

UNIVERSIDAD NACIONAL DE GENERAL SARMIENTO

TESINA DE LA LICENCIATURA DE SISTEMAS

Rastreo de dispositivos móviles mediante el uso de paquetes de control 802.11 con dispositivos de baja potencia

Autor:
Agustín C. Alexander

Supervisor:
Alexis Tcach Lufrano

*La tesina se presenta para cumplir con los requerimientos de grado de
Licenciado en Sistemas
en el*

Instituto de Industria (IDEI)

24 de junio de 2025

Cuando se empieza a hablar en términos absolutos en la política, significa que las atrocidades están a punto de comenzar. La humanidad ha logrado cosas asombrosas simplemente avanzando a través de discusiones, quejas, luchas y negociaciones. Es caótico y poco digno, pero es cuando estamos en nuestra mejor forma, porque todos tienen la oportunidad de hacer competir su voz. Incluso si otros intentan constantemente callarla. Cuando solo una voz tiene la posibilidad de ser expresada, lo que suele surgir de eso puede ser algo terrible.

Persepolis Rising - James S.A. Corey, 2017

Resumen

Existen múltiples organizaciones, como centros comerciales, que proveen Wi-Fi gratuito como incentivo para luego, aprovechando esa conexión, poder determinar el comportamiento de sus clientes; que negocios prefieren, cuánto tiempo pasan en el establecimiento y hasta qué gondolas les interesa más. Esto requiere sin embargo de que los individuos decidan conectarse a dicho Wi-Fi[12].

Nuestro trabajo consiste en investigar si existe la posibilidad de deducir el recorrido de transeúntes y vehículos usando un método similar sin necesidad de una conexión previa realizada por el usuario. Mas precisamente nos planteamos ver cuán factible es, utilizando dispositivos de bajo costo, estimar la posición y recorrido de un teléfono móvil sin necesidad de que este se encuentre conectado a una red Wi-Fi, es decir, de manera pasiva.

En este trabajo, describimos la implementación de un sistema de seguimiento que utiliza el RSSI (Received Signal Strength Indication) recibido a partir de sensores ESP8266, para inferir información de ubicación de un dispositivo móvil mediante paquetes de control 802.11. El sistema utiliza una red de puntos de referencia fijos, cada uno equipado con un sensor ESP8266, para medir la intensidad de las señales recibidas del dispositivo móvil y calcular su ubicación utilizando algoritmos de multilateración. El bajo coste y la amplia disponibilidad de dichos sensores, así como su diseño de bajo consumo, los hacen muy adecuados para el despliegue masivo de una red de sensores urbana en particular para su uso en aplicaciones de rastreo de localización.

Como resultado, se logró desarrollar un sistema funcional de localización pasiva con pruebas exitosas en escenarios reales y simulados, junto con diversas herramientas de software publicadas. Incluyendo bibliotecas de multilateración, firmware personalizado, y entornos de simulación. Estas ayudan a demostrar la viabilidad técnica de este enfoque para aplicaciones urbanas en el contexto de *Smart Cities* y en particular el monitoreo del flujo urbano.

Agradecimientos

Muchas gracias a Alexis Tcach y a Ivan Jourdan, quienes sin sus ideas y su guia este trabajo jamas hubiera sido posible. A Pedro Gutierrez, Sebastian Pintos, Bruno Capecce y Fernando Molachino por ayuda durante la fase experimental de este proyecto. Y finalmente a todos los que me incitaron a terminar este trabajo y me empujaron a mas....

Listado de abreviaciones

RSSI Received Signal Strength Indicator

Beacon Beacon Frame

GPS Global Positioning System

ESP8266 Espressif System's Wi-Fi Microchip

ESP32 Espressif Systems Wi-Fi + Bluetooth Microchip

BLE Bluetooth Low Energy

Wi-Fi Wireless Fidelity

GND Ground

dBm Decibels relative to one milliwatt

IOT Internet of Things

IP Internet Protocol

LPWAN Low Power Wide Area Network

LoRa Long Range

MAC Media Access Control Address

NB-IoT Narrowband IoT

NS3 Network Simulator 3

AP Access Point

AODV Ad hoc On-Demand Distance Vector Routing

PRQ Probe Request

RPI Raspberry Pi

SLSQP Sequential Least Squares Quadratic Programming

BSSID Basic Service Set Identifier

SSID Service Set Identifier

UDP User Datagram Protocol

UNGS Universidad Nacional de General Sarmiento

USB *Universal Serial Bus*

VIN *Voltage In*

WPA2 *Wi-Fi Protected Access 2*

WPA3 *Wi-Fi Protected Access 3*

Glosario

RSSI *Received Signal Strength Indicator*

Beacon *Beacon Frame*

GPS *Global Positioning System*

ESP8266 *Espressif System's Wi-Fi Microchip*

ESP32 *Espressif Systems Wi-Fi + Bluetooth Microchip*

BLE *Bluetooth Low Energy*

Wi-Fi *Wireless Fidelity*

GND *Ground*

dBm *Decibels relative to one milliwatt*

IOT *Internet of Things*

IP *Internet Protocol*

LPWAN *Low Power Wide Area Network*

LoRa *Long Range*

MAC *Media Access Control Address*

NB-IoT *Narrowband IoT*

NS3 *Network Simulator 3*

AP *Access Point*

AODV *Ad hoc On-Demand Distance Vector Routing*

PRQ *Probe Request*

RPI *Raspberry Pi*

SLSQP *Sequential Least Squares Quadratic Programming*

BSSID *Basic Service Set Identifier*

SSID *Service Set Identifier*

UDP *User Datagram Protocol*

UNGS Universidad Nacional de General Sarmiento

USB *Universal Serial Bus*

VIN *Voltage In*

WPA2 *Wi-Fi Protected Access 2*

WPA3 *Wi-Fi Protected Access 3*

Índice de figuras

2.1. Secuencia de estado de la conexión	5
2.2. Casos Trilateración	6
2.3. Script Python para ajuste de curva. Código disponible en GitHub.	9
2.4. RSSI segregado por distancia de la medición vs regresión logarítmica, prueba de campo .	10
2.5. RSSI recibido a distintas distancias	11
2.6. Medicion de RSSI original y filtrada con Kalman, prueba de campo con ESP8266 en campo de deportes UNGS	12
3.1. Visualización de la simulacion de NS3. El nodo objetivo se aleja del nodo Sniffer.	14
3.2. A la derecha, media de RSSI segregado por distancia de la medición vs regresión logarítmica, simulación usando NS3. A la izquierda, inversa al logaritmo de la distancia y distancia predicha en base a A y N calculados para la media del RSSI	16
3.3. Visualización de la simulación de NS3. El nodo objetivo realiza un recorrido mientras los otros ocho nodos Sniffer captan su señal.	17
3.4. Multilateración simulada	18
4.1. Flujo de datos en mediciones de perfilado	20
4.2. Perfilado	21
4.3. Perfilado	22
4.4. Media de RSSI segregado por distancia de la medición vs regresión logarítmica, prueba de campo	23
4.5. Flujo de datos en mediciones de multilateración	24
4.6. Experimento estático	25
4.7. Variación de la distancia con RSSI	26
4.8. Variación de la distancia con RSSI	27
4.9. Postes de Nodo Multilateración	28
4.10. Stepless	29
4.11. Flujo de datos	30
4.12. Perfilado	31
4.13. Configuración de la multilateración en Tiempo Real	32
4.14. Resultados de la multilateración en Tiempo Real	33
6.1. Herramienta gráfica de multilateración	38

