2021-09

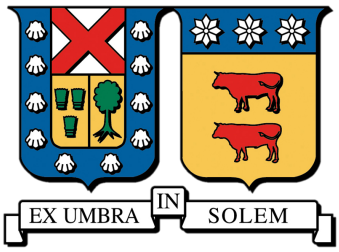
HACIA UN SISTEMA AUXILIAR DE DETECCIÓN DE UBICACIÓN USANDO REDES LORA

QUIROZ CÁRDENAS, DAMIÁN MARCELO

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE ELECTRÓNICA

VALPARAÍSO - CHILE



“HACIA UN SISTEMA AUXILIAR DE DETECCIÓN DE UBICACIÓN USANDO REDES LORA”

DAMIÁN MARCELO QUIROZ CÁRDENAS

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRÓNICO MENCIÓN COMPUTADORES

PROFESOR GUÍA: SR. NICOLÁS JARA

PROFESOR CORREFERENTE: SR. MOHAMED ABDELHAMID

SEPTIEMBRE - 2021

Agradecimientos

Este trabajo de Tesis realizado en la Universidad Técnica Federico Santa María, es un esfuerzo en el que han participado distintas personas, entregando su opinión y dando nuevos puntos de vista durante todo el tiempo invertido desde el comienzo al final de éste proceso. Con éste trabajo se me ha permitido apreciar nuevas experiencias y extender el conocimiento más allá de lo entregado por la institución a la cual estoy profundamente agradecido.

En primero lugar, me gustaría agradecer a mi asesor de Tesis, PhD. Nicolás Jara, por compartir su experiencia y conocimientos al momento de realizar este trabajo, y por su gran paciencia durante este periodo, ya que, debido a la situación transcurrida a nivel mundial, el desarrollo de éste trabajo ha sido bastante anormal. Le agradezco profundamente el apoyo que me ha brindado y la dedicación que le entregó a este proyecto.

Le agradezco al PhD. Pablo Reyes, ya que ha sido un gran apoyo durante el transcurso de este trabajo y durante los últimos años, entregando sus conocimientos y experiencia de forma incondicional, dando su punto de vista y corrigiendo puntos críticos que me permitieron crecer como profesional y como persona.

Mis agradecimientos a todos los profesores del departamento de Electrónica de la Universidad Técnica Federico Santa María, lo cuales han sido un pilar fundamental para mantenerme en éste rumbo y me han permitido mantener mi interés en la Electrónica y en los nuevos desafíos tecnológicos que aparecen constantemente.

Les entrego un agradecimiento especial a mis compañeros Luis Bahamondes, Matías Contreras, Francisco Frez y Sebastián Villanelo, los cuales han sido un apoyo incondicional durante mi estancia en la universidad y especialmente durante este proceso tan importante para cerrar mi ciclo como estudiante, les agradezco profundamente las noches de ayuda, de apoyo para orientarme en la redacción de éste documento y por las buenas intenciones que siempre han entregado al momento de ofrecer su ayuda.

Le agradezco a la Ing. De Apoyo para la Gestión del departamento de Electrónica, Gabriela Sanhueza, ya que durante los últimos años de mi carrera fue un gran apoyo que me permitió crecer como persona, me apoyo constantemente, me guió siempre que lo necesitaba y fue una de las personas que hizo que mi estancia en la Universidad fuera grata y provechosa.

Le agradezco al equipo de Secretarias del departamento de Electrónica, que siempre fueron de gran ayuda y dieron guía cuando la necesitaba.

Finalmente, agradezco profundamente a mi familia y pareja, que me apoyaron incondicionalmente durante todos estos años, me entregaron su amor y cariño cuando lo necesitaba, y que a pesar de la distancia y del tiempo que estuvimos separados esperaron pacientemente hasta el final de mi carrera.

A todos ustedes les entrego mis más profundos agradecimientos y cariño.

HACIA UN SISTEMA AUXILIAR DE DETECCIÓN DE UBICACIÓN USANDO REDES LORA

Damián Marcelo Quiroz Cárdenas

Memoria para optar al Título de Ingeniero Civil Electrónico, Mención Computadores

Universidad Técnica Federico Santa María

Profesor Guía: Sr. Nicolás Jara

Julio 2021

Resumen

Una de las principales problemáticas vistas por las empresas de comercialización y distribución de productos, son los robos y asaltos. Estos problemas generan una gran cantidad de pérdidas para dichas empresas ya que la mercancía robada casi nunca es recuperada. Como una alternativa para mitigar esos delitos, muchas empresas contratan servicios de ubicación mediante Sistema de Posicionamiento Global o Global Positioning System (GPS) para tener información de los vehículos cuando ocurren estos delitos. Como una contra medida ante la acción de las empresas, los delicuentes utilizan dispositivos que inutilizan las señales de GPS, dificultando aún más el trabajo de las empresas dedicadas a este rubro. Este trabajo propone una alternativa de solución al problema de localización orientada a vehículos. El texto se centra en el diseño de un sistema auxiliar de localización para un sistema que posee de forma integrada un GPSs, el que por distintos motivos podría fallar. Se trabajará en un prototipo de solución desde el diseño de Hardware y Firmware hasta la arquitectura de Software, apuntando a la cobertura de eventos especiales, donde no se cuente con la señal de GPSs o cuando se necesite enviar mensajes de alerta hacia otros vehículos. Para esto se evalúan las alternativas existentes en tecnologías como Wi-Fi, LoRa y Bluetooth. Por otra parte se investigan los algoritmos de posicionamiento existentes, que son compatibles con las tecnologías, con sus ventajas y desventajas para resolver el problema planteado. Al evaluar las alternativas, se escogerá una de ellas para dar solución al problema y continuar con el desarrollo del dispositivo y servidor.

Palabras Clave

GPS, Hardware, Firmware, Software, Wi-Fi,LoRa, Bluetooth, Algoritmos de posicionamiento.

TOWARDS AN AUXILIARY LOCATION DETECTION SYSTEM USING LORA NETWORKS

Damián Marcelo Quiroz Cárdenas

Final Project Report towards the fulfillment of the Civil Electronic Engineer,
Minor in Computers degree

Universidad Técnica Federico Santa María

Advisor: Mr. Nicolás Jara

July 2021

Abstract

One of the main problems encountered by marketing and distribution companies is theft and robbery. These problems generate a large amount of losses for these companies as stolen merchandise is almost never recovered. As an alternative to mitigate these crimes, many companies contract location services through GPS to have vehicle information when these crimes occur. As a countermeasure to the companies' actions, criminals use devices that disable GPS signals, making it even more difficult for the companies involved in this area. This document proposes an alternative solution to the problem of vehicle-oriented localisation. The text focuses on the design of an auxiliary localisation system for a system that has an integrated Global Positioning System (GPS), which for different reasons could fail. We will work on a prototype solution from the design of hardware and firmware to the architecture of the software, aiming at the coverage of special events, where the GPS signal is not available or when it is necessary to send alert messages to other vehicles. For this purpose, the existing alternatives in technologies such as Wi-Fi, LoRa and Bluetooth are evaluated. On the other hand, existing positioning algorithms are investigated, which are compatible with the technologies, with their advantages and disadvantages to solve the problem posed. After evaluating the alternatives, one of them will be chosen to solve the problem and continue with the development of the device and server.

Keywords GPS, Hardware, Firmware, Software, Wi-Fi, LoRa, Bluetooth, Positioning algorithms.

Glosario

3G es la abreviación de tercera generación de transmisión de voz y datos a través de telefonía móvil mediante Universal Mobile Telecommunications Systems (UMTSs). 3, 19, 53

AoA Angle of Arrival. 14, 20

AP Access Point. 18–21, 25

ASCII American Standard Code for Information Interchange. v, 63

AWS Amazon Web Services. 91, 99, 100, 116

Baud Rate Es una unidad de medida común de medición de transmisión de Símbolos en la electrónica e informática. Esta unidad es uno de los componentes que determina la velocidad de comunicación mediante un canal de datos. . 73

BDS BeiDou Navigation Satellite System. v

BeiDou-B1 BeiDou es la abreviación para BeiDou Navigation Satellite Systems (BDSs), que corresponde aun sistema de navegación satelital Chino con dos constelaciones de satélites separadas. La banda de frecuencias B1 corresponde a las frecuencias entre 1561.098 y 1575.42 MHz.. 60

Bluetooth Es una especificación industrial para redes inalámbricas de área personal (Wireless Personal Area Networks (WPANs)) creado por Bluetooth Special Interest Group, Inc. que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda Indistrial,Scientist and Medicals (ISMs) de los 2.4 GHz.. iii, iv, 1, 9, 22, 24, 25

BW Band Width. 30–32

comandos AT Es un lenguaje específico de comandos que consisten en cadenas de texto cortas que se pueden combinar para operar distintos módulos, cambiando sus parámetros sde configuración internos de forma fácil.. 54, 73, 81

CSS Chirp Spread Spectrum. viii, 29, 123

Dúplex Es un término utilizado en telecomunicación para definir a un sistema que es capaz de mantener una comunicación bidireccional, enviando y recibiendo mensajes de forma simultánea. . 86

DGPS Differencial Global Positioning System. 16

DIO Digital Input/Output. 76

EC2 Elastic Computing Cloud. 91, 92, 96

EDGE Enhanced Data Rates for GSM Evolution. 54

estándar NMEA Es un subset del estándar NMEA-183 utilizado por los equipos electrónicos marítimos, su abreviación corresponde a National Marine Electronics Associations (NMEAs).. 59–61, 74, 85

Firmware Es un programa informático que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo. Se puede describir como el Software que controla el Hardware.. iii, iv, 79

Framework En el desarrollo de software, un entorno de trabajo (o Framework) es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. . 79, 88, 96, 109, 110, 115, 117

Galileo-E1 Es un sistema de navegación y posicionamiento satelital Europea desarrollado por la Unión Europea para uso civil. La banda de frecuencias E1 corresponde a las frecuencias en torno a los 1575.42 MHz.. 60

GLONASS Global Navigation Satellite System. vii

GLONASS-L10F Es una banda de frecuencias del sistema de navegación satelital Global Navigation Satellite Systems (GLONASSs), entregando una alternativa al GPS, siendo el segundo sistema de navegación operativo con una cobertura global y con una precisión comparable a GPS. La banda L10F usa las frecuencias en torno a 1602 MHz.. 60

GPRS General Packet Radio Service. 3, 19, 53–57

GPS Global Positioning System. iii, iv, 1–3, 5, 9, 16, 17, 20, 53, 54, 56, 87–89, 106

GPS-L1 Son las señales más antiguas de GPS, las cuales se conforman de dos partes Coarse/Adquisition (C/A) para uso público y Precision Code (P-Code) para uso militar. Las señales L1 usan las frecuencias en torno a los 1575.52 MHz.. 59–61

GSM Global System for Mobile Communications. 3, 19, 54–57, 80, 81

Hardware Hace referencia a todos los componentes materiales y físicos de un dispositivo, es decir, aquellos que se pueden ver y tocar.. iii, iv, vi, 10, 16, 110, 115, 116

HSDPA High-Speed Downlink Packet Access. 54

HTTP (Hypertext Transfer Protocol. 96

HTTPS (Hypertext Transfer Protocol. 96

I2C Inter-Integrated Circuit. 68–71

Identificación por Radiofrecuencia Es un sistema de identificación de productos que utiliza las ondas de radio para comunicarse con un microchip, que puede estar montado sobre gran cantidad de soportes, como por ejemplo un tag o etiqueta Radio Frequency Identifications (RFIDs), una tarjeta o un transpondedor.. 9

IMEI International Mobile Equipment Identity. 87

Internet de las cosas Es un concepto que se refiere a una interconexión digital de objetos cotidianos con internet. Es, en definitiva, la conexión de internet más con objetos que con personas.. 1, 26

IoT Internet of Thing. 1, 2, 19, 26, 59

IP Internet Protocol. 4, 91

ISM Indistrial,Scientist and Medical. vi

Jammer Es un dispositivo electrónico que impide o dificulta las radiocomunicaciones en un determinado espectro de frecuencias mediante interferencias intencionadas.. 16

KNN K-Nearest Neighbors. 15, 26

LAN Local Area Network. 18

LoRa Es una técnica de modulación de espectro amplio (Spread Spectrum), basada en ta tecnología de modulación Chirp Spread Spectrums (CSSs). iii, iv, 1, 9, 26–29, 31–35, 38, 40, 87–90

LOS Line of Sight. 13

Map Matching Es el problema de cómo hacer coincidir las coordenadas geográficas registradas con un modelo lógico del mundo real.. 26

MCU Microcontroller Unit. 3, 5, 6, 86, 88, 117

NMEA National Marine Electronics Association. vi

PCB Printed Circuit Board. 105

Puntos de Acceso Son dispositivos que crean una red de área local inalámbrica (Wireless Local Area Networks (WLANs)), normalmente en una oficina o un edificio de grandes dimensiones. Un punto de acceso se conecta a un router, switch o hub por un cable Ethernet y proyecta una señal Wi-Fi en un área designada.. 19

QZSS Quasi-Zenith Satellite System. ix

QZSS-L1 Es la abreviación del sistema de satélites Michibiki que funciona en las frecuencias L1 de las señales de GPS (1575.42 MHZ), centrado principalmente en Japón.Las siglas corresponden a Quasi-Zenith Satellite System s (QZSSs).. 59, 60

RF Radio Frecuencia. 1, 3

RFID Radio Frequency Identification. vii, 9

RSS Received Signal Strength. 13, 14

RSSI Received Signal Strength Indicator. xi, 10, 15, 20, 24, 26, 27, 35, 40, 41

RTLS Real Time Location System. 20

SF Spreading Factor. xiv, 29–33, 76

Sistema de Posicionamiento Global Es un sistema que permite posicionar cualquier objeto (una persona, un vehículo) sobre la Tierra con una precisión de unos pocos metros.. iii

SMP Smallest M-Vertex Polygon. 15

SNR Signal to Noise Ratio. 29

Software Corresponde a la parte digital del ordenador, es decir, el conjunto de instrucciones, programas y reglas informáticas que el equipo requiere para funcionar.. iii, iv, vi, 116

SPI Serial Peripheral Interface. 65, 66, 68–72, 76, 77, 86, 89

SQL Structured Query Language. 97

SS Spread Spectrum. 29

SVM Support Vector Machine. 15

TCP Transmission Control Protocol. 4, 79, 81, 96

TDoA Time Difference of Arrival. 11, 26, 27, 88, 103, 107, 110, 115, 116

ToA Time of Arrival. 10, 12, 14

ToF Time of Flight. 40

Trilateración La trilateración es un método matemático para determinar las posiciones relativas de objetos usando la geometría de triángulos de forma análoga a la triangulación. 11

UART Universal asynchronous Receiver Trasmitter. 54, 55, 58–61, 63, 64, 68–75, 77, 89, 103

UMTS Universal Mobile Telecommunications System. v

USB Universal Serial Bus. 103

WCDMA Wideband Code Division Multiple Access. 54

Wi-Fi Tecnología que permite la interconexión inalámbrica de dispositivos electrónicos.. iii, iv, 1, 3, 9, 18–22, 24

WLAN Wireless Local Area Network. viii, 18

WPAN Wireless Personal Area Network. v

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El constante robo a vehículos de transporte es una de las principales fuentes de pérdida en las empresas dedicadas al comercio y venta de insumos. En Chile, el año 2020 una de las principales empresas de venta de cigarrillos declaro pérdidas del orden de los 25 mil millones de pesos en robos y asaltos. Por otra parte, el número de delitos de esta índole ha ido en aumento durante los últimos años. Dentro de las posibles soluciones a tales delitos, muchas empresas utilizan servicios de rastreo por GPS con distintas tecnologías que ayudan a dificultar o mitigar estas situaciones. Debido a esto, muchos delincuentes aprovechan dispositivos que inutilizan la señal de los aparatos de GPS y las tecnologías celulares como estrategia para cometer delitos, impidiendo que las empresas dedicadas al rubro del GPS cumplan con su deber por completo.
En base a lo anterior, como una alternativa distinta para atacar este problema se propone diseñar un dispositivo Internet of Thing (IoT) utilizando otros tipos de señales de Radio Frecuencia (RF) que estén menos expuestos a su inutilización, como lo son Wi-Fi, Bluetooth o LoRa.
Durante los últimos años el Internet de las cosas (IoT) ha crecido en gran medida y se ha incorporado a un sin fin de dispositivos electrónicos como electrodomésticos, sistemas de seguridad, automóviles y dispositivos industriales. Esta integración ha generado una gran demanda de tráfico en internet, y a su vez que los equipos electrónicos tengan la capacidad de conectarse a internet y, en el último tiempo, que tengan la capacidad de comunicarse entre ellos.
El crecimiento en la cantidad de dispositivos portátiles, y dispositivos de bajo consumo orientados al IoT, ha generado un gran interés en reemplazar la tecnología GPS por otras alternativas de bajo consumo, ya que debido a sus características, los dispositivos con la tecnología

GPS integrada tienen un gran consumo de batería, lo cual hace poco recomendable la integración de ésta tecnología en dispositivos que requieran una gran duración de batería. Si bien, en los años anteriores se han estudiado soluciones de bajo consumo con distintas tecnologías, siempre se debe evaluar el tipo de uso que se dará para determinar cual es la tecnología correcta a usar, debido a esto durante los próximos capítulos, se hará un análisis de las tecnologías y algoritmos utilizados para reemplazar la localización mediante GPS.

Este proyecto trata de aprovechar al máximo los beneficios que ha generado el IoT, buscando una solución alternativa de bajo costo y bajo consumo para problemas de localización en dispositivos móviles, en este caso vehículos en movimiento. Además, se busca agregar a esta solución la posibilidad de enviar mensajes entre dispositivos, para informar situaciones anormales que se presenten hacia el servidor encargado de procesar toda esta información.

1.1. Objetivos

1.1.1. Objetivo general

El trabajo de memoria consiste en el diseño e implementación de una solución de bajo costo y bajo consumo que permita aproximar la ubicación de dispositivos que no cuenten con una señal GPS, enviando mensajes de alerta a otros dispositivos para informar de su situación. En el desarrollo se considera el diseño del equipo de pruebas, la programación tanto del dispositivo de pruebas como del servidor que recibirá los datos, y el diseño de la base de datos donde se almacenará toda la información enviada desde los dispositivos. En la Figura 1.1 se muestra un diagrama de arquitectura general del proyecto, en el cual se presentan los dispositivos móviles como vehículos, el servidor que recibirá los datos y una base de datos que guardará la información enviados desde los dispositivos.

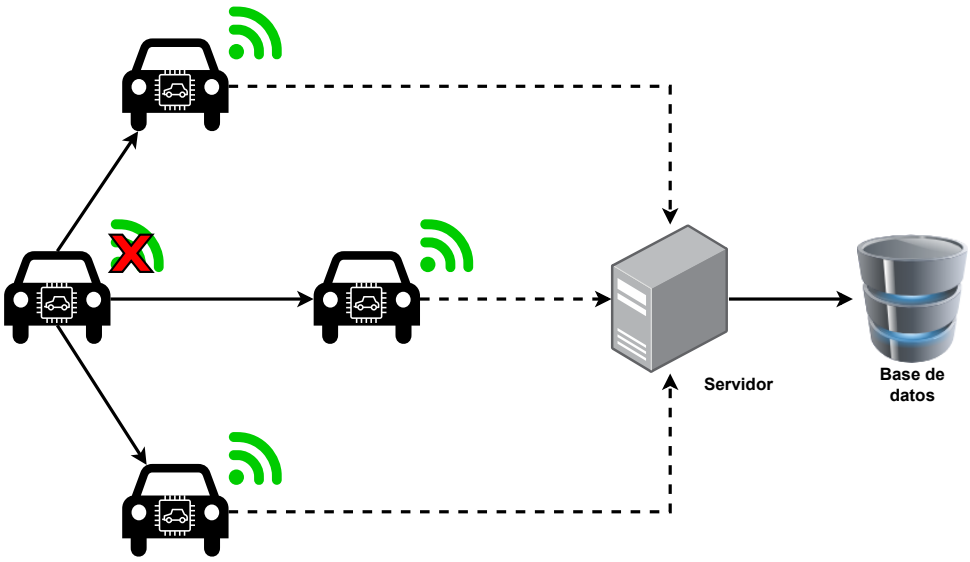


Figura 1.1: Arquitectura general del proyecto

Para lo que sigue del texto, se asumirá que el dispositivo a diseñar debe tener al menos los siguientes 4 componentes principales.

- Módulo para acceso a internet (General Packet Radio Service (GPRS)/Global System for Mobile Communications (GSM) y/o 3G).
- Módulo GPS integrado.
- Módulo con tecnología inalámbrica (RF, Wi-Fi u otro) que permita comunicación entre dispositivos y aproximar ubicación.
- Microcontrolador (Microcontroller Unit (MCU)) que permita la comunicación con los periféricos y desarrollo del algoritmo para envío de datos.

Según lo señalado anteriormente, en la Figura 1.2 se presenta un ejemplo del diagrama de conexiones esperado para el dispositivo de pruebas, donde cada flecha indica un flujo de información, es decir, el MCU se comunicará de forma bidireccional tanto con el módulo GPRS/GSM como con el módulo de RF, y recibirá datos de forma unidireccional desde el módulo GPS. Esta última relación se describe de esa forma ya que normalmente este tipo de módulos no recibe comandos ni información de vuelta, ya que vienen pre-configurados para entregar la información de ubicación.

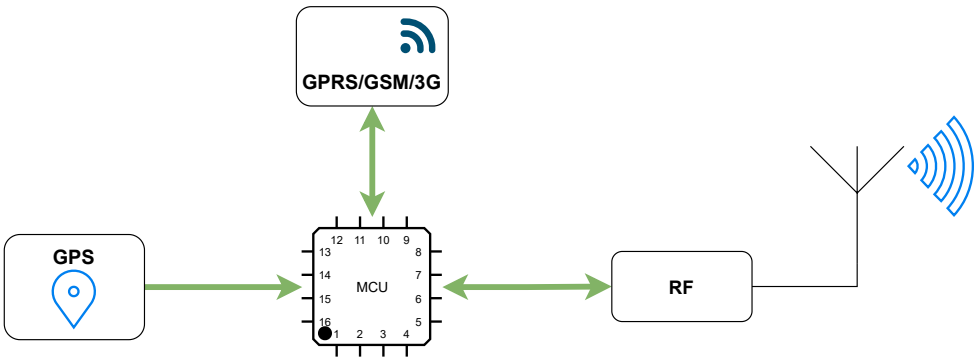


Figura 1.2: Diagrama de conexión de hardware con flujo de información.

Por otra parte, la comunicación con el servidor se realizará mediante un socket Transmission Control Protocol (TCP) ya que se considera la forma más sencilla para establecer una comunicación bidireccional entre los dispositivos y el servidor. Con esto mencionado, el servidor debe tener al menos las siguientes características para cumplir con los requerimientos del proyecto.

- Tener una Internet Protocol (IP) pública que permita la conexión al servidor desde cualquier dispositivo.
- Implementar una conexión TCP mediante algún puerto para que los dispositivos se comuniquen con el servidor.
- Implementar un algoritmo de decodificación de datos, según el formato que se defina para el envío de datos.
- Permitir la comunicación con una base de datos para guardar los datos decodificados.

Finalmente, la base de datos a diseñar debe ser cumplir con los únicos requisito de ser accesible desde nuestro servidor y permitir el guardado de todos los datos necesarios para calcular la localización de los dispositivos.

1.1.2. Objetivos Específicos

Para desarrollar un sistema capaz de cumplir con los requerimientos mencionados anteriormente, se establecen los siguientes objetivos específicos, que tratan de dar cumplimiento a cada uno de los requisitos.

Selección de una tecnología que cumpla con los requisitos del proyecto

Si bien lo que se busca es una solución de bajo costo y bajo consumo, se debe indagar en las tecnologías asociadas que permitan cumplir con estos requisitos. Para esto, se hará un análisis de las tecnologías inalámbricas existentes que permitan implementar una comunicación entre dispositivos, y que permitan desarrollar una solución de bajo costo y bajo consumo.

Selección de un algoritmo de localización acorde al proyecto y la tecnología seleccionada

Una vez seleccionada la tecnología a utilizar, se debe establecer un algoritmo de localización acorde con ella, para así cumplir con el objetivo de aproximar la posición de los dispositivos, aún sin contar con una señal de GPS activa. Para esto, se evaluarán distintas alternativas de algoritmos de localización, para seleccionar el que se acomode a las necesidades del proyecto.

Selección de componentes para el armado del prototipo

Al seleccionar la tecnología y el algoritmo a usar, se necesita establecer los componentes a utilizar para el armado del prototipo a utilizar, siempre buscando una opción de bajo costo ya que es un dispositivos de pruebas. La selección de componentes dependerá mucho del costo y la experiencia previa utilizando cada uno de ellos, ya que si bien es un dispositivo de pruebas, se espera optimizar tiempos de programación usando alternativas que ya se conozcan o se haya trabajado con ellas.

Interconexión de componentes

Cuando se tengan los componentes seleccionados, se debe diseñar el diagrama de conexión entre ellos para obtener el resultado mínimo correspondiente a la comunicación entre los componentes y la alimentación de cada uno de ellos. Si bien, cada componente es distinto, se espera que todos los periféricos seleccionados se comuniquen con el microcontrolador (MCU), el que actuará como “cerebro” para realizar las acciones necesarias con cada uno de los periféricos.

Programación del dispositivo de pruebas

Con la interconexión establecida, se debe programar el MCU para aprovechar los periféricos. En esta parte se espera encontrar y utilizar herramientas ya desarrolladas para optimizar tiempos de programación, ya que el desarrollo desde cero de herramientas de comunicación con periféricos puede provocar un excesivo gasto de tiempo en programación.
En esta parte también se considera establecer un formato de envío de datos, que será respetado por el servidor cuando se reciban los datos.

Diseño y programación del servidor de pruebas

Cuando el desarrollo del dispositivo de pruebas esté completo, se procederá a programar el servidor, que será el encargado de decodificar la información que envíen los dispositivos, e insertarla en una base de datos para luego utilizarla en el algoritmo de localización. El servidor debe respetar el formato de envío de datos que se establece desde el dispositivo para poder decodificar la información, de otra forma, se interpretará la información de forma errónea, lo que provocará que se use información "falsa" en el algoritmo de localización.

Diseño de la base de datos

Finalmente, se debe diseñar la base de datos de trabajo considerando los datos necesarios tanto para tener una historia de las ubicaciones de los dispositivos, como para utilizar el algoritmo de localización seleccionado. En esta parte del proyecto se completan los elementos mínimos mostrados en el diagrama de arquitectura presentado en la Figura 1.1, con lo que se termina la parte de desarrollo y se empieza el proceso de pruebas.

Pruebas de medición con algoritmo de localización

En la parte final del proyecto, se espera hacer pruebas reales donde se ubiquen dispositivos en ciertas partes de una ciudad, utilizando un dispositivo que tendrá el trabajo de enviar mensajes de alerta hacia los demás, para emular el comportamiento de un dispositivo con problemas de localización, y así finalmente probar el algoritmo dentro de este entorno.

1.2. Estructura de este documento

1.2.1. Estado del Arte

En este apartado, se dará una breve explicación de los algoritmos de localización existentes compatibles con el proyecto, para posteriormente analizar las tecnologías que permiten utilizar estos algoritmos presentando casos de uso existentes para luego listar sus ventajas y desventajas. Finalmente, se escogerá una de las tecnologías analizadas para implementar en este proyecto.

1.2.2. Tecnología escogida y selección de algoritmo

En esta parte del documento, nos centraremos en describir la tecnología escogida para tener un mejor entendimiento de ésta, mostrando más casos de uso para finalmente escoger un algoritmo de localización.

1.2.3. Diseño de Hardware

En este capitulo, nos centraremos en dar cumplimiento a los objetivos específicos que permitan construir un dispositivo de pruebas con la tecnología seleccionada y las capacidades mínimas de conexión a internet.

1.2.4. Diseño de Firmware

En este apartado se evaluarán alternativas que permitan acelerar el proceso de programación del dispositivo de pruebas, para finalmente diseñar el programa que hará posible la conexión con el servidor, la comunicación con los periféricos, y el envío de la información necesaria por el algoritmo escogido.

1.2.5. Diseño y arquitectura del Servidor

En este capítulo nos centraremos en el diseño del servidor, mostrando la programación interna básica que permite la decodificación de los datos recibidos desde los dispositivos de prueba, en

conjunto con la conexión a la base de datos.

1.2.6. Prototipo y Pruebas

En esta parte del documento se presentarán las pruebas realizadas, para analizar los resultados obtenidos utilizando los dispositivos y el algoritmo utilizado.

1.2.7. Desafíos y Problemas

Finalmente, en este capitulo se listarán los problemas detectados en el desarrollo del proyecto y los desafíos por cumplir para mejorar el funcionamiento y desarrollar un dispositivo más completo.

Capítulo 2

Estado del Arte

En la actualidad existe una variedad de técnicas y sistemas de localización, los que tienen distintas aplicaciones. En general, un sistema de localización conlleva una comunicación entre distintos dispositivos mediante tecnologías inalámbricas como Wi-Fi, GPS, Identificación por Radiofrecuencia (RFID), LoRa o Bluetooth, utilizando algoritmos que permiten calcular la ubicación aproximada de estos dispositivos empleando herramientas de telecomunicaciones. Cada una de las tecnologías mencionadas, tiene sus ventajas y desventajas dependiendo de los requerimientos del proyecto en que se deseen aplicar, por ejemplo, para localización en lugares cerrados la comunicación Bluetooth o Wi-Fi son buenas alternativas, mientras que en entornos abiertos o exteriores, la tecnología GPS es la más popular.
Por otra parte, se ha experimentado con distintos métodos de localización para buscar soluciones libres de GPS (GPS-Free) de menor consumo, debido a la variedad de éstos métodos y algoritmos es que se puede experimentar con ellos, combinándolos con opciones más avanzados de procesamiento o bien buscando enfoques distintos a metodologías que ya han sido estudiadas. Debido a esto, se hace imperativo investigar sobre los distintos algoritmos y metodologías de localización existentes.

2.1. Algoritmos de localización

Principalmente, los algoritmos de localización se dividen en 2 tipos de caracterización: Basados en alcance (Range Based) y libres de alcance (Range Free), esta clasificación se basa en

cómo los algoritmos obtienen la información entre dispositivos [1]. Los del primer tipo corresponden a algoritmos donde se hace necesario calcular distancias entre dispositivos, utilizando información de ángulos o tiempos de llegada, para utilizar algún método de estimación de posición. En los del segundo tipo como Distance Vector (DV), Centroid System o Aproximate Point in Triangulation (APIT), se utilizan características de conectividad (usualmente obtenida mediante la medición del RSSI) para estimar las posiciones de los dispositivos basándose en las señales recibidas desde dispositivos cercanos, por lo que estos métodos generalmente conllevan el uso de una gran densidad de dispositivos que emitan o reciban señales. En esta sección se realizará un análisis breve de las metodologías y algoritmos de localización existentes, mencionando sus principales usos y tecnologías compatibles, esto nos dará una visión más amplia para la elección de la tecnología a usar y qué métodos se pueden usar. Lamentablemente debido a la naturaleza de los métodos libres de distancia, para implementar cualquiera de estos métodos se requiere un gran trabajo y costo en cuanto a tiempo y Hardware por lo tanto no serán estudiados.

2.1.1. Time of Arrival (ToA)

La medición del ToA, corresponde al tiempo de llegada de cualquier tipo de señal hacia algún receptor. Esto corresponde al tiempo de transmisión más el tiempo de propagación entre un transmisor y un receptor, lo que permite deducir la distancia entre dos nodos. Para utilizar este algoritmo es imperativo que todos los dispositivos de la red estén sincronizados con el mismo reloj, ya que para calcular la distancia entre los dispositivos se utiliza la ecuación.

$$d = c\Delta t = c(t_r - t_t)$$

Donde c corresponde a la constante de propagación de la luz, t_r corresponde al tiempo en que se recibe el mensaje emitido desde el emisor, y t_t es el tiempo en que se emite el mensaje. Asumiendo que la señal recibida recorre la distancia más corta de propagación y sabiendo la ubicación de los dispositivos receptores, se puede dibujar un circulo de posibles posiciones del

dispositivo transmisor. utilizando la ecuación:

$$d_n = \sqrt{(x - x_n)^2 + (y - y_n)^2} \tag{2.1}$$

Luego, al intersectar el menos 3 círculos de posibles posiciones (Trilateración), se puede obtener el punto exacto donde se encuentra el dispositivo que emite la señal. En la Figura 2.1, se muestra este procedimiento donde los rombos azules corresponde a dispositivos receptores, y el rombo rojo corresponde a un dispositivo transmisor.

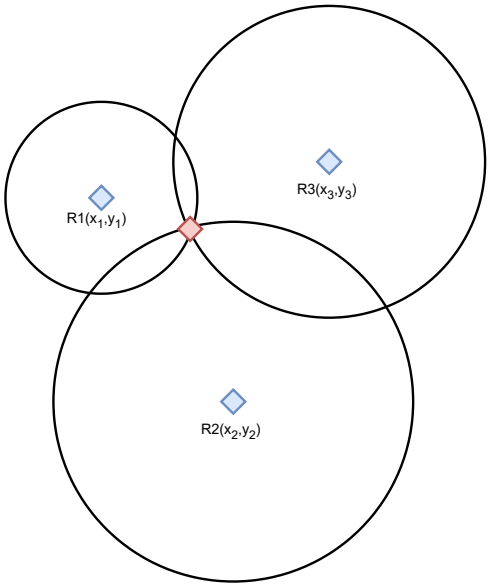


Figura 2.1: Representación geométrica de TOA para 3 receptores.

El principal problema que presenta este método, es que no siempre las señales recorren el camino más corto para llegar al receptor, debido a las múltiples reflexiones que se generan en el trayecto. Debido a esto, en entornos reales este método puede estar sujeto a errores, los que se pueden trabajar utilizando métodos como los presentados en [2], donde al modelo matemático de la ecuación 2.1 se le agrega un elemento que representa el error, con lo que se hace más complejo el cálculo pero se tiene un control del error inherente del método en entornos reales.

