.295,1.391,1,01,700,-33.019554,-71.550751,WiFi,-50,40,55,1780162772071,1780162772771,148
IoT,1780162778863651074,0.761,-0.383,1.367,1,01,710,-33.019497,-71.550735,WiFi,-50,40,55,1780162778143,1780162778853,149
IoT,1780162784895909746,0.692,-0.027,1.331,1,01,674,-33.019455,-71.550728,WiFi,-51,39,56,1780162784213,1780162784887,150
IoT,1780162791036091527,0.397,0.121,1.033,1,01,743,-33.019402,-71.550705,WiFi,-47,43,55,1780162790283,1780162791026,151
IoT,1780162797109978341,0.522,-0.013,0.886,1,01,749,-33.019363,-71.550713,WiFi,-47,43,56,1780162796351,1780162797100,152
IoT,1780162803106051388,0.495,0.057,1.149,1,01,675,-33.01931,-71.550705,WiFi,-48,42,57,1780162802418,1780162803093,153
IoT,1780162809295368895,0.601,-0.051,1.242,1,01,786,-33.019245,-71.550697,WiFi,-49,41,58,1780162808498,1780162809284,154
IoT,1780162815292928974,0.661,-0.029,1.282,1,01,699,-33.019188,-71.550705,WiFi,-49,41,58,1780162814580,1780162815279,155
IoT,1780162821480973139,0.523,0.205,0.999,1,01,810,-33.019131,-71.550697,WiFi,-49,41,58,1780162820660,1780162821470,156
IoT,1780162827517430915,0.451,0.076,0.84,1,01,768,-33.019047,-71.550713,WiFi,-46,44,58,1780162826742,1780162827510,157
IoT,1780162833540329626,0.743,-0.177,1.187,1,01,708,-33.019005,-71.550705,WiFi,-50,40,57,1780162832822,1780162833530,158
IoT,1780162839707849491,0.696,0.139,1.132,1,01,796,-33.018951,-71.550682,WiFi,-49,41,58,1780162838902,1780162839698,159
IoT,1780162845702073552,0.783,-0.085,0.961,1,01,715,-33.018909,-71.550667,WiFi,-49,41,58,1780162844977,1780162845692,160
IoT,1780162851741196139,0.829,0.008,0.751,1,01,676,-33.018875,-71.550659,WiFi,-52,38,59,1780162851055,1780162851731,161
IoT,1780162857825963185,0.747,-0.12,1.09,1,01,692,-33.018829,-71.550652,WiFi,-55,35,59,1780162857129,1780162857821,162
IoT,1780162864091399158,0.516,0.083,1.006,1,01,877,-33.018787,-71.550644,WiFi,-50,40,59,1780162863203,1780162864080,163
IoT,1780162869976326494,0.762,0.004,0.759,1,01,690,-33.018738,-71.550636,WiFi,-51,39,58,1780162869276,1780162869966,164
IoT,1780162876163155513,0.603,0.046,1.026,1,01,797,-33.018688,-71.550629,WiFi,-56,34,57,1780162875355,1780162876152,165
IoT,1780162882206926899,0.744,0.014,0.746,1,01,764,-33.018642,-71.550613,WiFi,-55,35,57,1780162881432,1780162882196,166
IoT,1780162888200212866,0.642,-0.011,0.922,1,01,692,-33.018597,-71.550606,WiFi,-53,37,56,1780162887503,1780162888195,167
IoT,1780162894294267248,0.775,0.01,1.255,1,01,716,-33.018543,-71.550598,WiFi,-51,39,56,1780162893573,1780162894289,168
IoT,1780162900342524724,0.525,0.155,1.34,1,01,689,-33.018486,-71.550591,WiFi,-51,39,56,1780162899640,1780162900329,169
IoT,1780162906421077716,0.52,-0.135,0.777,1,01,694,-33.018444,-71.550552,WiFi,-53,37,54,1780162905716,1780162906410,170
IoT,1780162912524940259,0.579,0.028,1.021,1,01,717,-33.018387,-71.550529,WiFi,-52,38,52,1780162911797,1780162912514,171
IoT,1780162918757601048,0.727,-0.187,1.15,1,01,876,-33.018341,-71.550522,WiFi,-52,38,52,1780162917876,1780162918752,172
IoT,1780162924644817291,0.746,-0.143,1.362,1,01,683,-33.018295,-71.550507,WiFi,-50,40,55,1780162923952,1780162924635,173
IoT,1780162930730606188,0.666,-0.019,1.084,1,01,694,-33.018246,-71.550491,WiFi,-50,40,55,1780162930026,1780162930720,174
IoT,1780162936893231633,0.818,0.097,1.033,1,01,781,-33.018192,-71.550484,WiFi,-50,40,55,1780162936100,1780162936881,175

IoT,1780162942936251012,0.55,-0.044,0.849,1,01,750,-33.018139,-71.550476,WiFi,-51,39,55,1780162942174,1780162942924,176
IoT,1780162949172972430,0.829,-0.113,1.283,1,01,919,-33.018082,-71.550453,WiFi,-49,41,55,1780162948249,1780162949168,177
IoT,1780162955112190632,0.64,-0.07,0.784,1,01,783,-33.018028,-71.550438,WiFi,-50,40,55,1780162954319,1780162955102,178
IoT,1780162961070381464,0.717,-0.119,1.049,1,01,673,-33.017975,-71.55043,WiFi,-50,40,55,1780162960387,1780162961060,179
IoT,1780162967151812451,0.511,0.013,0.92,1,01,687,-33.017925,-71.550415,WiFi,-50,40,56,1780162966455,1780162967142,180
IoT,1780162973238442776,0.689,-0.146,1.082,1,01,703,-33.017868,-71.5504,WiFi,-51,39,56,1780162972524,1780162973227,181
IoT,1780162979377042556,0.603,0.252,1.096,1,01,780,-33.017796,-71.550392,WiFi,-50,40,56,1780162978592,1780162979372,182
IoT,1780162985366863241,0.649,-0.003,0.835,1,01,695,-33.017746,-71.550377,WiFi,-49,41,55,1780162984662,1780162985357,183
IoT,1780162991420282403,0.758,-0.092,0.991,1,01,678,-33.017685,-71.550369,WiFi,-49,41,55,1780162990732,1780162991410,184
IoT,1780162997504303431,0.532,0.07,0.851,1,01,692,-33.017632,-71.550362,WiFi,-50,40,54,1780162996803,1780162997495,185
IoT,1780163003653017074,0.594,-0.097,0.774,1,01,769,-33.017574,-71.550354,WiFi,-50,40,54,1780163002873,1780163003642,186
IoT,1780163009643283401,0.673,-0.141,1.02,1,01,696,-33.017517,-71.550369,WiFi,-49,41,53,1780163008942,1780163009638,187
IoT,1780163015738019244,0.546,-0.079,0.993,1,01,713,-33.017502,-71.550362,WiFi,-50,40,53,1780163015012,1780163015725,188
IoT,1780163021877796729,0.518,-0.036,1.025,1,01,786,-33.01749,-71.550369,WiFi,-51,39,53,1780163021081,1780163021867,189
IoT,1780163027855560771,0.557,-0.015,1.044,1,01,693,-33.017479,-71.550362,WiFi,-48,42,53,1780163027152,1780163027845,190
IoT,1780163033948996961,0.58,0.025,1.016,1,01,715,-33.017471,-71.550346,WiFi,-50,40,53,1780163033224,1780163033939,191
IoT,1780163040002774481,0.521,0.025,1.048,1,01,698,-33.017467,-71.550346,WiFi,-50,40,53,1780163039298,1780163039996,192
IoT,1780163046084455175,0.506,0.017,1.064,1,01,697,-33.017456,-71.550354,WiFi,-49,41,52,1780163045376,1780163046073,193
IoT,1780163052195870915,0.496,0.061,1.058,1,01,730,-33.017452,-71.550346,WiFi,-50,40,52,1780163051454,1780163052184,194
IoT,1780163058221057108,0.47,0.066,1.065,1,01,680,-33.017445,-71.550346,WiFi,-49,41,52,1780163057531,1780163058211,195
IoT,1780163064296708571,0.374,0.084,1.077,1,01,678,-33.017441,-71.550354,WiFi,-48,42,49,1780163063609,1780163064287,196
IoT,1780163070412505089,0.505,0.058,1.06,1,01,719,-33.017437,-71.550354,WiFi,-50,40,50,1780163069688,1780163070407,197
IoT,1780163076558116051,0.445,0.055,1.093,1,01,781,-33.017445,-71.550362,WiFi,-49,41,56,1780163075767,1780163076548,198
IoT,1780163082580139547,0.449,0.083,1.076,1,01,720,-33.017448,-71.550362,WiFi,-48,42,56,1780163081848,1780163082568,199
IoT,1780163088953901682,0.381,0.177,1.134,1,01,1012,-33.017456,-71.550385,WiFi,-50,40,56,1780163087928,1780163088940,200
IoT,1780163094742611407,0.537,0.126,1.08,1,01,724,-33.017456,-71.550385,WiFi,-49,41,56,1780163094008,1780163094732,201
IoT,1780163100778161195,0.395,0.091,0.935,1,01,685,-33.017456,-71.550423,WiFi,-47,43,56,1780163100087,1780163100772,202
IoT,1780163106978180009,0.402,0.048,1.016,1,01,801,-33.017464,-71.550491,WiFi,-49,41,57,1780163106167,1780163106968,203
IoT,1780163112993369815,0.471,-0.066,1.019,1,01,737,-33.017471,-71.550545,WiFi,-49,41,57,1780163112247,1780163112984,204
IoT,1780163119056941809,0.557,0.092,0.876,1,01,723,-33.017483,-71.550575,WiFi,-48,42,57,1780163118324,1780163119047,205
IoT,1780163125099436839,0.522,0.093,0.827,1,01,685,-33.017517,-71.550613,WiFi,-51,39,57,1780163124402,1780163125087,206
IoT,1780163131337244629,0.54,0.068,0.999,1,01,851,-33.017567,-71.550621,WiFi,-50,40,57,1780163130480,1780163131331,207
IoT,1780163137259638907,0.696,-0.068,1.132,1,01,689,-33.017616,-71.550613,WiFi,-49,41,56,1780163136561,1780163137250,208