Índice de cuadros

3.1. Extracto de captura simulada de perfilado (capture.csv)	15
3.2. Extracto de captura simulada que luego se usará como <i>input</i> para multilateración. (Simulation.csv)	16

Índice general

Resumen	III
Agradecimientos	V
Lista de Abreviaciones	VII
Glosario	IX
Índice de Figuras	X
Índice de Cuadros	XI
Índice General	XIII
Dedicatoria	XIV
1. Introducción	1
1.1. Motivación y Antecedentes	1
1.2. Objetivo y alcance	2
1.3. Estructura del trabajo	3
2. Revisión de Literatura	4
2.1. Conexiones inalámbricas	4
2.1.1. Tipos de conexiones inalámbricas	4
2.1.2. Estados de conexión Wi-Fi	4
2.2. Técnicas de Posicionamiento	6
2.2.1. Intensidad de la Señal WiFi	6
2.2.2. Multilateración y Trilateración	6
2.2.3. Aleatorización de MAC Addresses	7
2.3. Medición de distancia en base al modelo de <i>Path Loss</i>	8
2.4. Ajuste Logarítmico	8
2.4.1. Modelo Matemático	10
2.4.2. Implementación	11
2.5. Filtrado de Kalman	11
2.6. Cuadrados Mínimos aplicado a Multilateración	13
3. Experimento Simulado	14
3.1. Perfilado	14
3.1.1. Implementación	14
3.1.2. Experimento	15
3.1.3. Conclusión	15
3.2. Multilateración	16
3.2.1. Implementación	16
3.2.2. Experimento	17
3.3. Conclusión	17

4. Experimentos de campo	20
4.1. Perfilado preliminar con “ <i>steps</i> ”	20
4.1.1. Implementación	20
4.1.2. Experimento	21
4.1.3. Conclusión	21
4.2. Infraestructura de pruebas de campo <i>wireless</i>	22
4.3. Interferencias y Rango Usable para Multilateración con ESP8266	25
4.3.1. Experimento estático	25
4.3.2. Precisión en el perfilado	26
4.4. Perfilado sin <i>steps</i>	27
4.4.1. Experimento	28
4.4.2. Conclusiones	29
4.5. Multilateración	30
4.5.1. Infraestructura	30
4.5.2. Experimento	31
4.5.3. Conclusión	32
5. Discusiones	34
5.1. Análisis de Errores y Mejoras	34
5.1.1. Interferencia Ambiental y Variabilidad de la Señal	34
5.1.2. Desafíos en la Multilateración y la Estimación de Distancias	34
5.1.3. Limitaciones del Hardware y la Infraestructura	35
5.2. Consideraciones Finales	35
6. Conclusiones y trabajo futuro	37
6.1. Herramientas de Software Desarrolladas	37
6.2. Hallazgos Principales	38
6.3. Limitaciones y Desafíos	38
6.4. Recomendaciones para Futuras Investigaciones	39
6.5. Comentarios Finales	39
Bibliografía	41

Para mi Padre, Alfredo Carlos Alexander, quien despertó en mi desde muy temprano mi pasión por la ingeniería y gracias a quien elegí esta carrera. Y a mi madre, María Teresa Campos, quien ante cada obstáculo siempre me alentó a continuar en mis estudios.

Capítulo 1

Introducción

1.1. Motivación y Antecedentes

En el contexto de la planificación urbana y las ciudades inteligentes o *Smart Cities*, el problema de encontrar espacios de estacionamiento es algo que se encuentra en estado de plena investigación y desarrollo. Se consultó la literatura y se encontró soluciones que utilizan sensores colocados en el pavimento o numerosas cámaras más algún tipo de software que permite detección de estacionamiento. Estas soluciones, si bien pueden ser efectivas, tienen como problema el uso de hardware costoso con gran poder de cómputo, cuya instalación requiere de gran esfuerzo y recursos.

Basándose en la experiencia previa adquirida durante la colaboración con el municipio de San Miguel en la investigación y desarrollo del proyecto *Smart Cities* [2], el grupo de investigación en redes AD-HOC de la UNGS, se puso como objetivo el desarrollo de una solución de bajo costo para *Smart Parking*.

Luego de analizar el estado del arte se concluyó que existía novedad y utilidad en un sistema *wireless* de rastreo de vehículos y transeúntes. Este podría permitir a su vez en trabajos futuros identificar si un vehículo se encuentra en movimiento, detenido o ha estacionado.

Un sistema completo como el que el equipo de investigación se propone analizar, permitiría detectar individuos que posean dispositivos móviles o cualquier otro tipo de dispositivo que posea Wi-Fi o incluso BLE. Una vez identificado, el sistema podrá ser capaz de detectar su ubicación a lo largo del tiempo y finalmente predecir si dicho dispositivo se encuentra dentro de un vehículo en movimiento, o es un transeúnte caminando, o bien es un vehículo que acaba de estacionar y a continuación marcar esa plaza como ocupada o detectar una zona que posee embotellamiento. Esta información es crucial para la administración de una ciudad que intenta introducirse al paradigma de *Smart City* ya que le permitiría actuar de manera acorde ante estos eventos.

Por ejemplo, utilizando una aplicación de *Smart Parking* sugerir al usuario dónde estacionar. Lo que es más, hasta podría permitirle a la ciudad, mediante el uso de dicha aplicación, ajustar las tarifas de estacionamiento según cuán congestionada se encuentre la zona. De esta manera permitiendo ahorro de combustible a los vehículos al no tener que recorrer mucha distancia para encontrar una plaza de estacionamiento y poder descomprimir los centros urbanos en horas de alta demanda.

Para que un dispositivo cliente se conecte a una red Wi-Fi, el primer paso es identificar los puntos de acceso (AP) disponibles dentro de su rango. Este proceso utiliza dos tipos principales de tramas: los *Beacon Frame*, enviados periódicamente por los AP, y las *Probe Request*, que los clientes transmiten activamente cuando necesitan más información sobre las redes cercanas.

Los Beacon son señales que los AP emiten a intervalos regulares, generalmente cada 100 ms, para anunciar la existencia de la red. Estas tramas incluyen datos esenciales como el nombre de la red (SSID), la dirección MAC del punto de acceso (BSSID) y detalles técnicos sobre los canales, velocidades y métodos de cifrado soportados, como WPA2 o WPA3. Este método permite a los dispositivos cliente identificar redes disponibles de manera pasiva, simplemente escuchando, lo cual es eficiente en términos de energía porque no implica que el cliente envíe datos.

En contraste, las *Probe Request* son tramas activas que los dispositivos cliente utilizan cuando necesitan información más específica o están buscando una red particular. Al enviar estas solicitudes, el cliente puede recibir respuestas de los AP en el área, lo que resulta útil en escenarios como el roaming entre diferentes puntos de acceso o cuando el escaneo pasivo no es suficiente debido a interferencias o congestión.