2.1.2. Time Difference of Arrival (TDoA)

Este método TDoA corresponde a la medición de la diferencia entre los tiempos de llegada de paquetes enviados desde un transmisor a 2 receptores distintos. Asumiendo que las ubicaciones

de los receptores son conocidas, que la señal viaja hacia los receptores por el camino más rápido y, a diferencia de ToA que requiere una sincronización total en toda la red de dispositivos, solo los dispositivos receptores deben estar sincronizados. El procedimiento general, al igual que en el caso de ToA, empieza por el cálculo de la distancia entre dos receptores utilizando la ecuación

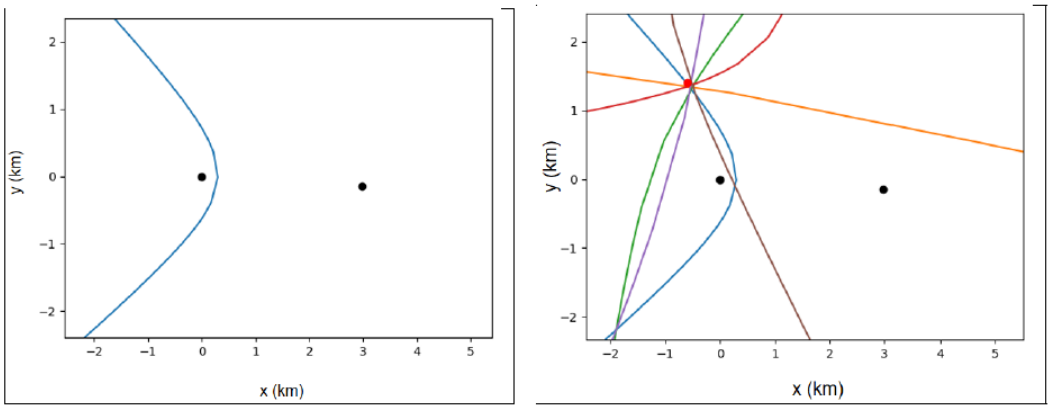
$$d_{ij} = c \cdot \Delta t_{ij} = c \cdot (t_j - t_i)$$

Donde t_j y t_i corresponden a los tiempos de llegada de los paquetes al j-ésimo nodo y al i-ésimo nodo de la red de dispositivos. Con estas mediciones de distancia, se pueden generar hipérbolas, las cuales corresponden a todos los posibles lugares donde el transmisor puede ubicarse. El problema a solucionar corresponde al sistema de ecuaciones no lineales generado por las n diferencias entre las distancias calculadas entre los nodos, es decir:

$$c \cdot \Delta t_{ij} = \sqrt{(x - x_j)^2 + (y - y_j)^2} - \sqrt{(x - x_i)^2 + (y - y_i)^2}$$

Al encontrar las intersecciones entre 3 o más de estas hipérbolas se puede estimar la posición exacta del dispositivo emisor, tal como se muestra en la Figura 2.2.

$$\begin{aligned}
c \cdot \Delta t_{ij_1} &= \sqrt{(x - x_{j_1})^2 + (y - y_{j_1})^2} - \sqrt{(x - x_i)^2 + (y - y_i)^2} \\
c \cdot \Delta t_{ij_2} &= \sqrt{(x - x_{j_2})^2 + (y - y_{j_2})^2} - \sqrt{(x - x_i)^2 + (y - y_i)^2} \\
c \cdot \Delta t_{ij_3} &= \sqrt{(x - x_{j_3})^2 + (y - y_{j_3})^2} - \sqrt{(x - x_i)^2 + (y - y_i)^2} \\
&\vdots \\
c \cdot \Delta t_{ij_n} &= \sqrt{(x - x_{j_n})^2 + (y - y_{j_n})^2} - \sqrt{(x - x_i)^2 + (y - y_i)^2}
\end{aligned}$$



(a) (b)

Figura 2.2: (a) Hipérbola con todos las posibles posiciones del dispositivo transmisor. (b) Intersección de hipérbolas para encontrar la ubicación exacta del dispositivo transmisor.[3]

Al igual que en el caso anterior, el principal problema que tiene este tipo de método, es que generalmente las distancias que recorren las señales entre un emisor y un receptor no es la más corta, ya que no se encuentra en linea vista (Line of Sight (LOS)).

2.1.3. Received Signal Strength (RSS)

Los métodos de cálculo de distancia basados en RSS se centran principalmente medir la intensidad de señal en el dispositivo receptor. Estos métodos están principalmente inspirados en la ecuación de propagación de señal en el vacío (o ecuación de Friis):

$$P_r = P_r D_t D_r \left(\frac{\lambda}{4\pi d}\right)^2 \tag{2.2}$$

Al utilizar esta ecuación como referencia y su modelo logarítmico, se pueden agregar elementos que permitan representar y caracterizar las condiciones ambientales.En la actualidad se han postulado varios modelos tanto para ciudades pobladas como para espacios rurales [4] permitiendo calcular distancias de forma más precisa dependiendo del entorno en que se mide la intensidad de señal.

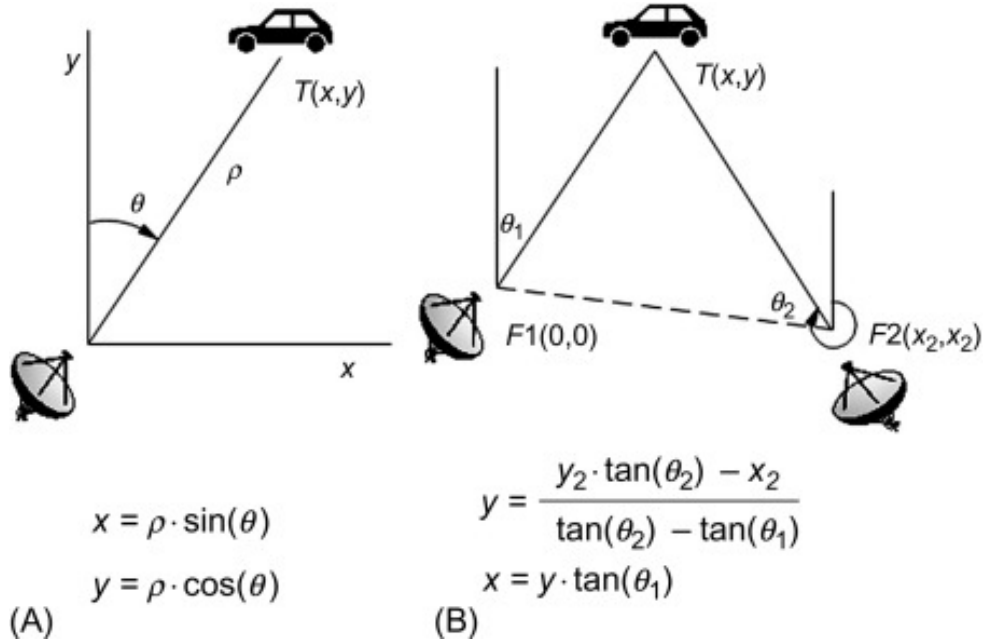


Figura 2.3: A.- Se combina AoA con un método de cálculo de distancia RSS o ToA. B.- Se calcula la ubicación encontrando la intersección de las lineas direccionales de dos antenas distintas. [5]

2.1.4. Angle of Arrival (AoA)

Esta metodología se puede usar para cálculo de ubicación de dispositivos por sí sola, o bien, normalmente se utiliza en compañía de alguna de las tecnologías mencionadas anteriormente. Cuando esta metodología se usa de forma complementaria a las anteriores permite disminuir el rango de error en los otros casos. En la actualidad existen varias formas de utilizar esta metodología para localización, a continuación en la Figura 2.3 se muestran dos formas de utilizar AoA para calcular la ubicación de un dispositivo. En el caso A, se calcula la posición de un objetivo utilizando solamente una antena receptora y alguna de las metodologías de cálculo de distancia hacia el objetivo mostradas anteriormente (RSS o ToA). En el caso B, permite calcular la ubicación de un objetivo utilizando los ángulos de recepción de las antenas, y las lineas direccionales de dos antenas distintas. Las coordenadas del objetivo se encuentran a partir de las coordenadas del receptor fijo conocido y los ángulos del haz de la antena en relación con una dirección de referencia común [5].

2.1.5. Fingerprint basado en RSSI

El método de Fingerprint basado en RSSI, corresponde a una técnica que divide el área de interés en secciones, para luego hacer mediciones de intensidad de señal en cada una de las secciones para formar un mapa de intensidades relacionado directamente con la ubicación del dispositivo de medición, esto corresponde a la parte “Offline” del algoritmo. Luego, utilizando este mapa de intensidades, la medición del RSSI del dispositivo que se desea encontrar y algún algoritmo de aproximación como métodos probabilísticos, K-Nearest Neighbors (KNN), Redes Neuronales, Support Vector Machine (SVM),o Smallest M-Vertex Polygon (SMP), se puede encontrar la ubicación del dispositivo dentro del mapa de intensidades anteriormente creado. En la Figura 2.4 se muestra el proceso para calcular el mapa de intensidades de un punto dentro de las divisiones realizadas. Como se aprecia, este punto genera una serie de mediciones de intensidades en distintos receptores, las cuales deben ser guardadas, para luego recrear el mapa de relaciones como en el proceso mostrado en la Figura 2.5.

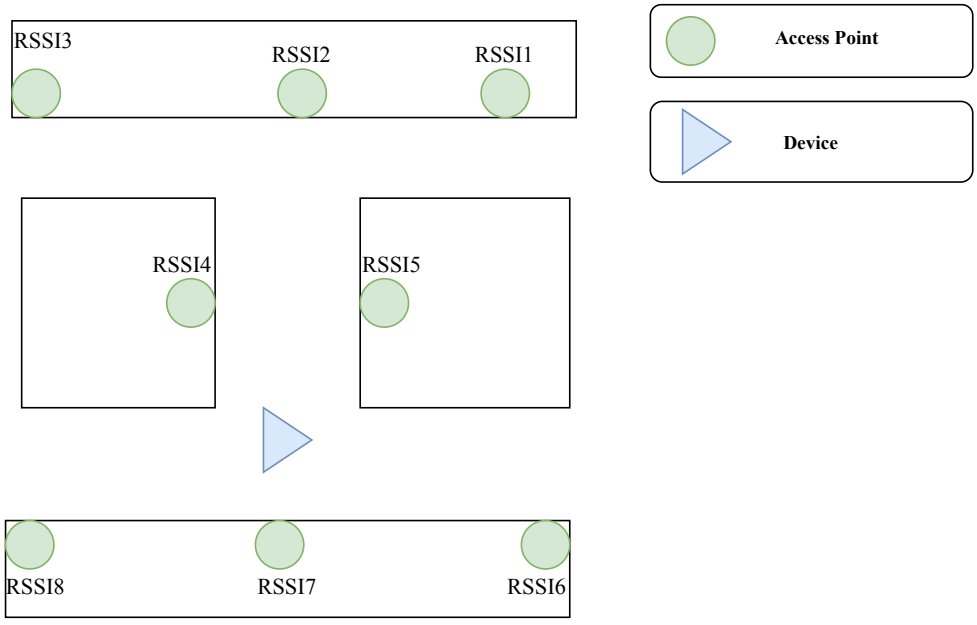


Figura 2.4: Ejemplo de mapa de intensidad de señal,medición de un solo punto de interés.

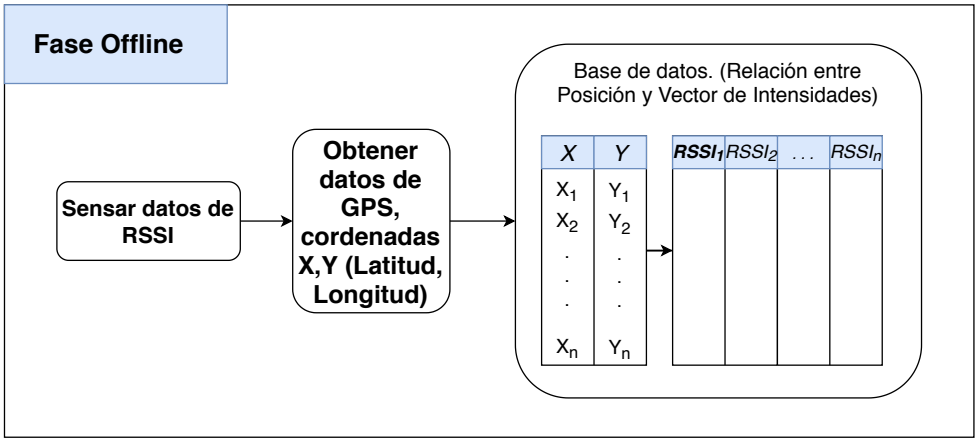


Figura 2.5: Proceso de Fase Offline para algoritmo Fingerprint.

2.2. Análisis de tecnologías inalámbricas

A continuación se realizará una análisis de las tecnologías existentes que permitan la implementación de algoritmos de localización. Se analizará la factibilidad técnica de cada una de estas tecnologías analizando los pros y contras de utilizar cada una de ellas, para finalmente decidir cual será la indicada para este proyecto.

2.2.1. GPS

La tecnología GPS es la más popular al hablar de posicionamiento, ya que tiene un gran alcance, cobertura y precisión. Esta tecnología usa un algoritmo de triangulación mediante satélites que se encuentran orbitando el planeta, logrando obtener información de la posición de cualquier tipo de dispositivo en casi cualquier lugar. Si bien esta tecnología es la más popular, se ha tratado de sustituir por otros tipos de tecnologías ya que todos sus beneficios conllevan un gran consumo energético y un alto costo de Hardware. Además, en algunas aplicaciones se tiene una probabilidad de obtener errores en la lectura de los datos de GPS [6], tal como se muestra en la Tabla 2.1. Si bien estos errores se pueden combatir usando un sistema diferencial de GPS (Differencial Global Positioning System (DGPS)), como se hace normalmente en agricultura, siendo una solución bastante útil cuando los dispositivos monitorizados se encuentran siempre cerca de un lugar de referencia conocido. Por otra parte, en la actualidad se pueden utilizar dispositivos de bloqueo de señal (Jammer), lo que se pueden conseguir entre 14 y 28

dólares (14.000 y 20.000 Pesos Chilenos), los que mediante la emisión de ruido en las frecuencias asociadas a GPS interrumpen la comunicación de estos dispositivos con los satélites para así evitar la localización mediante esta tecnología [7].

Error	Explicación
Reloj	El sistema GPS requiere una sincronización con el satélite y su reloj atómico, lo cual debe ser muy preciso. Por razones de costo, los GPS usan relojes que no están precisamente sincronizados, por lo que requieren de un cuarto reloj para aumentar su precisión.
Efemérides	La orbita de los satélites cambia a menudo, lo que requiere su ajuste y mantención. Debido a esto, pueden darse errores en los datos de localización del satélite que se usan para la triangulación.
Dilución de la precisión (DOP)	La configuración de los satélites respecto a un receptor puede afectar en cualquier momento a la precisión en la posición, en que si los satélites visibles se encuentran muy cerca disminuye la precisión, aumentándola si los satélites se encuentran distribuidos en el cielo. El DOP es cuantificado en la configuración del satélite, por lo que muchos GPS tienen valores para DOP horizontal, vertical, posición y tiempo. Menores valores de DOP indican una mejor configuración del satélite, y en general, si es menor a cuatro dan una buena determinación de la posición.

Atmósfera	Cuando las ondas de radio del satélite del GPS entran en la atmósfera de la Tierra su camino puede ser refractado, lo cual cambia la longitud del camino que toma al receptor, causando un error en la determinación de la distancia. Estos efectos atmosféricos son mayores en satélites bajos en el horizonte, ya que las ondas entran en la atmósfera en más de un ángulo. Debido a esto, algunos receptores GPS permiten al usuario ignorar o enmascarar satélites bajo cierto ángulo en el horizonte.
Múltiples trayectorias	Ocurre cuando la misma señal de radio es captada muchas veces por un receptor a través de diferentes trayectorias. Es un error común en presencia de edificios.

Tabla 2.1: Errores comunes asociados a la señal del GPS.

2.2.2. Wi-Fi

En los últimos años Wi-Fi se ha convertido en una de las tecnologías mas usadas en el mundo, debido a su uso en conexiones de red de acceso local (Local Area Network (LAN)) y su característica inalámbrica para las conexiones a internet muchos lugares incluyendo hoteles, restaurantes, aeropuertos, en los hogares e incluso estaciones de metro y plazas han implementado esta tecnología para ofrecer acceso a internet, por lo tanto se ha convertido en una de las soluciones más usadas de forma cotidiana al tener acceso a la web. Un sistema Wi-Fi o una red inalámbrica de acceso local (WLAN) hogareña sencilla consta de 3 partes principales, una linea de acceso a internet, proporcionada por un cable coaxial o los más sofisticados mediante fibra óptica, un punto de acceso (Access Point (AP)) y adaptador de red inalámbrico, el cual viene integrado en la mayoría de los dispositivos electrónicos en el último tiempo. Estos 3 actores permiten que los dispositivos finales (o Endpoint) tengan acceso a la red de forma inalámbrica tal como se muestra en la Figura 2.6.

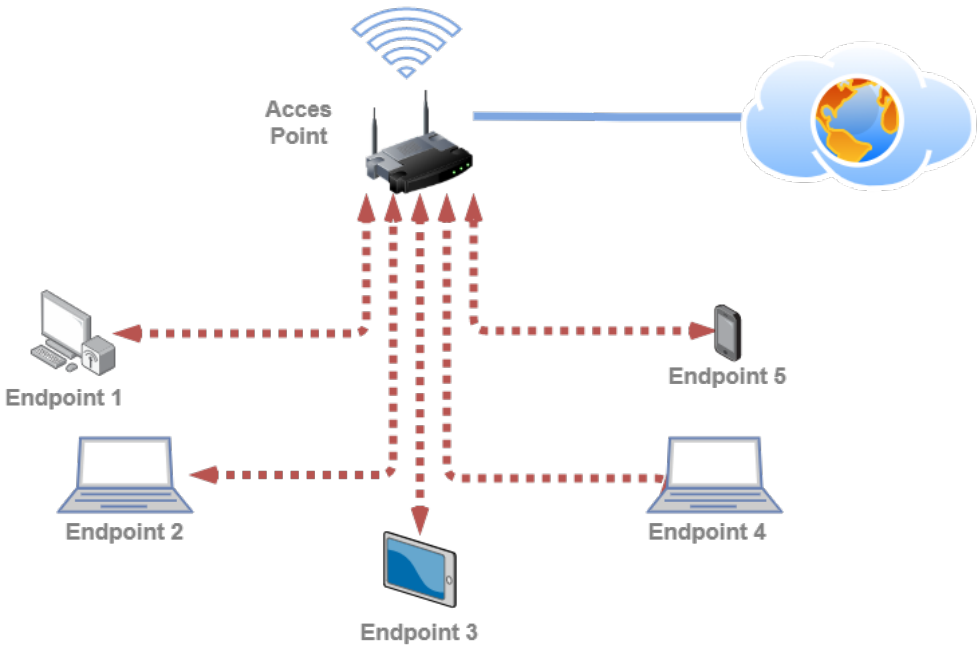


Figura 2.6: Diagrama de arquitectura de una red WLAN sencilla.

Debido a que el problema planteado está orientado a dispositivos móviles, la arquitectura de la red puede cambiar ya que el acceso a internet se obtiene directamente desde algún modem con conexión 3G/GPRSGSM.

Debido al crecimiento de las soluciones IoT, los dispositivos con capacidad de acceder a Wi-Fi han disminuido considerablemente su precio, por lo tanto la integración de esta tecnología en un dispositivo portátil no tiene un precio alto. Además, dependiendo del uso que se le de al dispositivo, esta tecnología puede tener un consumo de energía bajo o medio. Por otra parte, como bien se sabe, esta tecnología tiene una gran capacidad de comunicación de datos, lo que permite transmitir información de forma rápida a través de ella. Además, el alcance que posee una conexión de este tipo en exteriores, se ha extendido hasta aproximadamente los 100 metros.

Localización utilizando tecnología Wi-Fi

Si bien el acceso a internet es el principal uso que se le da a esta tecnología, también puede ser usada para localizar objetos y dispositivos mediante el uso de Puntos de Acceso (AP) distribuidos en un área designada, y en ubicaciones específicas, lo que permite detectar dispositivos

Wi-Fi en las cercanías de cada AP.

Existen varios trabajos que utilizan la tecnología Wi-Fi para localización, por ejemplo en [8] se utiliza un entorno con 3 AP en conjunto con dispositivos portátiles pequeños que son transportados por los usuarios, utilizando un algoritmo Fingerprint-RSSI(Ver Sección 2.1.5) para crear un sistema de localización en tiempo real (Real Time Location System (RTLS)) en un hogar de ancianos de aproximadamente 1800 m^2. Este sistema logra una precisión de aproximadamente 5 metros, aprovechando los beneficios de la tecnología. La finalidad de este trabajo fue localizar a los usuarios de forma fácil para disminuir los tiempos de respuesta ante emergencias, entre otros beneficios de la tecnología Wi-Fi.

Por otra parte en [9], se presenta un trabajo para detectar la ubicación exacta de un AP Wi-Fi, utilizando un algoritmo Triangulación de dos puntos (Ver Sección 7.5.2) para calcular la ubicación, utilizando las direcciones de AoA y un dispositivo GPS para obtener el ángulo exacto de orientación de la antena Wi-Fi. Esta solución permite la correcta determinación de un dispositivo Wi-Fi en exteriores. Esta solución utiliza una antena tipo Yagi como la mostrada en la Figura 2.7, la cual rota sobre un eje para encontrar las direcciones de máxima intensidad de señal son su respectivo ángulo.

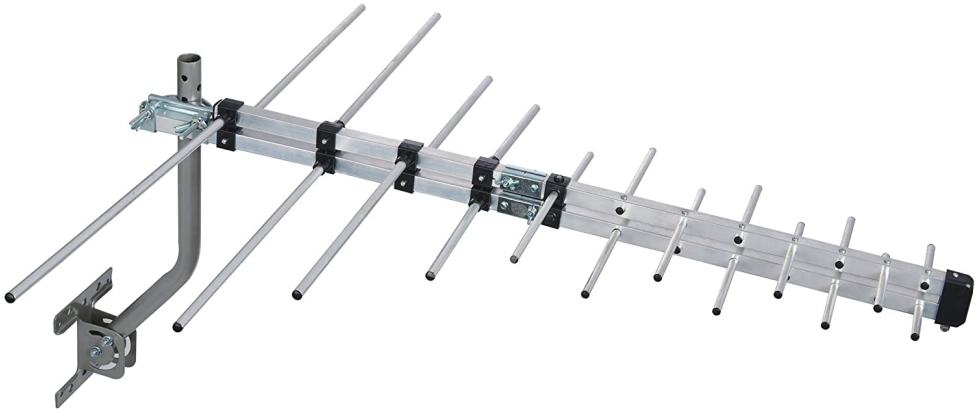


Figura 2.7: Antena Yagi de multiples elementos.

En [10] se trata de mostrar un sistema de ubicación en exteriores, buscando solucionar un problema existente en los actuales proveedores de esta tecnología como WiGLE [11], OpenBMap[12] o Skyhook[13]. Esta propuesta trata de mejorar el algoritmo de localización utilizado por estos proveedores, el cual consiste principalmente en la identificación de redes Wi-Fi de forma periódica, guardando los datos de cada red en una base de datos, para luego aproximar

la ubicación de los dispositivos en referencia a las redes cercanas utilizando un algoritmo de peso ponderado. Si bien es un sistema bastante innovador, se hace bastante difícil aplicarlo al proyecto.

Resumen, ventajas y desventajas

Como se mencionó durante esta sección, la tecnología Wi-Fi posee una serie de alternativas para implementar algoritmos de procesamiento. Si bien en esta sección se mostraron solo algunas de ellas, existen muchas otras variedades de soluciones que están diseñadas para lo mismo. Con los datos presentados, se arma la Tabla 2.2, donde se muestran las ventajas y desventajas de usar la tecnología Wi-Fi en este proyecto específico.

Resumen: Posicionamiento Wi-Fi		
N°	Ventajas	Desventajas
1	Es una solución de bajo consumo	Para la mayoría de los casos se requiere una gran cantidad de dispositivos Wi-Fi repartida en el área de interés, por lo que para un área extensa, el precio de implementación crece con la cantidad de dispositivos que se debe usar.
2	Permite implementar comunicación entre dispositivos de forma fácil	Debido a las limitaciones del hardware de la tecnología Wi-Fi, el área que puede cubrir un sólo AP es baja (aproximadamente 100m).
3	Se pueden usar varios métodos para calcular las posiciones de dispositivos	Al usar un método basado en Fingerprint se hace casi imposible extender la solución a un área extensa, ya que se requiere una gran cantidad de mediciones y datos de la parte Offline del proceso.

Tabla 2.2: Ventajas y desventajas de usar un algoritmo basado en Wi-Fi

2.2.3. Bluetooth

Bluetooth es una tecnología inalámbrica de corto alcance, desde los años 2000 aproximadamente se empezó a masificar con sus primeras versiones siendo integrada tanto en teléfonos móviles como computadores, con lo que se hizo bastante conocida por su capacidad de transferencia de datos y su alcance de aproximadamente 10 metros. Esta tecnología opera en el rango

de frecuencia de los 2.4GHz. La tecnología en sí, con sus especificaciones han ido cambiando y mejorando a lo largo de los años, añadiendo nuevas funcionalidades y seguridad de datos partiendo desde la versión Bluetooth 1.0 hasta la actual versión Bluetooth 5.2.
Al igual que en caso de la tecnología Wi-Fi, la masificación de ésta tecnología ha provocado que su adquisición tenga un menor costo en el tiempo, y debido a su gran utilidad y bajo consumo, se ha integrado en casi todos los dispositivos electrónicos portátiles.

Localización utilizando tecnología Bluetooth

Los trabajos presentados a continuación, se centran principalmente utilizando la tecnología para localización en entornos cerrados (Indoor) debido a la naturaleza de corto alcance de la tecnología y los desafíos que se han presentado en este tipo de problemas. Por ejemplo en [14] se presenta una propuesta de localización utilizando trilateración, pero en vez de utilizar geometría de circunferencias, se propone buscar la recta normal a las rectas que conectan los diferentes puntos de recepción mediante la medición de las intensidades de señal recibidas desde los receptores Bluetooth, la representación gráfica de esta propuesta se presenta en la Figura 2.8.

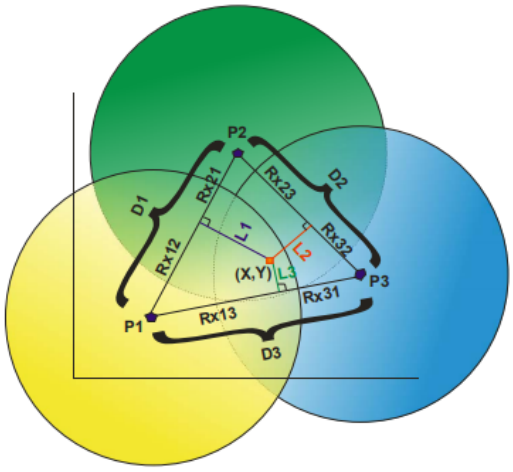


Figura 2.8: Representación gráfica de la solución por trilateración de intensidades Bluetooth. [14]

En esta figura, se asume que la posición de los puntos P1, P2 y P3, son conocidas, además las distancias D1, D2 y D3 también son conocidas. Utilizando el supuesto de que la intensidad de

señal es una relación proporcional a la distancia $P \propto \frac{1}{R^2}$ se puede calcular la relación entre las intensidades de señal en P1 y P2 utilizando la ecuación

$$\frac{P_{RSSI_1}}{P_{RSSI_2}} = \left(\frac{R_{X21}}{R_{X12}}\right)^2$$

Reescribiendo esta ecuación y sabiendo que $D1 = R_{X12} + R_{X21}$, se obtiene que

$$R_{X12} = \frac{D1}{1 + \sqrt{\frac{P_{RSSI_1}}{P_{RSSI_2}}}}$$

Las demás distancias pueden ser encontradas de forma similar, luego, al considerar las intersecciones las rectas L1,L2 y L3 en las lineas que conectan los receptores, podemos definir las ecuaciones de cada recta de la siguiente forma:

$$y_{L1} = \left(\frac{x_1 - x_2}{y_2 - y_1}\right) x_{L1} + \left(\frac{R_{x12}}{D1}\right)\left((y_2 - y_1) - \left(\frac{x_1 - X_2}{y_2 - y_1}\right)(x_2 - x_1)\right)$$

$$y_{L2} = \left(\frac{x_2 - x_3}{y_3 + y_2}\right) x_{L2} + \left(\frac{R_{x23}}{D2}\right)\left((y_3 - y_2) - \left(\frac{x_2 - X_3}{y_3 - y_2}\right)(x_3 - x_2)\right)$$

$$y_{L3} = \left(\frac{x_3 - x_1}{y_1 + y_3}\right) x_{L3} + \left(\frac{R_{x31}}{D3}\right)\left((y_1 - y_3) - \left(\frac{x_3 - X_1}{y_1 - y_3}\right)(x_1 - x_3)\right)$$

Al encontrar la intersección entre estas 3 rectas se obtendrá una región de error, la que corresponde a la posible ubicación del elemento transmisor, tal como se muestra en la Figura 2.9.

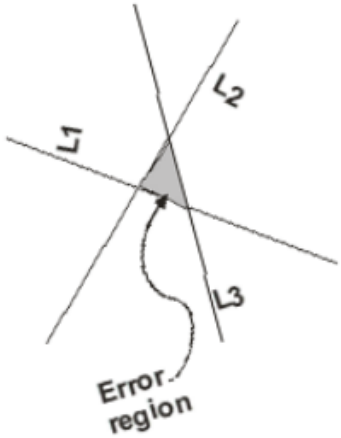


Figura 2.9: Región aproximada de ubicación del objetivo. [14]

Por otra parte, en [15] se muestra un sistema que combina la tecnología Wi-Fi y Bluetooth para implementar un sistema de localización de interiores. En esta solución se genera un modelo de aproximación de distancia para la tecnología Bluetooth utilizando las mediciones de RSSI tomando datos en el entorno donde se desea trabajar, y aplicando una regresión lineal para generar una ecuación que relacione los valores de RSSI con la distancia. En el caso de Wi-Fi se utiliza la ecuación de Friis 2.2 presentada en la Sección 2.1.3 para aproximar la distancia hacia los dispositivos emisores. Finalmente, utilizando un algoritmo que combina los datos de distancia provenientes de los dispositivos Bluetooth y Wi-Fi, se obtiene un sistema de ecuaciones no lineales sobre determinado que es solucionado utilizando mínimos cuadrados, mostrando un excelente rendimiento y precisión en comparación a otros algoritmos estudiados dentro del mismo documento.

Resumen, ventajas y desventajas

Como se acaba de mostrar en esta sección, la tecnología Bluetooth tiene varias alternativas de solución para implementar algoritmos de localización. Al igual que en el caso de Wi-Fi, se mostraron solo algunas alternativas de todas las que existen. En la Tabla 2.3, se presentan las ventajas y desventajas de utilizar esta alternativa para el propósito de este proyecto.

Resumen: Posicionamiento Bluetooth		
N°	Ventajas	Desventajas
1	Es una solución de bajo consumo	Para la mayoría de los casos se requiere una gran cantidad de dispositivos Bluetooth repartida en el área de interés, por lo que para un área extensa, el precio de implementación crece con la cantidad de dispositivos que se debe usar.
2	Permite implementar comunicación entre dispositivos de forma fácil	Debido a las limitaciones del hardware de la tecnología Bluetooth, el área que puede cubrir un sólo AP es baja (aproximadamente 100m como máximo).
3	Se pueden usar varios métodos para calcular las posiciones de dispositivos y se puede combinar con otras tecnologías para aumentar la precisión	Al utilizar algoritmos que requieran combinar otras tecnologías, se debe pensar en el precio de implementación ya que puede incrementarse significativamente.

Tabla 2.3: Ventajas y desventajas de usar un algoritmo basado en Bluetooth

2.2.4. LoRa

Esta tecnología se ha masificado debido a que ofrece comunicación de un largo alcance y bajo consumo de energía, a cambio de una tasa de transmisión de información más baja. En el último tiempo los sistemas LoRa son una gran alternativa de comunicación en entornos cerrados y abiertos en donde no sean necesarios grandes flujos de información, y en sistemas que requieran un muy bajo consumo como los sistemas móviles o proyectos orientado al Internet de las cosas (IoT).

Localización utilizando tecnología LoRa

Al igual que en los casos anteriores, se hará un análisis de los trabajos relacionados tratando de mostrar soluciones en varias alternativas. Por ejemplo, en [16] se presenta una alternativa que utiliza un algoritmo de trilateración basado en RSSI, donde se diseña un modelo de pérdida de intensidad (Path Loss) para un área amplia ubicando varias dispositivos receptores, que miden la intensidad de señal de un dispositivo emisor ubicado estratégicamente en posiciones específicas que permiten obtener datos para diseñar el modelo de intensidad de señal. Por su parte, en [17] también se muestra una solución basada en RSSI, pero recolectando datos y utilizando un algoritmo Fingerprint resuelto con KNN en zonas de larga escala urbana y rural, entregando un error promedio de 398.4 m y un error mediano de 273.03 m. Lo que representa un error apto para su uso en casos donde no sea tan importante la precisión y se le de importancia al consumo de batería.

Por otra parte, en [18] se muestra un sistema de localización utilizando una red LoRaWan que fusiona el algoritmo TDoA con una brújula electrónica (e-Compass) para aumentar la precisión del algoritmo TDoA usando Map Matching basado en los datos provenientes del e-Compass. Para esto se diseña un algoritmo en tiempo real que utiliza los datos de la brújula y de los tiempos de llegada a los dispositivos receptores. Además, se realizan pruebas para distintas formas de transportarse, como caminar, manejar y salir en bicicleta. Los resultados muestran que en los escenarios de caminata el error de cálculo de posición es en el 90% de los casos menor que 100m, pero al cambiar de medio de transporte el error comienza a crecer, mostrando que en escenarios de conducción de vehículos el error de cálculo es de aproximada-

mente 400m.

Finalmente, en [19] se muestra una solución similar al caso anterior, donde se utiliza el algoritmo TDoA en combinación con un algoritmo de seguimiento basado en el modelo de intensidad DVB-H presentado en [20]. Los resultados de esta alternativa muestran que para el 90% de los datos el error en situaciones tanto de caminata, ciclismo y conducción se centra entre los 400m y 500m.

Resumen, ventajas y desventajas

Según los datos mencionados en esta sección, la tecnología LoRa tiene alternativas principalmente basadas en TDoA y RSSI, con algunas variantes entre una solución y otra, entregando en casi todos los casos errores entre los 300 a 500 metros. En la Tabla 2.4, se muestran las ventajas y desventajas de utilizar esta tecnología centrándonos en este proyecto.

Resumen: Posicionamiento LoRa		
N°	Ventajas	Desventajas
1	Es una solución de bajo consumo	La tasa de transmisión de datos que se alcanza es mucho menor que con otras tecnologías.
2	Permite implementar comunicación entre dispositivos	Al utilizar algoritmos basados en RSSI, se requiere un gran trabajo en cuanto a tiempo para definir modelos de perdida de señal o bien mapas de intensidad en los casos donde se use Fingerprint.
3	Se pueden usar varios métodos para calcular las posiciones de dispositivos y se puede combinar con sensores y metodologías para aumentar la precisión	Funciona en bandas de frecuencia de uso libre, por lo que está sujeta a varias regulaciones.
4	Al utilizar algoritmos que requieran combinar otras tecnologías no influye el precio de implementación, sino que el algoritmo y capacidad computacional.	La precisión de esta tecnología ronda entre los 300 y 500 metros.

Tabla 2.4: Ventajas y desventajas de usar un algoritmo basado en LoRa

2.2.5. Comparación de tecnologías inalámbricas

En la Tabla 2.5, se muestra un resumen comparativo entre las distintas tecnologías inalámbricas vistas anteriormente indicando su nivel de consumo, cobertura, costo, seguridad y capaci-

dad de transmisión de datos [21].

Característica	WiFi	Bluetooth	GPS	LoRa
Consumo de energía	Medio	Bajo	Alto	Bajo
Cobertura	Medio	Medio	Alto	Medio o alto
Costo de HW	Medio	Medio	Alto	Bajo
Capacidad de datos	Alto	Medio o alto	No aplica	Bajo o medio
Costo de implementación	Alto	Alto	Medio	Medio o Bajo
Precisión [m]	<1m	<1m	<1m	<500 m

Tabla 2.5: Comparación entre las características de los sistemas de comunicación inalámbricos.

Considerando los datos presentados y el análisis realizado, se determina que **la tecnología más indicada para trabajar en el proyecto es LoRa**, ya que puede usarse para cubrir distancias mayores que las otras tecnologías a cambio de una menor tasa de transmisión de datos, y sacrificando la precisión de los algoritmos, que en una situación donde no se tiene ninguna información de posicionamiento, como la planteada para el uso del dispositivo, una estimación de la posición real es más que suficiente para actuar. Además esta solución mantiene las características de bajo consumo que se necesita y presentando un costo mucho menor en cuanto a todas alternativas de implementación mostradas.