IoT,1780163143383145075,0.548,-0.054,0.799,1,01,732,-33.017677,-71.550613,WiFi,-47,43,55,1780163142641,1780163143373,209
IoT,1780163149878785803,0.559,0.109,0.815,1,01,1142,-33.017719,-71.550644,WiFi,-50,40,55,1780163148725,1780163149867,210
IoT,1780163155617176821,0.495,0.058,0.984,1,01,799,-33.017769,-71.550667,WiFi,-49,41,55,1780163154807,1780163155606,211
IoT,1780163161593985067,0.313,0.058,0.931,1,01,700,-33.017834,-71.550674,WiFi,-54,36,55,1780163160889,1780163161589,212
IoT,1780163167676125344,0.567,-0.039,1.327,1,01,695,-33.017887,-71.55069,WiFi,-50,40,53,1780163166971,1780163167666,213
IoT,1780163173809906540,0.606,-0.057,1.263,1,01,743,-33.017941,-71.55069,WiFi,-51,39,54,1780163173057,1780163173800,214
IoT,1780163179861942659,0.431,0.019,0.952,1,01,711,-33.01799,-71.55069,WiFi,-50,40,54,1780163179140,1780163179851,215
IoT,1780163185958907371,0.657,-0.024,1.112,1,01,726,-33.018036,-71.550697,WiFi,-55,35,54,1780163185222,1780163185948,216
IoT,1780163192428675871,0.546,0.045,1.014,1,01,1120,-33.018085,-71.550705,WiFi,-50,40,55,1780163191302,1780163192422,217
IoT,1780163198076262172,0.455,-0.016,0.966,1,01,683,-33.018139,-71.55072,WiFi,-53,37,55,1780163197383,1780163198066,218
IoT,1780163204254734948,0.731,0.109,1.255,1,01,781,-33.018185,-71.550735,WiFi,-49,41,55,1780163203464,1780163204245,219
IoT,1780163210277603431,0.575,-0.068,0.839,1,01,726,-33.018234,-71.550758,WiFi,-50,40,55,1780163209542,1780163210268,220
IoT,1780163216318599394,0.529,-0.025,1.188,1,01,687,-33.018284,-71.550781,WiFi,-48,42,55,1780163215621,1780163216308,221
IoT,1780163222429865719,0.567,0.074,0.838,1,01,722,-33.018337,-71.550789,WiFi,-50,40,56,1780163221702,1780163222424,222
IoT,1780163228497351248,0.468,-0.178,0.947,1,01,704,-33.01836,-71.550797,WiFi,-51,39,56,1780163227782,1780163228486,223
IoT,1780163235023681082,0.42,-0.211,1.009,1,01,1154,-33.01836,-71.550804,WiFi,-49,41,57,1780163233860,1780163235014,224
IoT,1780163240708863115,0.615,-0.079,0.982,1,01,760,-33.01836,-71.550804,WiFi,-49,41,57,1780163239938,1780163240698,225
IoT,1780163246719314459,0.497,0.119,0.993,1,01,691,-33.018375,-71.550804,WiFi,-47,43,57,1780163246018,1780163246709,226
IoT,1780163252891891913,0.448,-0.066,1.046,1,01,784,-33.018433,-71.550812,WiFi,-49,41,55,1780163252103,1780163252887,227
IoT,1780163258877374201,0.583,-0.023,1.03,1,01,681,-33.018482,-71.550827,WiFi,-50,40,54,1780163258185,1780163258866,228
IoT,1780163265017688698,0.697,-0.093,1.136,1,01,741,-33.018536,-71.550835,WiFi,-48,42,54,1780163264268,1780163265009,229
IoT,1780163271121728194,0.8,-0.122,1.14,1,01,766,-33.018589,-71.55085,WiFi,-49,41,54,1780163270346,1780163271112,230
IoT,1780163277135651844,0.644,0.21,1.049,1,01,701,-33.018639,-71.550858,WiFi,-50,40,53,1780163276425,1780163277126,231
IoT,1780163283188360117,0.564,0.061,0.75,1,01,679,-33.018692,-71.550865,WiFi,-49,41,52,1780163282504,1780163283183,232
IoT,1780163289351232968,0.672,-0.034,1.164,1,01,757,-33.018753,-71.550865,WiFi,-49,41,52,1780163288584,1780163289341,233
IoT,1780163295354782537,0.509,0.094,0.834,1,01,679,-33.018791,-71.55088,WiFi,-48,42,52,1780163294666,1780163295345,234
IoT,1780163301477369082,0.698,-0.169,1.09,1,01,719,-33.018845,-71.550903,WiFi,-49,41,52,1780163300747,1780163301466,235
IoT,1780163307537817496,0.635,0.14,1.024,1,01,695,-33.018894,-71.550911,WiFi,-49,41,52,1780163306829,1780163307524,236
IoT,1780163313769195259,0.537,0.036,0.85,1,01,854,-33.018948,-71.550926,WiFi,-51,39,52,1780163312910,1780163313764,237
IoT,1780163319698760079,0.548,-0.071,0.88,1,01,697,-33.018993,-71.550949,WiFi,-49,41,51,1780163318992,1780163319689,238
IoT,1780163325779527407,0.561,-0.071,0.89,1,01,695,-33.019035,-71.550949,WiFi,-50,40,53,1780163325074,1780163325769,239
IoT,1780163331945767775,0.542,-0.143,1.389,1,01,781,-33.019073,-71.550949,WiFi,-49,41,53,1780163331155,1780163331936,240
IoT,1780163337952706573,0.384,0.054,1.025,1,01,703,-33.019104,-71.550949,WiFi,-48,42,54,1780163337238,1780163337941,241

IoT,1780163344074255698,0.509,0.031,0.985,1,01,750,-33.019154,-71.550957,WiFi,-50,40,51,1780163343319,1780163344069,242
IoT,1780163350099253585,0.424,-0.018,1.051,1,01,693,-33.019203,-71.550957,WiFi,-49,41,41,1780163349396,1780163350089,243
IoT,1780163356216226399,0.465,-0.001,1.272,1,01,731,-33.019257,-71.550957,WiFi,-49,41,47,1780163355475,1780163356206,244
IoT,1780163362358962903,0.574,-0.055,0.919,1,01,793,-33.01931,-71.550964,WiFi,-49,41,54,1780163361556,1780163362349,245
IoT,1780163368358237443,0.496,-0.113,0.86,1,01,709,-33.019363,-71.550957,WiFi,-50,40,53,1780163367639,1780163368348,246
IoT,1780163374439523841,0.661,-0.151,1.123,1,01,710,-33.019421,-71.55098,WiFi,-50,40,51,1780163373723,1780163374433,247
IoT,1780163380518250482,0.524,-0.154,0.851,1,01,704,-33.019466,-71.55101,WiFi,-49,41,51,1780163379804,1780163380508,248
IoT,1780163386657121343,0.594,0.041,0.815,1,01,756,-33.019497,-71.551025,WiFi,-49,41,51,1780163385886,1780163386642,249
IoT,1780163392684002060,0.715,-0.058,0.803,1,01,704,-33.019531,-71.551041,WiFi,-51,39,51,1780163391970,1780163392674,250
IoT,1780163399120209363,0.617,0.072,1.021,1,01,1056,-33.019569,-71.551048,WiFi,-51,39,51,1780163398054,1780163399110,251
IoT,1780163404857341054,0.622,-0.05,1.344,1,01,718,-33.019604,-71.551064,WiFi,-49,41,51,1780163404134,1780163404852,252
IoT,1780163411348375233,0.706,-0.141,1.29,1,01,1125,-33.019653,-71.551079,WiFi,-49,41,51,1780163410211,1780163411336,253
IoT,1780163416999774117,0.594,-0.121,1.013,1,01,705,-33.019703,-71.551086,WiFi,-49,41,51,1780163416283,1780163416988,254
IoT,1780163423081189814,0.471,0.026,0.878,1,01,720,-33.019779,-71.551109,WiFi,-50,40,52,1780163422351,1780163423071,255
IoT,1780163429122814234,0.485,-0.028,0.898,1,01,691,-33.019829,-71.551125,WiFi,-50,40,53,1780163428421,1780163429112,256
IoT,1780163435205747112,0.639,-0.186,1.264,1,01,709,-33.019875,-71.55114,WiFi,-49,41,54,1780163434491,1780163435200,257
IoT,1780163441261941357,0.625,0.2,0.899,1,01,693,-33.019936,-71.55114,WiFi,-50,40,56,1780163440559,1780163441252,258
IoT,1780163447355153391,0.64,-0.07,0.929,1,01,713,-33.019997,-71.55114,WiFi,-48,42,56,1780163446632,1780163447345,259
IoT,1780163453504062957,0.501,-0.034,0.985,1,01,788,-33.02005,-71.551147,WiFi,-49,41,56,1780163452703,1780163453491,260
IoT,1780163459484271503,0.603,-0.162,1.021,1,01,702,-33.020111,-71.551163,WiFi,-49,41,56,1780163458773,1780163459475,261
IoT,1780163465548613747,0.34,0.297,0.891,1,01,696,-33.020164,-71.55117,WiFi,-49,41,54,1780163464847,1780163465543,262
IoT,1780163471723082425,0.431,-0.115,1.017,1,01,792,-33.02021,-71.55117,WiFi,-48,42,55,1780163470920,1780163471712,263
IoT,1780163477695975927,0.309,-0.037,0.969,1,01,693,-33.020256,-71.551186,WiFi,-48,42,56,1780163476993,1780163477686,264
IoT,1780163483794894577,0.572,-0.109,0.919,1,01,721,-33.020309,-71.551186,WiFi,-49,41,55,1780163483064,1780163483785,265
IoT,1780163489846179103,0.42,-0.081,1.623,1,01,703,-33.020351,-71.551201,WiFi,-49,41,55,1780163489133,1780163489836,266
IoT,1780163495991878143,0.34,-0.158,1.011,1,01,779,-33.020412,-71.551224,WiFi,-49,41,56,1780163495207,1780163495986,267
IoT,1780163502012583316,0.389,0.031,0.844,1,01,725,-33.020481,-71.551247,WiFi,-50,40,57,1780163501276,1780163502001,268
IoT,1780163508065166943,0.57,0.137,0.801,1,01,711,-33.020535,-71.551254,WiFi,-49,41,57,1780163507344,1780163508055,269
IoT,1780163514167967964,0.48,0.046,0.907,1,01,743,-33.020588,-71.55127,WiFi,-51,39,57,1780163513414,1780163514157,270
IoT,1780163520362078092,0.634,-0.035,1.247,1,01,867,-33.020638,-71.551277,WiFi,-49,41,56,1780163519485,1780163520352,271
IoT,1780163526308416520,0.476,0.097,0.931,1,01,740,-33.020683,-71.551277,WiFi,-50,40,56,1780163525556,1780163526296,272
IoT,1780163532447381829,0.305,-0.017,1.526,1,01,814,-33.020733,-71.551285,WiFi,-51,39,55,1780163531623,1780163532437,273
IoT,1780163538418122336,0.697,0,1.186,1,01,711,-33.020794,-71.551292,WiFi,-50,40,51,1780163537697,1780163538408,274