Estos dos métodos, el escaneo pasivo mediante *Beacon Frame* y el escaneo activo con *Probe Request*, se utilizan en conjunto para garantizar la detección y conexión a redes. Los *Beacon Frame* reducen el consumo de energía al no requerir interacción activa, mientras que los *Probe Request* se adaptan a situaciones donde se requiere intervención activa. De esta manera, una vez descubierto el AP se puede proceder con los siguientes pasos de la conexión wifi.

1.2. Objetivo y alcance

El presente trabajo investiga la viabilidad de estimar la ubicación y el recorrido de dispositivos móviles de manera pasiva, sin necesidad de que estos se encuentren conectados a una red Wi-Fi. Para ello, se propone el uso de sensores de bajo costo basados en ESP8266, capaces de captar tramas de control 802.11 como los paquetes *Probe Request* y mediante el análisis del nivel de señal RSSI recibido en múltiples puntos, estimar la posición del dispositivo utilizando técnicas de multilateración. El sistema desarrollado demuestra que es posible realizar seguimiento pasivo de dispositivos móviles con precisión aceptable, lo cual abre la puerta a futuras aplicaciones de monitoreo urbano en entornos de *Smart Cities*.

Como parte de esta línea de trabajo, se desarrolló un sistema funcional de localización pasiva que permite estimar la ubicación de dispositivos móviles sin requerir conexión previa. Además del sistema mismo, se desarrollaron una serie de herramientas complementarias publicadas como software libre, incluyendo firmware personalizado para los dispositivos embebidos utilizados, bibliotecas de multilateración y entornos de simulación. Estos recursos permiten tanto la reproducción de los experimentos como su adaptación a distintos escenarios urbanos.

Si bien el objetivo final del equipo de investigación es poder determinar con certeza el recorrido que realiza un vehículo al buscar estacionamiento en una ciudad. Este trabajo se limita al objetivo investigar si existe la posibilidad de identificar el comportamiento de transeúntes sin necesidad de una conexión previa realizada por el usuario. Más precisamente nos planteamos ver cuán factible es, utilizando dispositivos de bajo costo, estimar la posición y recorrido de un teléfono móvil transportado por un individuo sin necesidad de que este se encuentre conectado, es decir, de manera pasiva. El seguimiento móvil mediante RSSI o *Received Signal Strength Indicator* y sensores ESP8266 se planteó como una solución de bajo costo para determinar la ubicación de un dispositivo móvil.

Para tener resultados consistentes y poder saber a ciencia cierta si nuestro sistema era viable y poder plantear experimentos sencillos. Se tomó la decisión de realizar la conexión con el dispositivo móvil de manera diferente a en el escenario ideal donde el dueño del dispositivo no realiza ninguna modificación en la configuración de este y aún así es posible rastrearlo. La modificación realizada, fue encender el modo AP Wi-Fi del teléfono. Esto nos permite aumentar la frecuencia de envío de paquetes de tipo *Probe Request*. Exploraremos el motivo y los mecanismos más adelante, sin embargo este mismo sistema puede funcionar sin intervención del dispositivo rastreado si se quisiera aunque con menor frecuencia de envío. A continuación, describimos la implementación de un sistema de seguimiento móvil que utiliza el RSSI recibido a partir de microcontroladores ESP8266, para inferir información de ubicación de un dispositivo móvil mediante paquetes de control 802.11. El sistema utiliza una red de puntos de referencia fijos, cada uno equipado con un ESP8266, para medir la intensidad de las señales recibidas del dispositivo móvil y calcular su ubicación utilizando algoritmos de multilateración. El bajo coste y la amplia disponibilidad de dichos sensores, así como su diseño de bajo consumo, los hacen muy adecuados para su uso en aplicaciones de seguimiento de localización. Demostraremos la eficacia del sistema de seguimiento móvil mediante experimentos que evalúan su precisión.

1.3. Estructura del trabajo

Este trabajo se organiza de la siguiente manera: En el Capítulo 1 se plantea la introducción. En el Capítulo 2, se realiza una revisión de la literatura sobre las tecnologías inalámbricas y las técnicas de posicionamiento. En el Capítulo 3 se describe un experimento simulado de perfilado y multilateración. El Capítulo 4 presenta un experimento de campo real, para el perfilado de señales Wi-Fi y multilateración. En el Capítulo 5, se discuten las fuentes de errores y las posibles mejoras. Finalmente, en el Capítulo 6 se presentan las conclusiones y las recomendaciones para futuros trabajos. Los Apéndices contienen detalles adicionales sobre los experimentos y los resultados.

Capítulo 2

Revisión de Literatura

2.1. Conexiones inalámbricas

2.1.1. Tipos de conexiones inalámbricas

Además de la conectividad Wi-Fi, existen otras tecnologías inalámbricas ampliamente utilizadas en dispositivos móviles, ordenadores personales y dispositivos Internet of Things. Entre estas, *Bluetooth Low Energy* (BLE), una variante de Bluetooth diseñada para un consumo de energía extremadamente bajo, es especialmente relevante para aplicaciones Internet of Things debido a su eficiencia energética y capacidad para operar en dispositivos alimentados por baterías durante largos períodos.

Otra tecnología inalámbrica importante es Zigbee, que utiliza el protocolo IEEE 802.15.4 para crear redes de área personal de baja potencia. Zigbee es común en aplicaciones de automatización del hogar y Internet of Things, proporcionando una solución eficaz para controlar dispositivos inteligentes dentro de un hogar o edificio.

Las redes celulares, incluyendo tecnologías como 4G LTE y 5G, también juegan un papel crucial en la conectividad inalámbrica, ofreciendo cobertura a gran escala y soporte para una amplia gama de servicios móviles e IoT. Estas redes permiten la conectividad de dispositivos móviles y Internet of Things en movimiento, soportando aplicaciones que requieren movilidad y acceso a internet de alta velocidad.

Finalmente, el *Low Power Wide Area Network* (LPWAN), incluyendo tecnologías como *Long Range* y *Narrowband IoT*, proporciona soluciones para aplicaciones Internet of Things que requieren comunicaciones a larga distancia y bajo consumo de energía. Estas redes son ideales para sensores y dispositivos Internet of Things distribuidos a lo largo de áreas extensas, como en aplicaciones de monitoreo ambiental y agricultura inteligente.

Se eligió utilizar Wi-Fi como la tecnología para este sistema de seguimiento debido a varias razones prácticas. En primer lugar, la tecnología Wi-Fi está presente en casi todos los teléfonos y muchos otros dispositivos, hasta incluso algunos autos. No hay necesidad de equipo extra o configuraciones especiales para detectar estos dispositivos, lo que lo hace muy accesible. Además, Wi-Fi tiene un alcance más largo en comparación con tecnologías como Bluetooth, lo que permite cubrir áreas más grandes sin complicaciones. Otro punto a favor es que los dispositivos suelen buscar redes Wi-Fi automáticamente, enviando solicitudes de sondeo de manera pasiva. Esto significa que se pueden rastrear sin que hagan algo activamente para ser detectados; esto convierte al Wi-Fi en una tecnología ideal para monitorear movimientos y presencia en diversos entornos.