Capítulo 3

Tecnología escogida y selección de algoritmo

3.1. Teoría de trabajo

LoRa es un mecanismo de modulación de Espectro Ensanchado (Spread Spectrum (SS)) de propiedad exclusiva derivada de la modulación de Espectro Ensanchado Escalonado (CSS) (Ver Sección 7.5.2), el cual intercambia la velocidad de transmisión de datos por la sensibilidad dentro de un ancho de banda fijo. Este sistema implementa una tasa de datos variable, utilizando factores de propagación (SF) ortogonales, que permiten al administrador del sistema intercambiar la tasa de transmisión de datos por el alcance o la potencia, para optimizar el rendimiento de la red en un ancho de banda constante.

Lora es una implementación de capa física y es indiferente a las implementaciones de capas superiores. Esto permite que LoRa coexista y opere con las capas de ya existentes en la arquitectura de redes.

En [22], Semtech indica que "En la teoría de la información, el teorema de Shannon-Hartley establece la velocidad máxima en que la información puede transmitirse por un canal de comunicaciones de un ancho de banda determinado en presencia de ruido". Además, como en las aplicaciones de SS, la relación de señal a ruido (Signal to Noise Ratio (SNR)) es muy pequeña ($S/N << 1$), se considera que, "Para transmitir información sin errores en un canal con un SNR fijo, solo es necesario incrementar el ancho de banda de la señal transmitida."

Con lo anterior, se demuestra en [22] que la relación entre la transmisión de datos (Bit Rate), la transmisión de símbolos (Symbol Rate), el ancho de banda (Band Width (BW)) y el factor de propagación (SF) para una modulación LoRa puede ser expresada de la siguiente forma:

$$R_s = \frac{BW}{2^{SF}} \tag{3.1}$$

$$R_b = SF \cdot \frac{\frac{4}{4+CR}}{Rs^{-1}} \tag{3.2}$$

Donde:

- R_s = Symbol Rate (símbolos/seg)
- BW = Ancho de banda de modulación (Hz)
- SF = Factor de propagación (7...12)
- R_b = Bit Rate (bits/seg)
- CR = Code Rate (1...4)
- $\frac{4}{4+CR}$ = Rate Code

Como se mencionó anteriormente, en este esquema de modulación se emplean SF ortogonales, lo que permite que se transmitan distintas señales al mismo tiempo, en el mismo canal con el mínimo deterioro en la sensibilidad en el receptor.

3.2. Spreading Factor (SF)

Como se mostró en la Ecuación 3.2, el SF tiene una relación directa con la velocidad de transmisión de datos. El SF es lo que determina la velocidad de un Chirp (definido en 7.5.2). La velocidad de un solo Chirp corresponde aproximadamente a 2^{SF}. En la Figura 3.1, el aumento del SF en un solo número aumenta la duración de un Chirp aproximadamente al doble [23]. Además, mientras mayor sea el valor del SF, mayor será el alcance de LoRa y mayor será el consumo de batería de los dispositivos.

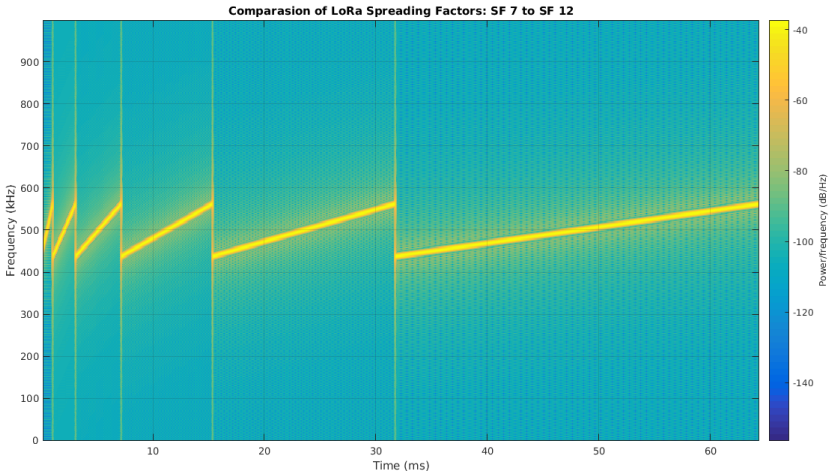


Figura 3.1: Comparación de la duración de un Chirp con SF entre 7 y 12

3.3. Limitaciones

Ya que LoRa es una tecnología relativamente nueva, se deben investigar las limitaciones de utilizarla para el proyecto, ya que si bien nos centraremos en desarrollar un dispositivo de pruebas, este debe cumplir con todos los estándares necesarios para funcionar dentro del País.

3.3.1. Regulaciones

Según Lora Alliance y la regulación Chilena [24], los dispositivos LoRa, tiene permitido funcionar dentro de los rangos de frecuencia de 433 – 434.79 MHz (EU433) y 915 - 928MHz2 (AU915-928) dentro de Chile, por lo tanto estaremos sujetos a las regulaciones de estos dos estándares que viene descritos en el documento [25].

Banda ISM EU433

Según este documento, el Ancho de banda (BW), frecuencia mínima (f_{min}), frecuencia máxima (f_{max}), Bit Rate, Data Rate (DR) y Ciclo de trabajo (Duty Cycle) que deben obedecer los dispositivos que transmitan en esta banda están definidos en la Tabla 3.1.

BW	$f_{min} - f_{max}$	DR / BitRate	Ciclo de Trabajo
125 KHz	433.175 MHz-433.575MHz	DR0-DR5 / 0.3-5 Kbps	<1 %

Tabla 3.1: Limitaciones de frecuencia y datos para la banda ISM EU433.

Banda ISM AU915-928

Las configuraciones de bandas de frecuencia para la esta banda, deben dividirse en los siguientes planes por canal de frecuencia:

- Subida: Si se usa un ancho de banda de 125KHz, el Data Rate puede ser de DR0 a DR5, el Code Rate de 4/5, y los canales serán divididos empezando desde la frecuencia mínima 915.2 MHz e incrementando de 200 KHz linealmente hasta 927.8 MHz.
- Suibida: Si se usa un ancho de banda de 50KHz, el Data Rate puede ser de DR6 o DR7, y los canales serán divididos partiendo desde 915.9 MHz e incrementando de 1.6 MHz linealmente hasta los 927.1 MHz

Para efectos prácticos, en esta banda de frecuencia se supondrá que se debe respetar el mismo ciclo de trabajo que para la Banda ISM EU433 (<1 %) ya que no se indica explícitamente un porcentaje dentro del documento.

3.3.2. Análisis del ciclo de trabajo

Como se mencionó anteriormente, la restricción del ciclo de trabajo para las redes LoRa en las bandas ISM es del del 1 %, pero ¿Cuál es el significado de esto?. A continuación se explicará a que se refiere esta restricción y que consecuencias tiene en la transmisión de datos.
Si se considera que el ciclo de trabajo es un valor d, y el tiempo de transmisión de datos, conocido como Tiempo en el aire (Time on Air), denotado por T_a, entonces esta restricción indica que cada dispositivo que transmita en un canal de la banda de frecuencia debe estar silenciado por al menos un periodo de $T_s = T_a(\frac{1}{d} - 1)$[26]. Para ejemplificar esto, en [27] se muestra una herramienta que permite calcular el ciclo de trabajo para paquetes LoRa en la banda ISM EU868. Esta herramienta permite cambiar el ancho de banda, el tamaño de los paquetes y el SF de transmisión, entregando el valor del Time on Air y el ciclo de trabajo en segundos. En la

Figura 3.2 analizada desde [26], se muestra la relación entre el Time on Air, el SF y el tamaño de los datos enviados en una transmisión utilizando un Code Rate de 4/5 y un ancho de banda de 125KHz. Se puede apreciar como el Time on Air aumenta a medida que se incrementa el tamaño del mensaje enviado y como es considerablemente superior, el doble que el SF según lo mencionado en las secciones anteriores, a medida que aumenta el valor del SF.

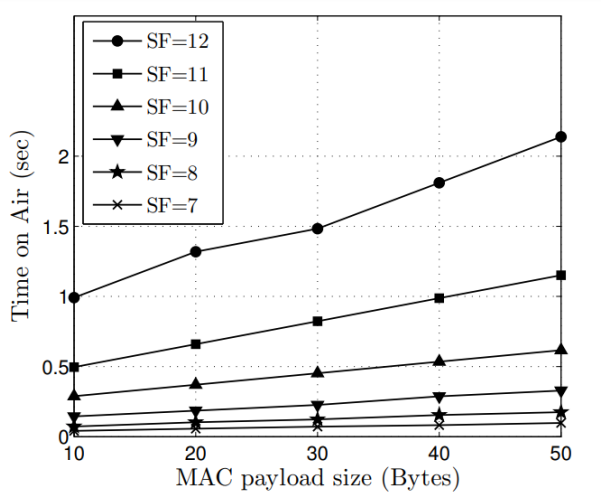


Figura 3.2: Time on Air de mensajes LoRa con SF entre 7 y 12, y tamaño de mensaje entre 10 y 50 bytes. [26]

Además, en las Figuras 3.3 y 3.4 también analizadas en [26], se muestra como la cantidad de paquetes recibidos por hora se relaciona con la cantidad de dispositivos conectados a la red LoRa, mostrando claramente que mientras más dispositivos se encuentren presentes en la red menor será la cantidad de paquetes que cada nodo puede transmitir y recibir. En la Figura 3.4, se muestra el caso específico de paquetes recibidos por hora y por dispositivo con 250, 500, 1000 y 5000 dispositivos en la red utilizando un mensaje de un largo de 10 bytes. Además, en esta misma figura, se muestra como el ciclo de trabajo limita la cantidad de mensajes a un límite específico para obedecer el estándar de la red.

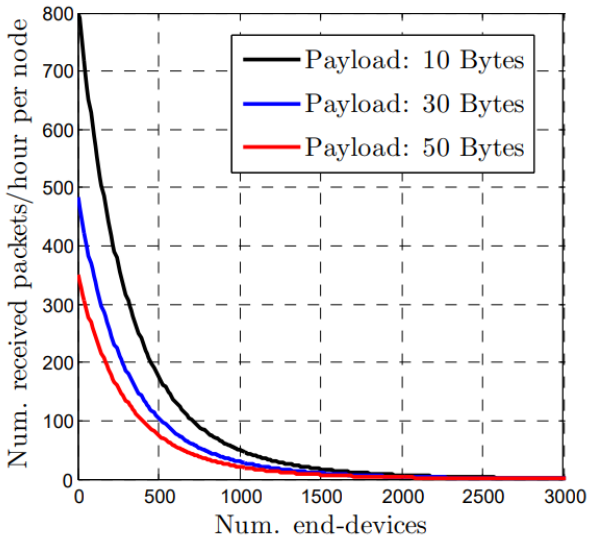


Figura 3.3: Numero de paquetes LoRa recibidos por hora cuando se envían nd/T_{ai} paquetes por segundo. Code Rate de 4/5 y ancho de banda de 125KHz.[26]

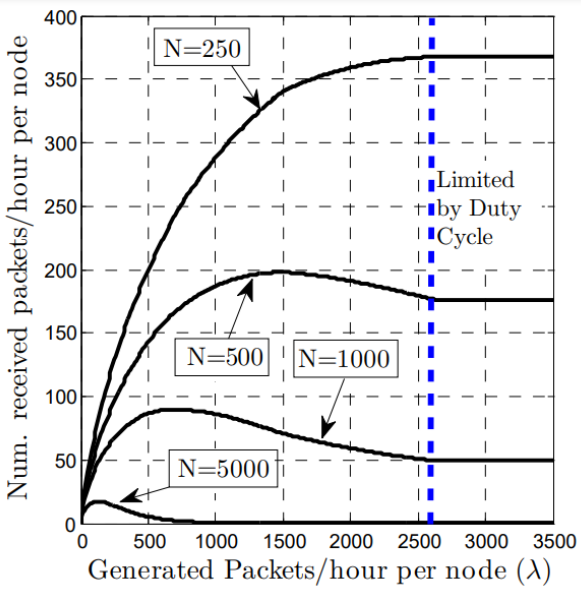


Figura 3.4: Número de paquetes de 10 bytes recibidos por hora y por nodo para 250,500,1000 y 5000 dispositivos.[26]

Según lo mencionado en esta sección, es de suma importancia saber las limitaciones presentes en la red LoRa para diseñar un sistema de transmisión de datos entre dispositivos, ya que si se abusa de la cantidad de mensajes a enviar, se puede llegar a saturar la red y quedar fuera del estándar.

3.4. Más trabajos relacionados

Si bien en el Capítulo 2 se analizan casos de uso para la tecnología LoRa en localización, a continuación se mostrarán otras soluciones que permiten exhibir las capacidades de la tecnología con distintos algoritmos y métodos de detección de ubicación para complementar los antes descrito y tomar una decisión final.

Localización LoRa basado en algoritmo Fingerprint

En [17] se presenta una solución basada en LoRa que utiliza el algoritmo de Fingerprint para detectar la posición de un dispositivo LoRa en un área designada. Al igual que en la sección 2.1.5 , este algoritmo consta de dos fases, una fase offline (o de entrenamiento) tal como la mostrada en la Figura 2.4 y una fase online, que corresponde a la que calcula la posición del dispositivo. A diferencia del algoritmo mostrado en la sección 2.1.5 , en esta solución los datos obtenidos para generar el mapa de intensidades son interpolados para generar una distribución completa de intensidades (esto ya que no es posible tomar datos de intensidad en cada punto del área donde se desea aplicar el algoritmo). Para realizar la interpolación se utiliza una interpolación de función de base radial (RBF) mostrada en la Figura 3.5. Esta elección fue hecha ya que en la solución planteada se hicieron pruebas donde se demuestra que éste tipo de interpolación minimiza el error, y tiene mejores resultados para extrapolar algunas áreas.
Como se mencionó anteriormente, a diferencia del algoritmo mostrado en 2.1.5, en esta solución además de generar un mapa de intensidades (utilizando interpolación RBF), en la fase online se genera un mapa de probabilidad que permite identificar un área en la que es posible encontrar al dispositivo. Esto se realiza comparando el valor de intensidad del dispositivo que se desea localizar con cada mapa de intensidad. Con lo que se obtienen una serie de regiones que son candidatas con las que mediante un factor de peso se genera un mapa de probabilidad. Para seleccionar las regiones que son candidatas, se utiliza un rango de 3 dBm en la medición de RSSI, es decir, si desde el dispositivo se observa un valor de RSSI de -51 dBm en un dispositivo puerta de enlace (Gateway), una región candidata será la que haya obtenido en ese mismo dispositivo valores entre -48 dBm y -54dBm.
Usando los datos observados de RSSI se calcula la probabilidad condicional utilizando un mé-

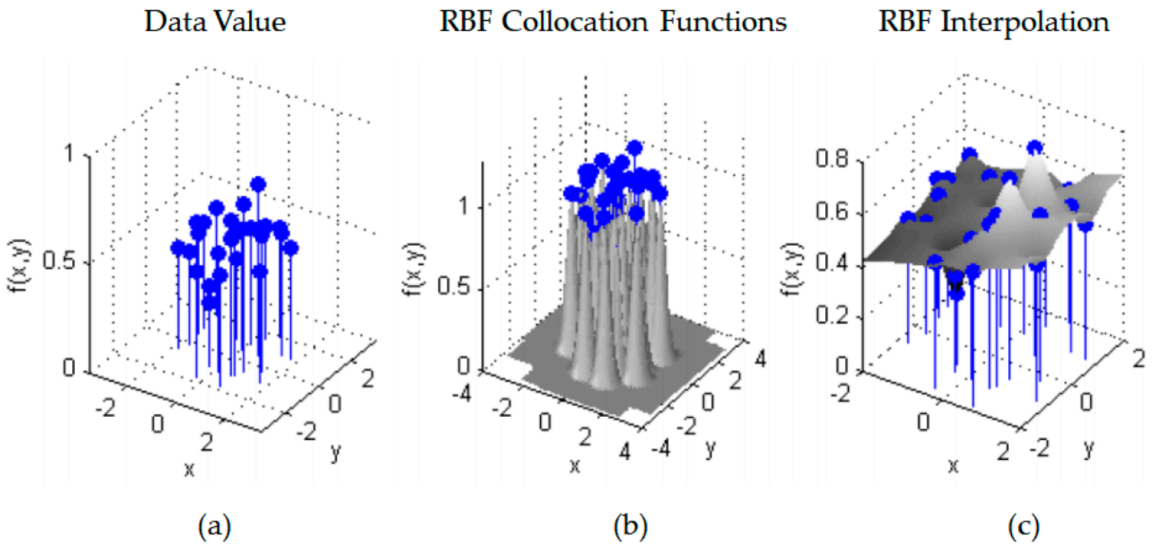


Figura 3.5: Ejemplo de interpolación en función de base radial. a) Muestra de puntos aleatorios. b) RBF centrada en cada punto de la muestra. c) Superficie de interpolación. [17]

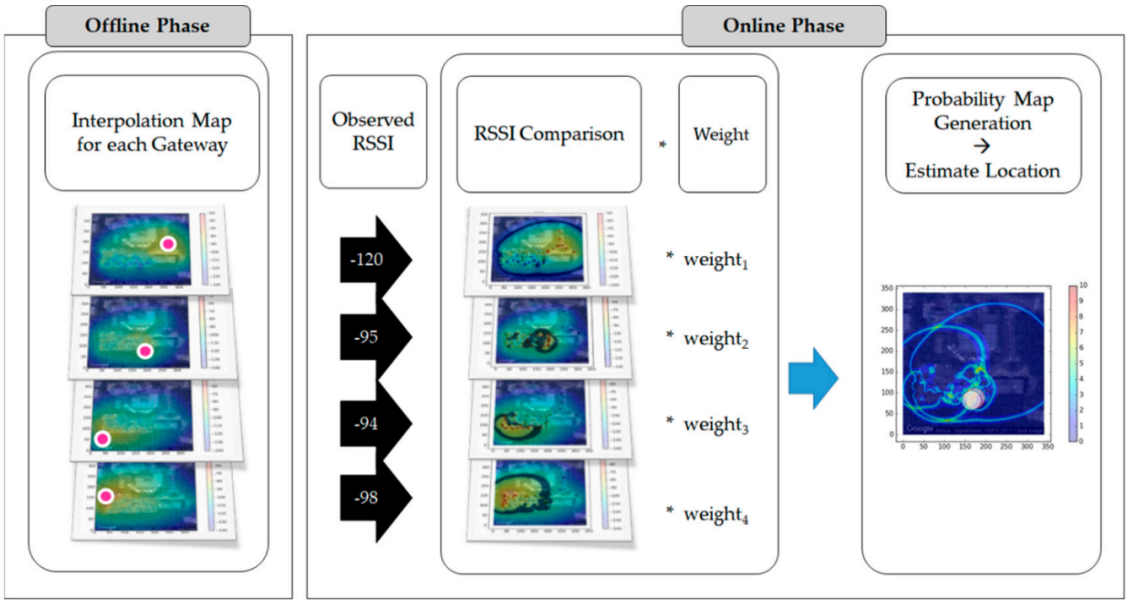


Figura 3.6: Proceso de generación de mapas de probabilidad. [17]

todo de inferencia Bayesiana. Posteriormente el área de prueba se divide en una grilla, donde cada celda es de 1m x 1m. A cada celda se le asigna un índice desde l_1 a l_n. Con este arreglo se define el conjunto $\mathbb{L}$ como los puntos $\{x, y\}$ tal que:

$$\mathbb{L} = \{l_1 = (x_1, y_1), l_2 = (x_2, y_1) \ldots, l_n = (x_c, y_r)\}$$

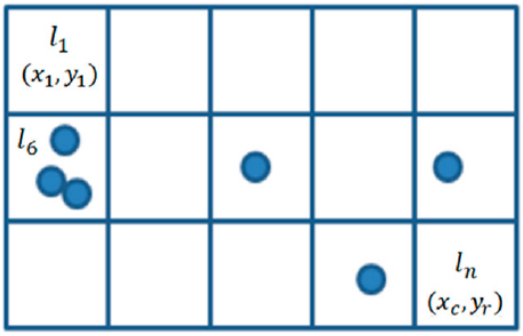


Figura 3.7: Espacio de vectores de la colección de datos RSSI. [17]

Con esto, cada vez que un dispositivo LoRa envié un paquete, todos las puertas de enlace tendrán la medición de un RSSI, generando un arreglo al cual se le aplica el algoritmo MLE (Maximun Likehood Estimation) para calcular la posición aproximada del dispositivo.

Ventajas y desventajas

Ventajas

1.- Es una solución de bajo costo en Hardware, ya que solo requiere los dispositivos LoRa.

2.- Tiene una buena precisión para una gran cantidad de datos.

3.- Al agregar el procesamiento de probabilidad, el tiempo de procesamiento de una posición disminuye.

4.- Se puede implementar la comunicación entre dispositivos.

5.- Es una solución de bajo consumo.

Desventajas

1.- Al igual que en la sección con las tecnologías Wi-Fi y Bluetooth se necesita un preprocesamiento casi imposible de realizar para la aplicación planteada.

2.- Junto con el punto anterior, la implementación solo sirve en un ambiente controlado y de un tamaño pequeño ($< 1Km^2$)).

3.- No se puede implementar con dispositivos en movimiento, ya que el algoritmo se adapta para localizar dispositivos fijos.

Localización LoRa basado en TDoA

En [3] se muestra un análisis del comportamiento de un sistema de localización LoRa usando el algoritmo TDoA en una red privada. Solución similar a la que se muestra en [28], que usa el algoritmo TDoA en un sistema con dispositivos fijos, el cual es descartado debido a las limitaciones que presenta. En [3] se usan módulos conectados a una serie de puertas de enlace LoRa que se encuentran sincronizadas en el orden de los nanosegundos. Para implementar éste algoritmo es un requerimiento fundamental que todos las puertas de enlace se encuentren sincronizadas, ya que el cálculo de la posición se realiza tomando la diferencia en los tiempos en que llega la señal proveniente desde un nodo hasta los diferentes dispositivos puerta de enlace, tal como se mencionó en la sección 2.1.2.

En cuando se calculan las hipérbolas generadas por las diferencias temporales, se debe suponer que *localmente* ambos receptores se encuentran en un mismo plano cartesiano. Situados en el Eje X y a una distancia D. Si consideramos la puerta de enlace i en la posición $X = -D/2$ y la puerta de enlace j en la posición $X = D/2$, entonces la hipérbola local que es representada en este plano cartesiano está descrita por la ecuación:

$$\frac{x^2}{\Delta d/4} - \frac{y^2}{D^2/4 - \Delta d^2/4} = 1$$

Donde Δd es respectivamente la diferencia de distancia calculada par un par de dispositivos puerta de enlace. Usando las asíntotas de estas hipérbolas se puede estimar el punto en que se encuentra el nodo. Pero como se mencionó, las coordenadas (x, y) deben ser traspasadas posteriormente a un plano global en que se encuentren todos los pares de dispositivos. Con lo que finalmente la intersección de las asíntotas de éstas hipérbolas será el punto donde se encuentre el nodo. Tal como se muestra en la Figura 3.8, donde los puntos negros son respectivamente las puertas de enlace, y el punto rojo representa un nodo que se desea localizar.

En esta solución se usan datos de dispositivos puertas de enlace propietarios , esto quiere decir que son dispositivos instalados por una empresa dedicada a éste rubro y a la expansión de esta tecnología. Al usar este tipo de dispositivos se consigue una sincronización del orden de los nanosegundos (ya que es uno de los servicios que ellos ofrecen). Desde los dispositivos se obtiene una serie de datos relacionados con la recepción de un mensaje desde los nodos, los

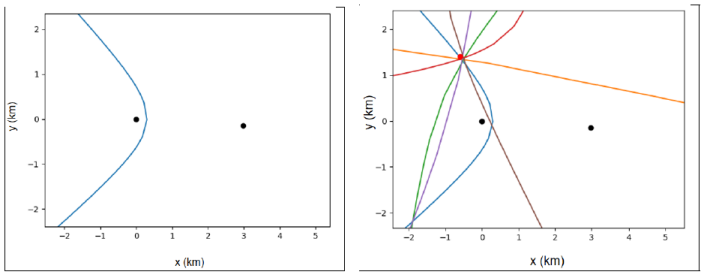


Figura 3.8: (a) Hipérbola que representa todos los posibles puntos donde se localiza un nodo (plano local). (b) Intersección de un conjunto de hipérbolas que representan todos los posibles puntos donde se localiza un nodo (plano global). [3]

que se encuentran en movimiento, como se muestra en la Figura 3.9. Entre los datos que entregan las puertas de enlace se encuentran, el tiempo en que se reciben los paquetes (timestamp) con una precisión de hasta 9 decimales, la relación señal a ruido (SNR) y el intensidad de señal recibida (RSSI).

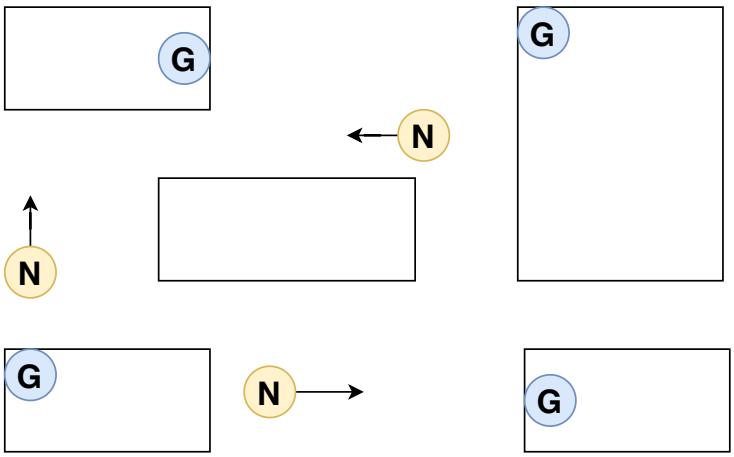


Figura 3.9: Distribución de dispositivos de puerta de enlace fijos y nodos en movimiento.

Ventajas y desventajas

Ventajas del método

1.- Al usar éste algoritmo se elimina el pre-procesamiento de datos, ya que no es necesario establecer un mapa de intensidades (como en los algoritmos anteriores). Considerando el lo mencionado anteriormente, y la aplicación en la que se desea implementar el po-

sicionamiento, el almacenamiento necesario para implementar este algoritmo es mucho menor comparado con los otros algoritmos presentados.

2.- El costo de implementar el algoritmo es relativamente bajo, ya que no se requiere tiempo extra ni capacidad de almacenamiento para la parte offline de los otros algoritmos y solo sería necesario contar con los dispositivos LoRa que conformen la red.

3.- Es un algoritmo relativamente rápido, ya que usa procesamiento matemático simple, lo que permite la posibilidad de posicionamiento en tiempo real.

4.- Se puede implementar comunicación entre dispositivos.

5.- Es una solución de bajo consumo.

Desventajas del método

1.- Si no se usa una red prioritaria (privada y pagada), es necesario crear una forma para sincronizar todos los dispositivos de la red.

2.- Las variaciones en el cálculo de posición pueden ser grandes dependiendo de la precisión con la que se midan los tiempos.

3.- Es necesario que la red esté conformada por al menos cuatro dispositivos y sólo uno pueda consultar su posición a la vez. Por lo que con menos dispositivos es imposible de implementar.

4.- Si se implementa en dispositivos en movimiento, y sin la intervención de puertas de enlace fijas, es necesario que cada dispositivo tenga información sobre su posición en todo momento.

Localización Lora basado en algoritmo híbrido

En [29] se presenta una solución híbrida, en la que se combina tanto un modelo de perdida de señal mediante RSSI (Path loss model) y un algoritmo de localización mediante cálculo del tiempo de vuelo Time of Flight (ToF). En esta solución se presentan varias limitaciones correspondientes al protocolo LoRa , presentadas por completo en [26], como las analizadas en las secciones anteriores, las que no son mencionadas en las otras soluciones.

En ésta alternativa se usa como base el modelo de propagación de señal Okumura–Hata para ciudades pequeñas, presentado a continuación:

$$L_H = 69{,}55 + 26{,}16 Log_{10}(f) - 13{,}82 Log_{10}(h_t) - a \cdot h_r + (44{,}9 - 6{,}55 Log(h_t)) Log(d) \quad (3.3)$$

Donde L_H son las perdidas en un ambiente urbano en decibeles (dB), f es la frecuencia de transmisión en Mhz, h_t es el largo del transmisor en metros (m), h_r es el largo del receptor en metros (m), a es el factor de corrección de la antena y d es la distancia entre el transmisor y receptor en kilómetros (km). Este modelo se usa en conjunto con el dato de intensidad de señal RSSI para estimar la distancia aproximada entre el nodo emisor y el nodo receptor resolviendo la ecuación:

$$d = 10^{\frac{(L_H - 69{,}55 - 26{,}16 Log_{10}(f) + 13{,}82 Log_{10}(h_t) - a \cdot h_r)}{(44{,}9 - 6{,}55 Log(h_t))}} \quad (3.4)$$

El tiempo de vuelo de una señal corresponde al tiempo desde que se emite el mensaje desde el transmisor hasta que se recibe en el receptor. Para calcular la distancia se usa la expresión:

$$\Delta d = c \cdot ToF \quad (3.5)$$

Para usar esta expresión, al igual que en el algoritmo TDoA se necesita que los dispositivos cuenten con una sincronización de relojes, para que los tiempos en que se envían y reciben los mensajes concuerden. Para evitar tener una sincronización entre relojes, esta solución utiliza un protocolo de dos pasos para calcular el tiempo. En la Figura 3.10 se presenta de forma gráfica el protocolo para calcular el ToF, el cual es explicado a continuación.

- El nodo A envia un “mensaje de rangoz guarda el tiempo en que se envía el mensaje como T_{rsA}
- El nodo B guarda el tiempo de recepción del mensaje proveniente del nodo A como T_{rsB}
- El nodo B envia un mensaje de rango en respuesta hacia el nodo A y guarda el tiempo en que envía el mensaje como $TrspB$ y calcula $T_{proc} = T_{rspB} - TrsB$
- El nodo A guarda el tiempo de recepción del mensaje proveniente desde el nodo B como

T_{rspA} y calcula $T_{trip} = T_{rspA} - T_{rsA}$

- El nodo A envia un mensaje de tiempo al nodo B con el tiempo T_{trip}
- El nodo B recibe el mensaje de tiempo desde el nodo A y calcula el tiempo de vuelo como :

$$ToF = \frac{T_{trip} - T_{proc}}{2} \tag{3.6}$$

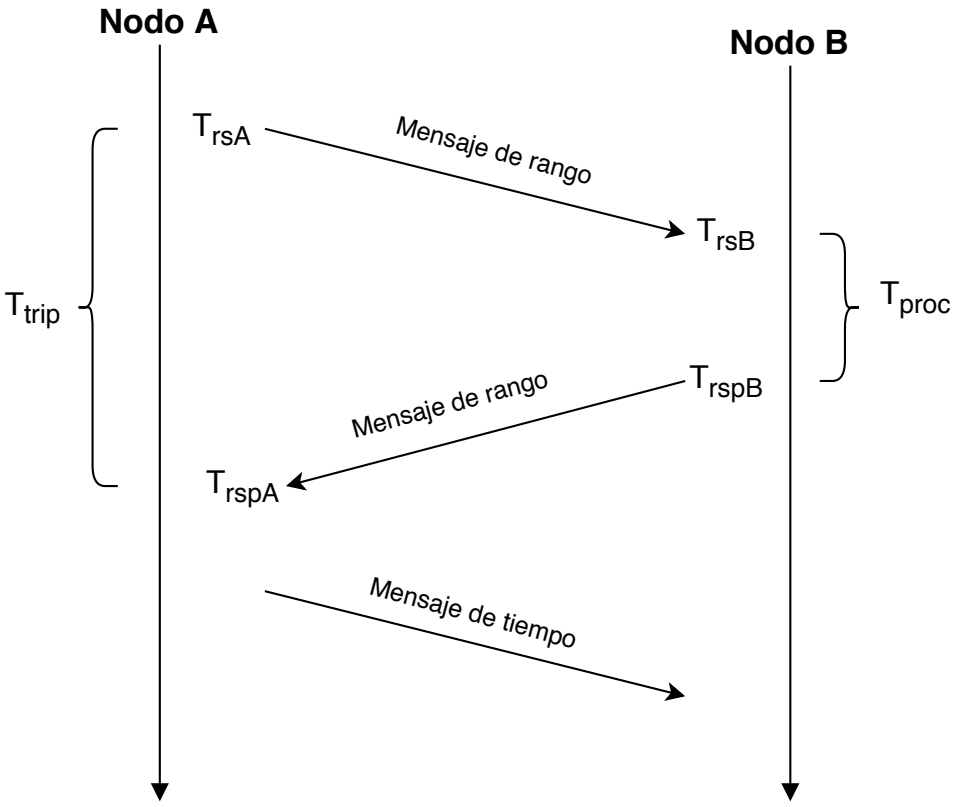


Figura 3.10: Protocolo de cálculo de ToF de forma visual. [29]

Como se observa, con esta implementación se independiza a los nodos de tener una sincronización entre ellos, y como en el algoritmo TDoA la precisión del algoritmo depende netamente de la precisión de los relojes usados en el hardware.

Finalmente para calcular las posiciones aproximadas de los dispositivos transmisores (en este caso los dispositivos que deseamos localizar) se usa el algoritmo de multilateración, similar al algoritmo TDoA, pero en este caso se usan circunferencias centradas en los dispositivos receptores, de radio $R_i = d_i$ correspondiente a la distancia calculada desde el dispositivo transmisor

al dispositivo receptor i, para así representar todas las posibles posiciones que tendría el dispositivo a localizar, luego al intersectar al menos tres de éstas circunferencias se puede estimar la posición aproximada del dispositivo transmisor. Esta alternativa permite combinar las mediciones tanto del modelo de pérdida de señal y las del algoritmo ToF. Como en los ambientes reales se reciben muchos datos con ruido o erróneos, se debe eliminar los datos que sean posiblemente ruidosos, para esto se usa una función que "recorta"los datos eliminando los que no se encuentren dentro de un límite específico y luego promediando los datos que se encuentren dentro de éste límite.

Ventajas y desventajas del método

Ventajas

1.- Es un método de bajo costo, ya que solo se necesitan los dispositivos LoRa para su implementación.

2.- Puede tener una buena precisión en entornos que correspondan al modelo de perdida de señal seleccionado.

3.- El procesamiento es relativamente rápido ya que se puede resolver mediante métodos iterativos.

4.- Se elimina la necesidad de que los dispositivos de la red se encuentren sincronizados.

5.- Se puede implementar comunicación entre dispositivos.

6.- Es una solución de bajo consumo.

Desventajas

1.- Debido al protocolo de rango, el tráfico de los dispositivos puede aumentar ya que el transmisor necesita conectarse a cada uno de los dispositivos cercanos para informar los parámetros temporales.

2.- Si el ambiente en que trabajan los dispositivos cambia , el modelo de pérdida de señal puede introducir mucho error en las mediciones.

3.- Si la cantidad de dispositivos de la red aumenta se puede saturar la red con mensajes con información temporal, con lo que se puede llegar a violar los estándares indicados en [26] y [25].