IoT,1780163544514204295,0.443,0.084,0.826,1,01,734,-33.020855,-71.551315,WiFi,-52,38,49,1780163543770,1780163544504,275
IoT,1780163550570232696,0.516,0.08,1.373,1,01,718,-33.020912,-71.551331,WiFi,-49,41,49,1780163549841,1780163550559,276
IoT,1780163556713516196,0.584,-0.064,0.93,1,01,788,-33.020969,-71.551338,WiFi,-51,39,50,1780163555915,1780163556703,277
IoT,1780163562734156721,0.357,0.064,0.893,1,01,736,-33.021027,-71.551331,WiFi,-50,40,52,1780163561987,1780163562723,278
IoT,1780163568768951292,0.488,-0.081,1.404,1,01,702,-33.021072,-71.551353,WiFi,-50,40,53,1780163568057,1780163568759,279
IoT,1780163575283136335,0.523,-0.051,1.459,1,01,1148,-33.021107,-71.551369,WiFi,-49,41,52,1780163574125,1780163575273,280
IoT,1780163580932322723,0.487,0.055,0.739,1,01,725,-33.021156,-71.551376,WiFi,-50,40,52,1780163580197,1780163580922,281
IoT,1780163586962260291,0.41,0.011,0.908,1,01,686,-33.021225,-71.551399,WiFi,-52,38,52,1780163586266,1780163586952,282
IoT,1780163593064956510,0.585,0.169,1.139,1,01,724,-33.021278,-71.551407,WiFi,-48,42,52,1780163592331,1780163593055,283
IoT,1780163599216378548,0.648,-0.1,1.061,1,01,809,-33.021328,-71.551407,WiFi,-49,41,52,1780163598394,1780163599203,284
IoT,1780163605171853796,0.434,0.145,1.016,1,01,700,-33.021389,-71.551414,WiFi,-51,39,52,1780163604460,1780163605160,285
IoT,1780163611269878512,0.466,0.077,0.889,1,01,736,-33.021439,-71.551437,WiFi,-49,41,52,1780163610523,1780163611259,286
IoT,1780163617396501591,0.211,-0.217,1.062,1,01,796,-33.021503,-71.551453,WiFi,-51,39,52,1780163616589,1780163617385,287
IoT,1780163623393797235,0.408,0.274,0.983,1,01,727,-33.021561,-71.551468,WiFi,-46,44,49,1780163622657,1780163623384,288
IoT,1780163629426672995,0.566,-0.114,0.871,1,01,694,-33.021629,-71.551483,WiFi,-50,40,49,1780163628723,1780163629417,289
IoT,1780163635513322117,0.691,-0.055,0.994,1,01,713,-33.021664,-71.551491,WiFi,-49,41,49,1780163634789,1780163635502,290
IoT,1780163641599728632,0.47,-0.3,1.265,1,01,739,-33.021713,-71.551491,WiFi,-52,38,49,1780163640853,1780163641592,291
IoT,1780163647651320852,0.576,-0.158,1.352,1,01,721,-33.021748,-71.551498,WiFi,-53,37,49,1780163646921,1780163647642,292
IoT,1780163653789269468,0.559,-0.202,1.277,1,01,791,-33.021793,-71.551498,WiFi,-51,39,49,1780163652989,1780163653780,293
IoT,1780163659763360972,0.633,-0.083,1.003,1,01,698,-33.021832,-71.551514,WiFi,-53,37,49,1780163659055,1780163659753,294
IoT,1780163665828730180,0.6,-0.077,0.909,1,01,699,-33.021854,-71.551514,WiFi,-51,39,49,1780163665120,1780163665819,295
IoT,1780163671927426778,0.673,-0.18,1.196,1,01,724,-33.021893,-71.551521,WiFi,-52,38,49,1780163671189,1780163671913,296
IoT,1780163678060392713,0.527,-0.049,0.856,1,01,795,-33.021938,-71.551529,WiFi,-54,36,49,1780163677254,1780163678049,297
IoT,1780163684068048313,0.411,-0.031,0.935,1,01,740,-33.021984,-71.551529,WiFi,-49,41,50,1780163683318,1780163684058,298
IoT,1780163690142834577,0.467,-0.03,1.077,1,01,751,-33.022022,-71.551537,WiFi,-50,40,53,1780163689380,1780163690131,299
IoT,1780163696183577553,0.326,0.122,1.022,1,01,729,-33.022034,-71.551544,WiFi,-41,49,43,1780163695444,1780163696173,300

Datos de prototipo LoRa

name,time,ax,ay,az,gps_ok,id_device,latencia,latitud,longitud,red,rssi,snr_rssnr,soc,time_device,time_server,tx
IoT,1781289162453757814,1.04,1.25,-0.49,1,02,507,-33.02214,-71.55135,LoRa,-79,8.75,57,1781289161932,1781289162439,53
IoT,1781289168484014540,1.05,1.42,-0.62,1,02,546,-33.02209,-71.55134,LoRa,-80,9.25,57,1781289167932,1781289168478,54
IoT,1781289174534823523,1.04,1.31,-0.62,1,02,589,-33.02211,-71.55133,LoRa,-85,6.25,57,1781289173932,1781289174521,55
IoT,1781289180360167502,1.04,1.27,-0.59,1,02,423,-33.02213,-71.55137,LoRa,-86,5.75,57,1781289179932,1781289180355,56
IoT,1781289186513973136,0.91,1.37,-0.58,1,02,571,-33.02215,-71.55141,LoRa,-88,3.75,57,1781289185932,1781289186503,57
IoT,1781289192385438755,1.16,1.34,-0.59,1,02,441,-33.02217,-71.55145,LoRa,-86,6.25,57,1781289191932,1781289192373,58
IoT,1781289204440260675,1.2,1.31,-0.69,1,02,498,-33.02215,-71.55156,LoRa,-84,8,57,1781289203932,1781289204430,60
IoT,1781289210419056856,1.23,1.23,-0.65,1,02,477,-33.02221,-71.55161,LoRa,-90,5,57,1781289209932,1781289210409,61
IoT,1781289216517414230,0.99,1.29,-0.77,1,02,575,-33.02218,-71.55161,LoRa,-84,7.75,57,1781289215932,1781289216507,62
IoT,1781289222457374973,1.04,1.24,-0.7,1,02,515,-33.02218,-71.55156,LoRa,-90,4.25,57,1781289221932,1781289222447,63
IoT,1781289234437078905,0.9,1.33,-0.45,1,02,494,-33.0223,-71.55156,LoRa,-92,-1,57,1781289233932,1781289234426,65
IoT,1781289240470434677,0.88,1.28,-0.48,1,02,533,-33.02229,-71.55157,LoRa,-90,5,56,1781289239932,1781289240465,66
IoT,1781289246410898033,1.08,1.28,-0.49,1,02,474,-33.02227,-71.55157,LoRa,-96,-4.75,56,1781289245932,1781289246406,67
IoT,1781289252393237536,1.1,1.28,-0.46,1,02,450,-33.02228,-71.55157,LoRa,-92,2.5,56,1781289251932,1781289252382,68
IoT,1781289258496199478,1.15,1.31,-0.53,1,02,557,-33.02232,-71.55159,LoRa,-91,2.5,57,1781289257932,1781289258489,69
IoT,1781289264442470922,1.21,1.32,-0.46,1,02,498,-33.02235,-71.55161,LoRa,-94,-0.75,57,1781289263932,1781289264430,70
IoT,1781289270420091270,1.21,1.18,-0.66,1,02,482,-33.02245,-71.55164,LoRa,-91,3.25,57,1781289269932,1781289270414,71
IoT,1781289276825283546,1.11,1.09,-0.51,1,02,887,-33.02263,-71.55165,LoRa,-95,-2.5,57,1781289275932,1781289276819,72
IoT,1781289294445033471,0.91,1.13,-0.68,1,02,502,-33.02269,-71.55165,LoRa,-95,-3.25,52,1781289293932,1781289294434,75
IoT,1781289312381485393,1.09,0.98,-0.55,1,02,437,-33.02303,-71.5517,LoRa,-96,-5.75,51,1781289311932,1781289312369,78
IoT,1781289336436208549,1.22,0.98,-0.32,1,02,487,-33.02343,-71.55177,LoRa,-96,-8.5,54,1781289335932,1781289336419,82
IoT,1781289360425265813,1.36,1.18,-0.65,1,02,487,-33.02356,-71.55167,LoRa,-96,-6,52,1781289359932,1781289360419,86
IoT,1781289427457952681,1.09,0.84,-0.39,1,02,1519,-33.02382,-71.55167,LoRa,-95,-5.75,52,1781289425932,1781289427451,97
IoT,1781289432478179666,1.18,1.14,-0.26,1,02,532,-33.02383,-71.55161,LoRa,-96,-8,52,1781289431932,1781289432464,98
IoT,1781289444566107551,0.92,1.12,-0.67,1,02,623,-33.02364,-71.55168,LoRa,-96,-5,53,1781289443932,1781289444555,100
IoT,1781289498876441061,1.02,1.08,-0.41,1,02,938,-33.02326,-71.55111,LoRa,-95,-5.25,46,1781289497932,1781289498870,109
IoT,1781289504473916561,0.82,0.88,-0.85,1,02,529,-33.02296,-71.55087,LoRa,-93,0.25,52,1781289503932,1781289504461,110
IoT,1781289510410232628,0.7,1.32,-0.81,1,02,470,-33.02281,-71.55074,LoRa,-90,1.75,52,1781289509932,1781289510402,111
IoT,1781289516432566680,0.77,1.31,-0.77,1,02,490,-33.02281,-71.55074,LoRa,-88,3.5,50,1781289515932,1781289516422,112
IoT,1781289522485927489,0.87,0.91,-0.79,1,02,548,-33.02281,-71.55074,LoRa,-90,1.5,50,1781289521932,1781289522480,113
IoT,1781289528527968883,0.85,0.94,-0.56,1,02,590,-33.02281,-71.55074,LoRa,-91,1.75,45,1781289527932,1781289528522,114
IoT,1781289534580185392,0.91,1.13,-0.71,1,02,634,-33.02281,-71.55074,LoRa,-89,4.75,45,1781289533932,1781289534566,115