2.1.2. Estados de conexión Wi-Fi

Todos los dispositivos móviles, ordenadores personales y dispositivos IoT que poseen conectividad Wi-Fi utilizan los mismos mecanismos de conexión. El esquema más común de topología de red es un *Access Point* y múltiples dispositivos en modo cliente conectados a este. Por ejemplo, una laptop

conectada a un router hogareño que le provee de internet.

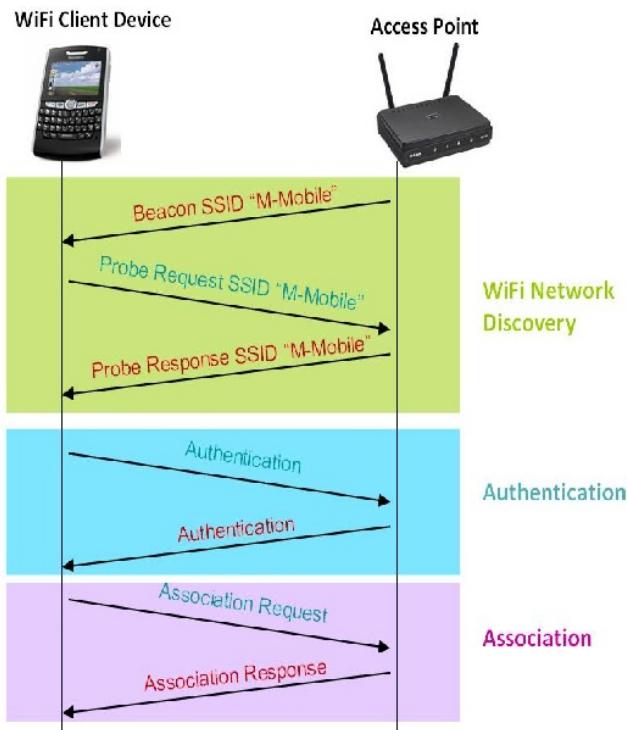


Figura 2.1: Secuencia de estado de la conexión. Los paquetes de tipo Probe-Request o Beacon pertenecen a frames de clase 1.

Si un dispositivo en modo cliente intenta conectarse a un *Access Point* primero debe encontrarse en rango. Para saber que *Access Points* están en rango para poder intentar la conexión, se utilizan dos paquetes que cumplen la misma función, pero lo realizan de maneras diferentes: Un paquete de tipo *Probe Request* y un paquete *Beacon*.

Un *Beacon* es un tipo de paquete que emiten los *Access Points* y tiene como objetivo anunciar a los clientes cercanos para que estos intenten conectarse si así lo desean. Un *Probe Request* en cambio, es un paquete emitido por un cliente para solicitar información acerca de un *Access Points* en particular o de todos los que se encuentren disponibles en un canal (banda inalámbrica para una frecuencia, ej., 2.4 GHz o 5 GHz).

Si un adaptador Wi-Fi se encuentra en modo “promiscuo” o *Sniffer*, podrá escuchar todos los paquetes y no solamente los que están dirigidos a dicho adaptador. Esto nos permitirá realizar una escucha pasiva, como mencionamos anteriormente. Por ende, un adaptador en dicho modo podrá recibir paquetes de todo tipo en particular, *Probe-Request* de clientes y *Beacon Frames*, de *Access Points* respectivamente, que se encuentren en rango.

Un cliente posee dos maneras de realizar un escaneo del área y determinar que *Access Points* se encuentran presentes: un escaneo pasivo o un escaneo activo.

Un escaneo pasivo (similar a un esquema de polling) consiste en sintonizar la antena de nuestro *Sniffer* una fracción de tiempo a la vez en distintos canales (channel hopping). En cada canal se reciben *Beacon Frames* pudiendo analizar así los *Access Points* cercanos.

Un escaneo activo, en cambio, envía *Probe Request Frames* y espera *Probe Response Frames* también usando opcionalmente *Channel Hopping*. Esta estrategia es utilizada ampliamente por dispositivos móviles donde la batería es un factor a tener en cuenta y cuanto menos tiempo se necesite estar haciendo un escaneo, ya sea activo o pasivo, del área, mejor para el ahorro de energía del dispositivo en cuestión.

Entonces, de esta manera, un dispositivo podría activamente solicitar respuesta de un *Access Point*

en particular o de cualquiera en el área en vez de esperar a recibir un *Beacon Frame*. Esto es beneficioso además, ya que asegura que ese *Access Point* está en rango efectivo, ya que este pudo enviar su *Probe Response* al cliente y este fue recibido exitosamente [1][20].

2.2. Técnicas de Posicionamiento

2.2.1. Intensidad de la Señal WiFi

Para poder obtener la distancia a la que se encuentra un dispositivo inalámbrico, existen múltiples estrategias [25] [13]. Una es enviar un mensaje (*Ping*) y esperar la respuesta (*Pong*). Con base en la diferencia entre el tiempo de emisión y el de respuesta, determinar la distancia según la velocidad de propagación de la onda electromagnética en el aire en promedio. Esta estrategia tiene el inconveniente de que sí requiere la cooperación del dispositivo del cual quiero obtener su posición, ya que necesito de su respuesta.

La otra alternativa es esperar a que el dispositivo inalámbrico cliente emita un paquete y medir su *Received Signal Strength Indicator* o RSSI. Este indicador representa una aproximación de la intensidad con la cual la antena inalámbrica receptora recibe el mensaje transmitido. Suele usarse a menudo en redes inalámbricas para determinar la calidad de la conexión entre dispositivos, pero puede aprovecharse para deducir su distancia (aplicando técnicas de predicción de la propagación del espectro electromagnético), como demuestra el estudio realizado por Barai et al. [5].

Esta característica puede ser usada para, por ejemplo, visualizar la intensidad de la señal de los *Access Points* que se encuentran en la cercanía, pero también sirve para que los *Access Points* puedan saber la distancia a la que se encuentran sus clientes.

2.2.2. Multilateración y Trilateración

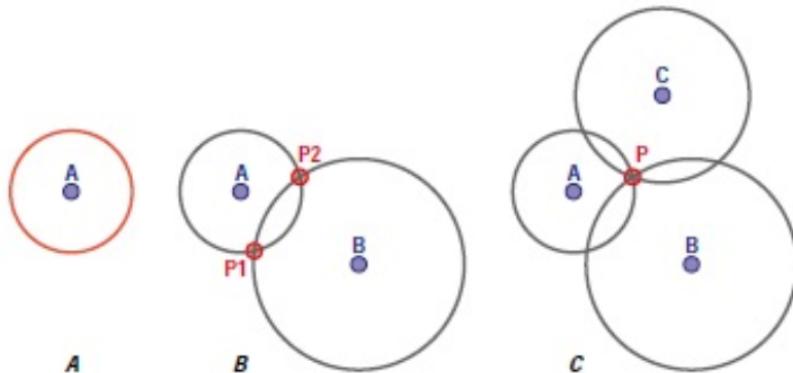


Figura 2.2: Tipos de intersecciones. Utilizando el método clásico de trilateración, se pueden observar por lo menos tres casos.