3.5. Criterios de selección

En esta sección se dará a conocer los criterios de selección para escoger entre las alternativas planteadas. Las alternativas anteriormente presentadas se centran en encontrar una solución al principal problema que se presenta en este proyecto, encontrar una aproximación de posición geográfica del dispositivo, permitiendo la comunicación entre los distintos dispositivos sin dependencia de redes celulares. Debido a esto es que los criterios de selección se centran principalmente en la factibilidad de construir los modelos planteados en el ambiente de trabajo ya establecido , el cual es una limitante estricta del proyecto. Es decir, los criterios se centran en qué tan factible es incorporar la solución en un ambiente de vehículos en movimiento, con una gran cantidad de dispositivos. Debido a esto, los criterios de selección son los siguientes:

- 1.- **Comunicación entre dispositivos**: Debido a que una de las finalidades de esta solución es entregar señales de alerta de estados no comunes, es de vital importancia que el sistema permita una comunicación entre los dispositivos que conforman la red.
- 2.- **Costo de implementación**: Los costos en cuanto a tiempo, volumen de datos y costo del hardware deben ser relativamente bajos.
- 3.- **Trabajo en tiempo real**: Ya que el sistema sobre el que se desea implementar esta solución, entrega información en tiempo real, se espera que la implementación también pueda trabajar con datos en tiempo real.
- 4.- **Estabilidad de la red**: Con la implementación se espera que la red se mantenga estable y no se sature cuando exista una gran cantidad de dispositivos.
- 5.- **Bajo Consumo**: Se espera que la solución planteada sea de bajo consumo ya que será implementada en vehículos, y si ésta tiene un consumo excesivo puede interferir en el funcionamiento general de los vehículos en que se implemente.

- 6.- **Escalabilidad**: La solución debe ser escalable tanto a ciudades grandes como a varias ciudades, la complejidad para implementar esta solución al cambiar de ambiente debe ser baja.

- 7.- **Dispositivo en movimiento**: Su implementación permite trabajar con dispositivos que se encuentren en movimiento.

Cada uno de estos puntos tiene una ponderación relativa que refleja la importancia dentro del proyecto. A continuación se presenta una tabla con las ponderaciones relativas de todos los criterios mencionados.

Criterio	Ponderación
Comunicación entre dispositivos	10 %
Costo de implementación	5 %
Trabajo en tiempo real	10 %
Estabilidad de la red	20 %
Bajo consumo	20 %
Escalabilidad	15 %
Dispositivos en movimiento	20 %
	100 %

Tabla 3.2: Ponderación relativa de criterios de selección.

Se ha dado poca relevancia al tiempo requerido para la implementación y al volumen de datos necesarios, ya que normalmente son parámetros que van de la mano con un desarrollo nuevo. Por otra parte los items de comunicación entre dispositivos, estabilidad de la red, bajo consumo y trabajo en tiempo real se les ha dado una mayor ponderación ya que son aspectos intrínsecos del entorno en que se desea implementar la solución.

3.6. Evaluación de alternativas

Para evaluar las alternativas planteadas se propone el siguiente sistema de evaluación

Muy malo	Malo	Regular	Bueno	Muy Bueno
0 - 0.2	0.3 -0.4	0.5 - 0.6	0.7 - 0.8	0.9 - 1.0

Tabla 3.3: Sistema de evaluación de alternativas

3.6.1. Puntuación de alternativas

En esta sección se muestra la puntuación asignada a cada una de las soluciones planteadas entregando una breve explicación de por qué obtuvo esta puntuación.

Localización LoRa basado en algoritmo Fingerprint

En esta solución se utiliza un algoritmo de Fingerprint basado en LoRa que para mejorar la precisión utiliza mapas de probabilidad de posición, entregando una solución precisa pero con algunas complicaciones.

Criterio	Puntuación	Explicación
Comunicación entre dispositivos	0.8	Al usar el protocolo LoRa se puede implementar comunicación entre dispositivos de forma directa si se genera también el Firmware necesario.
Costo de implementación	0.4	Al usar LoRa se reduce el costo de implementar soluciones basadas en Fingerprint ya que las puertas de enlace tienen más cobertura. No obstante la recolección de datos y las continuas actualizaciones necesarias aumentan considerablemente el costo con respecto a otros algoritmos.
Trabajo en tiempo real	0.4	Esta solución en particular no está diseñada para funcionar en tiempo real, ya que requiere actualizaciones periódicas del mapa de probabilidades para mantener la precisión del algoritmo.

Estabilidad de la red	1.0	En esta solución la red se puede mantener estable ya que solo hay comunicación hacia las puertas de enlace, lo que hace que al aumentar la cantidad de dispositivos no afecte en gran medida la comunicación.
Bajo consumo	1.0	Al usar el protocolo LoRa el sistema está diseñado para ser de bajo consumo.
Escalabilidad	0.2	Lamentablemente esta solución no es escalable ya que se requiere realizar la toma de datos en cada lugar que se implementará, realizar el mapa de probabilidad y además actualizarlo de forma periódica.
Dispositivos en movimiento	0.6	Esta solución permite movimiento de los dispositivos pero no está diseñada pensando en eso.
Calificación	69%	

Tabla 3.4: Puntuación total alternativa 2.

Localización LoRa basado en TDoA

En esta solución se utiliza el algoritmo TDoA para calcular posiciones de dispositivos en movimiento, esto introduce una cierta cantidad de error en los datos pero entrega mejores resultados en otros ámbitos de la aplicación que se desea implementar.

Criterio	Puntuación	Explicación
Comunicación entre dispositivos	0.8	Al igual que el caso anterior, al usar protocolo LoRa permite la comunicación entre dispositivos si se genera el Firmware correspondiente.

Costo de implementación	0.6	Al usar LoRa se reduce el costo de implementar este tipo de soluciones ya que las puertas de enlace tienen más cobertura. No obstante la necesidad de tener puertas de enlace cada cierta cantidad de kilómetros aumenta este costo de implementación.
Trabajo en tiempo real	1.0	Esta aplicación en específico está diseñada para calcular posiciones en tiempo real, por lo que es ideal para el problema.
Estabilidad de la red	1.0	Al igual que en el caso anterior , al usar LoRa y una comunicación solamente hacia las puertas de enlace mantiene la red estable.
Bajo consumo	1.0	al ser una solución con protocolo LoRa está diseñada para ser bajo consumo.
Escalabilidad	0.6	Si bien la solución puede ser fácilmente escalable, ya que no requiere procesamiento que dependa de cada entorno en que se instala, puede ser un problema poner una gran cantidad de dispositivo en la misma red.
Dispositivos en movimiento	1.0	Esta solución está diseñada y probada en un ambiente en que los dispositivos se encuentran en movimiento. Por lo que cumple a la perfección con la expectativa.
Calificación	90%	

Tabla 3.5: Puntuación total alternativa 3.

Localización Lora basado en algoritmo híbrido

En esta solución se presenta un algoritmo híbrido que integra una implementación usando ToF y un modelo de pérdida de señal por RSSI.

Criterio	Puntuación	Explicación
Comunicación entre dispositivos	0.8	Como se explico en los casos anteriores , al usar protocolo LoRa se puede implementar comunicación entre dispositivos siempre que se desarrolle el Firmware adecuado.
Costo de implementación	0.5	Si bien el costo de instalar puertas de enlace cada cierta cantidad de kilómetros es el mismo que en el método anterior, en esta solución se debe considerar el modelo de pérdida de señal para su desarrollo, lo que aumenta su costo de implementación (se necesita más tiempo de pruebas).
Trabajo en tiempo real	0.9	Al igual que en el caso anterior, esta solución está diseñada para trabajar en tiempo real. Pero la implementación que presenta la hace más lenta.
Estabilidad de la red	0.8	Si bien el protocolo LoRa promete una cantidad gigante de dispositivos en una red, la implementación de esta solución puede ocasionar una saturación de la red, ya que para calcular la posición de un dispositivo, éste debe conectarse secuencialmente con las N puertas de enlace más cercanas y realizar un protocolo de cálculo de tiempo.
Bajo consumo	1.0	Al igual que todos los casos anteriores, esta solución está diseñada para ser bajo consumo.

Escalabilidad	0.9	Si bien la solución puede ser escalable debido a su precisión y ´´fácilïmplementación, la estabilidad de la red puede jugar en contra.
Dispositivos en movimiento	1.0	Al igual que en la solución anterior, el sistema fue diseñado y probado para localizar dispositivos en movimiento, por lo que se hace ideal para la aplicación que se desea implementar.
Calificación	89 %	

Tabla 3.6: Puntuación total alternativa 4.

3.7. Algoritmo solución seleccionado

Localización LoRa basado en TDoA

Ya que de esta solución se introdujo la forma en que se calcula la posición de los dispositivos, en esta sección se presentarán las consideraciones a tener en cuanto a la implementación de la solución, orientada a la problemática que se desea resolver.

3.7.1. Dispositivos en movimiento

La solución escogida está diseñada para funcionar con dispositivos que se encuentren en movimiento, comunicándose directamente con puertas de enlace fijas en posiciones conocidas. El entorno de trabajo en que se desea implementar esta solución corresponde a una población de dispositivos que se encuentran en movimiento y que por algunas razones pueden perder la información de posición o necesitan informar situaciones no convencionales. Para cubrir esta diferencia se hace necesario que los dispositivos tengan información de su posición "en todo momento", esto ya es una característica del sistema, por lo que no es necesario implementar pasos extra para cumplirla. Los problemas que podrían presentarse vienen dados por los tiempos en que se obtiene la posición de las puertas de enlace y los tiempos en que se recibe la petición

de posición, ya que si se considera un dispositivo que se mueve a 120Km/h (caso extremo), si la posición tiene un retardo de 1 segundo se añade un error de 33 metros, solamente debido a la frecuencia con que se reciben las posiciones. Este caso está representado en la Figura 3.11, donde se muestra la posición usada para representar a la puerta de enlace (X, Y), y la posición real en la que se encuentra la puerta de enlace $(X + ex, Y + ey)$, donde ex es el error introducido en la coordenada X de la posición, y el valor ey es el error introducido en la coordenada Y.

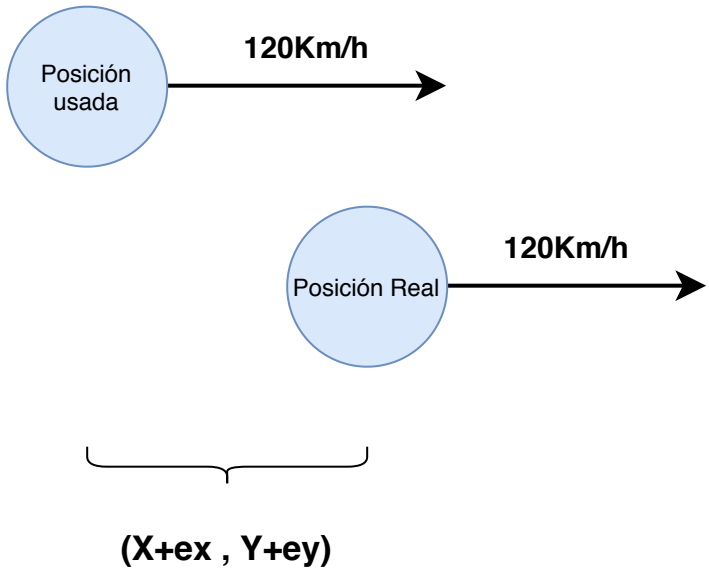


Figura 3.11: Representación del error introducido al usar una posición con frecuencia de 1 seg.

3.7.2. Sincronización entre dispositivos

Para implementarla solución se requiere que los dispositivos cuenten con sincronización, debido a las limitaciones de hardware no se puede implementar una sincronización del orden de los nanosegundos, pero si se puede implementar una sincronización del orden de los microsegundos, por lo tanto se debe encontrar una forma de sincronizar dispositivos de la red. Una posible solución para esto puede ser utilizando el reloj de sincronización de la señal de GPS, y que los datos que llegan desde esta señal tienen una hora con precisión de los milisegundos. Esta señal se puede usar para calibrar los relojes de todos los dispositivos (cuando tengan señal de GPS) y así sincronizar en el orden de los milisegundos con la señal GPS y en el orden de los microsegundos con el reloj interno del dispositivo. En esta implementación se deben considerar todos

los retardos asociados tanto a la adquisición de los datos de GPS como al movimiento de la señal para minimizar los errores de sincronización.

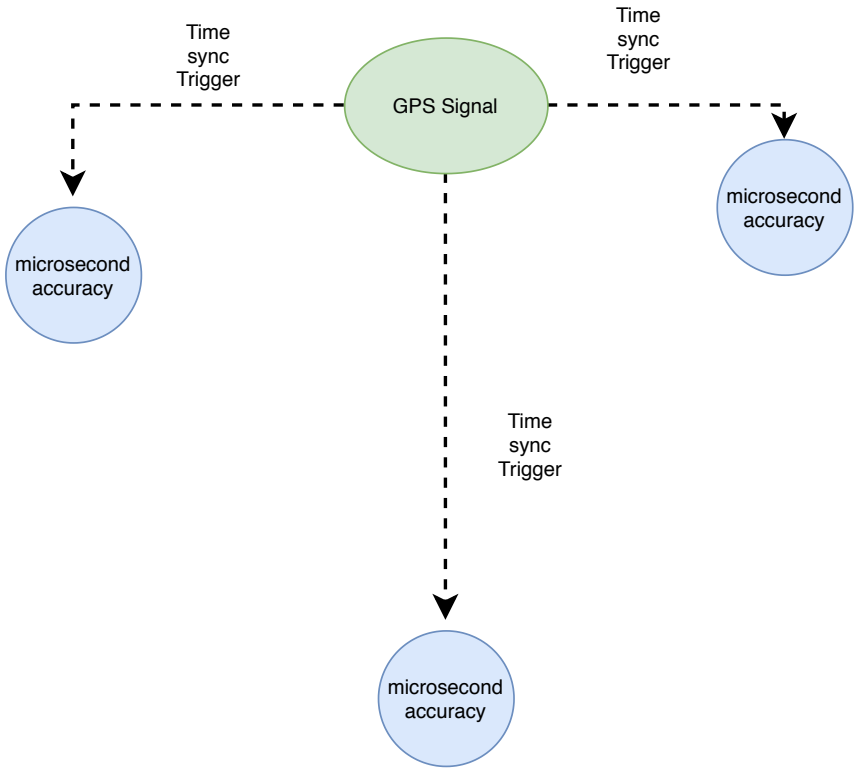


Figura 3.12: Sistema de sincronización de reloj propuesto.

Capítulo 4

Diseño de Hardware

En este capítulo nos dedicaremos a realizar un análisis de distintos componentes que cumplen las funciones básicas necesarias para desarrollar el circuito de pruebas, entre los que se encuentra el módulo de conexión GPRS/3G, el módulo GPS para comparar los resultados, un microcontrolador que funcionará como cerebro, y un módulo de comunicación LoRa para implementar el algoritmo seleccionado.

4.1. Análisis de componentes

4.1.1. Módulo de conexión GPRS/3G

Para la conexión GPRS/3G, existen distintas alternativas en el mercado, las cuales cumplen con creces el objetivo que se le desea dar al módulo. Para la selección nos centraremos principalmente en el precio y en la experiencia de trabajo, ya que al ser un dispositivo de pruebas, no es necesario gastar tanto en su diseño, y al tener experiencia trabajando con algunas alternativas, se hará el trabajo mucho más rápido y eficiente.

Módulo SIM5320

Este módulo, como el presentado en la Figura 4.1, posee un componente principal que pertenece a la empresa SIMCOM. En la página oficial [30] se muestran las características generales de éste módulo, donde se describe que las tecnologías de comunicación soportadas son Dual-

Band High-Speed Downlink Packet Access (HSDPA)/Wideband Code Division Multiple Access (WCDMA), Quad-Band GSM/GPRS/Enhanced Data Rates for GSM Evolution (EDGE) y un GPS integrado. Este módulo funciona mediante comandos AT a través de Universal asynchronous Receiver Trasmitter (UART), lo que permite una fácil manipulación de sus parámetros y conexión a internet. En [31] se muestra la serie de comandos asociados a éste módulo. Además según [32] se puede ver que el módulo permite una conexión TCP/IP de forma fácil. Anteriormente se ha trabajado con éste módulo, pero como se intenta realizar solamente un dispositivo de pruebas no es necesario invertir dispositivos con un alto precio, por lo tanto es una opción que se puede descartar fácilmente, ya que en el extranjero su precio ronda los 43 Dólares (30.000 pesos Chilenos) y en Chile su precio ronda los 85.000 pesos.

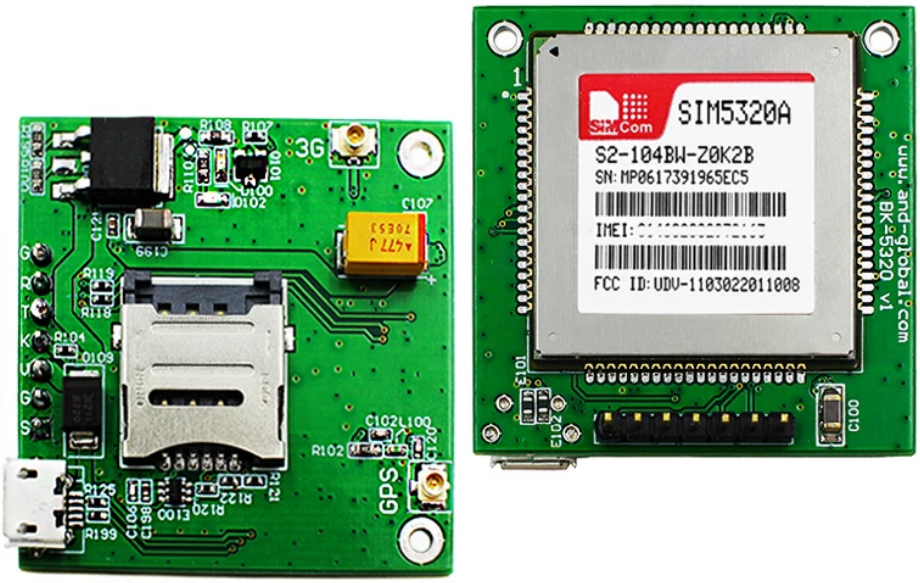


Figura 4.1: Módulo SIMCom SIM5320A

Módulo SIM900

En este módulo mostrado en la Figura 4.2, el componente principal también pertenece a la compañía SIMCom. En la página principal de SIMCom [33] se muestra que las tecnologías compatibles con éste módulo son Quad-band GSM/GPRS, además este caso el módulo no posee GPS integrado lo que lo hace un poco más barato que el SIM5320. Como su compañero, también funciona mediante comandos AT a través de UART, lo que lo hace fácil de configurar para establecer una conexión a internet. En [34] se muestran los comandos asociados a éste módulo para así configurarlo. Además, según [35] se puede configurar el módulo fácilmente para establecer una conexión TCP/IP. El precio de éste módulo en el extranjero ronda en los 11 Dólares (8.000 pesos Chilenos), por otra parte en Chile este módulo tiene un precio de 20.000 Pesos. Si bien es bastante más barato que el módulo anterior, sigue siendo un precio elevado para un módulo que no tiene GPS integrado.

Figura 4.2: Módulo SIMCom SIM900

Módulo A6

Este módulo, como el mostrado en la Figura 4.4, se hizo famoso hace algunos años, ya que pertenece a la compañía Ai-Thinker Inc, la que fabrica los módulos Wi-Fi ESP8266. En [36] se muestra que el módulo es compatible con las tecnologías GSM/GPRS, además al igual que en el caso anterior, es un módulo sin GPS integrado.Su funcionamiento es a través de comandos AT al igual que los casos anteriores, y como en el caso del módulo SIM900. En [37], se muestran los comandos asociados para configurarlo y además en el Anexo 14.2.6 del mismo documento se puede observar los comandos necesarios para configurar una conexión TCP/IP. El precio de éste módulo en el extranjero es de aproximadamente 7.8 Dólares (5.600 Pesos Chilenos), y dentro de Chile su precio es de 15.000 pesos. También existe una versión minimalista de 4.3 Dólares (3.000 Pesos Chilenos) del mismo módulo, mostrada en la Figura 4.3,

que dentro del país se puede conseguir por 11.000 pesos.

Figura 4.3: Módulo GSM Ai Thinker A6.

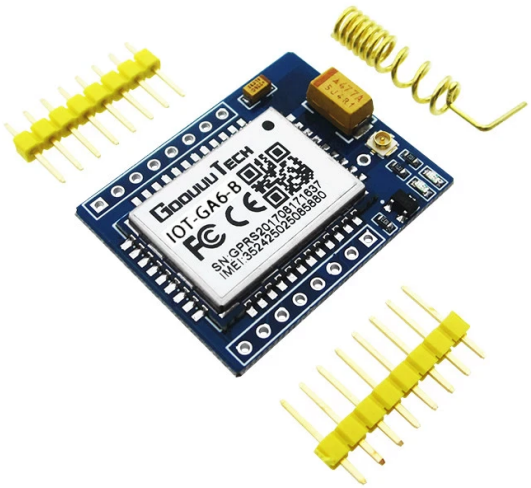


Figura 4.4: Módulo GSM Ai Thinker A6 mini.

Módulo SIM800L

En el módulo mostrado en la Figura 4.5, el componente principal pertenece a la compañía SIMCom. En [38] se muestra que las tecnologías compatibles de éste módulo son Quad-band GSM/GPRS al igual que en el módulo SIM900, además al igual que en el caso anterior este

componente no posee GPS integrado lo que lo hace barato. Además como en los casos anteriores, éste módulo funciona mediante comandos AT utilizando puerto UART al igual que los módulos anteriores, lo que hace que sea fácil de configurar. Este módulo tiene las mismas funcionalidades que su compañero SIM900, haciéndolo una mejor opción. En [39] se muestran los comandos asociados para configurarlo. Además, como se muestra en [40] el módulo puede ser configurado fácilmente para establecer una conexión TCP/IP.

Anteriormente se ha trabajado con éste módulo, demás su precio en el extranjero es de aproximadamente 2.5 Dólares (2.000 pesos chilenos) y dentro del país se puede conseguir por 9.000 pesos.

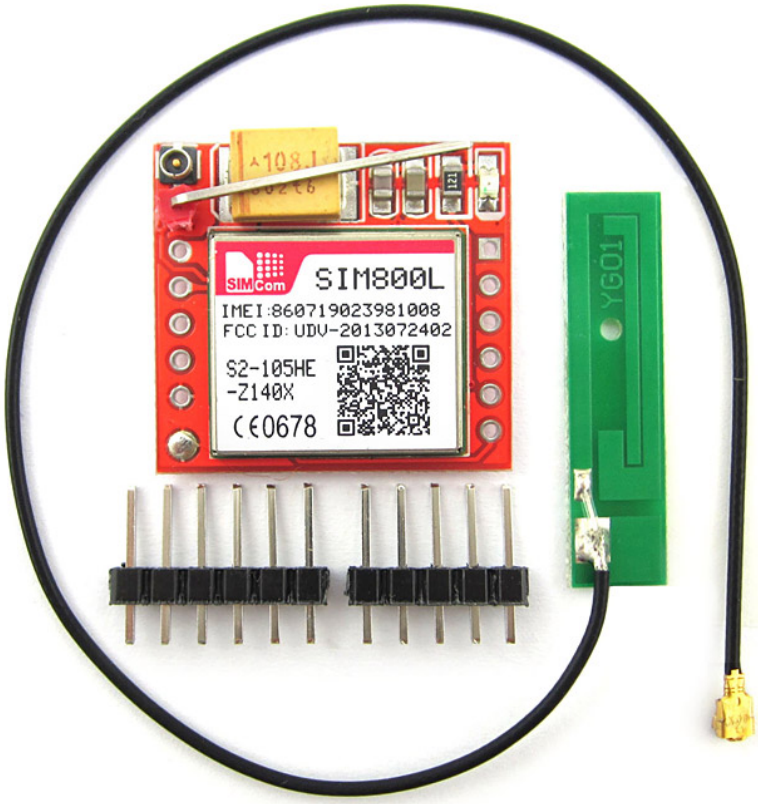


Figura 4.5: Módulo SIMCom SIM800L

Módulo GSM/GPRS Seleccionado

A continuación se presenta en la Tabla 4.1, una comparación entre los módulos presentados anteriormente, destacando los aspectos que más nos interesan para escoger. Entre ellos se da

énfasis en el precio, el protocolo de comunicación utilizado y la experiencia previa de trabajo.

Módulo	SIM5320	SIM900	Ai Thinker A6	SIM800
Tecnologías Soportadas	HSDPA/WCD-MA, GS-M/GPRS/EDGE	GPRS/GSM	GPRS/GSM	GPRS/GSM
Experiencia propia	Alta	Bajo	Nula	Alto
Precio Extranjero[USD]	43	11	4.3	2.5
Precio Nacional [CLP]	85.000	20.000	15.000	9.000

Tabla 4.1: Comparación entre las características de los módulos GSM/GPRS.

De acuerdo con esta comparativa la opción más viable para desarrollar el módulo de pruebas es el módulo **SIM800L** ya que tiene un costo bajo y se posee experiencia previa desarrollando aplicaciones con él.

4.1.2. Módulo GPS

Para la obtención de ubicaciones existen distintas alternativas en el mercado, al igual que en la Sección 4.1.1 se centrará la elección principalmente en el precio del módulo y la experiencia previa de trabajo.

Módulo Se868-a

Este módulo tiene por componente principal in circuito de la compañía Telit, especializada principalmente en el desarrollo de componentes IoT. En [41] se muestra que las Bandas de Frecuencia soportadas por éste módulo corresponden a GPS-L1 y QZSS-L1, haciéndolo compatible con la localización global, pero con una pequeña mejora de rendimiento y precisión para Japón. Este módulo funciona a través de UART, con una alimentación de 5V, entregando su información en el estándar NMEA [42]. El módulo se muestra en la Figura 4.6, alcanzando un precio dentro del país de aproximadamente 15.000 Pesos, considerándolo uno de los más baratos que existen en Chile para desarrollos pequeños.

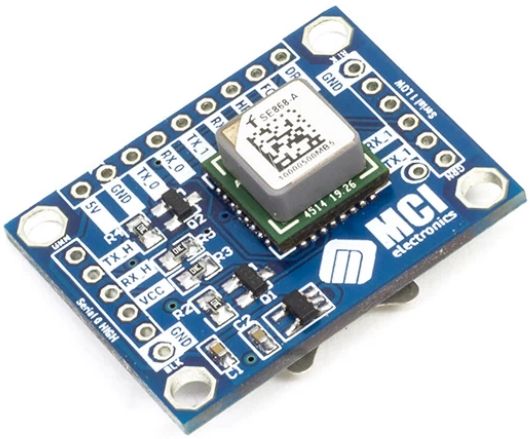


Figura 4.6: Módulo GPS SE868-A

Módulo Beitian BN-180

Este módulo esta basado en el circuito UBX-M8030 de la compañía Ublox, en la Figura 4.7 se muestra un ejemplar. En [43] se muestran las características del circuito, donde se muestra que es compatible con las tecnologías GPS, GLONASS, BeiDou y Galileo, en las bandas de frecuencia GPS-L1, QZSS-L1, GLONASS-L10F, BeiDou-B1 yGalileo-E1. Al igual que en el caso anterior, su comunicación es mediante un puerto UART entregando la información en el estándar NMEA. El precio del módulo en el país es de aproximadamente 40.000 Pesos, lo que se considera un precio bastante alto para un módulo GPS. Se entiende que es debido a la cantidad de tecnologías compatibles pero por su precio es fácilmente descartable.

Figura 4.7: Módulo GPS Beitian BN-180

Módulo Ublox Neo-6m

Este módulo está basado en un circuito de la familia NEO-6 de la compañía Ublox, como el mostrado en la Figura 4.8. En [44] se muestran las características de éste módulo, donde se indica que la única tecnología compatible con este componente es GPS en la banda L1 (GPS-L1). Además, al igual que en los casos anteriores, éste módulo entrega su información mediante un puerto UART utilizando el estándar NMEA. El precio de éste módulo en Chile es de unos 8.000 Pesos, lo que lo hace la opción más barata con la tecnología necesaria para el propósito de éste proyecto.

Figura 4.8: Módulo GPS NEO 6M

4.1.3. Módulo GPS Seleccionado

A continuación se presenta la Tabla 4.2, con los datos correspondientes a los módulos GPS presentados anteriormente para comparar sus características y decidir la mejor opción.

Módulo	Telit SE868	Beitian BN-180	uBlox Neo-6M
Tecnologías Soportadas	GPS/QZSS	GPS/QZSS, GLONASS, BeiDou, Galileo	GPS
Experiencia propia	Baja	Nula	Alta
Precio Nacional [CLP]	15.000	40.000	8.000

Tabla 4.2: Comparación entre las características de los módulos GPS.

De acuerdo a esta tabla comparativa, y de acuerdo a las características de elección que se establecieron, la alternativa más viable es el módulo **NEO-6M**, ya que es la opción con el precio más bajo y con la que se tiene mayor experiencia de trabajo.

4.1.4. Módulo de comunicación LoRa

Para el módulo de comunicación LoRa, se centrará la búsqueda en un dispositivo que funcione en las bandas de frecuencia permitidas en Chile para radiocomunicación de aficionados, según [24] estas frecuencias corresponden a las bandas entre los 430MHz- 438MHz y los 890MHz-928MHz. A continuación se evaluarán las alternativas disponibles en Chile y el extranjero.

MCI LoRaBee

Según la página oficial de MCI [45], éste módulo, mostrado en la Figura 4.9,está construido con el circuito integrado RN2903A de la compañía Microchip Technology Inc, el cual funciona a una frecuencia base de 915MHz. Además el módulo incorpora el protocolo LoRaWAN de Clase A, utilizando un voltaje de operación de 3.3V. Por otra parte según [46] el circuito RN2903A se comunica mediante un puerto UART con una interfaz de comandos ASCII. El precio de éste módulo en Chile es de aproximadamente 24.000 Pesos, además no viene con su antena por lo tanto se debe gastar un monto extra para comprar una antena que sea compatible.

Figura 4.9: Módulo LoRa MCI LoRaBee

Módulo Transmisor Receptor RF LoRa E32-433T30D

Este módulo , mostrado en la Figura 4.10, está basado en el circuito integrado SX1278 de la familia SX127X de Semtech Corporation, que según [47], los integrados SX1276/77 pueden transmitir en la banda de frecuencias 137MHz-1020MHz, el integrado SX1278 puede transmitir en la banda de frecuencias 137MHz-525MHz y el integrado SX1279 puede transmitir en la banda de frecuencias 137MHz-960MHz. Si bien en Chile se venden varias alternativas utilizando esta familia de circuitos integrados, las opciones se centran principalmente en la banda 433MHz. La comunicación de este módulo es mediante UART, con una alimentación de 3.3V a 5V. El precio de este módulo en Chile es de aproximadamente 15.000 Pesos, y en el extranjero ronda en los 7 Dólares (5.000 Pesos Chilenos).

Figura 4.10: Módulo LoRa E32-433T30D

Ai-Thinker Lora SX1278

Este módulo, mostrado en la Figura 4.11, fue diseñado por la empresa Ai-Thinker, también es una alternativa con el integrado SX1278, es más barata que el módulo anterior (basado en el mismo circuito integrado), ya que corresponde a una versión reducida del mismo. La comunicación de ésta versión del módulo es mediante un puerto Serial Peripheral Interface (SPI). El precio de éste módulo en Chile es de 9.000 Pesos, pero es una opción sin antena, por lo tanto se debe invertir un poco más en la antena. En el extranjero esta versión del módulo tiene un precio de 3.6 Dólares (2.800 Pesos Chilenos).

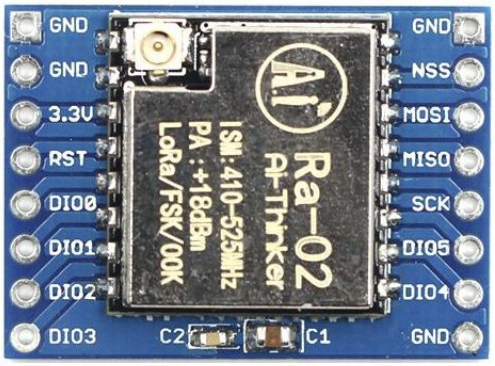


Figura 4.11: Módulo Ai-Thinker LoRa SX1278

Módulo Minimalista SX1278

Este módulo, mostrado en la Figura 4.12, al igual que la opción anterior, pertenece a la familia de los circuito Semtech SX1278, el cual se comunica mediante un puerto SPI, desde el que puede ser configurado para su uso. El precio de éste módulo en Chile es de 9.000 Pesos, pero en este caso la compra viene con su antena lo que nos ahorra un poco de dinero en esta compra. En el extranjero este módulo tiene un precio de 2.9 Dólares (2.200 Pesos Chilenos).

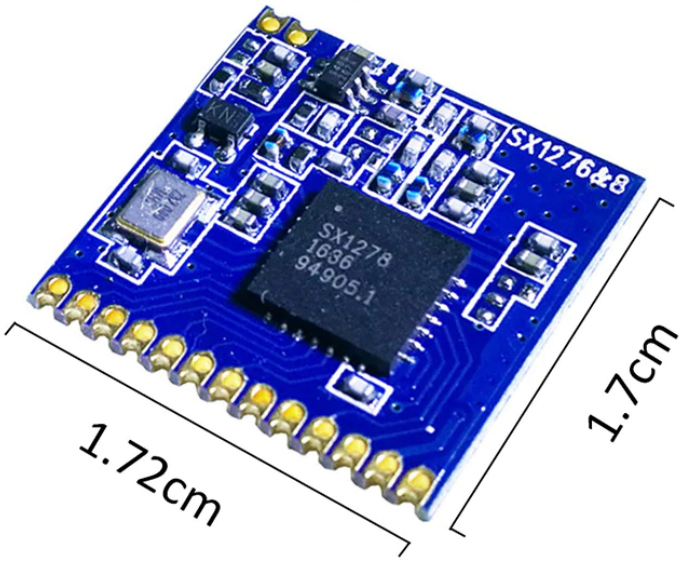


Figura 4.12: Módulo LoRa SX1278 Minimalista

4.1.5. Módulo LoRa seleccionado

A continuación se presenta la Tabla 4.3, con los datos correspondientes a los módulos LoRa presentados anteriormente para comparar sus características y decidir la mejor opción.

Módulo	MCI LoRaBee	LoRa E32-433T30D	Ai-Thinker Lora SX1278	Minimalista SX1278
Banda de Frecuencia [MHz]	915	433	433	433
Experiencia propia	Nula	Nula	Nula	Nula
Precio Nacional [CLP]	24.000	15.000	9.000	9.000
Precio Extranjero [USD]	-	7	3.6	2.9

Tabla 4.3: Comparación entre las características de los módulos LoRa.

De acuerdo a esta tabla comparativa, y de acuerdo a las características de elección que se establecieron, la alternativa más viable es el módulo **Minimalista SX1278**, ya que es la opción con el precio más bajo, y que nos ahorra el costo de la antena. Debido a que no se tiene experiencia previa de trabajo con ninguna de las opciones, se debe descartar esa variable.

4.1.6. Microcontrolador

Para controlar el resto de los módulos seleccionados para el proyecto, se necesita contar con "cerebro", esta función la cumplirá el microcontrolador seleccionado. Debido a esto, tenemos que escoger un componente que sea compatible con el resto de los módulos y que tenga puertos habilitados para todos los módulos. A diferencia del resto de los componentes, la elección del microcontrolador también se evaluará según su capacidad para tener un reloj interno del orden de los microsegundos, ya que esta funcionalidad nos permitirá llevar un control más preciso de los tiempos de llegada de los mensajes que llegan desde los otros dispositivos. Por lo tanto, los requerimientos mínimos para escoger un componente serán, un buen precio, experiencia previa, tener al menos 3 puertos UART disponibles, tener al menos un puerto SPI disponible y finalmente permitir interrupciones del orden de los microsegundos.