IoT,1781289540415321476,1.31,1.12,-0.5,1,02,472,-33.02281,-71.55074,LoRa,-90,5.75,47,1781289539932,1781289540404,116
IoT,1781289552387258634,1.17,0.95,-0.69,1,02,446,-33.02281,-71.55074,LoRa,-90,5,43,1781289551932,1781289552378,118
IoT,1781289558390482370,1.26,1.13,-0.83,1,02,453,-33.02281,-71.55074,LoRa,-85,7,42,1781289557932,1781289558385,119
IoT,1781289564479634927,1.19,0.94,-0.94,1,02,535,-33.02281,-71.55074,LoRa,-88,7.25,45,1781289563932,1781289564467,120
IoT,1781289570520207817,1.15,0.89,-0.66,1,02,577,-33.02281,-71.55074,LoRa,-85,7.5,43,1781289569932,1781289570509,121
IoT,1781289577099276146,1.25,0.92,-0.73,1,02,1155,-33.02281,-71.55074,LoRa,-90,4.25,41,1781289575932,1781289577087,122
IoT,1781289582434393056,1.31,0.9,-0.73,1,02,496,-33.02281,-71.55074,LoRa,-89,5,40,1781289581932,1781289582428,123
IoT,1781289588557618297,1.25,0.94,-0.69,1,02,605,-33.02281,-71.55074,LoRa,-93,1,44,1781289587932,1781289588537,124
IoT,1781289594482451864,1.25,0.95,-0.67,1,02,539,-33.02262,-71.55139,LoRa,-92,2.25,40,1781289593932,1781289594471,125
IoT,1781289600423088040,1.05,0.93,-0.68,1,02,478,-33.02269,-71.55138,LoRa,-89,5.5,43,1781289599932,1781289600410,126
IoT,1781289606462282318,1.17,0.93,-0.76,1,02,520,-33.02268,-71.55136,LoRa,-92,2.75,46,1781289605932,1781289606452,127
IoT,1781289612496980118,1.23,0.94,-0.53,1,02,559,-33.02266,-71.55128,LoRa,-92,3,42,1781289611932,1781289612491,128
IoT,1781289618419109368,1.22,1.08,-0.66,1,02,470,-33.02262,-71.55119,LoRa,-85,7.75,45,1781289617932,1781289618402,129
IoT,1781289624488639847,1.11,0.91,-0.66,1,02,545,-33.02258,-71.55116,LoRa,-90,5,43,1781289623932,1781289624477,130
IoT,1781289630527854731,1.28,1.02,-0.66,1,02,584,-33.02257,-71.55114,LoRa,-87,6.5,43,1781289629932,1781289630516,131
IoT,1781289636467145731,0.97,1.22,-0.67,1,02,524,-33.02257,-71.55113,LoRa,-93,2,53,1781289635932,1781289636456,132
IoT,1781289642448496342,0.97,1.17,-0.63,1,02,511,-33.02256,-71.55116,LoRa,-92,3,53,1781289641932,1781289642443,133
IoT,1781289648468621104,0.94,1.15,-0.6,1,02,526,-33.0226,-71.55116,LoRa,-88,6.75,54,1781289647932,1781289648458,134
IoT,1781289654389371699,0.83,1.14,-0.93,1,02,444,-33.02259,-71.55119,LoRa,-95,-3.75,54,1781289653932,1781289654376,135
IoT,1781289660429020066,0.88,1.09,-0.51,1,02,486,-33.02255,-71.55119,LoRa,-93,1.75,54,1781289659932,1781289660418,136
IoT,1781289666470116278,0.82,1.19,-0.58,1,02,527,-33.02254,-71.55121,LoRa,-88,6.75,55,1781289665932,1781289666459,137
IoT,1781289672505257390,0.9,1.1,-0.58,1,02,567,-33.02254,-71.55122,LoRa,-92,2.5,56,1781289671932,1781289672499,138
IoT,1781289678444004231,0.88,0.95,-0.59,1,02,506,-33.02255,-71.55122,LoRa,-89,6.75,54,1781289677932,1781289678438,139
IoT,1781289684442777168,0.92,0.93,-0.89,1,02,496,-33.02254,-71.55128,LoRa,-84,8.25,55,1781289683932,1781289684428,140
IoT,1781289690534205047,1.02,1.11,-0.5,1,02,591,-33.02254,-71.55132,LoRa,-82,9.25,55,1781289689932,1781289690523,141
IoT,1781289696473984160,1.11,0.99,-0.53,1,02,531,-33.02253,-71.55134,LoRa,-85,8,55,1781289695932,1781289696463,142
IoT,1781289702412921053,0.82,1.02,-0.87,1,02,470,-33.02251,-71.55138,LoRa,-77,10.25,54,1781289701932,1781289702402,143
IoT,1781289708392534212,1.04,0.95,-0.85,1,02,450,-33.02248,-71.55138,LoRa,-91,3.5,54,1781289707932,1781289708382,144
IoT,1781289714399036203,0.95,1.08,-0.72,1,02,460,-33.02246,-71.55138,LoRa,-87,7,55,1781289713932,1781289714392,145
IoT,1781289720538111827,0.91,0.86,-0.59,1,02,596,-33.02246,-71.55138,LoRa,-86,7.25,54,1781289719932,1781289720528,146
IoT,1781289726470546962,0.93,0.84,-0.61,1,02,533,-33.02246,-71.55137,LoRa,-75,9.5,55,1781289725932,1781289726465,147
IoT,1781289732519692903,1.05,0.96,-0.67,1,02,577,-33.02244,-71.55133,LoRa,-77,10,55,1781289731932,1781289732509,148
IoT,1781289738458946725,1.01,1.09,-0.31,1,02,515,-33.02238,-71.55132,LoRa,-69,9.5,56,1781289737932,1781289738447,149

IoT,1781289744493398123,0.78,0.97,-1.15,1,02,555,-33.02232,-71.55131,LoRa,-69,9.75,56,1781289743932,1781289744487,150
IoT,1781289750622416488,1.05,1.07,-0.62,1,02,676,-33.02227,-71.55132,LoRa,-71,10.5,56,1781289749932,1781289750608,151
IoT,1781289756475515197,1.13,1.08,-0.6,1,02,536,-33.02221,-71.55134,LoRa,-75,9.75,55,1781289755932,1781289756468,152
IoT,1781289762822905009,0.9,1.12,-0.86,1,02,885,-33.02213,-71.55135,LoRa,-76,9.25,56,1781289761932,1781289762817,153
IoT,1781289768416055159,0.99,1.12,-0.53,1,02,472,-33.02205,-71.55136,LoRa,-91,3.75,56,1781289767932,1781289768404,154
IoT,1781289774498975402,0.89,0.78,-0.8,1,02,559,-33.02201,-71.55136,LoRa,-80,9.5,55,1781289773932,1781289774491,155
IoT,1781289780444570608,0.93,0.83,-0.39,1,02,500,-33.02195,-71.55135,LoRa,-75,9.75,56,1781289779932,1781289780432,156
IoT,1781289786479752361,1.15,0.81,-0.72,1,02,539,-33.02186,-71.5513,LoRa,-78,9.25,54,1781289785932,1781289786471,157
IoT,1781289792518185802,0.89,0.78,-0.71,1,02,581,-33.02176,-71.55125,LoRa,-78,10,55,1781289791932,1781289792513,158
IoT,1781289798453171731,0.98,1.26,-0.39,1,02,516,-33.02168,-71.5512,LoRa,-81,9.25,55,1781289797932,1781289798448,159
IoT,1781289810445879964,0.85,0.84,-0.55,1,02,503,-33.02151,-71.55112,LoRa,-89,5.75,55,1781289809932,1781289810435,161
IoT,1781289816479677403,0.81,1.02,-0.49,1,02,542,-33.02142,-71.55112,LoRa,-83,8.5,55,1781289815932,1781289816474,162
IoT,1781289822523171256,0.92,1.22,-0.66,1,02,584,-33.02139,-71.55112,LoRa,-86,7.25,55,1781289821932,1781289822516,163
IoT,1781289828419530758,0.98,1.21,-0.44,1,02,481,-33.02135,-71.55111,LoRa,-87,6.75,55,1781289827932,1781289828413,164
IoT,1781289840446237871,1,1.19,-0.77,1,02,503,-33.02129,-71.55113,LoRa,-87,7,55,1781289839932,1781289840435,166
IoT,1781289846490525567,0.95,0.98,-0.46,1,02,548,-33.02127,-71.55114,LoRa,-88,5.75,54,1781289845932,1781289846480,167
IoT,1781289852526340839,0.91,1.02,-0.45,1,02,588,-33.0212,-71.55114,LoRa,-89,5.75,55,1781289851932,1781289852520,168
IoT,1781289864403290488,1.08,0.84,-0.51,1,02,466,-33.02113,-71.55114,LoRa,-94,-1,55,1781289863932,1781289864398,170
IoT,1781289870455252451,1.07,0.82,-0.7,1,02,510,-33.02111,-71.55115,LoRa,-90,4.5,54,1781289869932,1781289870442,171
IoT,1781289876589343710,0.94,0.87,-0.86,1,02,652,-33.0211,-71.55115,LoRa,-92,2.5,54,1781289875932,1781289876584,172
IoT,1781289882425936567,0.99,1.15,-0.29,1,02,489,-33.02106,-71.55114,LoRa,-94,-8.5,52,1781289881932,1781289882421,173
IoT,1781289888474706606,1.08,1.12,-0.79,1,02,532,-33.02101,-71.55112,LoRa,-90,4.5,54,1781289887932,1781289888464,174
IoT,1781289906490748476,1.04,1.07,-0.53,1,02,553,-33.02082,-71.55105,LoRa,-96,-5.25,56,1781289905932,1781289906485,177
IoT,1781289924472029959,0.94,0.78,-0.36,1,02,527,-33.0206,-71.55099,LoRa,-94,-0.25,56,1781289923932,1781289924459,180
IoT,1781289936391168559,1.02,1.13,-0.56,1,02,454,-33.0205,-71.55096,LoRa,-95,-0.25,55,1781289935932,1781289936386,182
IoT,1781289942432509918,1.09,1.09,-0.61,1,02,495,-33.02044,-71.55093,LoRa,-95,-2.5,55,1781289941932,1781289942427,183