Utilizando un *Sniffer* para recibir los paquetes de tipo *Probe Request* o *Beacon Frame* que se encuentran en un área y obteniendo el RSSI de cada paquete, es posible estimar la distancia a la que se encuentra un objetivo desde nuestro *Sniffer* (caso A). Sin embargo, si se tiene más de un *Sniffer* (B y C) y de cada uno conozco su ubicación precisa, puede entonces estimar la posición geográfica del cliente mediante multilateración.

La trilateración y la multilateración son técnicas matemáticas que se utilizan para determinar las coordenadas desconocidas de un punto en el espacio, basándose en las distancias desde ese punto a varios puntos conocidos. En el caso del GPS, un receptor mide la diferencia en el tiempo de llegada de las señales de al menos cuatro satélites, lo que permite calcular su posición tridimensional (latitud, longitud

y altitud). De manera similar, en un entorno de localización basado en Wi-Fi, los puntos conocidos serían los sensores Wi-Fi que registran la presencia de otros dispositivos, y la ubicación desconocida sería la del dispositivo objetivo [7].

La figura 2.2 ilustra el concepto de trilateración, que es la base para entender la multilateración. En un contexto ideal, con tres círculos que se intersectan en un único punto, se puede determinar la ubicación exacta del dispositivo. Sin embargo, en la práctica, debido a factores como la interferencia de señales, los reflejos y la atenuación, las estimaciones de distancia basadas en RSSI pueden no ser precisas [17], lo que lleva a la necesidad de utilizar múltiples mediciones y aplicar técnicas de optimización para encontrar la mejor estimación de la posición[16][3].

La multilateración es un proceso avanzado de localización que extiende los principios de la trilateración para estimar la posición de un objeto en dos o tres dimensiones. Mientras que la trilateración se basa en las distancias medidas desde tres puntos conocidos (en 2D) o cuatro puntos (en 3D) hasta el punto de interés, la multilateración utiliza señales de cuatro o más puntos para realizar la estimación, lo que mejora significativamente la precisión y la fiabilidad de la localización, especialmente en entornos complejos.

El proceso de multilateración implica la resolución de un sistema de ecuaciones que representan las distancias estimadas desde los nodos conocidos hasta el dispositivo. Este sistema puede ser linealizado y resuelto mediante el método de los cuadrados mínimos, como veremos más adelante en la sección 2.6, para encontrar la ubicación que mejor se ajuste a las mediciones observadas.

Entonces, hasta ahora podemos identificar a un dispositivo en particular y si se encuentra en rango de al menos uno de nuestros sensores que

A partir de ahora llamaremos *Sniffers* y sabremos su ubicación con mayor o menor precisión según cuantos *Sniffers* tengamos en el área. Para poder seguir efectivamente la trayectoria de un individuo, es necesario poder identificar inequívocamente al dispositivo que se quiere rastrear a lo largo del tiempo. Esto es posible ya que cada paquete *Probe Request* y *Beacon Frame* contiene la *MAC Address* del dispositivo que sirve como identificador único.

2.2.3. Aleatorización de MAC Addresses

Existen dispositivos que implementan técnicas de randomización de direcciones MAC [15]; en este caso, se asigna periódicamente una dirección MAC aleatoria como medida de protección de la privacidad. Esta práctica complica el seguimiento continuo de dichos dispositivos y dificulta la estimación precisa del número de dispositivos presentes en una determinada área, puesto que el mismo dispositivo podría ser contabilizado en varias ocasiones bajo distintas direcciones MAC.

Sin embargo, esta técnica no es aún ampliamente usada al momento de escribir este trabajo [1] e incluso se cuenta con la posibilidad de distinguir rápidamente si una *MAC Address* es local (randomizada) o global (no randomizada) pudiendo así descartar “el ruido”.

Además, los dispositivos móviles [10] suelen emitir los paquetes *Probe Request* en grandes ráfagas sin cambiar la *MAC Address* incluso cuando estos tienen activada la randomización. Esto implica que, si bien no es posible saber en principio si es el mismo dispositivo el que desapareció de la ubicación anterior y luego reapareció en una nueva, al tener más de un *Probe Request* puedo saber su velocidad y dirección.

Lo que es más, sabiendo estos dos datos, es posible predecir que dos *MAC Address* son distintas (randomizadas) sean en verdad pertenecientes al mismo dispositivo.

No obstante, aunque el dispositivo no randomice, es necesario para poder saber el recorrido completo de un dispositivo en un área, haciendo necesario algún tipo de algoritmo de predicción de desplazamiento. Ya que a no ser que todos los *Sniffers* cubran el área en cuestión de manera completa, y no haya ninguna anomalía en la recepción. Van a haber secciones del recorrido que no van a poder ser capturadas por los *Sniffer* ya sea por un problema de rango, de espacios ciegos, o bien porque el dispositivo cliente no emite la suficiente cantidad de *Probe Request* como para poder hacer un seguimiento más exhaustivo. Esto sin mencionar las anomalías de mediciones, ya sea por interferencia o por paredes en el medio que bajan la señal recibida a pesar de que el dispositivo no esté más lejos.

En este trabajo, sin embargo, no exploramos el problema de la randomización ni el problema de los espacios ciegos. Nos limitaremos a ubicar en el espacio y el tiempo a un dispositivo en modo *Access*

Point emitiendo paquetes *Beacon Frame*. De esta manera nos aislamos del problema de la randomización, además de tener una buena frecuencia regular de datos sin interrupciones, que es fundamental para los experimentos realizados.

2.3. Medición de distancia en base al modelo de *Path Loss*

Para poder realizar el paso de multilateración, primero es necesario conocer la distancia desde cada uno de los *Sniffer* hacia el “*Objetivo*”. Para ello se usará en este trabajo la medición de RSSI y luego, para convertir de RSSI a distancia, basándonos en trabajos previos [11] [18], se decidió usar un modelo derivado de la fórmula de pérdida en el trayecto de propagación o *Path Loss*. Mediante el cual la pérdida de señal se incrementa proporcionalmente al logaritmo de la distancia recorrida.

Este modelo describe cómo la intensidad de una señal electromagnética disminuye a medida que se propaga a través del espacio. La pérdida ocurre debido a diversos factores como la dispersión, absorción y difracción de la señal, y se incrementa proporcionalmente a la distancia recorrida desde la fuente emisora. Esta atenuación de la señal es esencialmente una consecuencia de la expansión de las ondas electromagnéticas en el espacio, lo que resulta en una reducción de la densidad de potencia a medida que la distancia aumenta.

La fórmula general de *Path Loss* en el espacio libre se puede expresar como:

$$P(d) = P(d_0) - 10n * \log\left(\frac{d}{d_0}\right) - X_\sigma \quad (2.1)$$

Donde:

d = distancia (m)

d_0 = distancia de referencia inicial (m)

$P(d)$ = es la potencia recibida en la distancia d

$P(d_0)$ = potencia en la distancia de referencia inicial (dBm)

X_σ = es una variable de ajuste que está relacionada directamente con la incertidumbre del modelo.