Atmel Atmega328P

Este microcontrolador ha sido uno de los más usados en el último tiempo, ya que es el componente principal en la familia de placas de desarrollo ARDUINO UNO, mostrada en la Figura 4.13, que se ha hecho bastante conocida en los últimos años ya que permite integrar la electrónica de una forma fácil y rápida. El microcontrolador creado por Atmel, marca adquirida por Microchip en 2016, según su hoja de datos [48] el dispositivo posee tan solo un puerto Universal Synchronous/Asynchronous Receiver Trasmitter (UART), un puerto SPI y dos puertos Inter-Integrated Circuit (I2C). Si bien posee temporizadores con la capacidad de generar interrupciones en el orden de los microsegundos, no posee la cantidad necesaria de periféricos para conectar todos los módulos anteriormente seleccionados, por lo tanto se puede descartar esta opción.

Figura 4.13: Arduino UNO, microcontrolador Atmega328p

Atmel Atmega2560

Este microcontrolador, junto con el mencionado en la Sección 4.1.6, ha sido utilizado para las placas de desarrollo ARDUINO, por lo tanto tiene una gran cantidad de documentación y librerías para todo tipo de sensores y módulos. En la Figura 4.14 se muestra la placa de desarrollo ARDUINO MEGA, que contiene el microcontrolador Atmel Atmega2560. Según la documentación oficial de Microchip [49], este microcontrolador posee 4 puertos UART, 2 puertos SPI y un puerto I2C disponibles para su uso. Además posee un reloj interno capaz de realizar interrupciones en el orden de los microsegundos, por lo tanto lo hace un buen candidato para el propósito de éste proyecto. Su precio nacional es de 14.000 Pesos, y su precio en el extranjero ronda los 10 Dólares (7.000 Pesos Chilenos).

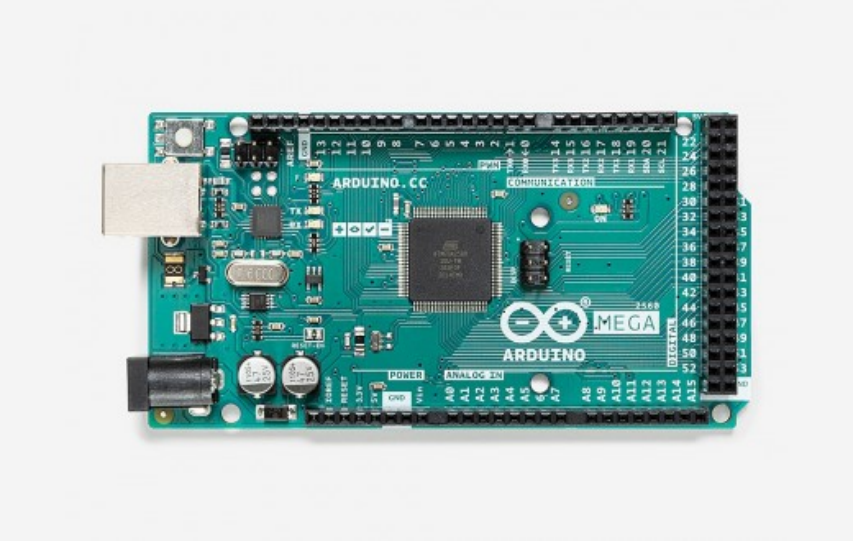


Figura 4.14: Arduino Mega, microcontrolador Atmega2560

STM32F407VET6

Este microcontrolador desarrollado por la empresa STMicroelectronics, es una de las tantas opciones que ofrece esta compañía, ya que tiene una gran gama de variedades de microcontroladores orientados a distintos rubros y con distintas funcionalidades. Se listó esta opción ya que se hizo conocida por la placa de desarrollo STM32F407VE Black board, mostrada en la Figura 4.16, de bajo precio que logró sacar todas sus funcionalidades a la luz. Según la documentación oficial de STMicroelectronics [50], este microcontrolador posee disponibles hasta 3 puertos I2C, hasta 3 puertos UART y 2 puertos UART y hasta 3 puertos SPI. Además posee un reloj interno capaz de realizar interrupciones en el orden de los microsegundos. Si bien posee más puertos disponibles que el microcontrolador anterior, el precio de la placa de desarrollo en Chile tiene el mismo precio de 14.000 Pesos, y en el extranjero su precio es de 15 Dólares (11.000 Pesos Chilenos), por lo que su adquisición puede llegar a ser más cara que la opción anterior si se quiere adquirir en el extranjero.

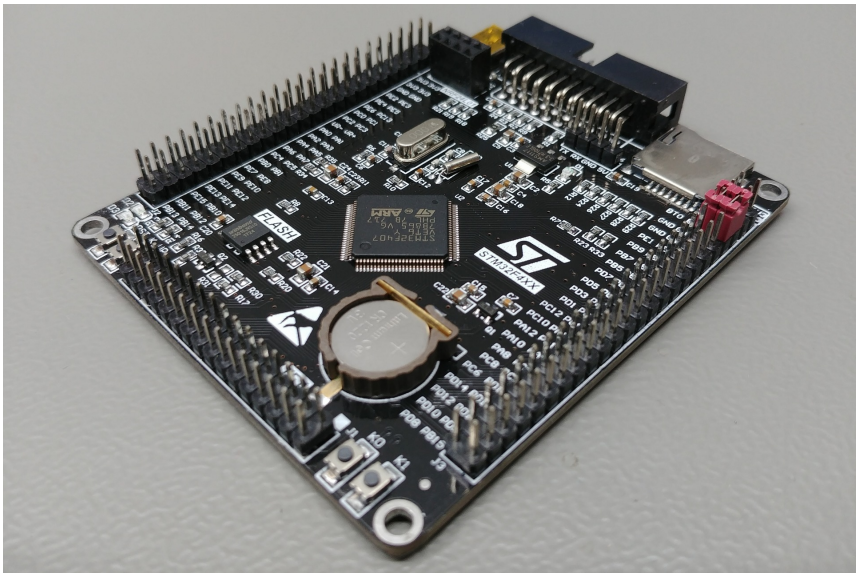


Figura 4.15: STM32F407 Black Board, microcontrolador STM32F407VET6

STM32F411CEU6

Este microcontrolador también desarrollado por STMicroelectronics, es de la misma gama que el presentador en la Sección 4.1.6. Si bien es de la misma gama, este modelo en específico se hizo conocido ya que al igual que en el caso anterior, se desarrolló una versión de bajo costo que permite aprovechar las funcionalidades del microcontrolador, a un bajo precio. La placa STM32F411 Black Pill, con el microcontrolador STM32F411CE es una placa de tan solo 5.2 x 2.0 cm que deja todos los puertos disponibles para su uso, entregando según la documentación oficial [51] 3 puertos UART, 2 puertos I2C y hasta 3 puertos SPI disponibles. Además al igual que en los casos anteriores, posee un reloj interno que permite realizar interrupciones del orden de los microsegundos. Por último, el precio de esta placa de desarrollo en Chile es de 10.000 pesos, y en el extranjero es de 6 Dólares (4.500 Pesos Chilenos).

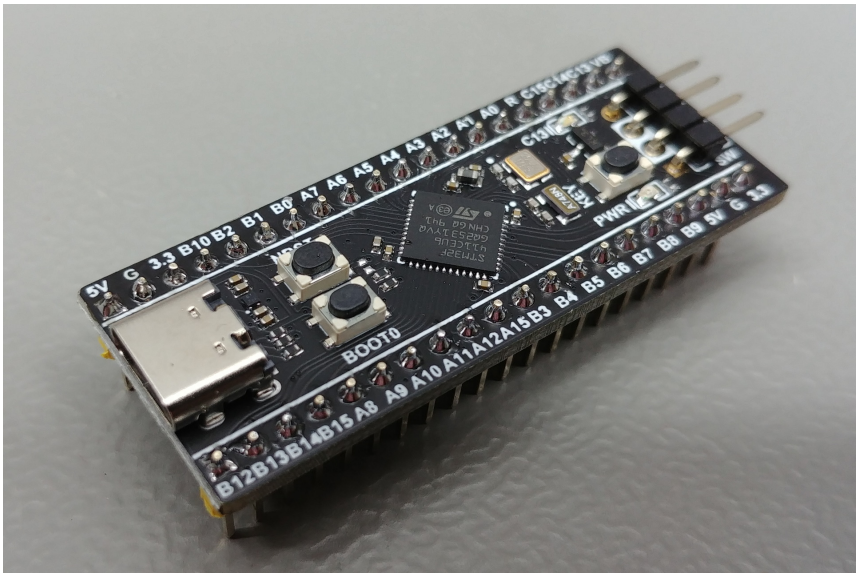


Figura 4.16: STM32F411 Black Pill, microcontrolador STM32F411CEU6

4.2. Microcontrolador Seleccionado

A continuación, en la Tabla 4.4, se presenta la comparación entre las alternativas listadas en las secciones anteriores, mostrando las cualidades que se desean evaluar, sin olvidar que los requisitos mínimos de funcionamiento corresponde a que posea al menos 2 puertos UART, 1 puerto SPI y un reloj con la capacidad de medir tiempos de al menos el orden de los microsegundos. Además se presenta el nivel de experiencia con cada opción para realizar una elección que permita garantizar la optimización de tiempos de desarrollo.

Característica	Atmega328p	Atmega2560	STM32F407VET6	STM32F411CEU6
Puertos UART/USART	1	4	5	3
Puertos I2C	2	1	3	2
Puertos SPI	1	2	3	3
Capacidad de Reloj	μs	μs	μs	μs
Precio Nacional [CLP]	-	14.000	14.000	10.000
Precio Internacional [USD]	-	10	15	6
Experiencia Previa	Alta	Media	Media	Alta

Tabla 4.4: Comparación entre las características de los módulos LoRa.

Con los datos presentados en la Tabla 4.4 se puede deducir que la mejor opción en cuanto a precio, experiencia y los requerimientos mínimos es el microcontrolador **STM32F411CEU6**, ya que cumple con las expectativas y se posee una experiencia previa programando esta familia de microcontroladores.

4.3. Conexión entre componentes

Esta sección está dedicada especialmente a la evaluación y elección de pines a usar para interconectar los componentes escogidos en la Sección **??**, permitiendo la correcta comunicación con el microcontrolador.

4.3.1. Detalle de componentes

A continuación se mostrarán los detalles técnicos de cada módulo, mostrando la configuración de pines de cada uno para visualizar la forma de interconectar los componentes.

Detalles del módulo GPRS/GSM SIM800L

Tal como se mencionó en la Sección 4.1.1, éste módulo se comunica a través de un puerto UART utilizando comandos AT. Según su documentación oficial [39] su configuración por defecto es aun Baud Rate de 115200 bauds/s, por lo tanto el puerto dedicado para la conexión de este componente debe funcionar a 115200 bauds/s. En la Figura **??**, se muestra la configuración de pines del módulo, para la comunicación con el microcontrolador, los pines RXD y TXD son necesarios. Además, ya que éste módulo no tiene un regulador interno de voltaje, se debe conectar su alimentación en el pin Vcc a una batería de 3.7V, la que también cumplirá la función de entregar la corriente necesaria para la transmisión de datos.

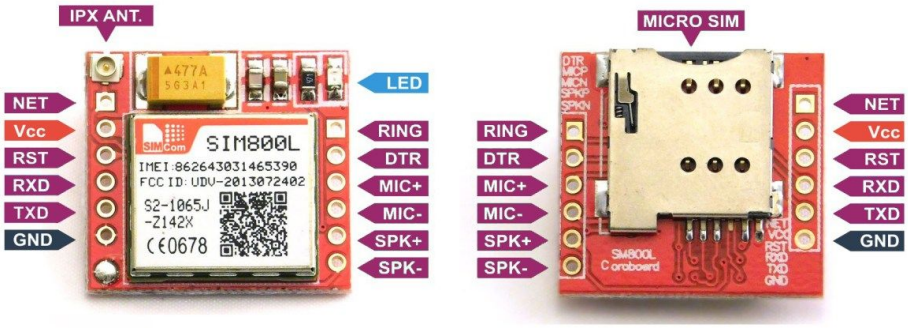


Figura 4.17: Configuración de pines del modulo SIM800L

Detalles del módulo GPS Ublox NEO-6M

El módulo NEO-6M, como se explicó en la Sección 4.1.2 funciona a través del puerto UART, entregando la información de los satélites en el estándar NMEA. En su documentación oficial [44] se indica además la configuración de pines necesaria para cambiar la velocidad de transmisión del módulo. Tal como se muestra en las Figuras 4.18 y 4.19, la configuración de los pines CFG_COM0 y CFG_COM1 es la que controla la velocidad de transmisión del módulo, además, como se muestra en la Figura 4.20, estos pines están desconectados, por lo tanto la velocidad por defecto es de 9600 Baud/s. Ya que esta velocidad de transmisión es una de las más bajas, se realiza una modificación al hardware de los módulos GPS, llevando el pin CFG_COM1 a GND, para aumentar la velocidad de transmisión a 34800 Baud/s. Esta conexión se muestra en la Figura 4.21.

13	All	GND	I	Ground
14	All	MOSI/CFG_COM0	O/I	SPI MOSI / Configuration Pin. Leave open if not used.
15	All	MISO/CFG_COM1	I	SPI MISO / Configuration Pin. Leave open if not used.

Figura 4.18: Explicación de pines, UBLOX NEO-6M

CFG_COM1	CFG_COM0	Protocol	Messages	UARTBaud rate	USB power
1	**1**	**NMEA**	**GSV, RMC, GSA, GGA, GLL, VTG, TXT**	**9600**	**BUS Powered**
1	0	NMEA	GSV, RMC, GSA, GGA, GLL, VTG, TXT	38400	Self Powered
0	1	NMEA	GSV[14], RMC, GSA, GGA, VTG, TXT	4800	BUS Powered
0	0	UBX	NAV-SOL, NAV-STATUS, NAV-SVINFO, NAV-CLOCK, INF, MON-EXCEPT, AID-ALPSERV	57600	BUS Powered

Figura 4.19: Configuración de pines para Baud Rate por defecto.

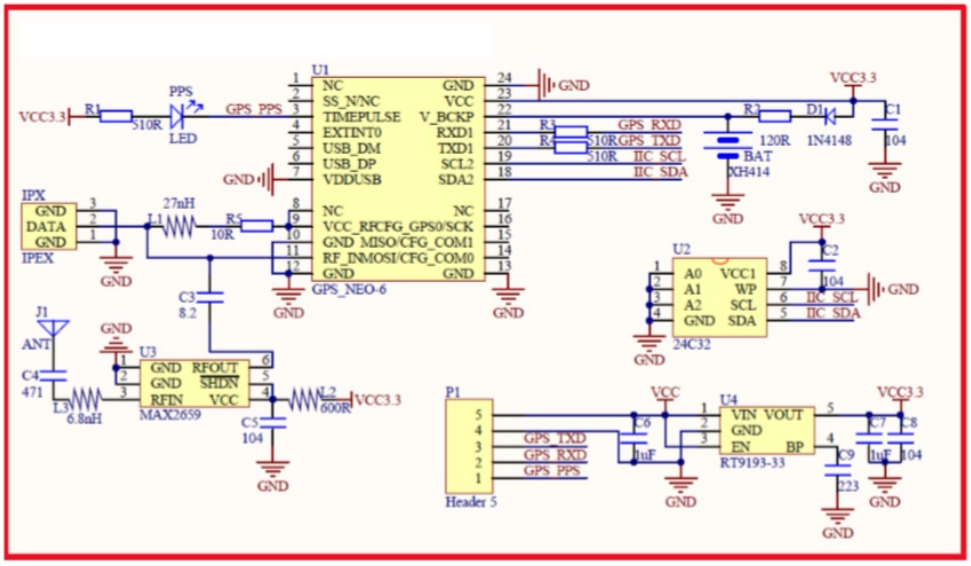


Figura 4.20: Diagrama esquemático interno del módulo NEO-6M

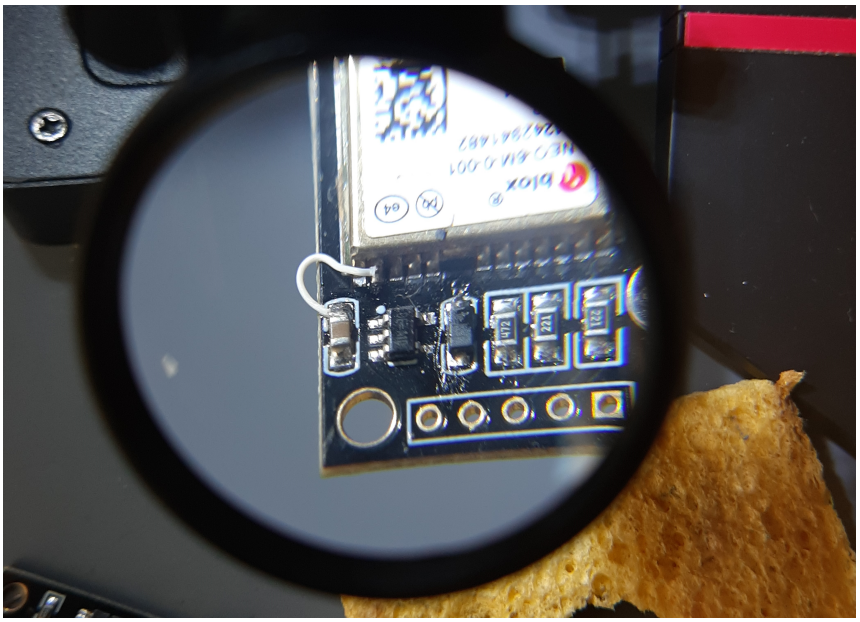


Figura 4.21: Conexión del pin CFG_COM1 a GND para Baud Rate de 38400 por defecto.

Finalmente, en la Figura 4.22 se muestra la configuración de pines del Módulo NEO-6M, donde se destacan claramente los pines RX y TX correspondientes al puerto UART, encargados de la comunicación con el microcontrolador.

Figura 4.22: Detalle de pines del módulo NEO-6M

Detalles del módulo LoRa SX1278

El módulo LoRa SX1278, como se mencionó en la Sección 4.1.4, se comunica a través de una interfaz SPI con el microcontrolador, esto permite transacciones rápidas tanto para el envío como para la recepción de datos. Mediante este puerto el módulo puede recibir las configuraciones de banda de frecuencia, SF,Code Rate y palabra de sincronización. En la Figura 4.23 se muestra la configuración de pines del módulo, donde se indica claramente la ubicación del puerto SPI y varios pines Digital Input/Output (DIO) configurables con ciertas funcionalidades. Además, en la Figura 4.24 se muestra la descripción de cada uno de éstos pines.

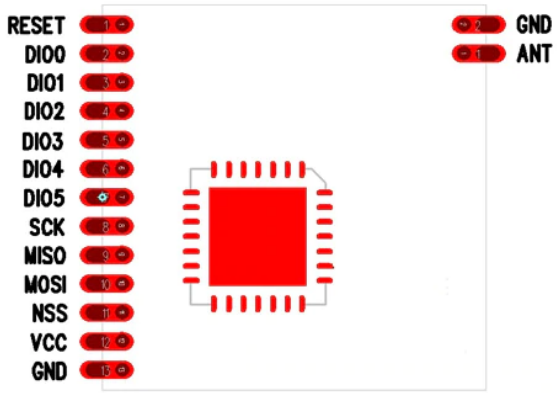


Figura 4.23: Configuración de pines del módulo SX1276

NO.	Pin Name	Type and Description	Remark
1	RESET	Chip reset	Chip reset input
2	DI00	Digital I/O port, can be customized	For transmission indicator, similar to NIRQ feet
3	DI01/DCLK	Digital I/O port, can be customized, optional DCLK	For test sensitivity, floating
4	DI02/DATA	Digital I/O port, can be customized, optional DATA	For test sensitivity, floating
5	DI03	Digital I/O port, can be customized	Can be selected
6	DI04	Digital I/O port, can be customized	Can be selected
7	DI05	Digital I/O port, can be customized	Can be selected
8	SCK	SPI clock	SPI clock input
9	MISO	SPI output	SPI data output
10	MOSI	SPI input	SPI data input
11	NSS	SPI chip selection	Chip select input
12	VCC	Working power supply	3.3V
13	GND	GND	GND
14	GND	GND	GND
15	ANT	Antenna	Antenna

Figura 4.24: Descripción detallada de pines del módulo SX1276

Detalles de placa STM32F411 Black Pill

Esta placa de desarrollo, como se explicó en la Sección 4.1.6, tiene las capacidades necesarias para comunicarse con el resto de los módulos, además, según lo indicado en la hoja de datos[51], la versión que usa éste módulo es la de 48 pines. Además, en el esquemático oficial de la placa [52] se muestran las conexiones a los pines del microcontrolador. Con esta información se pueden destacar los pines que se pueden usar para las conexiones de los puertos UART y SPI. Estos pines se destacan en la Figura 4.25

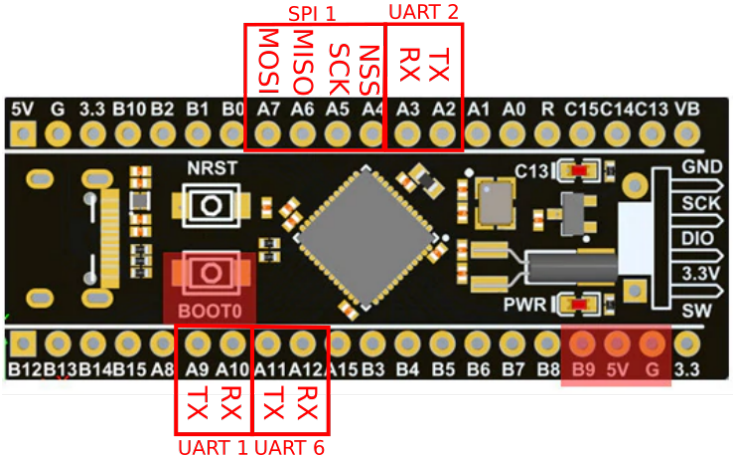


Figura 4.25: Detalle de pines de la placa STM32F411 Black Pill

4.3.2. Diagrama de conexión de módulos con el microcontrolador

A continuación, en la Figura 4.26 se muestra un diagrama de conexiones con la conexión de los puertos de cada módulo con el microcontrolador.

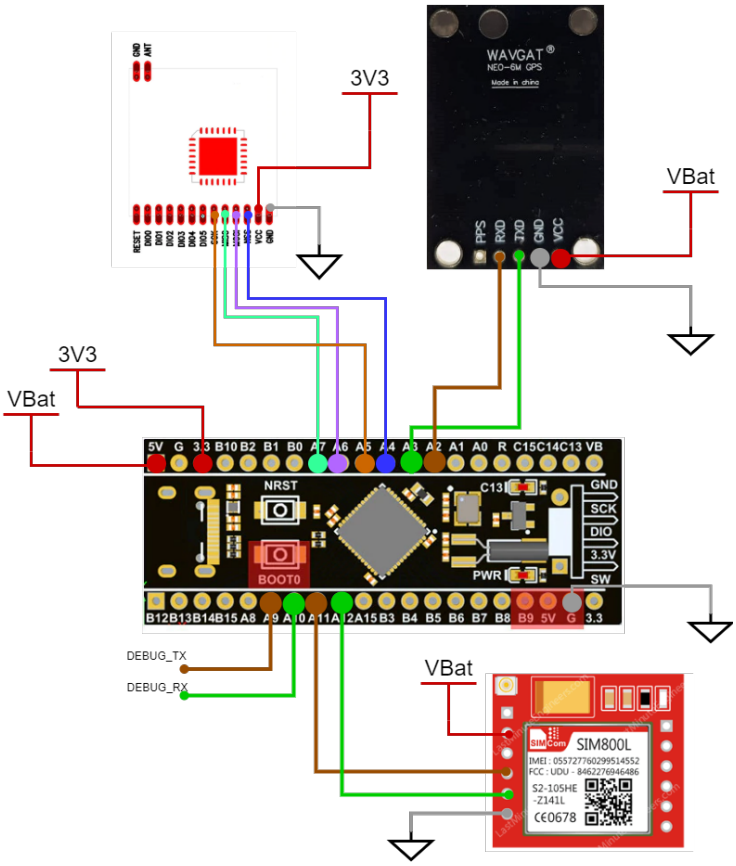


Figura 4.26: Diagrama de conexiones de los módulos y el microcontrolador.

Capítulo 5

Diseño de Firmware

En este capitulo nos centraremos en la elección de las herramientas necesarias para la programación del Firmware en el microcontrolador, encargado de utilizar los módulos seleccionados. Además, buscaremos librerías que permitan agilizar la programación, disminuyendo la cantidad de código propio que se debe usar para finalmente unificar el funcionamiento de cada módulo y programar el funcionamiento básico de transmisión de datos. Por último se establecerá el el formato de envío de los datos del dispositivo de pruebas hacia un servidor de pruebas.

5.1. Selección de herramientas

Para la programación del microcontrolador se utilizará el Framework PlatformIO [53]. Este Framework contiene una gran cantidad de herramientas para la programación de microcontroladores en distintas arquitecturas. Demás es compatible con

5.1.1. Librerías para módulo SIM800L

Las alternativas para la programación de éste módulo son limitadas, buscando el envío de datos mediante TCP a un servidor. Debido a lo anterior se exploran tan solo dos opciones ya desarrolladas, considerando como tercera opción el desarrollo de una librería de forma manual que cumpla con los requisitos de transmisión de datos a un servidor mediante TCP.

SIM800L: Cristian Steib

En la Tabla 5.1, se presentan las funciones que incluye esta librería con su descripción. Si bien esta librería contiene todo lo necesario para hacer funcionar el módulo [54], enviar mensajes de texto y hacer llamadas, el objetivo principal que se desea con ella no está presente. Además, esta librería está hecha para funcionar con una placa de desarrollo Arduino UNO como la mostrada en la Sección 4.1.6, por lo tanto para utilizarla en este proyecto, se deben agregar funcionalidades y adaptarla al microcontrolador seleccionado.

SIM800L: Métodos y Funciones		
Función	Retorna	Descripción
begin()	None	Inicializa la librería
reset()	None	Reinicia el módule y espera al mensaje SMS Ready
sendSms(number,text)	true or false	Envía un mensaje de texto al número indicado con el texto en formato String
readSms(index)	String	Lee un SMS indexado en la memoria interna del módulo.
getNumberSms(index)	String	Entrega el número de un SMS en formato String.
delAllSms()	true or false	Borra todos los SMS
answerCall()	true or false	Responde una llamada entrante.
callNumber(number)	None	Realiza una llamadad de voz al número indicado.
hangoffCall()	true or false	Cuelga una llamada en curso.
getCallStatus()	uint8_t	Devuelve el estado de una llamada 0 : Lista , 2: Desconocido, 3: Sonando , 4: Llamada en curso.
setPhoneFunctionality()	None	Inicial la funcionalidad total del módulo.
activateBearerProfile()	None	Activa el perfil de internet
deactivateBearerProfile()	None	Desactiva el perfil de internet
RTCtime(int *day,int *month, int *year,int *hour,int *minute, int *second)	None	Actualiza el valor de las variables entregadas como parámetros mostrando la hora guardada por el módulo.
dateNet()	String	Entrega la hora desde la red GSM
updateRtc(utc)	true or false	Actualiza la hora del reloj interno del módulo.

Tabla 5.1: Métodos y funciones de SIM800L de Cristian Steib

SIM800L: Vittorio Esposito

Esta librería corresponde a una extensión de la mostrada en la Sección 5.1.1, si bien contiene más funciones que en el caso anterior, el funcionamiento principal que se desea no está presente. Además, como se mencionó anteriormente , corresponde a una versión adaptada para la placa de desarrollo Arduino UNO por lo tanto al igual que en el caso anterior, se deberían agregar funcionalidades y adaptarla al microcontrolador seleccionado.

Librería personalizada

Considerando las opciones presentadas anteriormente, se decide desarrollar una versión simplificada que cumpla el rol específico necesitado en éste proyecto. Esto nos permitirá ignorar gran cantidad de las funcionalidades que están incluidas en los casos anteriores y centrarnos principalmente en establecer una conexión a internet mediante GSM y luego comunicarse con un servidor mediante TCP. Para simplificar el trabajo, se decide generar una máquina de estados que permita guiar al módulo, insertando los comandos AT necesarios para establecer una conexión GSM, guiándonos por la máquina de estados interna del módulo, presentada en la Figura 5.1, y utilizando la información entregada por el documento oficial [39]. Luego de establecer una conexión a internet, se debe enviar información a un servidor utilizando una conexión TCP como se muestra en [40]. Si bien para hacer que esta máquina de estados funcione correctamente se hicieron varias pruebas, el resultado final se muestra en la Figura 5.2.

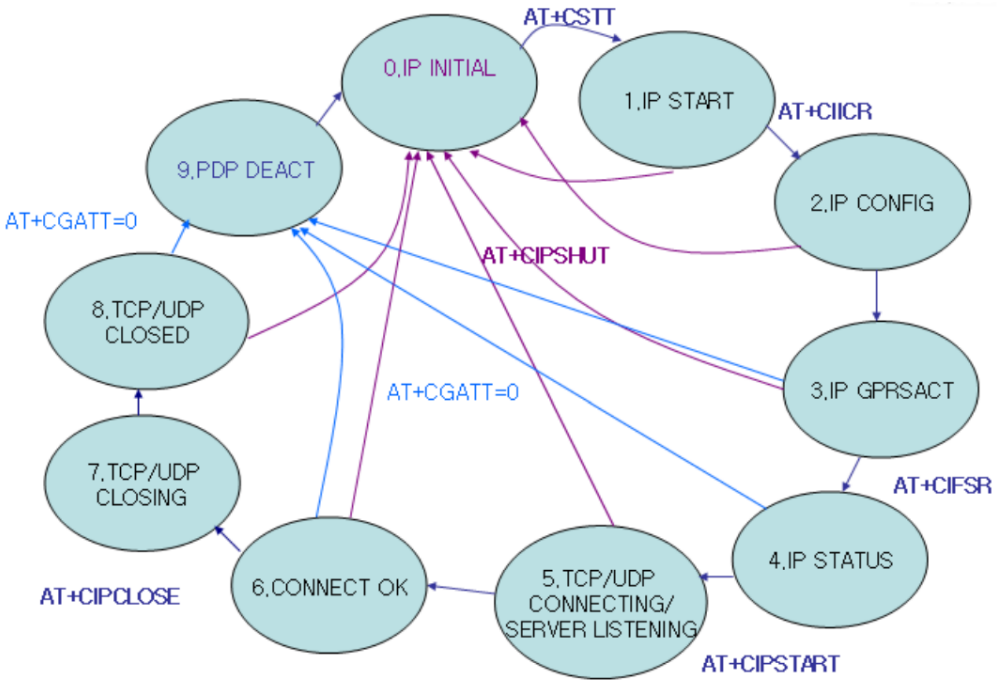


Figura 5.1: Máquina de estados SIM800L para conexión individual.

En la Tabla 5.2, se muestra una descripción de cada estado de la máquina desarrollada, describiendo lo que se busca con cada uno de ellos. Por otra parte, en la Figura 5.2 se marca con colores 3 estados en específico, los que corresponden al estado Inicial (State 0: Verde) , estado intermedio de espera (State 9: Amarillo) y un estado final (State 21: Rojo), en el que se cierra la conexión a internet. Dentro de cada estado se describe el comando AT utilizando dentro del módulo. El Contador de errores (Error Counter) es un contador que se incrementa cada vez que al enviar un comando al módulo se recibe una respuesta inesperada, por lo tanto se dan oportunidades de que el módulo responda, y si la respuesta no es la esperada 5 veces seguidas reinicia el proceso.

Estado	Descripción
0	Se usa para saber si el módulo tiene comunicación con el MCU.
1	Se activan todas las funciones de RF del módulo.
2	Se desactiva el estado de .ECHO"del módulo, por lo que no responderá con los comandos AT que se le envían, sino que solo mostrará la respuesta al comando.
3	Se activan los códigos de error extensos, con lo que se puede tomar decisiones dependiendo del error que entregue el módulo.
4	Se solicita el IMEI del módulo, ya que será usado para identificar desde que dispositivo vienen los datos.
5	Se conecta el dispositivo a la red
6	Se define el contexto del protocolo de paquetes de datos (PDP context), como Protocolo IP usando y con un Nombre de punto de acceso (APN).
7	Se cierra el protocolo de paquete de datos por defecto hasta realizar una transmisión.
8	Se activa la característica de que: Cuando se envían datos se despliega el caracter "$>$" y cuando se terminó el envío de forma correcta aparece el mensaje "SEND OK".
9	Se espera a que el arreglo de datos contenga datos.
10	Se activa el protocolo de paquetes de datos definido en el estado 6 (PDP context), y se escoge como protocolo por defecto.
11	Se obtiene la dirección IP obtenida desde la red GPRS.
12	Se define que el módulo no trabajará con multiples conexiones TCP.
13	Se define la conexión a internet mediante TCP/IP utilizando un APN, nombre de usuario y contraseña.
14	Se activa la conexión a internet mediante TCP/IP con los datos definidos anteriormente.
15	Se verifica la IP Local obtenida mediante TCP/IP.
16	Se verifica el estado de conexión del módulo, desde la máquina de estados interna (Mostrada en la Figura 5.1, para proseguir con el envío de datos.
17	Se abre la conexión TCP/IP con el servidor definido en "IPADDR" en el puerto definido como"PORT".
18	Se envían los datos al servidor.
19	Se espera por la confirmación de la recepción o se reenvían los datos si se cumple un tiempo límite.
20	Se cierra la conexión TCP y se vuelve a esperar por más datos. O se cierra la conexión TCP y se procede a apagar el módulo.
21	Se cierran todas las conexiones y se apaga la antena del módulo.

Tabla 5.2: Descripción de los estados definidos en la máquina de estados personalizada.

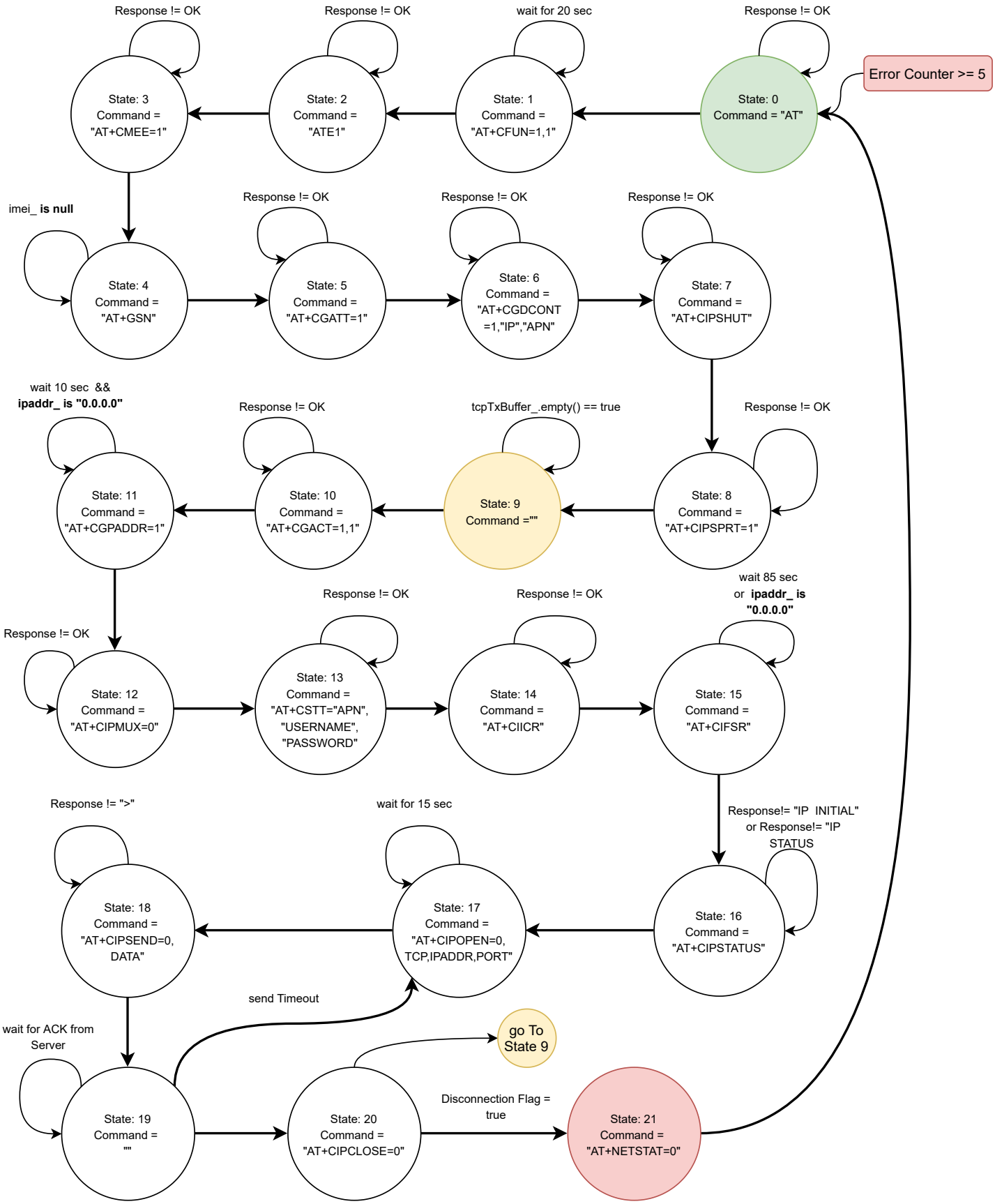


Figura 5.2: Maquina de estados de librería personalizada.