Datos de prototipo 4G

name,time,ax,ay,az,gps_ok,id_device,latencia,latitud,longitud,red,rssi,snr_rssnr,soc,time_device,time_server,tx
IoT,1779311597616591534,-0.076,0.293,1.26,1,03,380,-33.021954,-71.55172,4G,-51,11,0,1779311597235,1779311597615,64
IoT,1779311603640976314,0.044,0.381,0.964,1,03,389,-33.021988,-71.551804,4G,-51,3,0,1779311603245,1779311603634,65
IoT,1779311609721410322,-0.014,0.194,1.247,1,03,465,-33.022018,-71.551834,4G,-51,4,0,1779311609249,1779311609714,66
IoT,1779311615636718735,-0.215,0.178,0.903,1,03,375,-33.022049,-71.551826,4G,-51,7,0,1779311615255,1779311615630,67
IoT,1779311621644598068,-0.186,0.079,0.987,1,03,378,-33.022076,-71.551796,4G,-51,6,0,1779311621259,1779311621637,68
IoT,1779311627673127775,-0.257,0.093,1.008,1,03,401,-33.022129,-71.551796,4G,-51,5,0,1779311627265,1779311627666,69
IoT,1779311633665131904,-0.132,0.056,0.912,1,03,392,-33.02216,-71.551895,4G,-51,8,0,1779311633269,1779311633661,70
IoT,1779311639715719051,0.08,0.203,1.049,1,03,425,-33.022259,-71.551865,4G,-51,3,0,1779311639283,1779311639708,71
IoT,1779311645680796004,-0.216,0.222,0.999,1,03,388,-33.022297,-71.551903,4G,-51,4,0,1779311645289,1779311645677,72
IoT,1779311651685953435,-0.111,0.23,0.993,1,03,382,-33.022358,-71.551933,4G,-51,2,0,1779311651297,1779311651679,73
IoT,1779311657747330029,-0.144,0.179,0.834,1,03,435,-33.022511,-71.551849,4G,-51,0,0,1779311657301,1779311657736,74
IoT,1779311663772208690,-0.005,0.054,1.197,1,03,458,-33.02253,-71.551903,4G,-51,-2,0,1779311663308,1779311663766,75
IoT,1779311669712991719,-0.048,0.319,1.091,1,03,395,-33.022591,-71.551888,4G,-51,-3,0,1779311669312,1779311669707,76
IoT,1779311675752525611,-0.193,0.118,1.01,1,03,433,-33.022682,-71.551743,4G,-51,0,0,1779311675316,1779311675749,77
IoT,1779311681805838936,-0.305,0.205,1.138,1,03,480,-33.022758,-71.551674,4G,-51,-3,0,1779311681319,1779311681799,78
IoT,1779311687717503532,0.001,0.131,0.952,1,03,386,-33.022861,-71.551781,4G,-51,-2,0,1779311687324,1779311687710,79
IoT,1779311693712097840,-0.177,0.014,1.038,1,03,373,-33.022915,-71.551819,4G,-51,1,0,1779311693334,1779311693707,80
IoT,1779311699824011822,-0.217,0.149,0.991,1,03,472,-33.022934,-71.551865,4G,-51,2,0,1779311699344,1779311699816,81
IoT,1779311705743438764,0.098,0.213,1.183,1,03,394,-33.022976,-71.55188,4G,-51,2,0,1779311705345,1779311705739,82
IoT,1779311711756106114,-0.26,0.17,0.851,1,03,400,-33.023018,-71.551842,4G,-51,-3,0,1779311711349,1779311711749,83
IoT,1779311717751579022,0.05,0.291,1.312,1,03,391,-33.023003,-71.551811,4G,-51,3,0,1779311717353,1779311717744,84
IoT,1779311723783373392,-0.09,0.063,0.945,1,03,420,-33.023052,-71.551804,4G,-51,3,0,1779311723358,1779311723778,85
IoT,1779311729827718710,0.252,0.016,1.038,1,03,457,-33.023113,-71.551811,4G,-51,3,0,1779311729363,1779311729820,86
IoT,1779311735764861807,-0.207,0.257,0.909,1,03,391,-33.02317,-71.551811,4G,-51,2,0,1779311735369,1779311735760,87
IoT,1779311741767493929,-0.206,0.255,1.199,1,03,382,-33.023205,-71.551811,4G,-51,0,0,1779311741378,1779311741760,88
IoT,1779311747849136918,0.062,0.099,1.107,1,03,455,-33.023285,-71.551826,4G,-51,-1,0,1779311747387,1779311747842,89
IoT,1779311753780331134,0.08,0.166,0.982,1,03,385,-33.023346,-71.551857,4G,-51,-13,0,1779311753391,1779311753776,90
IoT,1779311759841800531,-0.464,0.257,0.771,1,03,434,-33.02343,-71.551872,4G,-51,-1,0,1779311759398,1779311759832,91
IoT,1779311765868772120,-0.124,0.113,1.271,1,03,466,-33.023506,-71.55188,4G,-51,0,0,1779311765399,1779311765865,92
IoT,1779311771792583228,-0.264,0.157,1.081,1,03,380,-33.023563,-71.551926,4G,-51,4,0,1779311771405,1779311771785,93
IoT,1779311777807154415,-0.231,0.201,1.1,1,03,392,-33.023575,-71.551926,4G,-51,3,0,1779311777408,1779311777800,94
IoT,1779311783785082585,-0.319,0.138,1.092,1,03,369,-33.023602,-71.551949,4G,-51,1,0,1779311783411,1779311783780,95

IoT,1779311789936585364,-0.33,0.225,1.023,1,03,509,-33.023613,-71.551964,4G,-51,8,0,1779311789420,1779311789929,96
IoT,1779311795822319829,0.122,0.162,1.131,1,03,389,-33.02364,-71.551994,4G,-51,15,0,1779311795429,1779311795818,97
IoT,1779311801841719832,-0.111,0.116,1.197,1,03,401,-33.023651,-71.552017,4G,-51,10,0,1779311801434,1779311801835,98
IoT,1779311807861103475,-0.158,0.152,1.201,1,03,414,-33.023682,-71.55204,4G,-51,8,0,1779311807440,1779311807854,99
IoT,1779311813832881783,-0.163,0.222,1.131,1,03,384,-33.023678,-71.552063,4G,-51,10,0,1779311813444,1779311813828,100
IoT,1779311819940330003,-0.206,0.229,0.942,1,03,486,-33.023689,-71.552048,4G,-51,7,0,1779311819447,1779311819933,101
IoT,1779311825845853619,-0.218,0.369,1.022,1,03,393,-33.023769,-71.551941,4G,-51,3,0,1779311825448,1779311825841,102
IoT,1779311831842810959,-0.026,0.094,1.238,1,03,382,-33.023853,-71.551918,4G,-51,7,0,1779311831451,1779311831833,103
IoT,1779311837857317479,-0.202,0.114,0.899,1,03,389,-33.023891,-71.551872,4G,-51,4,0,1779311837461,1779311837850,104
IoT,1779311843900283557,-0.163,0.024,1.333,1,03,426,-33.023899,-71.551826,4G,-51,2,0,1779311843470,1779311843896,105
IoT,1779311849871985155,-0.183,0.075,0.917,1,03,389,-33.023903,-71.551781,4G,-51,2,0,1779311849474,1779311849863,106
IoT,1779311855871146041,-0.024,-0.341,1.27,1,03,386,-33.023903,-71.55175,4G,-51,-6,0,1779311855478,1779311855864,107
IoT,1779311862299574689,0.122,0.12,1.15,1,03,809,-33.023884,-71.551743,4G,-51,-1,0,1779311861483,1779311862292,108
IoT,1779311867883130885,0.09,0.275,1.258,1,03,388,-33.023834,-71.55172,4G,-51,-4,0,1779311867486,1779311867874,109
IoT,1779311873863617027,0.123,0.244,1.34,1,03,369,-33.023769,-71.551712,4G,-51,-7,0,1779311873491,1779311873860,110
IoT,1779311879945218688,-0.367,0.203,1.157,1,03,443,-33.02372,-71.551735,4G,-51,2,0,1779311879495,1779311879938,111
IoT,1779311885905664624,-0.176,0.233,1.349,1,03,396,-33.023674,-71.55175,4G,-51,-3,0,1779311885503,1779311885899,112
IoT,1779311893565313803,-0.268,0.213,1.183,1,03,2045,-33.023636,-71.551765,4G,-51,-9,0,1779311891512,1779311893557,113
IoT,1779311897921707010,-0.228,0.096,0.955,1,03,401,-33.023609,-71.551704,4G,-51,-4,0,1779311897517,1779311897918,114
IoT,1779311904008965637,-0.126,0.174,1.196,1,03,483,-33.023575,-71.551682,4G,-51,3,0,1779311903521,1779311904004,115
IoT,1779311909923106078,0.102,-0.175,1.279,1,03,391,-33.023514,-71.551636,4G,-51,-6,0,1779311909525,1779311909916,116
IoT,1779311915916069076,0.196,0.396,1.206,1,03,384,-33.023479,-71.551613,4G,-51,-3,0,1779311915528,1779311915912,117
IoT,1779311922346064929,-0.025,0.115,1.327,1,03,801,-33.023407,-71.551613,4G,-51,0,0,1779311921532,1779311922333,118
IoT,1779311927962640355,-0.188,0.215,1.144,1,03,418,-33.02335,-71.551582,4G,-51,-9,0,1779311927537,1779311927955,119
IoT,1779311934013337673,-0.051,-0.023,1.181,1,03,462,-33.023296,-71.551598,4G,-51,-3,0,1779311933547,1779311934009,120
IoT,1779311939955841373,0.085,0.176,1.16,1,03,392,-33.023247,-71.55159,4G,-51,-4,0,1779311939556,1779311939948,121
IoT,1779311945958014833,-0.084,-0.055,1.255,1,03,385,-33.023186,-71.551559,4G,-51,2,0,1779311945566,1779311945951,122
IoT,1779311952239847858,-0.273,0.219,1.008,1,03,662,-33.023121,-71.551529,4G,-51,0,0,1779311951570,1779311952232,123
IoT,1779311957954652916,-0.15,0.032,1.403,1,03,374,-33.023052,-71.551514,4G,-51,-2,0,1779311957574,1779311957948,124
IoT,1779311963983367115,-0.271,0.188,1.212,1,03,402,-33.022995,-71.551498,4G,-51,4,0,1779311963577,1779311963979,125
IoT,1779311969992281885,-0.336,0.268,1.111,1,03,398,-33.022968,-71.551491,4G,-51,1,0,1779311969582,1779311969980,126
IoT,1779311975998743104,-0.204,0.048,1.314,1,03,405,-33.022919,-71.551483,4G,-51,0,0,1779311975586,1779311975991,127
IoT,1779311982000283540,-0.147,0.244,1.183,1,03,400,-33.022869,-71.551476,4G,-51,1,0,1779311981596,1779311981996,128