Luego podemos asumir que la distancia de referencia será 1 metro y agregar una variable A' que ajustaremos para calibrar nuestra fórmula, reemplazando a $P(d_0)$ y X_σ .

$$RSSI = A' - 10n * \log(d) \quad (2.2)$$

Finalmente, podemos simplificar aún más dividiendo por 10 y reemplazando $A'/10$ por A

$$RSSI = A - n * \log(d) \quad (2.3)$$

Ahora sólo resta hallar el valor de A y N para nuestros

Sniffers en el ambiente donde deseamos realizar la medición. Sin embargo, debido a las interferencias que ya mencionamos, en algunos escenarios puede ser necesario algún proceso que permita filtrar la señal para luego después proceder a hacer el siguiente paso de ajuste logarítmico.

2.4. Ajuste Logarítmico

El ajuste logarítmico juega un papel crucial en la modelización de la relación entre la intensidad de la señal recibida (RSSI) y la distancia en aplicaciones de localización y seguimiento. Este proceso implica ajustar un modelo matemático que mejor se acomode a los datos observados, permitiendo así predecir la distancia a partir de los valores de RSSI.

Para llevar a cabo el ajuste logarítmico, se empleó la biblioteca **lmfit** en *Python*, la cual ofrece herramientas avanzadas para la optimización y solución de ecuaciones lineales y no lineales. En particular, se utilizó el método de *Cuadrados mínimos Secuenciales* (SLSQP), que permite encontrar los valores

```

1 def fit_parameters(distance, signal, A=0, N=0):
2 """
3 Ajusta los parámetros del modelo logarítmico para estimar la relación entre
4 la distancia y el RSSI.
5 distance (list): Lista de distancias medidas.
6 signal (list): Lista de valores de señal al RSSI correspondientes a las distancias.
7 A (float): Valor inicial de A (por defecto 0).
8 N (float): Valor inicial de N (por defecto 0).
9
10 Retorna:
11 tuple: Valores ajustados de A y N, y la métrica de calidad del ajuste.
12 """
13
14 # Define una estimación inicial para a y n.
15 initial_guess = dict(a=A, n=N)
16 # Crear un modelo utilizando la función distance_to_rssi
17 regressor = lmfit.Model(distance_to_rssi)
18 # Ajustar el modelo a los datos usando el método de Cuadrados Mínimos Secuenciales
19 # (SLSQP)
20 results = regressor.fit(signal, d=np.array(distance), **initial_guess, method="slsqp")
21 # Calcular el residual como la norma de la diferencia entre la señal observada y la
22 # predicha.
23 residual = np.linalg.norm(signal - distance_to_rssi(distance,
24                                         results.values['a'],
25                                         results.values['n']))
26
27 # Retornar los parámetros ajustados y la métrica de calidad del ajuste (100 -
28 # residual)
29 return results.values['a'], results.values['n'], 100 - residual
30
31 def distance_to_rssi(d, a, n):
32 """
33 Modelo que describe la relación entre la distancia y el RSSI.
34 d (float): Distancia entre el emisor y el receptor.
35 a (float): Parámetro A del modelo.
36 n (float): Parámetro N del modelo.
37 Retorna:
38 float: Valor de RSSI correspondiente a la distancia d.
39 """
40
41 # Ecuación del modelo para convertir la distancia en RSSI
42
43 return - (n * np.log10(d) + a)
44
45 def rssi_to_distance(rssi_values, a, n):
46 """
47 Convierte valores de RSSI a estimaciones de distancia utilizando el modelo ajustado.
48 Retorna:
49 list: Lista de distancias estimadas correspondientes a los valores de RSSI.
50 """
51 distances = []
52 for rssi in rssi_values:
53     # Usar la inversa de la ecuación distance_to_rssi para encontrar la distancia
54     distances.append(10 ** (-1 * (rssi + a) / n))
55
56 return distances

```

Figura 2.3: Script Python para ajuste de curva. Código disponible en GitHub.

óptimos de los parámetros del modelo que minimizan la diferencia entre los valores predichos por el modelo y los observados en la realidad.

El proceso de ajuste comenzó con la selección de una función a la que ajustar, en este caso, la función logarítmica ya mencionada que describe cómo el RSSI varía con la distancia. Basada en la teoría de pérdida de trayectoria (*Path Loss*).

El ajuste del modelo se realizó sobre un conjunto de datos que consiste en mediciones de RSSI y las correspondientes distancias reales, utilizando como valores iniciales una aproximación de los parámetros **A** y **N**, que representan la intensidad de señal a una distancia de referencia y el decaimiento de la señal con la distancia, respectivamente.

2.4.1. Modelo Matemático

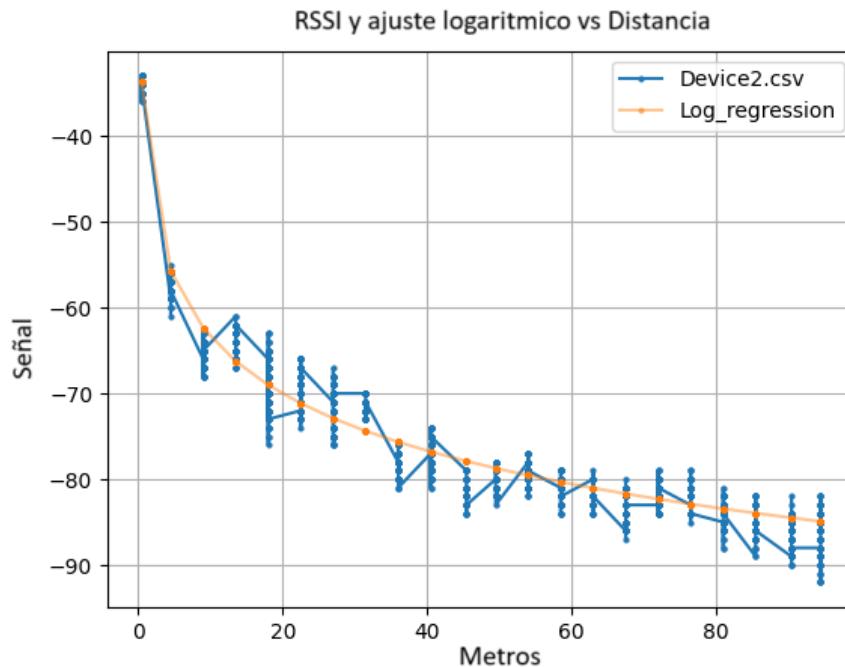


Figura 2.4: RSSI segregado por distancia de la medición vs regresión logarítmica, prueba de campo, con ESP8266 en campo de deportes UNGS

La relación entre RSSI y la distancia se modela con la misma ecuación ya mencionada en la sección anterior. Con una pequeña diferencia en los signos para obtener números positivos.

$$\text{RSSI} = -A - N \cdot \log(d) \quad (2.4)$$

donde:

- d representa la distancia entre el dispositivo emisor y el receptor.
- A es el valor de RSSI medido a una distancia de referencia de 1 metro. Que en vez de averiguar de manera experimental, procedemos a calcularlo usando las técnicas mencionadas.
- N es el coeficiente de pérdida de trayectoria, que refleja cómo la señal se atenúa con el incremento de la distancia.