5.1.2. Librerías para módulo NEO-6M

En el caso del módulo NEO-6M, existe una librería que lleva bastantes años siendo desarrollado y utilizada, ya que es compacta y se puede adaptar a varias plataformas.

TinyGPS++: Mikañ Hart

Esta librería [55] contiene un procesamiento interno de los comandos del estándar NMEA, interpretando los datos transmitidos por el módulo y entregando un set de funciones y métodos que permiten tomar éstos datos y usarlos en lenguaje entendible para el humano. En la Tabla 5.3 se muestran las funciones y métodos que entrega ésta librería.

TinyGPS++: Métodos y Funciones		
Función	Retorna	Descripción
encode(char c)	bool	Procesa los mensajes recibidos en estándar NMEA.
location	TinyGPSLocation	Entrega la posición en una estructura con formato TinyGPSLocation
date	TinyGPSDate	Entrega la fecha en una estructura con formato TinyGPSDate
time	TinyGPSTime	Entrega la fecha en una estructura con formato TinyGPSTime
speed	TinyGPSSpeed	Entrega la velocidad en una estructura con formato TinyGPSSpeed
course	TinyGPSCourse	Entrega el curso en una estructura con formato TinyGPSCourse
altitude	TinyGPSAltitude	Entrega la altura en una estructura con formato TinyGPSAltitude
satellites	TinyGPSInteger	Entrega los satélites visibles en una estructura con formato TinyGPSInteger

Tabla 5.3: Métodos y funciones de la librería TinyGPS++

5.1.3. Librerías para módulo LoRa SX1278

En el caso de los módulos LoRa SX1278, existe una variedad de librerías que pueden entregar el resultado esperado, pero solamente se evaluarán dos de ellas.

Arduino-LoRa: Sandeep Mistry

Esta librería [56] tiene varias funcionalidades interesantes que podrían servir para el desarrollo del proyecto, entre ellas, la más importante es un ejemplo donde demuestran una comunicación Dúplex entre dispositivos LoRa. Si bien, como se muestra en la Figura 5.4, se deben agregar dos conexiones extra hacia el MCU para utilizar está opción, lamentablemente haciendo varias pruebas se llega a la conclusión de que la librería es incompatible con el MCU que se seleccionó actualmente, ya que al activar el periférico SPI, se cambias las configuraciones base del reloj interno del mismo. Esto implica que se debe hacer un trabajo extra bastante extenso al tratar de adecuar la librería a la MCU.

Semtech SX1276/77/78/79	Arduino
VCC	3.3V
GND	GND
SCK	SCK
MISO	MISO
MOSI	MOSI
NSS	10
NRESET	9
DIO0	2

Tabla 5.4: Caption

RadioLib: Jan Gromeš

Esta librería [57] tiene funcionalidades para varios módulos de radio frecuencia de distinta índole, entre ellos el módulo SX1278 que es el que se intenta utilizar, además se ha trabajado para tener compatibilidad con distintas plataformas, entre ellas la STM32F4, la cual corresponde al MCU que se utilizará. A diferencia que en el caso anterior, no se encuentra un ejemplo de comunicación Dúplex, por lo que esta parte se deberá trabajar de forma manual, pero el trabajo a realizar es bastante menor que el de adecuar una librería a otra plataforma. Con esto, se selecciona la librería RadioLib para trabajar con el módulo.

Para el desarrollo de la comunicación Dúplex, se usa el pin DIO0 dedicado a informar cuando se realiza una transmisión y cuando se detecta una recepción según [47] y según la configuración mostrada en la Figura 5.3, donde RxDone y TxDone son las señales de recepción y transmisión respectivamente. De esta forma, configurando un pin de interrupción, se puede detectar

tanto cuando se termina de enviar un dato y cuando se reciben datos desde otros módulos.

Operating Mode	DIOx Mapping	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
ALL	00	ModeReady	CadDetected	CadDone	FhssChangeChannel	RxTimeout	RxDone
	01	ClkOut	PllLock	ValidHeader	FhssChangeChannel	FhssChangeChannel	TxDone
	10	ClkOut	PllLock	PayloadCrcError	FhssChangeChannel	CadDetected	CadDone
	11	-	-	-	-	-	-

Figura 5.3: Configuración de pines del Módulo SX1278

Formato de envío

Debido a la facilidad que permite la programación en C++ de convertir información a su equivalente Binario, se opta por enviar toda la información al servidor en este formato, para esto se definen distintos paquetes de datos que serán entregados al servidor y posteriormente procesados. Primero, se define el paquete International Mobile Equipment Identity (IMEI) que enviará la información de identificación del módulo SIM800L para así distinguir los datos de los distintos dispositivos. Además, se definirá un paquete GPS que contendrá la información de latitud, longitud en un arreglo de datos, la hora en la que se tomó cada dato. Finalmente se define el paquete LoRa, que contiene la información de los mensajes recibidos desde otros dispositivos, en conjunto con la información del dispositivo que recibe, donde se tiene el IMEI del dispositivo que envió el mensaje, el IMEI del dispositivo que recibe el mensaje, la hora de recepción en el dispositivo que recibe y la posición del dispositivo en ese momento. En la Tabla 6.1 se muestra una pequeña descripción de cada uno de los paquetes que se usarán

Paquete	ID	Composición	Descripción
IMEI	40	[ID,TIME , IMEI] : [1 byte, 4 bytes, 7 bytes]	Contiene el Imei del módulo SIM800L junto con el tiempo en que fue tomado este dato
GPS	42	[ID, TIME,N, [LAT, LON]] : [1 byte, 4 bytes, 1 byte, [3 bytes, 3 bytes]	Contiene un arreglo de posiciones de GPS obtenidas desde el módulo NEO-6M junto con el tiempo en que fueron enviados estos datos y la cantidad de posiciones que fueron enviadas en este paquete.
LoRa	91	[ID, TIMESTAMP, MICROSECONDS, IMEI1,IMEI2, LAT,LON] : [1 byte, 4 bytes, 4 bytes,7 bytes, 7 bytes, 3 bytes, 3 bytes]	Contiene un paquete LoRa con el Imei del Dispositivo que envía el dato, el Imei del dispositivo que recibe los datos, y los tiempos locales del dispositivo que recibe los datos, además la posición en que se encontraba el dispositivo cuando recibió el mensaje LoRa.

Tabla 5.5: Tabla de paquetes que serán enviados al servidor, con su descripción e ID de paquete.

5.1.4. Sincronización de dispositivos

Como se comentó en los capítulos anteriores, para utilizar el método TDoA, se necesita tener una sincronización entre los dispositivos receptores. Debido a que la complejidad de resolver este requerimiento mediante Hardware conlleva la compra de un MCU más caro, o tratar de aumentar la precisión temporal utilizando otro Framework de programación. Debido a lo anterior, se decidió realizar una sincronización por GPS con la precisión de microsegundos que permite el MCU.

5.1.5. Unificación de librerías

Para unificar las distintas librerías, tanto la personalizada como las externas utilizadas, se necesita definir cada elemento en los puertos correspondientes, y además se necesita tener claridad en las funciones que se utilizarán de cada una de estas. Para el desarrollo del dispositivo de pruebas, se usará un tiempo predefinido de 1 minuto para enviar paquetes LoRa desde el dispositivo que se encuentre “perdido”, mientras el resto de dispositivos permanecerán en un estado de escucha permanente esperando que lleguen los paquetes. Además, se establecerá un tiempo fijo de 5 minutos para el envío de los datos al servidor, es decir, se recolectarán datos de GPS y paquetes LoRa durante 5 minutos, y se iniciará el proceso para enviarlos al servidor. La codificación de datos desde el módulo NEO-6M se debe realizar en todo momento,

ya que se recibirán datos constantemente desde el módulo. Desde el módulo NEO-6M se tomará el dato de latitud y longitud, además de la hora. El dato de hora se usará para determinar el momento en que llegan los paquetes LoRa desde otros dispositivos, ya que se necesita una precisión de al menos microsegundos

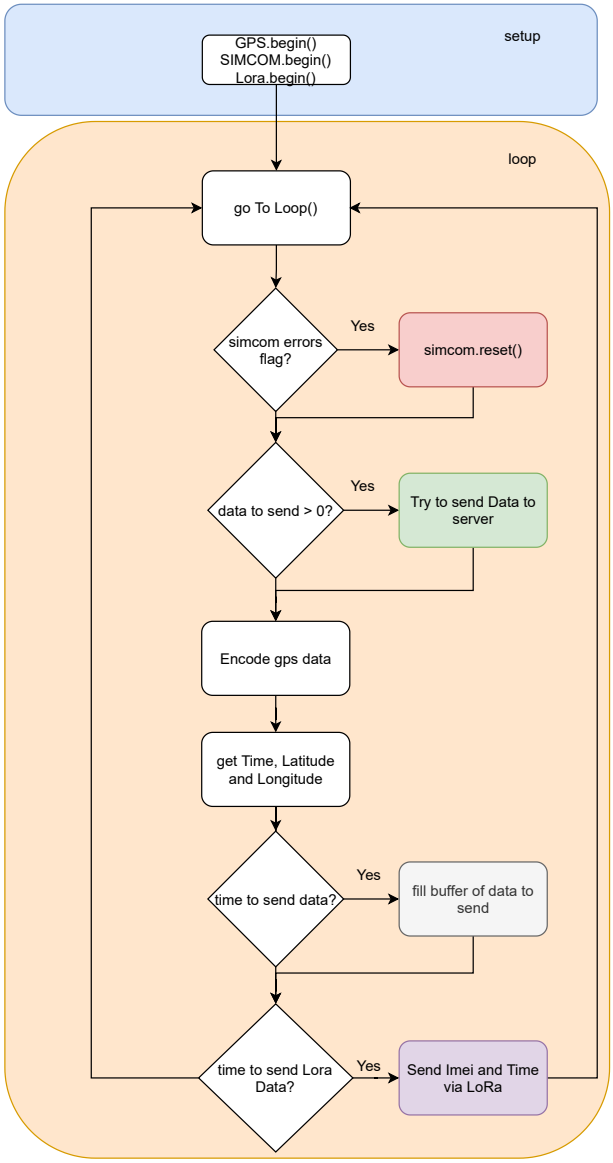


Figura 5.4: Diagrama de Flujo del programa unificado

El programa consta de dos funciones principales, `setup()` y `loop()`. En el la función `setup()` se inicializan las librerías de los módulos habilitando los puertos de los periféricos (SPI y UART), y dejando todas las variables en su estado inicial. En la función `loop()`, se efectúa el algoritmo completo de conexión a internet, toma de datos de GPS, envío y detección de datos

LoRa. El algoritmo consta de varias partes, primero se procesa la maquina de estados del módulo SIM800L, y se evalúa si existen errores para reiniciar el procedimiento. Luego, se evalúa si existen datos listos para enviar, luego se decodifican los datos del módulo NEO-6M, obteniendo la latitud , longitud y hora. Después se evalúa si es momento de enviar los datos para llenar un arreglo con los datos que serán entregados al servidor. Finalmente, en el dispositivo "perdido" se evalúa si ha pasado un minuto, para enviar un mensaje via LoRa hacia los demás dispositivos. En la Figura 5.4 se presenta el diagrama de flujo descrito anteriormente.

Capítulo 6

Diseño y arquitectura del Servidor

En este capitulo nos centraremos en la programación y levantamiento del servidor de pruebas encargado de procesar y guardar los datos que serán enviados desde los dispositivos diseñados en los capítulos anteriores. Para lograr esto se usará un servidor gratuito proporcionado por Amazon Web Services (AWS), con su servicio Elastic Computing Cloud (EC2), así obtendremos una IP pública para poder enviar los datos necesarios desde cualquier dispositivo y desde cualquier lugar.

6.1. Procedimiento a seguir

A continuación se describe el procedimiento para levantar un servidor en AWS, y desarrollar una arquitectura completa para el guardado de los datos en la nube. Para usar los servicios que entrega Amazon debemos tener una cuenta creada previamente, así podremos acceder a los servicios gratuitos que serán suficientes para el propósito de este proyecto.

6.1.1. Creación de un servidor virtual en la nube utilizando EC2

Si bien los servicios virtuales de Amazon son relativamente nuevos, ya tienen varios años de crecimiento donde han expandido la cantidad de servicios y la calidad de los mismos. La consola EC2 permite levantar “instancias” , las que corresponden a maquinas virtuales situadas en un servidor a las que se les concede una IP pública y una serie de recursos para su uso.

Crear una instancia

Para crear una instancia se debe abrir la consola EC2 e ir a la opción “Lanzar instancia”, tal como se muestra en la Figura 6.1.

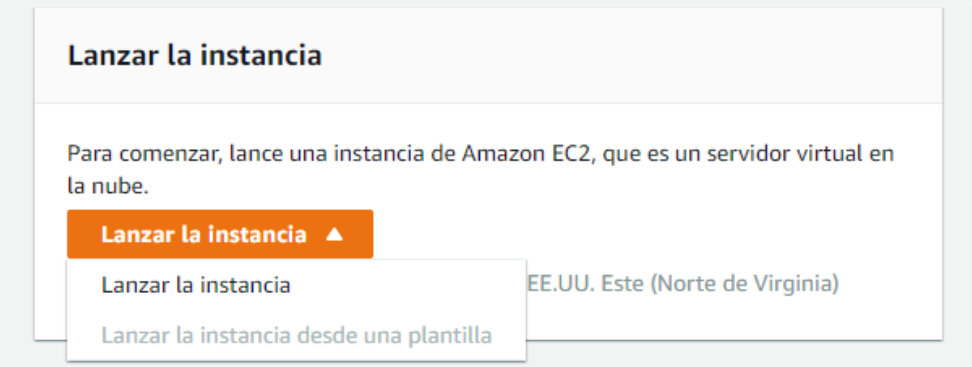


Figura 6.1: Botón y opción de lanzar instancia

Al seleccionar esta opción nos aparecerán una serie de opciones de instancia que se pueden lanzar, entre las cuales hay servidores con sistema operativo Ubuntu, Amazon Linux, Open Suse entre otros. Sin importar la opción que se escoja, es mejor trabajar bajo una arquitectura Unix, por lo tanto para este proyecto se seleccionará el sistema operativo Amazon Linux, ya que es la opción apta para la capa gratuita que ofrece AWS tal como se muestra en la Figura 6.2.

Figura 6.2: Sistema operativo seleccionado para lanzar la instancia.

A continuación se debe seleccionar el tipo de instancia que se usará. Tal como en el paso anterior, se seleccionará el tipo de instancia apto para la capa gratuita de AWS, la que corresponde al tipo t2.micro como se muestra en la Figura 6.3

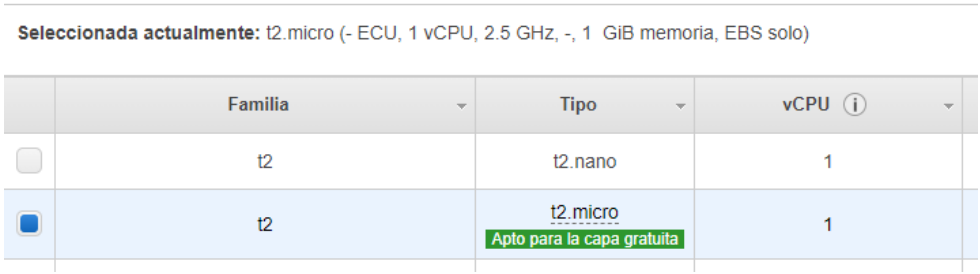


Figura 6.3: Tipo de instancia seleccionada para mantenerse dentro de la capa gratuita de AWS.

A continuación se debe revisar la instancia para lanzarla. En este paso se permite agregar grupos de seguridad de red previamente configurados, o configurar el grupo de seguridad de la instancia para abrir puertos que permitan la comunicación con el resto de la red. Por defecto las instancias tiene el puerto 22 abierto para las conexiones mediante SSH como se muestra en la Figura 6.4, lo que permite la configuración y programación del servidor desde una máquina externas.

Figura 6.4: Configuración por defecto del puerto 22 para conexiones SSH.

Al lanzar la instancia se solicita generar un par de llaves para la conexión SSH , como se muestra en la Figura 6.5, entre un cliente externo y el servidor, sin este par de claves es imposible conectarse al servidor, por lo tanto debe ser guardada para utilizarla posteriormente.

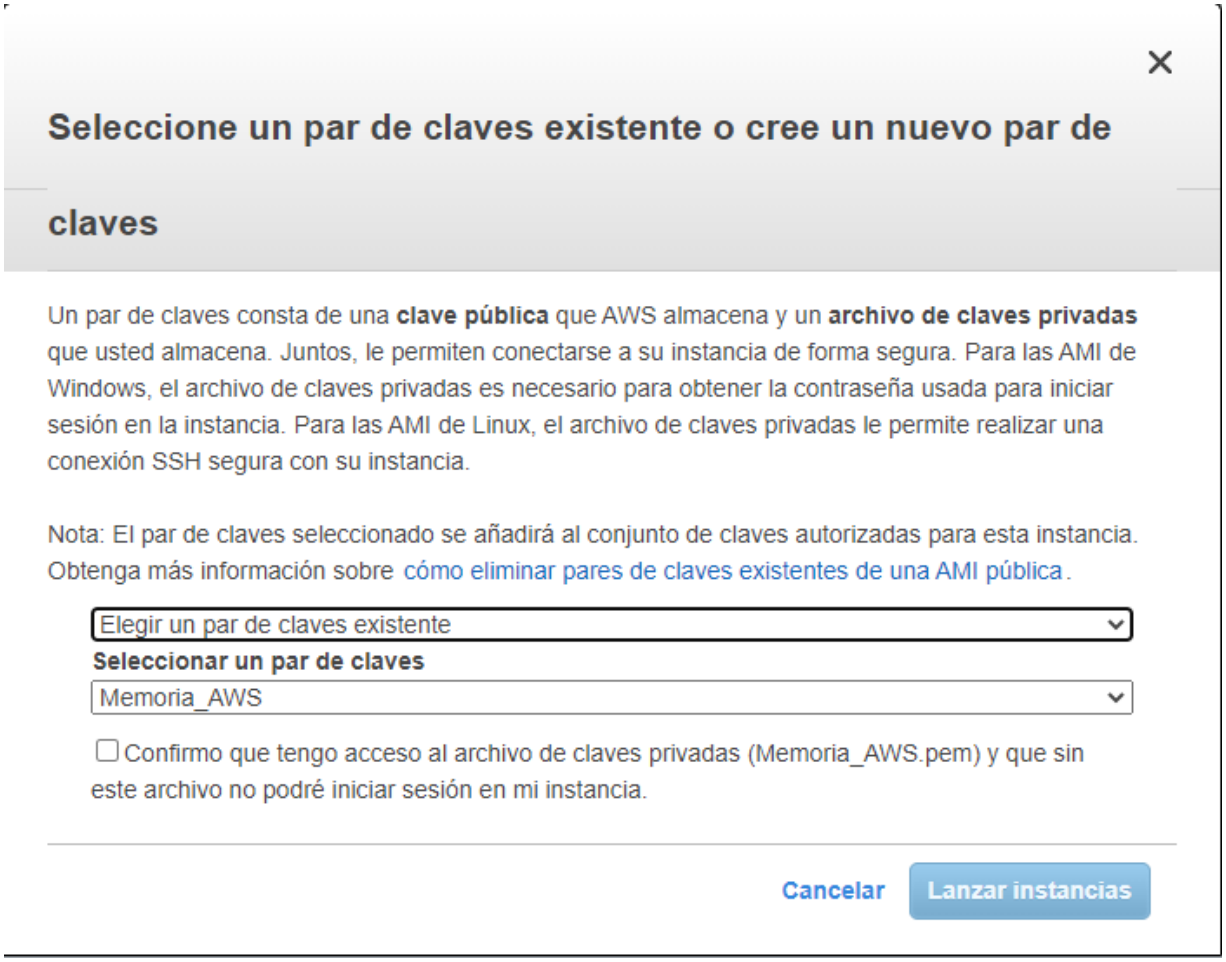


Figura 6.5: Generación del par de llaves para la conexión entre el cliente y el servidor.

6.1.2. Descripción de la arquitectura del servidor

Debido a los requerimientos necesarios del proyecto es necesario que la información enviada desde los dispositivos sea guardada en una base de datos para luego analizarla y realizar los cálculos necesarios. Debido a esto se debe utilizar una arquitectura que contemple la incorporación de esta base. Esto quiere decir que se debe tener un servidor que reciba datos a través de TCP, los decodifique según el formato determinado anteriormente y los guarde en una base de datos, es decir, una arquitectura como la mostrada en la Figura 6.6

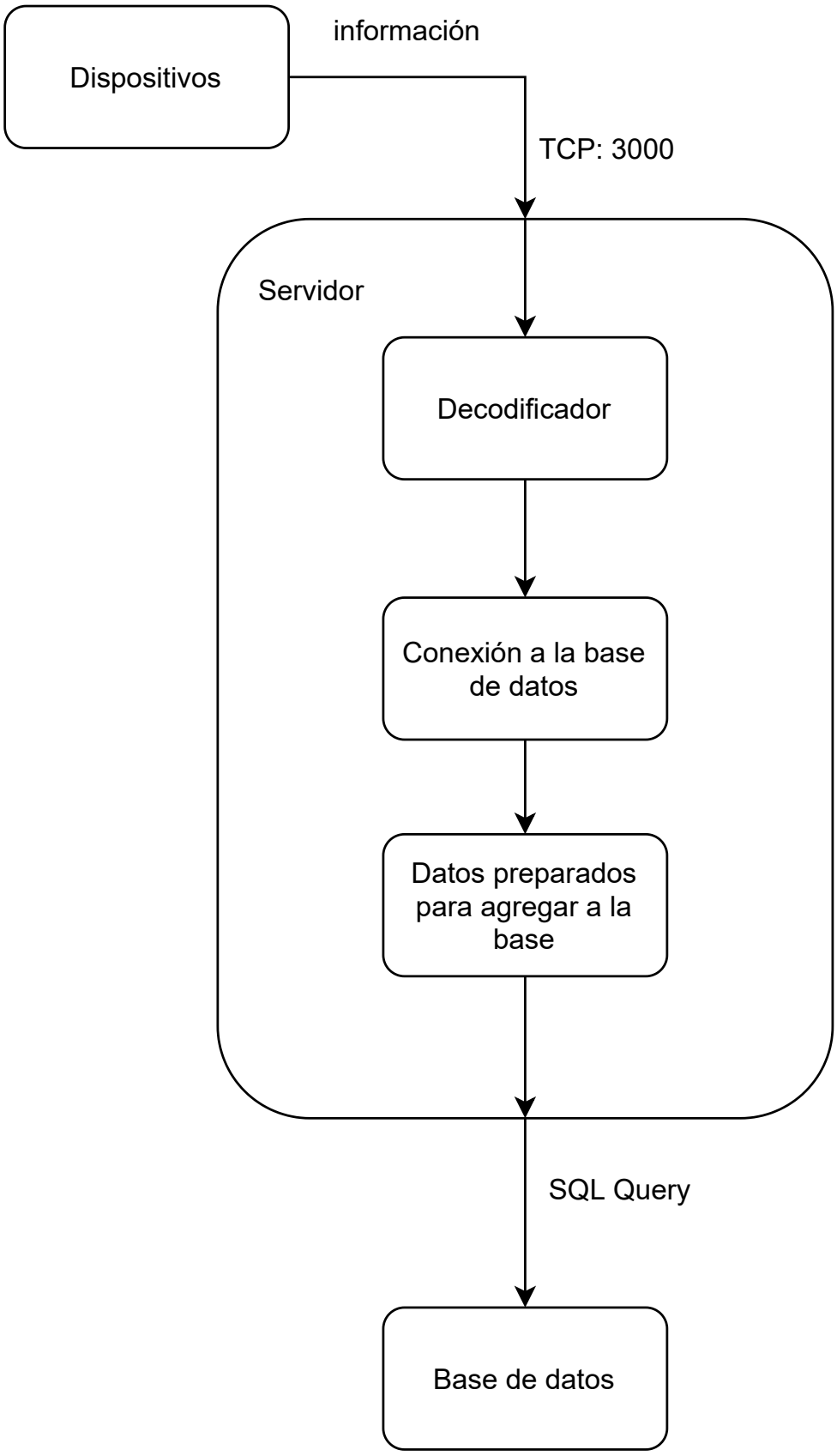


Figura 6.6: Arquitectura del servidor a programar

6.1.3. Programación del servidor

Para la programación del servidor se utilizará el lenguaje de programación Javascript con el entorno de trabajo (Framework) Nodejs, ya que se posee conocimientos previos sobre este lenguaje, y se considera más sencillo para el desarrollo del proyecto. Además para montar el servidor se utilizará el entorno de trabajo de aplicaciones Web (Web Application Framework) "Express" facilitando la conexión entre los dispositivos y el servidor.

Configuración inicial de la instancia EC2

Cuando se trabajo con aplicaciones Web, las conexiones desde la red se realizan a través de los puertos 80 para (Hypertext Transfer Protocol (HTTP) y 443 para (Hypertext Transfer Protocol (HTTPS) en caso de que se requiera tener una interfaz Web. Para esta aplicación se utilizará una comunicación TCP con el servidor, donde se decide utilizar el puerto 3001 para recibir la información desde los dispositivos. Para esto, se necesita agregar este puerto a las excepciones del Grupo de seguridad de la instancia. Esto se realiza ingresando a la consola EC2 y seleccionando la opción Grupos de seguridad. En esta opción se mostrarán los grupos de seguridad previamente creados, se debe seleccionar el grupo de seguridad asociado a la instancia levantada previamente. Finalmente utilizando la opción "Editar reglas de entrada" se debe agregar el puerto 3001 utilizando el protocolo TCP desde todos los orígenes, tal como se muestra en la Figura 6.7

Reglas de entrada (3)

Tipo	Protocolo	Intervalo de puertos	Origen
SSH	TCP	22	0.0.0.0/0
TCP personalizado	TCP	3000 - 3001	0.0.0.0/0
TCP personalizado	TCP	3000 - 3001	::/0

Figura 6.7: Configuración de Grupos de seguridad de la consola EC2

Bloque decodificador

Tal como se definió anteriormente, los mensajes provenientes desde los dispositivos tendrán la estructura mostrada en la Tabla 6.1.

Paquete	ID	Composición	Descripción
IMEI	40	[ID,TIME , IMEI] : [1 byte, 4 bytes, 7 bytes]	Contiene el Imei del módulo SIM800L junto con el tiempo en que fue tomado este dato
GPS	42	[ID, TIME,N, [LAT, LON]] : [1 byte, 4 bytes, 1 byte, [3 bytes, 3 bytes]	Contiene un arreglo de posiciones de GPS obtenidas desde el módulo NEO-6M junto con el tiempo en que fueron enviados estos datos y la cantidad de posiciones que fueron enviadas en este paquete.
LoRa	91	[ID, TIMESTAMP,MICROSECONDS, IMEI1,IMEI2, LAT,LON] : [1 byte, 4 bytes, 4 bytes, 7 bytes, 7 bytes, 3 bytes, 3 bytes]	Contiene un paquete LoRa con el Imei del Dispositivo que envía el dato, el Imei del dispositivo que recibe los datos, y los tiempos locales del dispositivo que recibe los datos, además la posición en que se encontraba el dispositivo cuando recibió el mensaje LoRa.

Tabla 6.1: Tabla de paquetes que serán enviados al servidor, con su descripción e ID de paquete.

Por lo tanto el Bloque decodificador, corresponde a un ciclo For o While que recorre los datos que provienen desde el dispositivo preguntando por el ID del dato que está leyendo, y sabiendo que cada paquete tiene un largo definido, se hace la descomposición del mismo obteniendo así el valor que fue enviado. Este proceso se describe en la Figura 6.8.

Bloque de conexión a la base de datos

Este bloque simplemente establece una conexión interna entre el servidor y la base de datos, entregando las credenciales necesarias para insertar datos en las tablas de esta misma.

Bloque de preparación de datos

Este bloque se encarga de preparar las consultas de Structured Query Language (SQL) que permiten insertar los datos recibidos desde los dispositivos en la base de datos, utilizando cada paquete decodificado e ingresando estos datos en las tablas correspondientes.

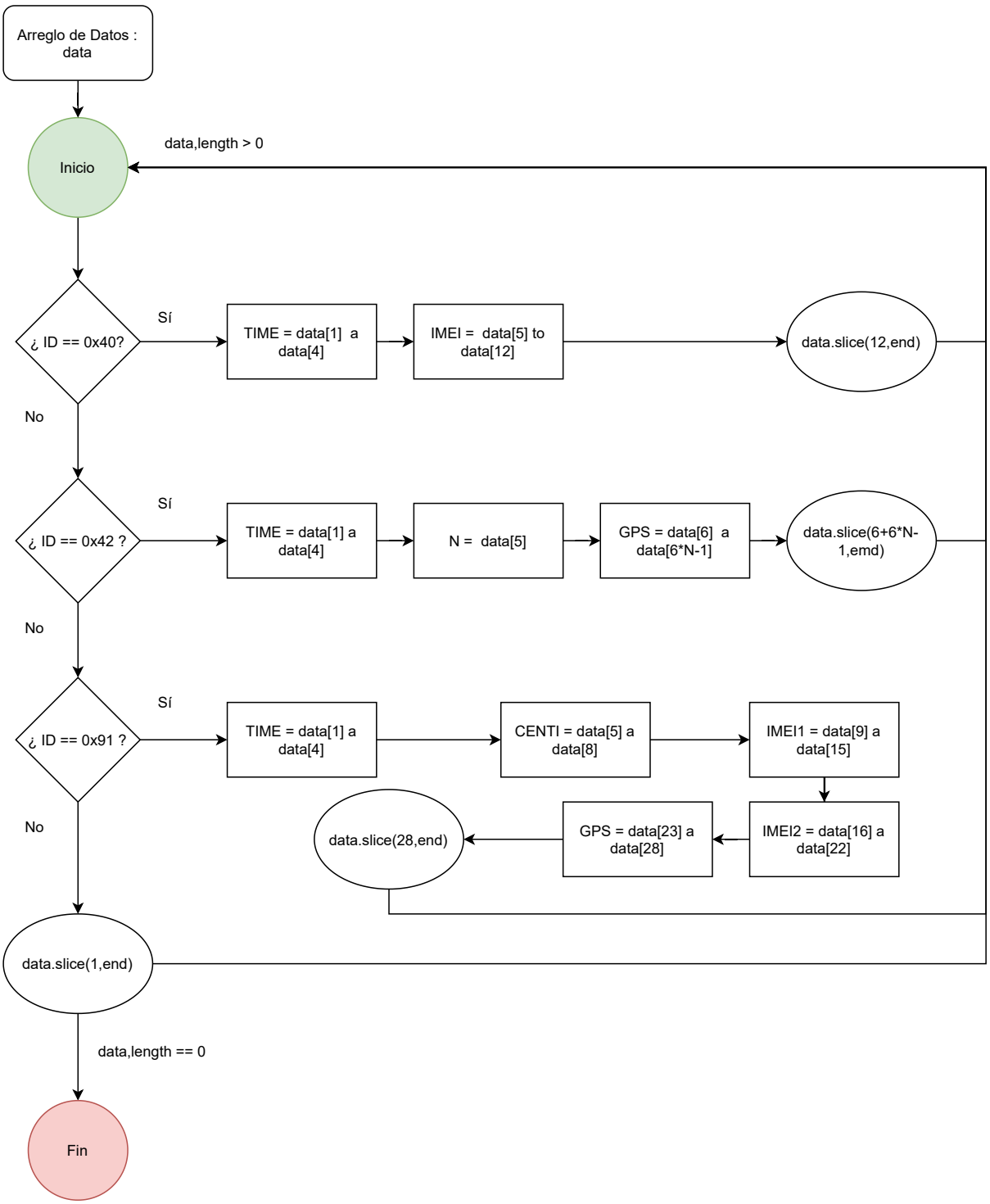


Figura 6.8: Diagrama de flujo del bloque decodificador

6.2. Base de datos

En esta sección se dará una descripción de los pasos a seguir para levantar una base de datos en la plataforma de AWS, y se mostrará la estructura de las tablas diseñadas para guardar la información enviada desde dispositivos.

6.2.1. Creación de la base de datos

Para crear una base de datos en la plataforma de AWS, se debe acceder al panel de servicios y buscar la sección de Base de datos como el mostrado en la Figura 6.9.

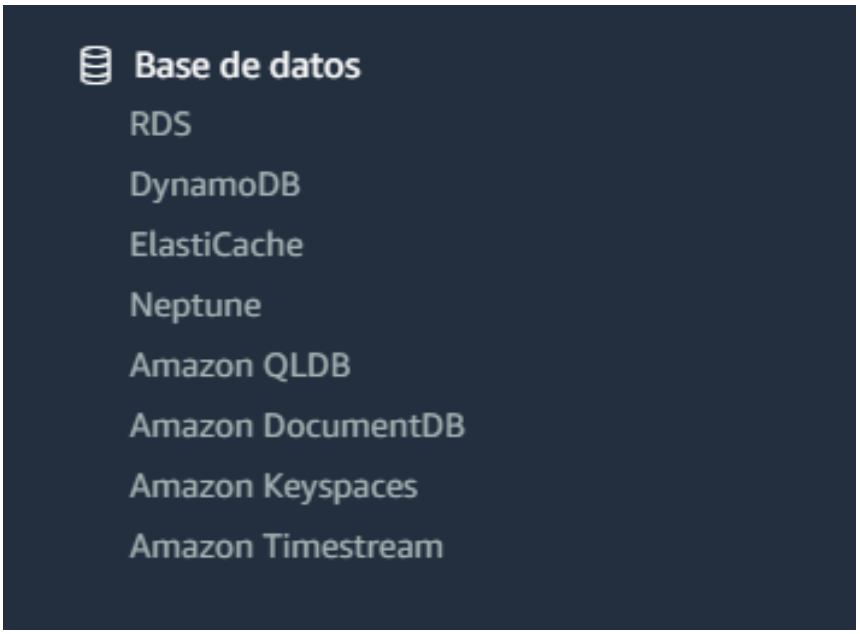


Figura 6.9: Sección de base de datos de AWS

Existen varias opciones de bases de datos [58] , en esta ocasión nos centraremos en una estructura que permita usar el motor de bases de datos MySQL ya que se tiene experiencia trabajando con él. De las alternativas mostradas en la Figura 6.10, entre ellas las que ofrecen una estructura diseñada para conectarse con MySQL son Amazon RDS, Amazon Redshift, Amazon Aurora. Ya que Amazon Aurora es un motor de bases de datos ofrecido por AWS, y Amazon Redshift es un procesador de datos, el servicio a usar es Amazon RDS para crear una base de datos basada en MySQL.