IoT,1779311987985638866,0.175,0.029,1.189,1,03,374,-33.022808,-71.551468,4G,-51,3,0,1779311987605,1779311987979,129
IoT,1779311994014536611,-0.186,0.247,1.016,1,03,402,-33.022747,-71.551453,4G,-51,1,0,1779311993609,1779311994011,130
IoT,1779312000065123006,-0.018,0.259,1.06,1,03,443,-33.022697,-71.551437,4G,-51,1,0,1779311999613,1779312000056,131
IoT,1779312006017165286,-0.158,0.011,0.991,1,03,387,-33.022636,-71.551414,4G,-51,2,0,1779312005623,1779312006010,132
IoT,1779312012039906412,0.029,0.145,0.997,1,03,410,-33.022564,-71.551399,4G,-51,5,0,1779312011626,1779312012036,133
IoT,1779312018017763045,-0.27,0.218,1.076,1,03,382,-33.022488,-71.551376,4G,-51,1,0,1779312017629,1779312018011,134
IoT,1779312024444533590,-0.291,0.251,0.927,1,03,808,-33.022411,-71.551353,4G,-51,6,0,1779312023633,1779312024441,135
IoT,1779312030036779444,-0.016,0.216,0.772,1,03,387,-33.022369,-71.551331,4G,-51,5,0,1779312029643,1779312030030,136
IoT,1779312036028120773,0.24,0.091,1.351,1,03,373,-33.022316,-71.5513,4G,-51,4,0,1779312035651,1779312036024,137
IoT,1779312042033646388,-0.118,0.249,1.182,1,03,372,-33.022285,-71.551277,4G,-51,5,0,1779312041655,1779312042027,138
IoT,1779312048087713339,-0.188,0.121,1.275,1,03,421,-33.022259,-71.551262,4G,-51,6,0,1779312047660,1779312048081,139
IoT,1779312054123553872,-0.062,-0.097,0.794,1,03,457,-33.022228,-71.551254,4G,-51,7,0,1779312053663,1779312054120,140
IoT,1779312060071886977,0.042,-0.194,1.081,1,03,397,-33.022141,-71.551247,4G,-51,4,0,1779312059667,1779312060064,141
IoT,1779312066085887934,-0.067,0.144,0.872,1,03,400,-33.022076,-71.551239,4G,-51,5,0,1779312065676,1779312066076,142
IoT,1779312072081148379,0.115,0.084,1.299,1,03,397,-33.022049,-71.551231,4G,-51,9,0,1779312071680,1779312072077,143
IoT,1779312078084362232,0.063,0.069,1.026,1,03,387,-33.021973,-71.551216,4G,-51,0,0,1779312077689,1779312078076,144
IoT,1779312084128269735,-0.187,0.161,0.922,1,03,428,-33.021908,-71.551208,4G,-51,3,0,1779312083696,1779312084124,145
IoT,1779312090698866312,-0.098,0.236,1.321,1,03,993,-33.021832,-71.551193,4G,-51,2,0,1779312089698,1779312090691,146
IoT,1779312096103086917,-0.231,0.1,0.953,1,03,397,-33.021751,-71.551163,4G,-51,2,0,1779312095699,1779312096096,147
IoT,1779312102148986879,0.258,0.085,1.067,1,03,442,-33.021706,-71.55117,4G,-51,0,0,1779312101702,1779312102144,148
IoT,1779312108199204993,0.256,0.01,1.009,1,03,487,-33.021648,-71.551201,4G,-51,-3,0,1779312107704,1779312108191,149
IoT,1779312114092857912,-0.11,0.216,1.19,1,03,383,-33.021599,-71.551193,4G,-51,0,0,1779312113706,1779312114089,150
IoT,1779312120128145286,-0.215,0.211,1.159,1,03,402,-33.021542,-71.551193,4G,-51,5,0,1779312119718,1779312120120,151
IoT,1779312126119788204,-0.361,0.047,1.163,1,03,390,-33.0215,-71.551178,4G,-51,2,0,1779312125722,1779312126112,152
IoT,1779312132152219783,-0.287,0.019,1.109,1,03,417,-33.021454,-71.551163,4G,-51,2,0,1779312131730,1779312132147,153
IoT,1779312138116726632,-0.159,0.17,1.3,1,03,378,-33.021385,-71.55114,4G,-51,-1,0,1779312137731,1779312138109,154
IoT,1779312144131898580,-0.346,0.105,1.019,1,03,394,-33.021324,-71.551125,4G,-51,-3,0,1779312143734,1779312144128,155
IoT,1779312150183377464,-0.275,0.219,1.093,1,03,438,-33.02124,-71.551102,4G,-51,-4,0,1779312149737,1779312150175,156
IoT,1779312156215036624,-0.171,0.247,1.172,1,03,472,-33.02116,-71.551079,4G,-51,-6,0,1779312155739,1779312156211,157
IoT,1779312162173215719,-0.126,0.285,1.394,1,03,425,-33.021107,-71.551079,4G,-51,-10,0,1779312161741,1779312162166,158
IoT,1779312168169119434,0.17,0.195,1.215,1,03,407,-33.021069,-71.551086,4G,-51,-13,0,1779312167755,1779312168162,159
IoT,1779312174237741510,-0.403,-0.048,0.999,1,03,471,-33.021015,-71.551102,4G,-51,-14,0,1779312173763,1779312174234,160
IoT,1779312180224106154,-0.241,0.275,1.153,1,03,445,-33.020924,-71.551079,4G,-55,-13,0,1779312179772,1779312180217,161

IoT,1779312186167167430,0.303,0.143,1.169,1,03,380,-33.020844,-71.551056,4G,-55,-7,0,1779312185776,1779312186156,162
IoT,1779312192158500386,-0.116,0.128,1.111,1,03,375,-33.020794,-71.551041,4G,-55,-8,0,1779312191779,1779312192154,163
IoT,1779312198180684428,-0.409,0.215,0.937,1,03,392,-33.020748,-71.551041,4G,-55,-8,0,1779312197781,1779312198173,164
IoT,1779312204248318554,-0.219,0.157,1.013,1,03,456,-33.020714,-71.551025,4G,-55,3,0,1779312203788,1779312204244,165
IoT,1779312210181005975,0.32,0.277,1.328,1,03,383,-33.020638,-71.550995,4G,-53,-1,0,1779312209790,1779312210173,166
IoT,1779312216203947219,-0.236,0.121,1.024,1,03,395,-33.020599,-71.550995,4G,-53,1,0,1779312215801,1779312216196,167
IoT,1779312222267966394,-0.159,0.124,1.289,1,03,456,-33.020515,-71.550972,4G,-53,1,0,1779312221808,1779312222264,168
IoT,1779312228202705937,-0.109,0.135,1.281,1,03,378,-33.020462,-71.550941,4G,-53,6,0,1779312227813,1779312228191,169
IoT,1779312234192508209,0.022,0.126,1.312,1,03,375,-33.020409,-71.550934,4G,-53,1,0,1779312233814,1779312234189,170
IoT,1779312240208172583,-0.136,0.227,1.089,1,03,385,-33.020348,-71.550919,4G,-51,11,0,1779312239815,1779312240200,171
IoT,1779312246215884464,-0.075,0.164,1.164,1,03,385,-33.020344,-71.550842,4G,-51,10,0,1779312245823,1779312246208,172
IoT,1779312252215941552,-0.005,0.041,0.932,1,03,388,-33.020264,-71.550865,4G,-51,0,0,1779312251824,1779312252212,173
IoT,1779312258214764975,-0.256,0.023,1.043,1,03,373,-33.020176,-71.55088,4G,-51,0,0,1779312257834,1779312258207,174
IoT,1779312264249176109,-0.14,0.085,0.942,1,03,401,-33.020107,-71.550858,4G,-51,2,0,1779312263844,1779312264245,175
IoT,1779312270297940271,-0.041,0,1.189,1,03,427,-33.020042,-71.550835,4G,-57,4,0,1779312269864,1779312270291,176
IoT,1779312276244938780,-0.123,0.045,0.902,1,03,379,-33.01997,-71.550812,4G,-57,-8,0,1779312275860,1779312276239,177
IoT,1779312282269398393,-0.089,-0.233,1.136,1,03,397,-33.019905,-71.550804,4G,-57,-5,0,1779312281865,1779312282262,178
IoT,1779312288320585298,-0.069,0.052,0.838,1,03,438,-33.019855,-71.550789,4G,-57,1,0,1779312287875,1779312288313,179
IoT,1779312294278656168,-0.02,0.28,1.472,1,03,389,-33.019791,-71.550781,4G,-57,2,0,1779312293885,1779312294274,180
IoT,1779312300289602660,0.211,0.013,1.223,1,03,396,-33.019733,-71.550789,4G,-55,0,0,1779312299886,1779312300282,181
IoT,1779312306277989463,-0.153,0,1.181,1,03,377,-33.019684,-71.550781,4G,-55,-1,0,1779312305894,1779312306271,182
IoT,1779312312379116306,-0.131,0.198,1.16,1,03,478,-33.019623,-71.550758,4G,-55,7,0,1779312311896,1779312312374,183
IoT,1779312318307261786,-0.068,0.156,1.175,1,03,387,-33.019581,-71.550758,4G,-55,6,0,1779312317906,1779312318293,184
IoT,1779312324320164787,-0.193,0.26,0.958,1,03,390,-33.019531,-71.550766,4G,-55,7,0,1779312323926,1779312324316,185
IoT,1779312330418558229,-0.095,0.201,0.949,1,03,477,-33.019485,-71.550758,4G,-51,0,0,1779312329934,1779312330411,186
IoT,1779312336325134104,0.121,0.105,1.127,1,03,378,-33.019444,-71.550766,4G,-51,3,0,1779312335940,1779312336318,187
IoT,1779312342343560770,-0.18,0.06,1.06,1,03,398,-33.019375,-71.550743,4G,-51,1,0,1779312341942,1779312342340,188
IoT,1779312348431669722,-0.088,0.217,1.214,1,03,478,-33.019299,-71.550705,4G,-51,0,0,1779312347944,1779312348422,189
IoT,1779312354340825695,0.114,0.266,1.012,1,03,392,-33.019234,-71.550674,4G,-51,1,0,1779312353945,1779312354337,190
IoT,1779312360345264738,0.024,0.156,1.364,1,03,392,-33.019192,-71.550644,4G,-51,1,0,1779312359946,1779312360338,191
IoT,1779312366343004124,-0.164,0.268,1.18,1,03,381,-33.01915,-71.550652,4G,-51,-5,0,1779312365954,1779312366335,192
IoT,1779312372383754354,-0.252,0.288,1.11,1,03,425,-33.019089,-71.550644,4G,-51,3,0,1779312371955,1779312372380,193
IoT,1779312378361863608,-0.152,0.171,1,1,03,390,-33.019047,-71.550636,4G,-51,-1,0,1779312377962,1779312378352,194