Esta fórmula ajusta el RSSI observado a un modelo logarítmico, donde a y n son parámetros clave que se optimizan para ajustar el modelo a los datos recogidos.

Para convertir los valores de RSSI en estimaciones de distancia, se utiliza la inversión de la fórmula anterior:

$$d = 10^{(-1 \cdot (\text{RSSI} + a) / n)} \quad (2.5)$$

Para evaluar la bondad del ajuste, se calculó el residual como la norma euclídea de la diferencia entre los valores observados y los predichos por el modelo. Este residual proporciona una medida cuantitativa de la discrepancia entre el modelo y los datos reales, sirviendo como indicador de la precisión del ajuste.

$$\text{residual} = \|\text{RSSI observado} - \text{RSSI predicho}\| \quad (2.6)$$

El valor de R, definido como $100 - \text{residual}$, ofrece un indicador para interpretar la calidad del ajuste; valores cercanos a 100 indican un ajuste muy preciso, mientras que valores más bajos sugieren una mayor discrepancia entre el modelo y los datos reales.

2.4.2. Implementación

Se implementaron scripts en *Python* para realizar el ajuste logarítmico y el cálculo del residual, como el extracto que se puede ver en 2.3 y cuyo código completo se encuentra en la sección 6.1. Estos scripts permiten automatizar el proceso de ajuste y evaluación para diferentes conjuntos de datos, facilitando la experimentación y el análisis.

La figura 2.4 muestra un ejemplo de gráfica resultante de aplicar dicho script, ilustrando cómo el modelo ajustado se alinea con las mediciones reales de RSSI segregadas por distancia.

2.5. Filtrado de Kalman

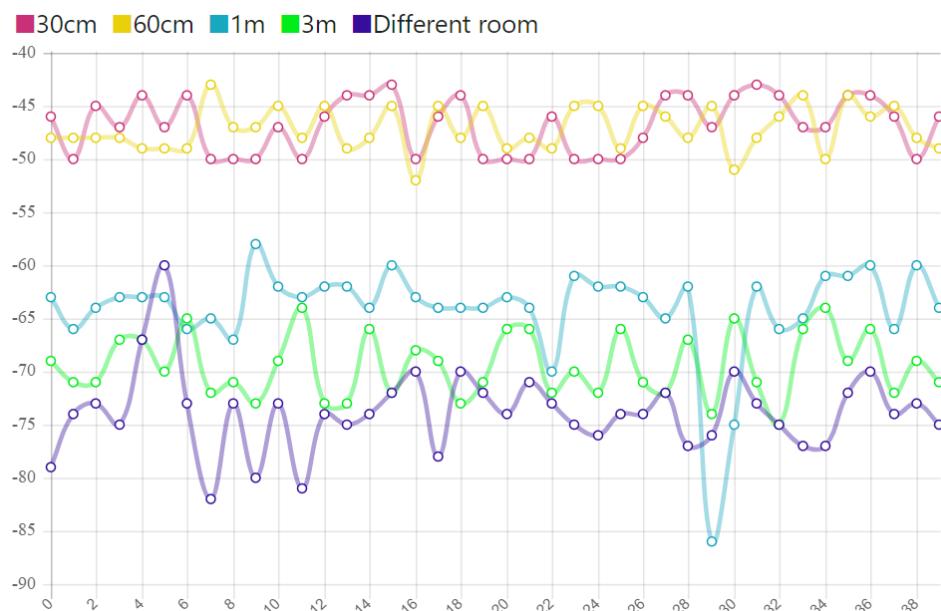


Figura 2.5: RSSI recibido a distintas distancias. (Kalman filters explained: Removing noise from RSSI signals. Wouter Bulten)

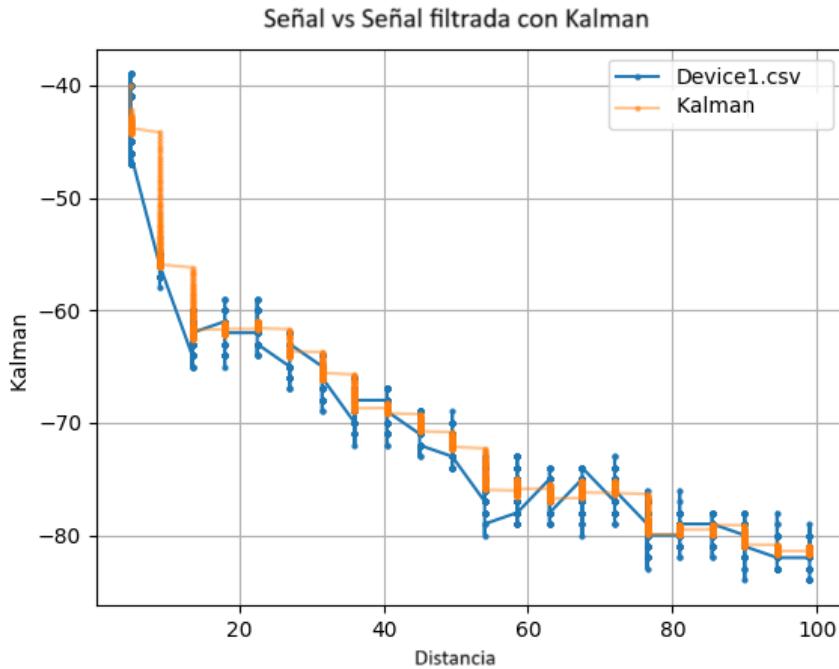


Figura 2.6: Medicion de RSSI original y filtrada con Kalman, prueba de campo con ESP8266 en campo de deportes UNGS

El filtro Kalman es un algoritmo recursivo muy versátil que se utilizó en este trabajo para estimar con más precisión la distancia a partir del RSSI recibido. Sobre todo en presencia de ruido e interferencia. Se decidió utilizar dicho filtro basándonos en el trabajo de Wouter Bulten et al. [6] y del cual deriva un script en Python adaptado del original en Javascript por Sifan Ye [22]. También se consultó como referencia otros trabajos similares como el de Tsanousa et al. [21].

Dicho filtro consiste en un estimador de estado que realiza una estimación de alguna variable no observada (en nuestro caso el RSSI real derivado de las mediciones observadas) basándose en mediciones ruidosas y teniendo en cuenta el historial de las mediciones anteriores. Un filtro de Kalman tradicional asume modelos lineales. Es decir, el paso del estado actual al siguiente debe ser una transformación lineal.

Se usará la identidad como matriz de transformación A_t , y como no hay control sobre el movimiento, la matriz de control B_t se asume nula. Lo mismo ocurre con V_t el cual dejaremos en 0, ya que inicialmente asumiremos que no hay control sobre el objetivo. Finalmente, como el estado es modelado directamente, la matriz de observación C_t también se igualará a la identidad. Por otro lado, ϵ_t y θ_t representan el ruido de proceso y de medición, respectivamente.