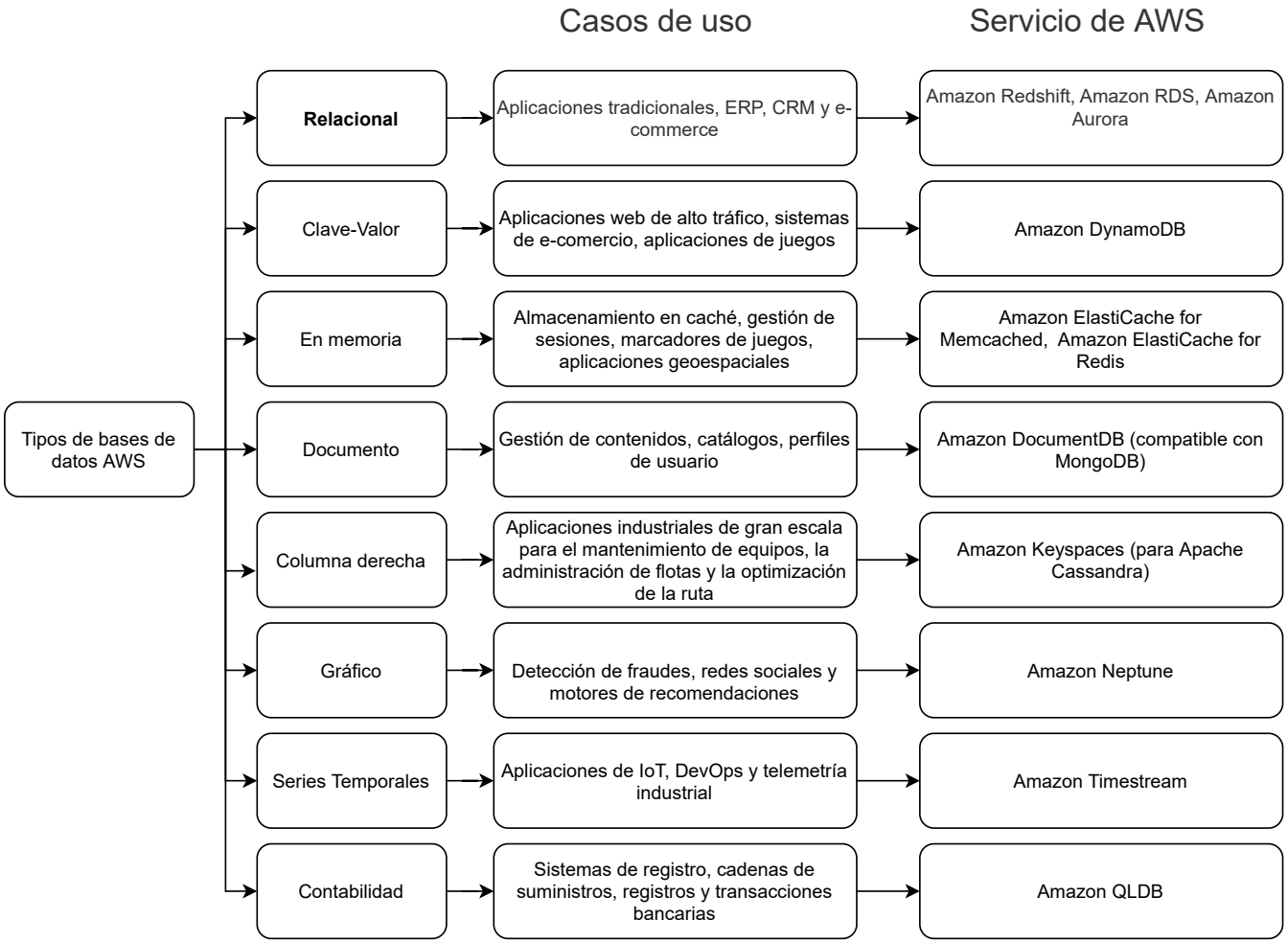


Figura 6.10: Clasificación de bases de datos en Amazon Web Services

Para crear una base de datos en Amazon RDS, se debe seleccionar la opción RDS mostrada en la Figura 6.9, ir a la sección “Crear base de datos” mostrada en la Figura 6.11. En la creación de la base se debe seleccionar el método de creación (proporcionado por AWS), el motor de base de datos, que será MySQL en este caso, la versión del motor seleccionado y las configuraciones iniciales de las credenciales de acceso, instancias, almacenamiento y conectividad. Con estos elementos configurados se creará la instancia de la base, a la que nos podremos conectar desde el motor seleccionado. Un paso importante para acceder a la base de datos desde dispositivos externos a los proporcionados por AWS, es la selección de acceso público de la base, esta selección se muestra en la Figura 6.12.

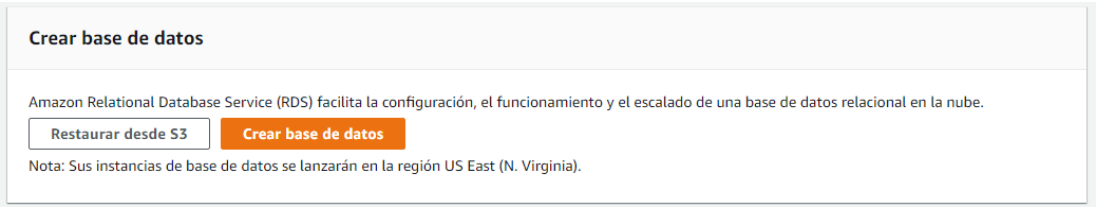


Figura 6.11: Sección crear base de datos de AWS.

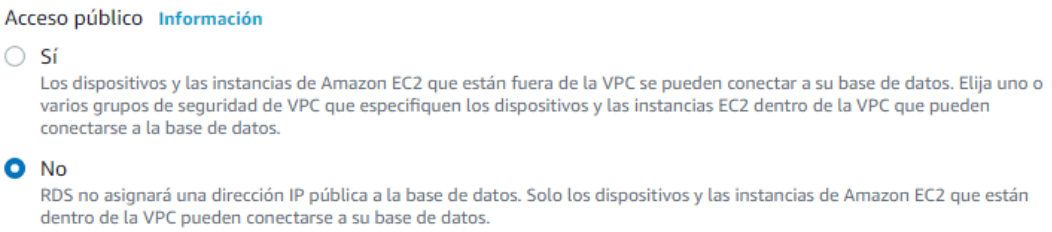


Figura 6.12: Selección de tipo de acceso a la base de datos

6.2.2. Modelo de base de datos

Según la estructura presentada en la Tabla 6.1 se necesita almacenar los datos de 3 tipos de paquetes Imei, GPS y Lora. El Imei debe tener una tabla separada, ya que cada dispositivo entregará un dato de Imei aparte, esto funcionará como un identificador de dispositivos. Todos los datos que sean enviados desde los dispositivos serán guardados en una tabla, la que contendrá los datos de GPS de cada dispositivo. Los paquetes LoRa, serán datos que envíen los dispositivos con conexión cuando les llegue un mensaje de alerta desde algún dispositivo "perdido". Esta configuración se muestra en la Figura 6.13 con un diagrama relacional de las tablas creadas.

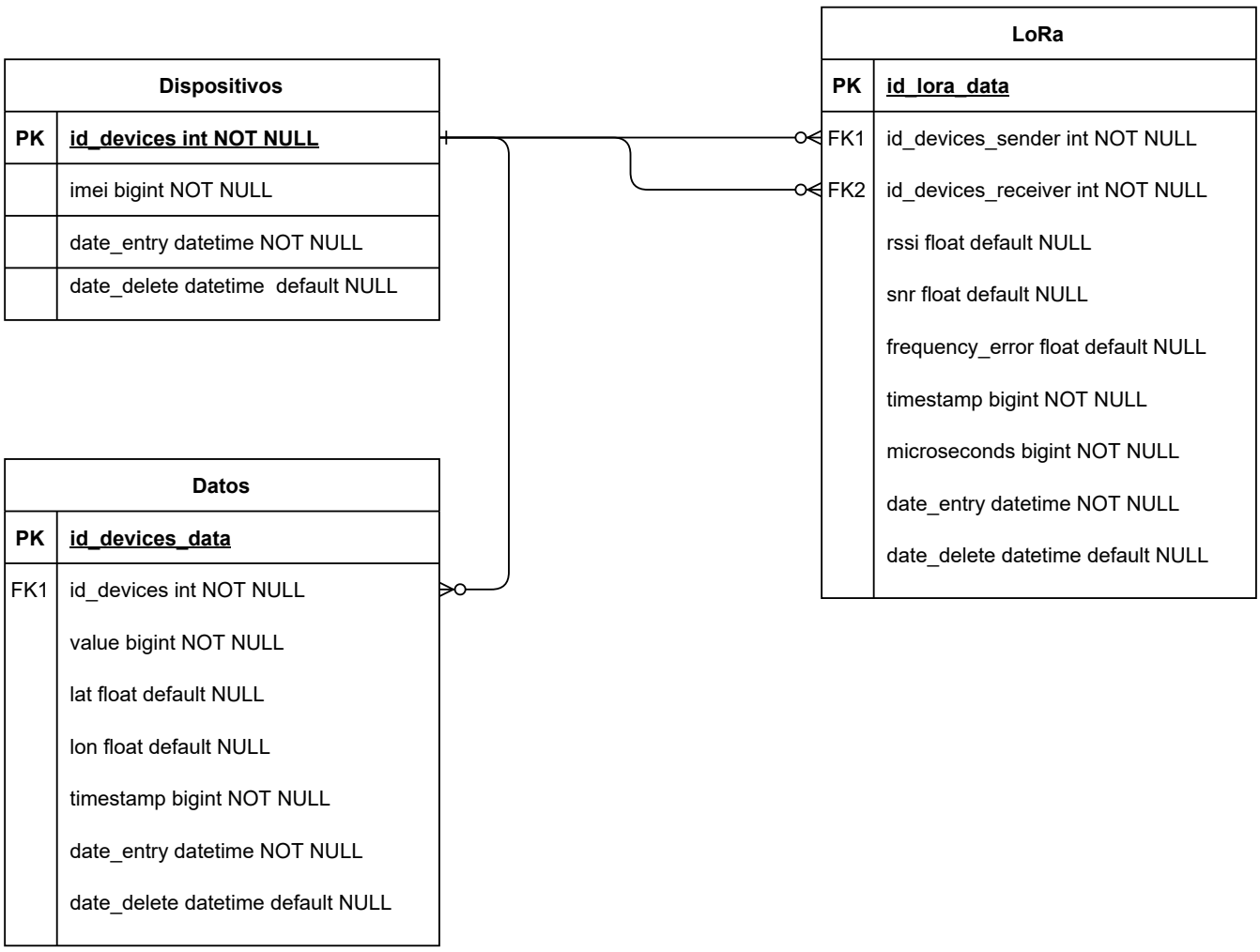


Figura 6.13: Diagrama de relaciones de base de datos

Capítulo 7

Prototipo y pruebas

En este capítulo se mostrarán los resultados finales correspondientes al armado del dispositivo de pruebas, el funcionamiento general del sistema implementado utilizando un ambiente controlado de testeo para comprobar que los dispositivos envían la información necesaria para detectar la ubicación utilizando el algoritmo TDoA. Lamentablemente debido a la situación a nivel mundial no se pueden realizar todas las pruebas deseadas con los dispositivos, por lo tanto se complementarán los datos utilizando simulaciones analizadas en profundidad en la sección 7.3.

7.1. Armado del Prototipo

A continuación se describe el procedimiento para armar los dispositivos de pruebas, mostrando los cambios realizados al diagrama de conexión mostrado en la Figura 4.26 para hacer funcionar de forma definitiva los módulos.

7.1.1. Modificaciones y diagrama final

Para utilizar el dispositivo en situaciones reales, se agregó una batería externa de 3.7V y 3000mA que permita energizar todos los módulos, además como se necesita cargar la batería se incluyó un módulo de carga. Para esto se utilizará el módulo TP4056 mostrado en la Figura 7.1. Así mismo los pines dedicados a DEBUG serán conectados a un conversor UART a Universal Se-

rial Bus (USB) para mostrar mensajes desde el dispositivo de pruebas, lo que permite revisar su estado desde un computador.

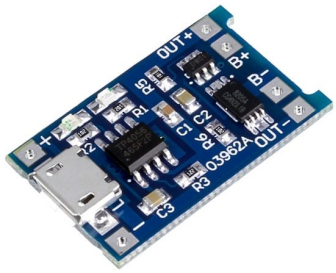


Figura 7.1: Módulo de carga de batería TP4056

A continuación en la Figura 7.2 se presenta el diagrama final con las conexiones necesarias para el funcionamiento del dispositivo de pruebas y los cambios mencionados en la sección anterior.

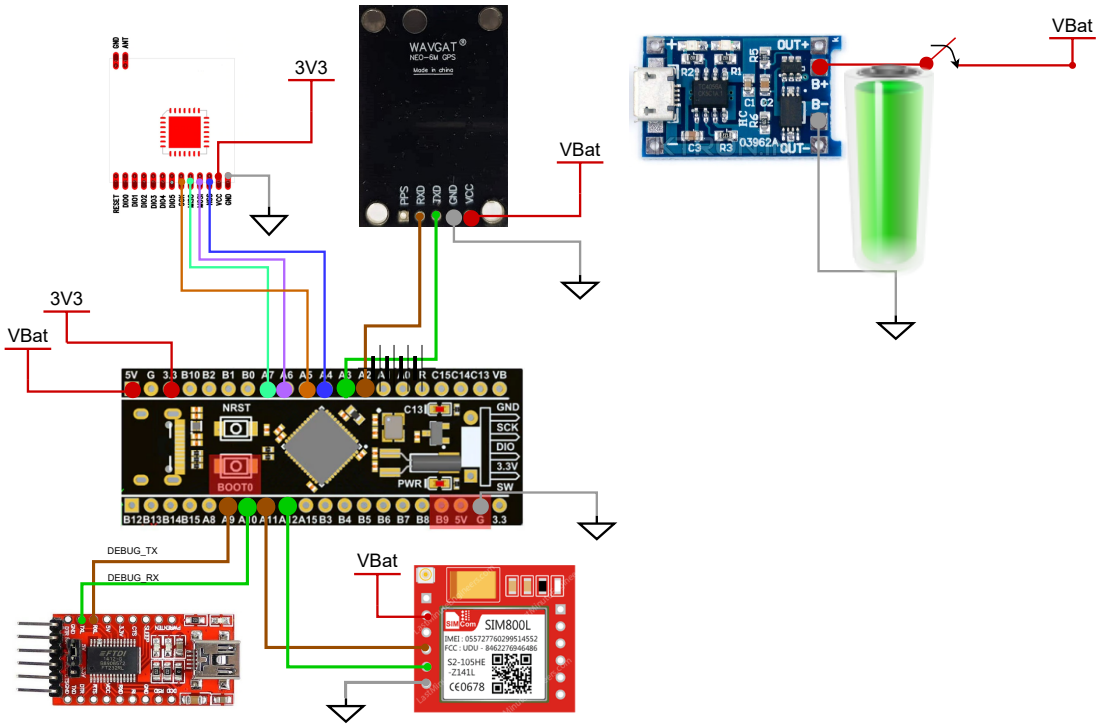


Figura 7.2: Diagrama de conexión del dispositivo con los últimos cambios para su funcionamiento.

7.1.2. Implementación

En base el diagrama mostrado en la Figura 7.2 y en con una placa perforada de circuitos impresos o Printed Circuit Board (PCB) como la mostrada en la Figura 7.3, se montan y conectan los módulos seleccionados de manera física. Finalmente el dispositivo de pruebas es empaquetado con una caja impermeable de 10x10cm para poder moverlo sin problemas al momento de hacer pruebas. El resultado de la conexión de todos los elementos dentro de esta placa se muestra en la Figura 7.4.

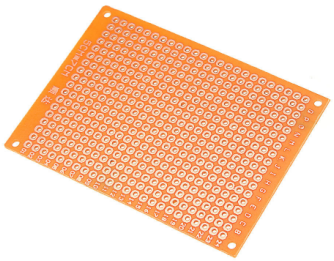


Figura 7.3: Placa perforada para montar y conectar los módulos.

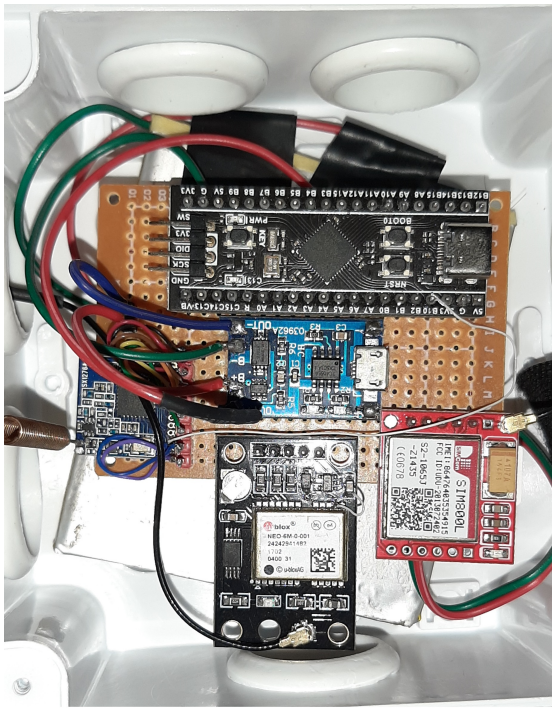


Figura 7.4: Resultado de conexión y empaquetado del dispositivo de pruebas.

7.2. Pruebas de detección de ubicación

Con los dispositivo de prueba armados, se realiza una prueba en un área controlada pequeña con 4 dispositivos, 3 de ellos como receptores y uno como emisor emulando una situación de emergencia, en la que se encuentra sin información de GPS. Esta situación se emula enviando mensajes a través de LoRa solicitando determinar la ubicación del dispositivo emisor. Algunos de los datos obtenidos desde los tres dispositivos receptores se muestran en la Tabla 7.1, correspondiente a las posiciones obtenidas desde el dispositivos en formato de latitud y longitud, y en la Tabla 7.2 , se entregan los datos obtenidos desde los dispositivos receptores, donde la información se diferencia con los prefijos *p* para las posiciones en formato de latitud y longitud, *t* para el tiempo en segundos en formato epoch linux, y *m* para los microsegundos.

N	Latitud	Longitud
1	-33.43364	-70.647582
2	-33.433639	-70.647585
3	-33.4336401	-70.647586
4	-33.433637	-70.647583
5	-33.433638	-70.647584
6	-33.43364	-70.647581

Tabla 7.1: Datos guardados en la base de datos, obtenidos desde el dispositivo emisor.

N	R1	R2	R3
1	p:-33.434729, -70.642970 t: 1624717612 m: 246781	p:-33.435734, -70.6474558 t: 1624717612 m: 246786	p:-33.434935, -70.6458243 t: 1624717612 m: 246785
2	p:-33.434730, -70.642971 t: 1624717672 m: 456783	p:-33.4357339, -70.647456 t: 1624717672 m: 456785	p:-33.434936, -70.6458251 t: 1624717672 m: 456786
3	p:-33.434730, -70.642968	p:-33.4357341, -70.6474561	p:-33.434937, -70.6458248

	t: 1624717732 m: 643785	t: 1624717732 m: 643785	t: 1624717732 m: 643788
4	p:-33.434731, -70.642972 t: 1624717792 m: 871785	p:-33.4357338, -70.6474559 t: 1624717792 m: 871786	p:-33.434937, -70.6458245 t: 1624717792 m: 871789
5	p:-33.434729, -70.642971 t: 1624717852 m: 923587	p:-33.4357341, -70.647456 t: 1624717852 m: 923588	p:-33.434935, -70.6458252 t: 1624717852 m: 923586
6	p:-33.434732, -70.642970 t: 1624717912 m: 355947	p:-33.4357339, -70.6474557 t: 1624717912 m: 355946	p:-33.434937, -70.6458249 t: 1624717912 m: 355946

Tabla 7.2: Datos guardados en la base de datos, obtenidos desde los dispositivos receptores.

Estos datos se deben transformar a coordenadas euclidianas para trabajarlos de mejor manera, para esto se utiliza en Python la herramienta *utm*, que permite transformar coordenadas en formato latitud y longitud al formato euclidiano. Luego de esto, se puede calcular la posición aproximada del dispositivo emisor, los resultados se muestra en el mapa de la Figura 7.5. Para esta prueba todos los dispositivos se encuentran en una ubicación fija. En la figura, el triangulo corresponde la posición del dispositivo emisor, y las X de color verde corresponden a las ubicaciones de los dispositivos receptores. Cada punto es una estimación de ubicación utilizando TDoA, donde las lineas que unen a los puntos y el triangulo corresponden a las distancias entre las ubicaciones estimadas por el algoritmo y la posición real del dispositivo emisor. Cada linea de distancia se muestra con diferentes colores debido a que corresponde a diferentes mediciones.

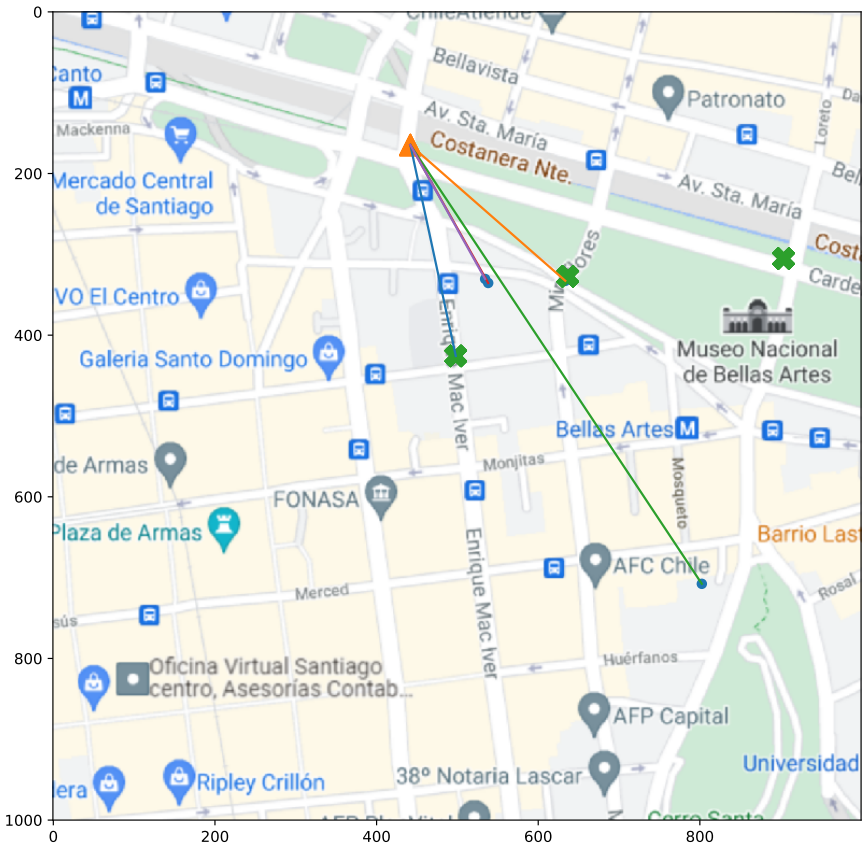


Figura 7.5: Ubicación de los dispositivos receptores, dispositivo emisor, y ubicación calculada. La marca triangular es el emisor, y las marcas con X son los dispositivos receptores, los puntos son las ubicaciones calculadas.

Según esto los cálculos de ubicación muestran una diferencia en cuanto a la posición real del dispositivo emisor. En la Tabla 7.3 se muestran los resultados de esta prueba simple, indicando el error medio, el error mediano y el máximo error incurrido en estas mediciones. Según estos datos el error medio de medición es de aproximadamente 312 metros y el máximo error corresponde aproximadamente a 652 metros.

Error	Valor
Error mediano	254.9 m
Error Medio	311.9 m
Máximo Error	651.8 m

Tabla 7.3: Resultados de prueba con dispositivos fijos en un ambiente controlado.

7.3. Complemento de pruebas con simulación

En esta sección, se busca generar información de dispositivos en movimiento para llevar las pruebas al ambiente más parecido al real, donde se instalarán los dispositivos en vehículos que se mueven a través de una ciudad. Por lo tanto, para complementar los resultados obtenidos en la prueba de la sección anterior, se realizaron simulaciones de un entorno que se parezca en lo posible al esperado. Para esto se utilizó el programa OMNeT++ con el Framework FloRa para posicionar los dispositivos en un cuadrado de 1000x1000 m el que corresponde a una porción del mapa de Santiago. Además para recuperar los datos necesarios que entrega este programa se desarrolla un código en Python (Ver Anexo 3 7.5.2) que permite identificar las actividades de cada dispositivo y recuperar los datos que entrega cada uno.

7.3.1. Configuración del Framework

Para utilizar el Framework FloRa y generar un ambiente de dispositivos en movimiento primero se debe cambiar el elemento de movilidad interno de éste. El procedimiento consiste en reemplazar el elemento interno Stationary mobility por algún otro de OMNeT++ que permita programar movimientos. En este caso se usó el elemento Turtle Mobility, con el que se ajustan movimientos de acuerdo a los comandos mostrados en el Anexo 2 (7.5.2). El siguiente paso corresponde a generar archivos de configuración y conexiones que permitan simular la red y el movimiento de cada elemento. La configuración visual de la red de simulación se muestra en la Figura 7.6, donde los dispositivos receptores corresponden a las antenas marcadas como *loRaGW1*, *loRaGW2* y *loRaGW3*, y el dispositivo emisor corresponde al nodo marcado como *loRaNode*. Los routers, conexiones con la nube (Internet Cloud) y servidor, son elementos necesarios por OMNeT++ para simular la red.

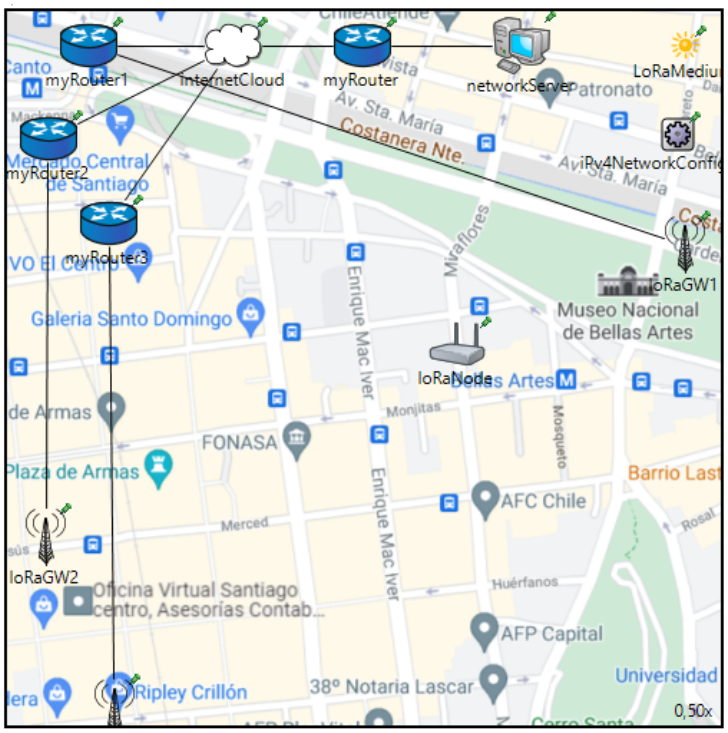


Figura 7.6: Configuración inicial de dispositivos en el Framework FloRa

Durante este proceso se realizan distintas pruebas, primero para entender el funcionamiento del Framework con movimientos simples, y luego se realizan pruebas más largas para simular movimientos complejos similares al entorno esperado.

7.4. Resultados de Simulaciones

Debido a que los resultados de la sección 7.2 muestran que los errores obtenidos al estimar la posición del emisor son bastante grandes, se utilizan los resultados de las simulaciones para explorar una mejora en la sincronización temporal del dispositivo desarrollado. Esta mejora se plantea según las características del microcontrolador, lo que nos permitiría aumentar la precisión de sincronización sin cambiar elementos de Hardware. A continuación se presentan los resultados de simulaciones realizadas utilizando 3 distintas precisiones temporales en el algoritmo TDoA, la primera de estas precisiones corresponde a la que tiene cada uno de los dispositivo armados, es decir, una precisión de los microsegundos, en la segunda revisión se agrega un decimal de precisión al algoritmo llevando la precisión temporal a los 0.1 microsegundos,

y en la última revisión se agrega un segundo decimal de precisión llegando a un precisión de 0.01 microsegundos.

Revisión con precisión de 1 microsegundo

En la Tabla 7.4 se muestran los resultados de una simulación de dispositivos en movimiento para una simulación de 290 segundos con una frecuencia de envío de 30 segundos entre paquetes LoRa en la primera prueba. En la Figura 7.7 se muestra este resultado gráficamente, donde los triángulos corresponden a las posiciones del dispositivo emisor, los puntos a las estimaciones del algoritmo y las lineas a las distancias entre la posición del emisor y las estimaciones.

Error	Valor
Error mediano	290.55 m
Error Medio	327.39 m
Máximo Error	687.49 m

Tabla 7.4: Resultados de simulación con dispositivos en movimiento y precisión de microsegundos.

Figura 7.7: Resultados de simulación con dispositivos en movimiento y una precisión de microsegundos. La marca triangular es el emisor, y los puntos son las estimaciones, cada linea corresponde a la distancia entre la estimación y la posición real del emisor.

Revisión con precisión de 0.1 microsegundos

En la Tabla 7.5 se muestran los resultados de una simulación de dispositivos en movimiento para una simulación de 290 segundos con una frecuencia de envío de 30 segundos entre paquetes LoRa y una precisión de 0.1 microsegundos. En la Figura 7.8 se muestra este resultado gráficamente , donde los triángulos corresponden a las posiciones del dispositivo emisor, los puntos a las estimaciones del algoritmo y las lineas a las distancias entre la posición del emisor y las estimaciones.

Error	Valor
Error mediano	82.56 m
Error Medio	126.37 m
Máximo Error	569.98 m

Tabla 7.5: Resultados de simulación con dispositivos en movimiento y precisión de 0.1 microsegundos.

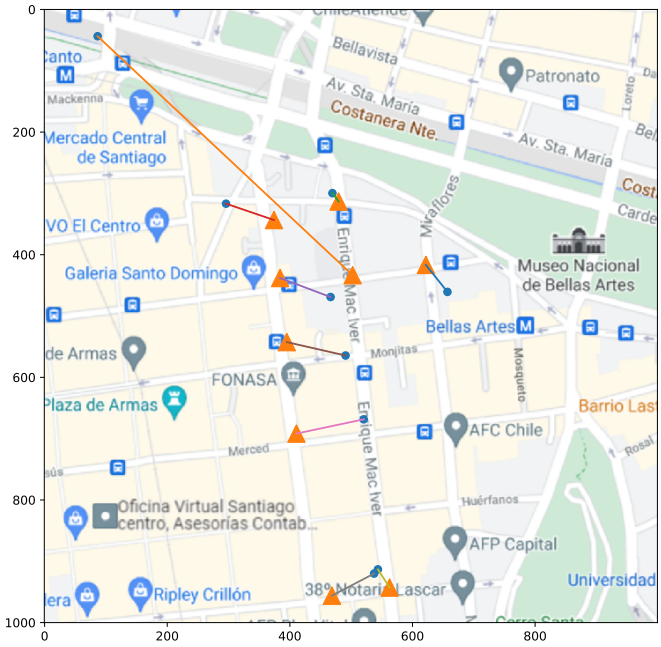


Figura 7.8: Resultados de simulación con dispositivos en movimiento y una precisión de microsegundos. La marca triangular es el emisor, y las marcas con o son las estimaciones.

Revisión con precisión de 0.01 microsegundos

En la Tabla 7.6 se muestran los resultados de una simulación de dispositivos en movimiento para una simulación de 290 segundos con una frecuencia de envío de 30 segundos entre paquetes LoRa y una precisión de 0.01 microsegundos. En la Figura 7.9 se muestra este resultado gráficamente , donde los triángulos corresponden a las posiciones del dispositivo emisor, los puntos a las estimaciones del algoritmo y las lineas a las distancias entre la posición del emisor y las estimaciones.

Error	Valor
Error mediano	9.63 m
Error Medio	55.39 m
Máximo Error	436.94 m

Tabla 7.6: Resultados de simulación con dispositivos en movimiento y precisión de 0.01 microsegundos.

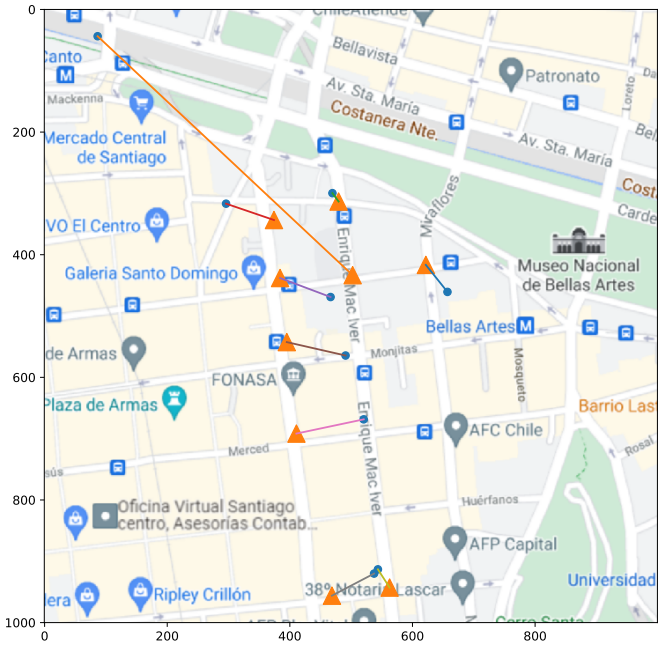


Figura 7.9: Resultados de simulación con dispositivos en movimiento y una precisión de 0.01 microsegundos. La marca triangular es el emisor, y las marcas con o son las estimaciones.

7.4.1. Comentarios

Como se puede observar en los resultados mostrados, al aumentar la precisión temporal del dispositivo, disminuye el error en la distancia del punto estimado comparado con la ubicación real del dispositivo. Esto se debe a las características intrínsecas del algoritmo utilizado, ya que se usan diferencias temporales para estimar curvas. Además, según los datos mostrados en la Tabla 7.7, al aumentar la precisión de 1 microsegundo a 0.1 microsegundos, el error medio disminuye a aproximadamente en un 60 %, y el error máximo disminuye en un 17 %. Finalmente al aumentar la precisión desde 1 microsegundo a 0.01 microsegundos, el error medio disminuye en aproximadamente un 83 %, y el error máximo disminuye en aproximadamente un 36 %.

Precisión	Error mediano	Error Medio	Máximo Error
1 μs	290.55	327.39	687.49
0.1 μs	82.56	126.37	569.98
0.01 μs	9.63	55.39	436.94

Tabla 7.7: Tabla comparativa de errores en las 3 revisiones realizadas.