IoT,1779312384354620761,-0.118,0.035,0.838,1,03,381,-33.018982,-71.550629,4G,-51,-11,0,1779312383969,1779312384350,195
IoT,1779312390371213469,0.052,0.063,1.182,1,03,392,-33.018913,-71.550606,4G,-51,-8,0,1779312389971,1779312390363,196
IoT,1779312396450563781,-0.261,0.253,0.879,1,03,471,-33.018856,-71.550591,4G,-51,0,0,1779312395973,1779312396444,197
IoT,1779312402417415056,-0.236,0.213,0.904,1,03,431,-33.018787,-71.550575,4G,-51,0,0,1779312401979,1779312402410,198
IoT,1779312408369780292,-0.091,0.177,1.186,1,03,382,-33.018715,-71.55056,4G,-51,1,0,1779312407981,1779312408363,199
IoT,1779312414384931778,0.162,0.191,1.13,1,03,398,-33.018642,-71.550545,4G,-51,-7,0,1779312413983,1779312414381,200
IoT,1779312420422013598,-0.108,0.176,1.138,1,03,430,-33.018574,-71.550537,4G,-51,0,0,1779312419986,1779312420416,201
IoT,1779312426385614366,-0.186,0.137,1.082,1,03,389,-33.018513,-71.550522,4G,-51,7,0,1779312425990,1779312426379,202
IoT,1779312436737463968,-0.212,0.215,1.135,1,03,4733,-33.018448,-71.550514,4G,-51,6,0,1779312431998,1779312436731,203
IoT,1779312510530163224,-0.32,0.19,1.023,1,03,468,-33.017826,-71.550407,4G,-51,4,0,1779312510054,1779312510522,216
IoT,1779312516571906389,-0.166,0.284,1.228,1,03,499,-33.017769,-71.550354,4G,-51,-1,0,1779312516066,1779312516565,217
IoT,1779312522613748842,0.01,0.11,1.118,1,03,535,-33.017704,-71.550339,4G,-51,-4,0,1779312522075,1779312522610,218
IoT,1779312528532205751,-0.164,0.145,1.141,1,03,448,-33.017635,-71.550316,4G,-51,-2,0,1779312528077,1779312528525,219
IoT,1779312534523959847,-0.051,0.208,0.938,1,03,437,-33.017601,-71.550354,4G,-51,-3,0,1779312534082,1779312534519,220
IoT,1779312540624683299,-0.214,0.148,0.932,1,03,527,-33.017578,-71.550362,4G,-51,-3,0,1779312540089,1779312540616,221
IoT,1779312546569197690,0.259,0.163,1.1,1,03,469,-33.017555,-71.550446,4G,-51,-7,0,1779312546093,1779312546562,222
IoT,1779312552558872854,-0.155,0.274,1.138,1,03,451,-33.017529,-71.550522,4G,-51,-8,0,1779312552102,1779312552553,223
IoT,1779312558668992305,-0.194,0.086,1.079,1,03,554,-33.017532,-71.550575,4G,-51,19,0,1779312558108,1779312558662,224
IoT,1779312564573736708,-0.167,0.166,0.898,1,03,453,-33.017574,-71.550591,4G,-51,17,0,1779312564112,1779312564565,225
IoT,1779312570576626824,-0.114,0.13,1.303,1,03,442,-33.017601,-71.550591,4G,-51,15,0,1779312570127,1779312570569,226
IoT,1779312576579416614,0.16,0.233,1.216,1,03,455,-33.017639,-71.550598,4G,-51,19,0,1779312576117,1779312576572,227
IoT,1779312582623317278,-0.13,0.238,1.374,1,03,489,-33.017662,-71.550613,4G,-51,17,0,1779312582127,1779312582616,228
IoT,1779312588584824103,0.006,0.114,1.327,1,03,446,-33.0177,-71.550621,4G,-51,9,0,1779312588131,1779312588577,229
IoT,1779312594581244849,-0.271,0.226,0.834,1,03,439,-33.017727,-71.550629,4G,-51,2,0,1779312594135,1779312594574,230
IoT,1779312600646904181,-0.239,0.348,1.239,1,03,497,-33.017731,-71.550629,4G,-51,9,0,1779312600141,1779312600638,231
IoT,1779312606632548201,-0.157,0.104,1.101,1,03,477,-33.017769,-71.550636,4G,-51,13,0,1779312606147,1779312606624,232
IoT,1779312612617145383,-0.293,0.199,1.238,1,03,457,-33.017803,-71.550644,4G,-51,10,0,1779312612149,1779312612606,233
IoT,1779312618669193689,-0.266,0.178,1.205,1,03,509,-33.017853,-71.550652,4G,-51,11,0,1779312618153,1779312618662,234
IoT,1779312624709241868,-0.324,0.218,1.196,1,03,539,-33.017906,-71.550667,4G,-51,11,0,1779312624163,1779312624702,235
IoT,1779312630631211903,0.041,0.235,1.138,1,03,451,-33.017963,-71.550667,4G,-51,12,0,1779312630173,1779312630624,236
IoT,1779312636634782388,0.009,0.223,1.229,1,03,449,-33.018017,-71.550667,4G,-51,13,0,1779312636177,1779312636626,237
IoT,1779312642738074000,-0.298,0.171,1.099,1,03,549,-33.018051,-71.550674,4G,-51,14,0,1779312642181,1779312642730,238
IoT,1779312648633082957,-0.028,-0.02,1.095,1,03,441,-33.018082,-71.550682,4G,-51,6,0,1779312648185,1779312648626,239

IoT,1779312654638502091,0.127,0.027,1.158,1,03,446,-33.018116,-71.55069,4G,-51,2,0,1779312654189,1779312654635,240
IoT,1779312660753907205,-0.064,0.167,1.036,1,03,552,-33.018154,-71.550705,4G,-51,19,0,1779312660194,1779312660746,241
IoT,1779312666659878985,-0.072,0.223,1.201,1,03,445,-33.018192,-71.55072,4G,-51,4,0,1779312666208,1779312666653,242
IoT,1779312672660518835,-0.03,0.085,0.958,1,03,436,-33.018303,-71.550743,4G,-51,15,0,1779312672217,1779312672653,243
IoT,1779312678668963359,-0.291,0.197,1.113,1,03,447,-33.018406,-71.550766,4G,-51,7,0,1779312678218,1779312678665,244
IoT,1779312684714650795,-0.051,0.334,1.281,1,03,485,-33.018452,-71.550766,4G,-51,5,0,1779312684222,1779312684707,245
IoT,1779312690694458531,-0.02,0.085,1.111,1,03,463,-33.018505,-71.550766,4G,-51,4,0,1779312690227,1779312690690,246
IoT,1779312696693322410,-0.283,0.262,1.005,1,03,457,-33.018559,-71.550781,4G,-51,8,0,1779312696230,1779312696687,247
IoT,1779312703166718053,-0.27,0.117,1.028,1,03,926,-33.018574,-71.550774,4G,-51,6,0,1779312702234,1779312703160,248
IoT,1779312708684983354,-0.273,0.16,0.975,1,03,442,-33.018604,-71.550781,4G,-51,7,0,1779312708239,1779312708681,249
IoT,1779312714694970955,-0.154,0.155,0.81,1,03,443,-33.018642,-71.550789,4G,-51,4,0,1779312714244,1779312714687,250
IoT,1779312721182276759,-0.112,0.203,0.86,1,03,927,-33.018715,-71.550812,4G,-51,16,0,1779312720245,1779312721172,251
IoT,1779312726711833381,-0.012,0.058,0.919,1,03,449,-33.018814,-71.550827,4G,-51,16,0,1779312726255,1779312726704,252
IoT,1779312732739353150,-0.027,0.05,0.893,1,03,471,-33.018887,-71.550842,4G,-51,19,0,1779312732261,1779312732732,253
IoT,1779312738774557776,-0.048,0.137,0.918,1,03,505,-33.018925,-71.550858,4G,-51,15,0,1779312738265,1779312738770,254
IoT,1779312744823252808,0.057,0.466,1.244,1,03,546,-33.018955,-71.550858,4G,-51,9,0,1779312744269,1779312744815,255
IoT,1779312750720423513,0.098,0.295,1.186,1,03,440,-33.019001,-71.550873,4G,-51,13,0,1779312750275,1779312750715,256
IoT,1779312756725767903,-0.111,0.23,0.914,1,03,436,-33.019016,-71.55088,4G,-51,11,0,1779312756281,1779312756717,257
IoT,1779312762746561325,0.039,0.27,1.205,1,03,454,-33.019035,-71.550888,4G,-51,6,0,1779312762285,1779312762739,258
IoT,1779312769206340008,0.111,0.156,1.002,1,03,907,-33.019066,-71.550911,4G,-51,10,0,1779312768295,1779312769202,259
IoT,1779312774747008033,-0.007,0.024,1.106,1,03,435,-33.019085,-71.550919,4G,-51,11,0,1779312774305,1779312774740,260
IoT,1779312780865585313,-0.028,0.179,1.08,1,03,551,-33.019096,-71.550926,4G,-51,18,0,1779312780311,1779312780862,261
IoT,1779312786780188617,-0.123,0.344,1.012,1,03,458,-33.019108,-71.550926,4G,-51,8,0,1779312786315,1779312786773,262
IoT,1779312792783544650,-0.268,0.149,0.853,1,03,460,-33.019161,-71.550941,4G,-51,12,0,1779312792316,1779312792776,263
IoT,1779312798781435342,0.025,-0.004,1.284,1,03,457,-33.01918,-71.550941,4G,-51,12,0,1779312798320,1779312798777,264
IoT,1779312804829349400,-0.037,0.168,0.942,1,03,498,-33.019203,-71.550941,4G,-51,8,0,1779312804324,1779312804822,265
IoT,1779312810792128115,-0.166,0.23,1.082,1,03,459,-33.019253,-71.550934,4G,-51,13,0,1779312810329,1779312810788,266
IoT,1779312816790628772,-0.12,0.089,1.273,1,03,453,-33.019299,-71.550919,4G,-51,6,0,1779312816330,1779312816783,267
IoT,1779312822853927025,-0.132,0.219,1.19,1,03,506,-33.019348,-71.550926,4G,-51,11,0,1779312822340,1779312822846,268
IoT,1779312828794270504,-0.285,0.285,1.154,1,03,441,-33.019413,-71.550934,4G,-51,9,0,1779312828346,1779312828787,269
IoT,1779312834804818798,-0.172,0.105,0.964,1,03,447,-33.019444,-71.550957,4G,-51,8,0,1779312834350,1779312834797,270
IoT,1779312840868825252,0.01,0.11,1.075,1,03,510,-33.019539,-71.550987,4G,-51,3,0,1779312840355,1779312840865,271
IoT,1779312846807993838,0.052,0.346,1.366,1,03,440,-33.019588,-71.55101,4G,-51,12,0,1779312846360,1779312846800,272