7.5. Problemas y Desafíos

Este apartado está dedicado a mencionar los problemas que se visualizaron en las pruebas y comparativas mostradas anteriormente, los cuales fueron detectados a partir de los resultados obtenidos. A continuación se listan los problemas para proponer desafíos que permitan solucionarlos o mejorarlos.

7.5.1. Problemas

Precisión del algoritmo

Debido a los resultados mencionados en la sección anterior, se deben explorar alternativas para mejorar la precisión del algoritmo, ya que la sincronización implementada está en el orden de los microsegundos, las diferencias de posición son bastante grandes. Si bien, el dispositivo se puede utilizar para estimar posiciones, el error medio es bastante alto, y pueden darse casos donde la ubicación estimada y la ubicación real difieran aún más. Esto considerando que ya se

agrega un error aproximado de 33 metros en la posición de los dispositivos cuando se encuentran en movimiento, se añade otro error de aproximadamente 300 metros solamente con una diferencia de 1 microsegundo. Por lo tanto se debe afrontar esta problemática para mejorar los resultados que se presentaron en los apartados anteriores.

Modelo de propagación

Al inicio del proyecto se decidió utilizar un algoritmo de ubicación basado únicamente en TDoA debido a que utilizar un algoritmo híbrido que incluya modelos de propagación o alguna otra alternativa que permita obtener más información para estimar ubicaciones agrega una capa extra de desarrollo y procesamiento. Si bien esto es cierto, al no tener un modelo de propagación, u otro tipo de estimación incluido en el modelo, los errores ocasionados por la precisión del algoritmo son más evidentes. Agregar esta capa de desarrollo puede introducir una mejora en las predicciones de ubicación, la que en conjunto con una mejora de precisión temporal, permitiría que los resultados mejoren considerablemente.

7.5.2. Desafíos

Para mejorar la precisión del reloj interno de cada dispositivo utilizando el mismo Hardware seleccionado, se pueden explorar distintas alternativas. Una de ellas podría ser cambiar las configuraciones por defecto de los relojes que introduce el Framework utilizado. Este cambio conlleva revisar la algoritmia de las mediciones temporales ya implementadas, pero realizando este cambio se podría llegar a obtener precisiones del orden de los 10ns, las cuales ya se simularon mostrando mejores resultados en cuanto a la situación actual. Otra forma de afrontar el problema de sincronización sería cambiar el Framework de programación a uno que permita mayor libertad en cuanto a las configuraciones. Esto implicaría rehacer la mayor parte del código ya diseñado para que funcione correctamente con el nuevo Framework.
Además, como se mencionó en el apartado anterior, agregar una capa de extra de estimaciones sería una buena opción para mejorar aún más la precisión del algoritmo, pero debido a que estas implementación requieren de un desarrollo que podría llevar más tiempo y pruebas,se debe pensar que es una opción nn el largo plazo.

Conclusiones

A lo largo del documento se planteó el desarrollo de un dispositivo de pruebas que permitiría la localización de otros dispositivos que se encuentren sin su información de ubicación disponible mediante la tecnología LoRa. La selección de la tecnología y de las herramientas de programación surgieron de una etapa de investigación en cuanto a soluciones existentes y programas que permitieran facilitar el uso del Hardware escogido. La etapa investigativa de las herramientas y dispositivos a utilizar permitió aprender sobre distintas opciones, con distintos precios para el mismo propósito, esto permitió ampliar la gama de opciones a seleccionar cuando se quiera afrontar otro proyecto de éste tipo. Además, durante el proceso de selección de herramientas de Software se aprendió como utilizar de forma sencilla las herramientas entregadas por AWS para montar un servidor básico con las conexiones necesarias para guardar datos en una base de datos. Con lo anterior, se exploró además un área nueva que permitió ampliar el abanico de conocimientos sobre el almacenamiento de los datos en la web.

Durante la etapa de pruebas, se hizo bastante llevar a cabo la prueba sencilla presentada, ya que debido a la contingencia nacional, no se contaba con la ayuda de las personas necesarias para realizar este tipo de ensayos. Si bien se pudo hacer en un entorno controlado, se esperaba probar el dispositivos en distintas situaciones. Durante el proceso de simulación se evidenció que la solución planteada permite estimar ubicaciones, pero que los errores de calculo en estas estimaciones son del orden de los 300 a 600 metros, por lo que se deben realizar cambios que permitan aumentar la precisión temporal del dispositivo diseñado para disminuir los errores por tiempo innatos del algoritmo TDoA. Además, se plantean distintas mejoras que podrían llevarse a cabo para compensar la falta de precisión existente en el desarrollo actual. Si bien los errores obtenidos utilizando una precisión de 1 microsegundo simulan el estado actual del dispositivo desarrollado, estos errores alcanzan los 687 metros, y este número se puede dismi-

nuir aumentando la precisión del reloj interno del dispositivo. Los resultados muestran que al cambiar la precisión desde $1\mu s$ a $0{,}1\mu s$ el error disminuye desde los 687 metros a los 569 metros como máximo, y con errores medios del orden de los 327 metros a 127 metros, es decir, la diferencia se reduce a menos de la mitad solamente mejorando el reloj interno. Y si se cambia la precisión desde $1\mu s$ a $0{,}01\mu s$, el error máximo disminuye a los 436 metros, y el error medio disminuye a los 55 metros, lo que muestra una gran mejora. Finalmente, como se planteó en los capítulos anteriores, con las características del dispositivo seleccionado, este aumento de precisión es factible pero se plantea como un desarrollo futuro, debido a que conlleva cambios que provocarían que el Framework utilizado tenga inconsistencias a nivel bajo, las cuales deben ser reparadas manualmente si se cambian las configuraciones del reloj interno del MCU.

Referencias

[1] Albert Bel Pereira. *A Pragmatic Approach of Localization and Tracking Algorithms in Wireless Sensor Networks*. Jul. de 2012.

[2] Jagedeesha S Ravindra S. *TIME OF ARRIVAL BASED LOCALIZATION IN WIRELESS SENSOR NETWORKS: A LINEAR APPROACH*. Ago. de 2013.

[3] D. Bissett. «Analysing TDoA Localisation in LoRa Networks». En: 2018.

[4] Michael Kisangiri Michael S. Mollel. *Comparison of Empirical Propagation Path Loss Models for Mobile Communication*. 2014.

[5] Alan Bensky. *Short-range Wireless Communication(Third Edition)*. 2019.

[6] Conrad Heatwole. Robert (Bobby) Grisso Mark Alley. *Precision Farming Tools: Global Positioning System (GPS)*. 2009.

[7] INTERTANKO. *Jamming and Spoofing of Global Navigation Satellite Systems (GNSS)*. 2019.

[8] Frederik Schrooyen. *Real Time Location System over WiFi in a Healthcare Environment.* 2006.

[9] Ahmet Ibrahim y Dogan Ibrahim. «Real-time GPS based outdoor WiFi localization system with map display». En: *Advances in Engineering Software* (2010). ISSN: 0965-9978. DOI: https://doi.org/10.1016/j.advengsoft.2010.06.005. URL: https://www.sciencedirect.com/science/article/pii/S0965997810000657.

[10] Jun Luo Jin Wang Nicholas Tan. *WOLoc: WiFi-only Outdoor Localization Using Crowdsensed Hotspot Labels*. Ene. de 2017.

[11] WiGLE. *"WiGLE: Wireless Network Mapping*. URL: https://wigle.net/. (accedido: 30-06-2021).

[12] OpenBMap. *OpenBMap Project*. URL: https://radiocells.org/. (accedido: 30-06-2021).

[13] Skyhook. *Skyhook Precision Location*. URL: http://www.skyhookwireless.com/products/precision-location. (accedido: 30-06-2021).

[14] Uzair Khaleeq uz Zaman Muhammad Irfan Aziz Thomas Owens. *RSSI Based Localization of Bluetooth Devices Using Trilateration: An Improved Method for the Visually Impaired*. Jun. de 2018.

[15] Ramon F. Brena Carlos E. Galvan-Tejada Jose C. Carrasco-Jiménez. *Bluetooth-WiFi based combined positioning algorithm, implementation and experimental evaluation.* 2013.

[16] Hussein Kwasme y Sabit Ekin. «RSSI-Based Localization Using LoRaWAN Technology». En: *IEEE Access* 7 (2019), págs. 99856-99866. DOI: 10.1109/ACCESS.2019.2929212.

[17] Wongeun Choi y col. «Low-Power LoRa Signal-Based Outdoor Positioning Using Fingerprint Algorithm». En: *ISPRS Int. J. Geo Inf.* 7 (2018), pág. 440.

[18] Michiel Aernouts Nico Podevijn Jens Trogh. *LoRaWAN Geo-Tracking Using Map Matching and Compass Sensor Fusion*. Abr. de 2018.

[19] Jens Trogh Nico Podevijn David Plets. *TDoA-Based Outdoor Positioning with Tracking Algorithm in a Public LoRa Network*. Mayo de 2018.

[20] David Plets y col. «Analysis of propagation of actual DVB-H signal in a suburban environment». En: *2007 IEEE Antennas and Propagation Society International Symposium*. 2007, págs. 1997-2000. DOI: 10.1109/APS.2007.4395915.

[21] Clarinox Technologies Pty Ltd. *Real Time Location Systems*. https://www.clarinox.com/site/assets/files/1210/realtime_main.pdf. Nov. de 2009.

[22] Semtech Corporation. *AN1200.22 - Lora Modulation Basics*. http://www.semtech.com/images/datasheet/an1200.22.pdf. Mayo de 2015.

[23] MQ Support Center. *What is the Spreading Factor (SF)*. URL: https://support.machineq.com/s/article/What-is-the-Spreading-Factor-SF. (accedido: 05-07-2021).

[24] Subsecretaría de Telecomunicaciones de Chile. *Estudio relativo a la actualización del Plan General de Uso del Espectro Radioeléctrico. Pags 34, 35*. https://www.subtel.gob.cl/images/stories/articles/subtel/asocfile/6_espectro_vol2.pdf. Oct. de 2005.

[25] LoRa Alliance. *RP002-1.0.3 LoRaWAN® Regional Parameters*. Mayo de 2021. URL: https://lora-alliance.org/wp-content/uploads/2021/05/RP-2-1.0.3.pdf. (accedido: 05-07-2021).

[26] Ferran Adelantado y col. «Understanding the Limits of LoRaWAN». En: *IEEE Communications Magazine* 55.9 (2017), págs. 34-40. DOI: 10.1109/MCOM.2017.1600613.

[27] LoRaTools. *Air Time Calculator*. URL: https://loratools.nl/%5C#/airtime. (accedido: 05-07-2021).

[28] Bernat Carbonés Fargas y Martin Nordal Petersen. «GPS-free geolocation using LoRa in low-power WANs». En: *2017 Global Internet of Things Summit (GIoTS)*. 2017, págs. 1-6. DOI: 10.1109/GIOTS.2017.8016251.

[29] Jonas Danebjer y Valthor Halldórsson. «A Hybrid Approach to GPS-Free Geolocation over LoRa». En: 2018.

[30] SIMCom. *SIMCOM5320 Specifications*. URL: https://simcom.ee/modules/wcdma-hspa/sim5320/. (accedido: 30-06-2021).

[31] SIMCom. *AT Command Set SIMCOM 5320 ATC EN V2.05*. https://simcom.ee/documents/SIM5320/SIMCOM_SIM5320_ATC_EN_V2.05.pdf. 2017.

[32] SIMCom. *TCPIP Application Note for WCDMA Solution V3.6*. https://simcom.ee/documents/SIM5320/tcpip_application_note_for_wcdma_solution_v3.6.pdf. 2013.

[33] SIMCom. *SIM900 Specifications*. URL: https://simcom.ee/modules/gsm-gprs/sim900/. (accedido: 30-06-2021).

[34] SIMCom. *SIM900 AT Commands Manual*. https://simcom.ee/documents/SIM900/SIM900_AT%20Command%20Manual_V1.11.pdf. 2015.

[35] SIMCom. *SIM900 TCPIP Application Note*. https://simcom.ee/documents/SIM900/SIM900_AT%20Command%20Manual_V1.11.pdf. 2011.

[36] Ai-Thinker Inc. *A6/A7/A6C User Manual*. https://www.smart-prototyping.com/image/data/9_Modules/101756%20Wireless%20A6C/A6_A7_A6C_datasheet-EN.pdf.

[37] Ai-Thinker Inc. *A6 Module AT command set*. http://www.alselectro.com/files/A6-AT-Commands.pdf.

[38] SIMCom. *SIM800 Specifications*. https://simcom.ee/modules/gsm-gprs/sim800/.

[39] SIMCom. *SIM800 Series AT Commands Manual*. https://simcom.ee/documents/SIM800/SIM800%20Series_AT%20Command%20Manual_V1.10.pdf. 2016.

[40] SIMCom. *SIM800 Series TCPIP Application Note*. https://simcom.ee/documents/SIM800x/SIM800%20Series_TCPIP_Application%20Note_V1.02.pdf. 2016.

[41] Telit. *Jupiter SE868-AS Product Description*. https://www.telit.com/wp-content/uploads/2017/09/Telit_Jupiter_SE868-AS_Datasheet.pdf. 2019.

[42] Inc. SiRF Technology. *NMEA Reference Manual*. Dic. de 2007.

[43] uBlox. *Product Summary UBX-M8030*. https://www.u-blox.com/en/docs/UBX-15029937. 2019.

[44] Ublox. *NEO-6 u-blox 6 GPS Modules*. https://www.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_(GPS.G6-HW-09005).pdf. Dic. de 2011.

[45] MCI Electronics. *MCI LoRaBee 915MHz*. https://www.mcielectronics.cl/shop/product/mci-lorabee-915mhz-modulo-de-comunicacion-inalambrica-lora-con-formato-xbee-26322?category=256.

[46] Microchip Technology Inc. *RN2903*. https://ww1.microchip.com/downloads/en/DeviceDoc/RN2903-Data-Sheet-DS50002390J.pdf. Mayo de 2018.

[47] Semtech Corporation. *SX1276/77/78/79 Datasheet*. https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R0000001Rc1/QnUuV9TviODKUgt_rpBlPz.EZA_PNK7Rpi8HA5..Sbo. Mayo de 2020.

[48] Microchip Technology Inc. *ATmega328P Datasheet*. https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. Sep. de 2020.

[49] Microchip Technology Inc. *ATmega640/V-1280/V-1281/V-2560/V-2561/V*. https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf. Mayo de 2020.

[50] STMicroelectronics. *STM32F405xx STM32F407xx*. https://www.st.com/resource/en/datasheet/stm32f407ve.pdf. Ago. de 2020.

[51] STMicroelectronics. *STM32F411xC STM32F411xE*. https://www.st.com/resource/en/datasheet/stm32f411ce.pdf. Dic. de 2017.

[52] WeAct. *WeAct MiniF4 Schematic*. https://stm32-base.org/assets/pdf/boards/original-schematic-STM32F411CEU6_WeAct_Black_Pill_V2.0.pdf.

[53] PlatformIO. *What is PlatformIO*. https://docs.platformio.org/en/latest/what-is-platformio.html.

[54] Samet Yılmaz. Cristian Steib Matheus Marabesi. *Library SIM800l Module for Arduino UNO*. https://github.com/cristiansteib/Sim800l. Abr. de 2016.

[55] Mikañ Hart. *TinyGPS, A compact Arduino NMEA (GPS) parsing library*. https://github.com/mikalhart/TinyGPS. Ago. de 2013.

[56] Sandeep Mistry. *Arduino LoRa, An Arduino library for sending and receiving data using LoRa radios*. https://github.com/sandeepmistry/arduino-LoRa. Ago. de 2016.

[57] Jan Gromeš. *RadioLib, Universal wireless communication library for Arduino*. https://github.com/jgromes/RadioLib. Mar. de 2019.

[58] Amazon Web Services. *Servicios de bases de datos*. URL: https://aws.amazon.com/es/products/databases/. (accedido: 22.07.2021).

Anexos

Anexo 1: Chirp Spread Spectrum

En las comunicaciones digitales, el espectro ensanchado Chirp (CSSs) es una técnica de espectro ensanchado que utiliza pulsos Chirp lineales de banda ancha modulados en frecuencia para codificar la información. Un Chirp es una señal sinusoidal cuya frecuencia aumenta o disminuye con el tiempo (a menudo con una expresión polinómica para la relación entre tiempo y frecuencia). La representación matemática de una señal Chirp es la siguiente:

$$x(t) = sin(\phi(t)) \tag{7.1}$$

Donde $\phi(t)$ es una función dependiente de la frecuencia. En la Figura 7.10 se muestra una señal sinusoidal Chirp de modulación lineal.

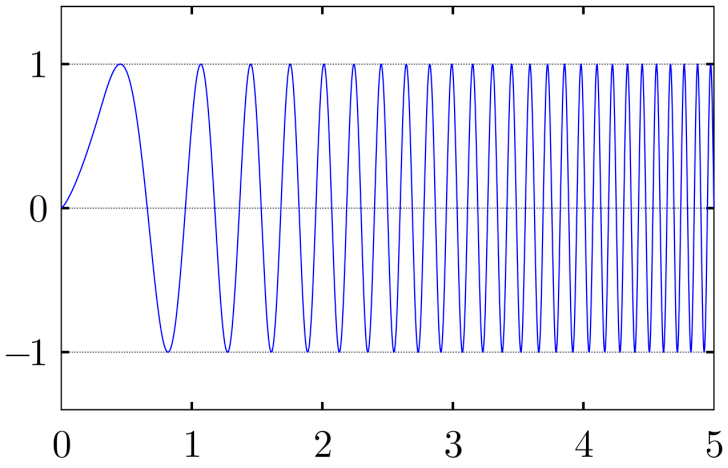


Figura 7.10: Señal sinusoidal Chirp lineal

Normalmente una señal de este tipo se mueve entre dos frecuencias límite, las que están definidas por el ancho de banda de modulación, por ejemplo, en la Figura 7.11 se muestra el espectrograma de una señal Chirp lineal, que se mueve entre las frecuencias de 0 a 2KHz y repite este proceso cada 2 seg aproximadamente.

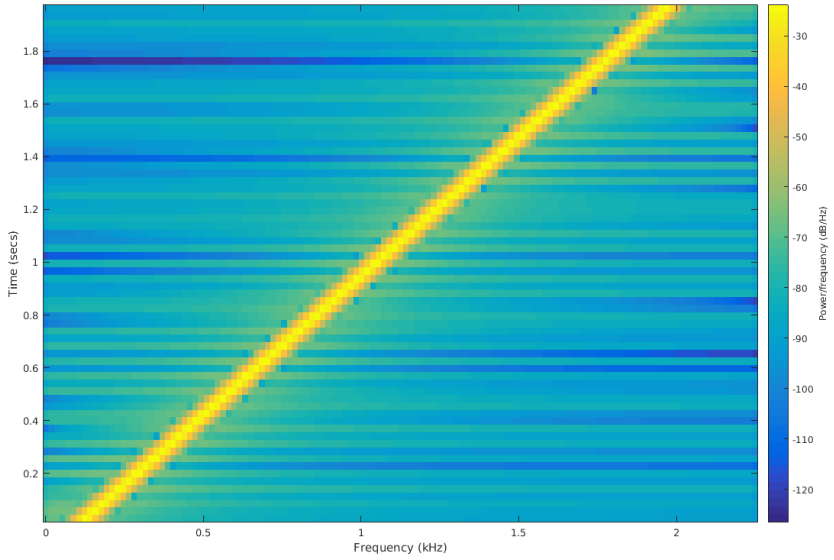


Figura 7.11: Espectrograma de una señal Chirp Lineal

Anexo 2: Archivos de configuración FloRa

En esta sección se disponene los distintos archivos de configuración utilizados en OMNET++ para simular un entorno con distintos dispositivos en movimiento durante 2 envíos de datos.

Archivo .NED

```
package loranetwork.simulations;

import inet.examples.ethernet.lans.cable;
import inet.examples.inet.netperfmeter.coreChannel;
import inet.examples.inet.netperfmeter.accessChannel;
import inet.node.ethernet.Eth100M;
import ned.IdealChannel;
import inet.node.ethernet.Eth1G;
import inet.node.ethernet.Eth40G;
import inet.examples.mobileipv6.fiberline;
import inet.examples.inet.ipv4hook.MyHost;
import inet.examples.inet.ipv4hook.MyRouter;
```

```
import inet.examples.inet.tcptimestamps.NormalPath;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.networklayer.internetcloud.InternetCloudTaggerChannel;
import inet.networklayer.ipv4.IPv4;
import inet.node.gpsr.GPSRRouter;
import inet.node.internetcloud.InternetCloud;
import inet.physicallayer.idealradio.IdealRadioMedium;
import loranetwork.LoRaPhy.LoRaMedium;
import loranetwork.LoraNode.LoRaGW;
import loranetwork.LoraNode.LoRaNode;
import ned.DatarateChannel;


@license(LGPL);
//
// TODO documentation
//
network TesisNetwork
{
    @display("bgb=1000,1000;bgu=m;bgi=background/1km2,s;bgl=3");
    types:
        channel Aire extends IdealChannel
        {
        }

    submodules:
        loRaGW3: LoRaGW {
            @display("p=150.59999,975.888");
        }
        loRaGW1: LoRaGW {
            @display("p=208.832,116.464;is=n");
        }
        loRaGW2: LoRaGW {
            @display("p=54.216,736.936");
        }
        loRaNode: LoRaNode {
            @display("is=n;p=630.51196,465.856");
        }
        myRouter: MyRouter {
            @display("p=497.98398,48.191998");
        }
        networkServer: MyHost {
            @display("p=720.87195,48.191998");
        }
        LoRaMedium: LoRaMedium {
            @display("p=949.784,48.191998");
        }
        internetCloud: InternetCloud {
            @display("p=317.26398,48.191998");
        }
        iPv4NetworkConfigurator: IPv4NetworkConfigurator {
            @display("p=939.74396,172.68799");
        }
        myRouter1: MyRouter {
            @display("p=114.45599,48.191998");
        }
        myRouter2: MyRouter {
            @display("p=58.232,178.71199");
        }
        myRouter3: MyRouter {
            @display("p=142.568,297.184");
        }
```

```
    connections:
        networkServer.ethg++ <--> Eth40G <--> myRouter.ethg++;
        internetCloud.pppg++ <--> fiberline <--> myRouter.pppg++;
        loRaGW1.ethg++ <--> Eth100M <--> myRouter1.ethg++;
        myRouter1.pppg++ <--> Eth100M <--> internetCloud.pppg++;
        loRaGW2.ethg++ <--> Eth100M <--> myRouter2.ethg++;
        myRouter2.pppg++ <--> Eth100M <--> internetCloud.pppg++;
        loRaGW3.ethg++ <--> Eth100M <--> myRouter3.ethg++;
        myRouter3.pppg++ <--> Eth100M <--> internetCloud.pppg++;
}
```

Archivo .INI

```
[General]
network = loranetwork.simulations.TesisNetwork
sim-time-limit = 2.2min
simtime-resolution = ps

**.loRaNode.LoRaNic.radio.energyConsumerType = "LoRaEnergyConsumer"
**.loRaNode.**.energySourceModule = "IdealEpEnergyStorage"
**.loRaNode.LoRaNic.radio.energyConsumer.configFile = xmldoc("energyConsumptionParameters.xml")
**.loRaNode.**.initFromDisplayString = false
**.loRaNode.**.evaluateADRinNode = false
**.loRaNode.**initialLoRaSF = 9
**.loRaNode.**initialLoRaBW = 125 kHz
**.loRaNode.**initialLoRaCR = 4
**.loRaNode.**initialLoRaTP = (2dBm + 3dBm*intuniform(0, 4))
**.loRaNode.**numberOfPacketsToSend = 0
**.loRaNode.**timeToNextPacket = 60s
**.loRaNode.mobility.turtleScript = xmldoc("loraNode.xml")



**.loRaGW1.numUdpApps = 1
**.loRaGW1.packetForwarder.localPort = 2000
**.loRaGW1.packetForwarder.destPort = 1000
**.loRaGW1.packetForwarder.destAddresses = "networkServer"
**.loraGW1.networkLayer.configurator.networkConfiguratorModule=""
**.loRaGW1.mobility.turtleScript = xmldoc("loraGW1.xml")
**.loRaGW1.packetForwarder.indexNumber = 0

**.loRaGW2.numUdpApps = 1
**.loRaGW2.packetForwarder.localPort = 2000
**.loRaGW2.packetForwarder.destPort = 1000
**.loRaGW2.packetForwarder.destAddresses = "networkServer"
**.loRaGW2.networkLayer.configurator.networkConfiguratorModule=""
**.loRaGW2.mobility.turtleScript = xmldoc("loraGW2.xml")
**.loRaGW2.packetForwarder.indexNumber = 1

**.loRaGW3.numUdpApps = 1
**.loRaGW3.packetForwarder.localPort = 2000
**.loRaGW3.packetForwarder.destPort = 1000
**.loRaGW3.packetForwarder.destAddresses = "networkServer"
**.loraGW3.networkLayer.configurator.networkConfiguratorModule=""
**.loRaGW3.mobility.turtleScript = xmldoc("loraGW3.xml")
**.loRaGW3.packetForwarder.indexNumber = 2
**.networkServer.numUdpApps = 1
**.networkServer.**.evaluateADRinServer = false
**.networkServer.udpApp[0].typename = "NetworkServerApp"
```

```
**.networkServer.udpApp[0].destPort = 2000
**.networkServer.udpApp[0].localPort = 1000
**.networkServer.udpApp[0].adrMethod = ${"avg"}

**.sigma = 0
**.constraintAreaMinX = 0m
**.constraintAreaMinY = 0m
**.constraintAreaMinZ = 0m
**.constraintAreaMaxX = 1500m
**.constraintAreaMaxY = 1500m
**.constraintAreaMaxZ = 0m


**.delayer.config = xmldoc("cloudDelays.xml")
**.radio.radioMediumModule = "LoRaMedium"
**.LoRaMedium.pathLossType = "LoRaLogNormalShadowing"
**.minInterferenceTime = 0s
**.displayAddresses = false
```

Archivo cloudDelays.xml

```
<internetCloud symmetric="true">
  <parameters name="good">
    <traffic src="**" dest="**" delay="10ms" datarate="1Gbps" drop="uniform(0,1)&lt;0" />
  </parameters>
</internetCloud>
```

Archivo energyConsumptionParameters.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
        <supplyVoltage value="3.3"/>
        <receiverReceivingSupplyCurrent value="9.7"/>
        <receiverBusySupplyCurrent value="9.7"/>
        <idleSupplyCurrent value="0.0001"/>
        <txSupplyCurrents>
                <txSupplyCurrent txPower="2" supplyCurrent="24"/>
                <txSupplyCurrent txPower="3" supplyCurrent="24"/>
                <txSupplyCurrent txPower="4" supplyCurrent="24"/>
                <txSupplyCurrent txPower="5" supplyCurrent="25"/>
                <txSupplyCurrent txPower="6" supplyCurrent="25"/>
                <txSupplyCurrent txPower="7" supplyCurrent="25"/>
                <txSupplyCurrent txPower="8" supplyCurrent="25"/>
                <txSupplyCurrent txPower="9" supplyCurrent="26"/>
                <txSupplyCurrent txPower="10" supplyCurrent="31"/>
                <txSupplyCurrent txPower="11" supplyCurrent="32"/>
                <txSupplyCurrent txPower="12" supplyCurrent="34"/>
                <txSupplyCurrent txPower="13" supplyCurrent="35"/>
                <txSupplyCurrent txPower="14" supplyCurrent="44"/>
        </txSupplyCurrents>
</root>
```

Archivo loraGW1.xml

```
<movement>
        <set speed="5" angle="196" x="917.65594" y="317.26398"/>
        <forward t="52"/>
```

```
        <set speed=”10” angle=”285”/>
        <forward t=”18”/>
        <set speed=”5” angle=”22”/>
        <forward t=”51”/>
        <set speed=”5” angle=”108”/>
        <forward t=”70”/>
</movement>
```

Archivo loraGW2.xml

```
<movement>
    <set speed=”8” angle=”352” x=”46.184” y=”736.936”/>
        <forward t=”45”/>
        <set speed=”5” angle=”84”/>
        <forward t=”53”/>
        <set speed=”5” angle=”352”/>
        <forward t=”53”/>
        <set speed=”5” angle=”263”/>
        <forward t=”100”/>
</movement>
```

Archivo loraGW3.xml

```
<movement>
    <set speed=”5” angle=”353” x=”150.59999” y=”975.888”/>
        <forward t=”84”/>
        <set speed=”5” angle=”260”/>
        <forward t=”51”/>
        <set speed=”5” angle=”353”/>
        <forward t=”67”/>
        <set speed=”0.01”  angle=”275”/>
        <forward t=”15”/>
        <set speed=”5”/>
        <forward t=”20”/>
        <set speed=”0.01”  angle=”300”/>
        <forward t=”25”/>
</movement>
```

*

Archivo loraNode.xml

```
<movement>
        <set speed=”5” angle=”260” x=”630.51196” y=”465.856”/>
        <forward d=”50”/>
        <set speed=”5” angle=”172”/>
        <forward t=”24”/>
        <set speed=”0.1”/>
        <forward t=”10” />
        <set speed=”5” angle=”264”/>
        <forward t=”24”/>
        <set speed=”5” angle=”185”/>
        <forward t=”24”/>
        <set speed=”5” angle=”84”/>
        <forward t=”27”/>
        <set speed=”0.01”/>
        <forward t=”20”/>
```

```
        <set speed="5" angle="84"/>
        <forward t="80"/>
</movement>
```

Anexo 3: Script de Python

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from scipy.optimize import fsolve
import math
import statistics
import sys
c = 299792458 # m/s  speed of light
image = Image.open('1km2.png')
x_ = []
y_ = []
lnx_ = []
lny_ = []
gwx = []
gwy = []
distances = []
moduleNames = ['loRaGW1','loRaGW2','loRaGW3','loRaNode']
filterReceiver = '.LoRaGWNic.radio:□LoRaGWRadio□Reception□ended:'
filterTransmiter = '.LoRaNic.radio:□Transmission□ended:'
receiverDataTypes = ['power','carrierFrequency','bandwidth','startTime',
                     'endTime','startPosition','endPosition','startOrientation','endOrientation']
transmitterDataTypes = ['startTime','endTime','startPosition','endPosition']

def myFunction(z,d):
    x = z[0]
    y = z[1]

    F = np.empty((2))
    F[0] = math.sqrt((d[1][1]-x)**2 + (d[1][2]-y)**2) -
    math.sqrt((d[0][1]-x)**2 + (d[0][2]-y)**2) - c*(d[1][0] - d[0][0])
    F[1] = math.sqrt((d[2][1]-x)**2 + (d[2][2]-y)**2) -
    math.sqrt((d[0][1]-x)**2 + (d[0][2]-y)**2) - c*(d[2][0] - d[0][0])
    return F


def detectSource(line,filter):
  d = line.split(filter)[0].split('.')
  return d[len(d)-1]

def parseType(line, str, type):
  p = line.split(str)
  if type == 0:
    p = p[1].split(',')[0]
    pd = p[3:len(p)].split('□')
    return {'value': float(pd[0]),'unit':pd[1]}
  elif type == 1:
    p = p[1].split(',')[0]
    pd = p[3:len(p)].split('□')
    return {'value': float(pd[0]),'unit':'s'}
  elif type == 2:
    p = p[1].split(')')[0]
```

```
    pd = p[4:len(p)].split(',')
    return {'x': float(pd[0]), 'y': float(pd[1]),'z':float(pd[2])}

def parseReceiverPacket(line):
  power = parseType(line, receiverDataTypes[0],0)
  frequency = parseType(line,receiverDataTypes[1],0)
  bandwidth = parseType(line,receiverDataTypes[2],0)
  startTime =  parseType(line,receiverDataTypes[3],1)
  endTime = parseType(line,receiverDataTypes[4],1)
  startPosition = parseType(line, receiverDataTypes[5],2)
  endPosition = parseType(line, receiverDataTypes[6],2)
  return {'power':power, 'frequency': frequency,
          'bandwidth': bandwidth,'startTime':startTime,'endTime':endTime,
          'startPosition':startPosition,'endPosition':endPosition}
def parseTransmitterPacket(line):
  startTime = parseType(line,transmitterDataTypes[0],1)
  endTime = parseType(line,transmitterDataTypes[1],1)
  startPosition = parseType(line,transmitterDataTypes[2],2)
  endPosition = parseType(line,transmitterDataTypes[3],2)
  return {'startTime':startTime, 'endTime': endTime, 'startPosition': startPosition,'endPosition':endPosition}

lines = []
data = {moduleNames[0]:[],moduleNames[1]:[],moduleNames[2]:[],moduleNames[3]:[]}
with open(sys.argv[1]) as f:
    lines = f.readlines()

count = 0
for line in lines:
  if(filterReceiver in line):
    source = detectSource(line,filterReceiver)
    d = parseReceiverPacket(line)
    data[source].append(d)
  elif filterTransmiter in line:
    source = detectSource(line,filterTransmiter)
    t = parseTransmitterPacket(line)
    data[source].append(t)

for i in range(len(data['loRaNode'])):
  dataset = []
  g1 = data['loRaGW1'][i]
  g2 = data['loRaGW2'][i]
  g3 = data['loRaGW3'][i]
  ln = data['loRaNode'][i]
  dataset = [g1,g2,g3]
  mintime = 1000000000
  mink = 0
  d = [[],[],[]]
  for k in range(len(dataset)):
    if(dataset[k]['endTime']['value'] < mintime):
      mink = k
  sum = 1
  for k in range(len(dataset)):
    if(float(sys.argv[4]) > 0.0):
      mu, sigma =  0, float(sys.argv[4])
      error =  np.random.normal(mu,sigma,3)
    else:
      mu, sigma = 0, 0.0000018
      error =  np.random.normal(mu,sigma,3)

    gwx.append(dataset[k]['endPosition']['x'])
    gwy.append(dataset[k]['endPosition']['y'])
    if(k != mink):
```

```
        if int(sys.argv[3]) == 1:
          d[sum].append(round(dataset[k]['endTime']['value'] + error[k],int(sys.argv[2])))
        else:
          d[sum].append(round(dataset[k]['endTime']['value'],int(sys.argv[2])))

        d[sum].append(dataset[k]['endPosition']['x'])
        d[sum].append(dataset[k]['endPosition']['y'])
        sum = sum + 1
      else:
        if int(sys.argv[3] == 1):
          d[0].append(round(dataset[k]['endTime']['value'] + error[k],int(sys.argv[2])))
        else:
          d[0].append(round(dataset[k]['endTime']['value'],int(sys.argv[2])))
        #

        d[0].append(dataset[k]['endPosition']['x'])
        d[0].append(dataset[k]['endPosition']['y'])

    print(d)
    zGuess = np.array([1,1])
    z = fsolve(myFunction,zGuess,d)

    dist1 = math.dist([d[0][1], d[0][2]],z)
    diff = math.dist([ln['startPosition']['x'],ln['startPosition']['y']],z)
    if dist1 <= 1000:
      x_.append(z[0])
      y_.append(z[1])
      lnx_.append(ln['startPosition']['x'])
      lny_.append(ln['startPosition']['y'])
      distances.append(diff)

fig, ax = plt.subplots()
fig = plt.imshow(image, origin='lower',cmap='Greys_r')
ax.set_ylim(1000,0)
ax.scatter(x_,y_)
ax.scatter(lnx_,lny_,marker="^",s=200)
for i in range(len(lnx_)):
  ax.plot([x_[i],lnx_[i]],[y_[i],lny_[i]])
if(len(distances) > 0):
  print(distances) # all distances
  print(np.percentile(distances,50)) # median error
  print(np.mean(distances)) # mean error
  print(np.max(distances)) # max error
plt.show()
f.close()
```