IoT,1779312852868556881,0.115,-0.014,1.466,1,03,490,-33.019657,-71.551025,4G,-51,9,0,1779312852370,1779312852860,273
IoT,1779312858893099591,0.001,0.167,0.993,1,03,515,-33.019764,-71.551025,4G,-51,7,0,1779312858374,1779312858889,274
IoT,1779312864824489112,-0.173,0.139,1.143,1,03,434,-33.019859,-71.551018,4G,-51,7,0,1779312864383,1779312864817,275
IoT,1779312870850426523,-0.063,0.395,1.154,1,03,455,-33.019974,-71.551003,4G,-51,7,0,1779312870392,1779312870847,276
IoT,1779312876923733381,-0.319,0.252,1.218,1,03,520,-33.020039,-71.55101,4G,-51,7,0,1779312876396,1779312876916,277
IoT,1779312882870903102,-0.268,0.168,1.153,1,03,463,-33.020107,-71.551018,4G,-51,7,0,1779312882401,1779312882864,278
IoT,1779312888844804545,-0.251,0.044,1.023,1,03,436,-33.020161,-71.551025,4G,-51,3,0,1779312888405,1779312888841,279
IoT,1779312894943100543,-0.004,0.265,1.548,1,03,525,-33.020214,-71.551033,4G,-51,6,0,1779312894411,1779312894936,280
IoT,1779312900868902967,0.229,-0.089,1.111,1,03,450,-33.020294,-71.551048,4G,-55,4,0,1779312900415,1779312900865,281
IoT,1779312906869228605,-0.349,0.028,1.206,1,03,443,-33.020351,-71.551064,4G,-55,4,0,1779312906419,1779312906862,282
IoT,1779312912972965985,-0.104,0.049,1.2,1,03,541,-33.020409,-71.551094,4G,-55,2,0,1779312912424,1779312912965,283
IoT,1779312918899057474,-0.051,0.165,1.24,1,03,462,-33.020451,-71.551109,4G,-55,-4,0,1779312918433,1779312918895,284
IoT,1779312924884948054,-0.142,0.204,1.093,1,03,440,-33.020477,-71.55114,4G,-55,-3,0,1779312924438,1779312924878,285
IoT,1779312930877010201,-0.044,0.155,0.883,1,03,430,-33.020508,-71.551163,4G,-51,1,0,1779312930443,1779312930873,286
IoT,1779312936942441631,-0.232,0.194,1.052,1,03,487,-33.020554,-71.551186,4G,-51,-2,0,1779312936448,1779312936935,287
IoT,1779312942970670901,-0.184,-0.002,1.132,1,03,512,-33.020599,-71.551193,4G,-51,-9,0,1779312942452,1779312942964,288
IoT,1779312949003702105,0.051,0.216,1.24,1,03,544,-33.020638,-71.551201,4G,-51,-10,0,1779312948456,1779312949000,289
IoT,1779312954902251566,-0.202,0.248,1.103,1,03,435,-33.020679,-71.551208,4G,-51,0,0,1779312954460,1779312954895,290
IoT,1779312960935177479,-0.202,0.245,1.016,1,03,461,-33.020733,-71.551224,4G,-51,-2,0,1779312960470,1779312960931,291
IoT,1779312966941544545,-0.158,0.066,1.054,1,03,455,-33.020767,-71.551231,4G,-51,-1,0,1779312966480,1779312966935,292
IoT,1779312973026321117,0.006,0.215,1.306,1,03,530,-33.020817,-71.551216,4G,-51,-4,0,1779312972490,1779312973020,293
IoT,1779312978954777749,0.128,0.173,1.098,1,03,455,-33.020863,-71.551216,4G,-51,-10,0,1779312978496,1779312978951,294
IoT,1779312985062782635,-0.22,0.227,1.165,1,03,556,-33.020916,-71.551208,4G,-51,-9,0,1779312984500,1779312985056,295
IoT,1779312990966397900,-0.28,0.15,0.906,1,03,454,-33.020958,-71.551208,4G,-51,-9,0,1779312990507,1779312990961,296
IoT,1779312997061163854,-0.102,0.083,0.879,1,03,539,-33.020985,-71.551208,4G,-51,0,0,1779312996515,1779312997054,297
IoT,1779313003079108914,0.133,0.052,1.218,1,03,551,-33.021015,-71.551224,4G,-51,0,0,1779313002521,1779313003072,298
IoT,1779313009675014235,-0.094,0.134,1.023,1,03,1135,-33.021042,-71.551247,4G,-51,0,0,1779313008535,1779313009670,299
IoT,1779313015306523369,-0.137,0.054,0.959,1,03,756,-33.02108,-71.55127,4G,-51,1,0,1779313014543,1779313015299,300
IoT,1779313021009737579,-0.14,0.203,0.937,1,03,452,-33.021126,-71.551285,4G,-51,-1,0,1779313020553,1779313021005,301
IoT,1779313027019926590,0.068,0.154,0.988,1,03,456,-33.021168,-71.551277,4G,-51,2,0,1779313026557,1779313027013,302
IoT,1779313033017936600,-0.008,0.002,1.014,1,03,450,-33.021214,-71.55127,4G,-51,-2,0,1779313032561,1779313033011,303

Bibliografía

[1] "Llega servicio de scooters a lo barnechea con bicicletas eléctricas," 2024. [Online]. Available: https://www.zoomtecnologico.com/2024/10/15/servicio-de-scooters/

[2] Lime AI, "Iot platform specifications," 2025. [Online]. Available: https://www.lime.ai/products/iot

[3] Whoosh, "Frequently asked questions." [Online]. Available: https://whoosh.bike/faq/en

[4] El Dínamo, "El ruso que quiere tener 200 mil scooters eléctricos y ser líder en micromovilidad en chile," 2026. [Online]. Available: https://www.eldinamo.cl/economia/negocios-economia/2026/04/27/el-ruso-que-quiere-tener-200-mil-scooters-electricos-y-ser-lider-en-micro-movilidad-en-chile

[5] Gobierno de Chile, "Modelos de negocios: Empresas de micromovilidad." [Online]. Available: https://energia.gob.cl/electromovilidad/modelos-de-negocios/empresas-de-micromovilidad

[6] "Xm iot device with gps tracking for electric scooter rental sharing scooter." [Online]. Available: https://www.omnismartiot.com/Electric-Scooter-Rental/xm-iot-device-with-gps-tracking-for-electric-scooter-rental-sharing-scooter.html

[7] “Gps vl110c 4g vehículos ligeros y flotas localización en tiempo real.” [Online]. Available: https://guardiangpschile.cl/producto/gps-vl110c-4g-vehiculos-ligeros-flotas-motos/

[8] “Gps tracker mlab 4g lte ip67.” [Online]. Available: https://camont.cl/producto/gps-tracker-mlab-4g-lte-ip67/

[9] R. Aburukba, A. R. Al-Ali, A. H. Riaz, A. A. Nabulsi, D. Khan, S. Khan, and M. Amer, “Fog computing approach for shared mobility in smart cities,” *Energies*, vol. 14, no. 23, p. 8174, Dec. 2021.

[10] A.-V. Palacean, D.-C. Trancă, C. Contașel, R. Tătărăoiu, and C. Duțescu, “Iot enabled optimized architectures for gps anti-theft tracking devices,” in *Proc. 20th RoEduNet Conference: Networking in Education and Research*, Iasi, Romania, Nov. 2021.

[11] R. Sanchez-Iborra, L. Bernal-Escobedo, and J. Santa, “Eco-efficient mobility in smart city scenarios,” *Sustainability*, vol. 12, no. 20, p. 8443, Oct. 2020.

[12] A. M. Joy, A. Ravi, J. Roy, G. V. Karthikeyan, and T. P. Rajan, “Iot enabled electronic security remote monitoring solution for electric bikes,” in *Proc. IEEE International Power and Renewable Energy Conference (IPRECON)*, Kothamangalam, India, Dec. 2022.

[13] G. K. Isik, T. Tryfonas, and G. Oikonomou, “E-scooter sharing platforms: Understanding their architecture and cybersecurity threats,” in *Proc. IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, Bilbao, Spain, Sep. 2023.

[14] D. Croce, D. Garlisi, F. Giuliano, A. L. Valvo, S. Mangione, and I. Tinnirello, “Performance of lora for bike-sharing systems,” in *Proc. IEEE International Conference on Electrical Engineering and Technology Applications (EETA)*, Turin, Italy, Jul. 2019.

[15] S. P. Sawant, D. Y. Thakor, D. K. Khandelwal, and S. N. Katkar, “Bike skid detection and smart alert system with route condition assistance,” in *Proc. IEEE International Conference on Computing, Embedded Systems and Communications (ICCES)*, Coimbatore, India, Jun. 2023.

[16] nPerf, “Mapa de cobertura de claro móvil en viña del mar,” https://www.nperf.com/es/map/CL/3868121.Vina-del-Mar/163636.Claro-Movil/signal, 2026.

[17] SensorsIot, “Repositorio iotstack,” GitHub repository. [Online]. Available: https://github.com/SensorsIot/IOTstack

[18] M. Dubarry and A. Devie, “Initial conditioning characterization test and other preliminary testing,” Hawaii Natural Energy Institute, University of Hawaii at Manoa, Honolulu, HI, USA, Tech. Rep. HNEI-06-15, Feb. 2015. [Online]. Available: https://www.hnei.hawaii.edu/wp-content/uploads/Initial-Conditioning-Characterization-Test.pdf