En consecuencia, el modelo de transición y observación se puede representar de la siguiente manera:

$$X_t = A_t * X_{t-1} + B_t + V_t + \epsilon_t = X_{t-1} + \epsilon_t$$

El paso predictivo lo calcularemos de la siguiente manera, donde R es el ruido de proceso que provendría del sistema en sí. Como sabemos que en este caso el ruido proviene de la medición y no del sistema, se asume un valor bajo (0.008). \sum se puede utilizar como indicador de la certeza de nuestra predicción. Entonces nos queda que la certeza en el paso t es igual a la certeza en el paso anterior más el ruido de proceso actual definido con la siguiente fórmula:

$$\mu'_t = \mu_{t-1}$$

μ describe la predicción y x es el verdadero valor del estado. μ' a diferencia de μ indica que todavía no hemos incorporado la información sobre la medición. Es decir, es una predicción tentativa. Lo mismo aplica con Z y Z' .

$$\sum t' = \sum(t - 1)' + R_t$$

La ganancia de Kalman, sera nuestra función de ponderación. Para ello usaremos Q como el ruido de medición; este indicador de ruido es más importante para este caso y deberá ser ajustado si se sabe que la interferencia puede deberse al movimiento del objetivo. Sin embargo, como inicialmente asumimos un modelo estático, entonces $Q=3$. La fórmula de la ganancia de Kalman entonces será la siguiente:

$$K_t = K'_t(\sum t' + Q_t)^{-1}$$

Finalmente, el paso de actualización será de la siguiente manera:

$$\begin{aligned} \mu_t &= \mu'_t + K_t(Z_t - \mu'_t) \\ \sum t &= \sum t' - (K_t \sum t') \end{aligned}$$

En este último paso podemos notar que estamos calculando μ'_t , o sea la predicción final del sistema. Además, también hallamos el valor de $\sum t$, nuestra certeza de predicción. Este filtro logra adaptarse al entorno, decidiendo en base al histórico cuánta importancia darle a las nuevas mediciones por sobre las anteriores, ignorando el ruido a medida que este se agranda.

Sin embargo, como este sistema no tiene manera de saber si el ruido de medición Q es producto del ambiente o producto de movimiento orgánico del objetivo, se podrá modificar luego el valor de V_t y B_t para indicar movimiento cuando lo haya y dejar de asumir así que se trata de un sistema estático. Se explorará esto más adelante en la sección de Multilateración.

En este trabajo usaremos las técnicas de filtrado *Kalman* como paso previo de procesamiento antes de utilizar las capturas obtenidas por los nodos. De esta manera eliminamos gran parte de la interferencia momentánea o errores de medición y le damos más foco a la trayectoria de la señal en vez de los accidentes temporales que producen su fluctuación.

2.6. Cuadrados Mínimos aplicado a Multilateración

Para resolver el problema de ubicar en 2D al objetivo solo teniendo las distancias estimadas hacia cada uno de los **Sniffers** es necesario algún algoritmo de localización. En este caso, como no se tiene el dato de la dirección en la que viene la señal, una triangulación no es posible. Para ello se usará en este trabajo el método de trilateración, cuya versión extendida a n nodos se llama multilateración, mediante el método de **Cuadrados mínimos**. Se evaluó durante el desarrollo de este trabajo la posibilidad de usar filtros de partículas como método de localización; sin embargo, dada su complejidad, queda para trabajo futuro analizar qué impacto de performance y precisión podría tener sobre experimentos simulados y reales.

La ecuación objetivo a resolver se plantea de la siguiente manera:

$$(x - x_0)^2 + (y - y_0)^2 = (dist_0 - r)^2 \quad (2.7)$$

$$\dots \quad (2.8)$$

$$(x - x_n)^2 + (y - y_n)^2 = (dist_n - r)^2 \quad (2.9)$$

Donde **Y**, **X** y **R** son las incógnitas a resolver, **Xi** e **Yi** son la posición del *Sniffer* que capturó la señal, **dist_i** es la distancia predecida en base al RSSI capturado, aplicando nuestra fórmula de Path Loss con el **A** y **N** hallados anteriormente. Se asume los mismos **A** y **N** para cada nodo; sin embargo, existen trabajos donde, durante la etapa *offline*, es decir, cuando se calibra la señal, se usa un **A** y un **N** distinto por cada nodo e incluso uno distinto por cada orientación de nodo [9]. A pesar de esto, se decidió optar por un modelo más simple, ya que en simulaciones dio buenos resultados.

Capítulo 3

Experimento Simulado

Antes de plantear un experimento real, decidimos probar nuestro sistema en un caso experimental. Para eso, se decidió usar NS3 (*Network Simulator 3*). NS3 es un simulador de redes que permite no solo emular protocolos de enrutamiento como AODV (*Ad hoc On-Demand Distance Vector Routing*) sino también aplicarle movilidad a los nodos y utilizar distintos modelos de *Path Loss*. Para este experimento simulado, se decidió configurar una red *Ad Hoc* entre todos los nodos. Y se usó un modelo de propagación ideal: el **ns3::FriisPropagationLossModel**. Uno de los nodos, nuestro **Objetivo**, se configuró como el nodo transmisor. Este emite mensajes constantemente de tipo *broadcast* que, al estar conectados en una red *ad hoc*, todos reciben. Cada nodo tiene un script que, al recibir un mensaje de *broadcast* del nodo **Objetivo**, guarda la ubicación real de este y el **RSSI** con el que se recibió el paquete. Mientras esto ocurre, usando un modelo de movilidad, se le aplica un recorrido al nodo **Objetivo**. Mientras tanto, los otros nodos, que hacen las veces de *Sniffer*, permanecen quietos durante todo el experimento. Al finalizar la simulación, se recopilan los datos en un archivo *.csv* para luego ser analizados por herramientas propias.

3.1. Perfilado

3.1.1. Implementación

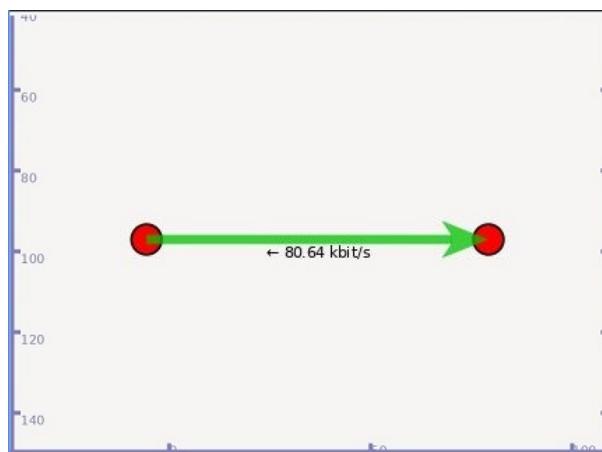


Figura 3.1: Visualización de la simulación de NS3. El nodo objetivo se aleja de uno de los nodos *Sniffer*.

En el caso del experimento de perfilado simulado, se utilizó un solo nodo *Sniffer* que recibe paquetes de un nodo *Objetivo* a medida que este se aleja. El resultado de la simulación es un archivo *CSV* compuesto

por filas (**distancia**, **rssi**). Finalmente, tomamos estos datos de RSSI, más los datos reales de distancia (calculados en la misma simulación) y realizamos un ajuste de curva hasta encontrar los valores de **N** y **A**. Estos nos servirán más adelante para predecir la distancia de dicho nodo en base a su RSSI. El código para este experimento se encuentra en el siguiente archivo **1DDistanceProfiling.cc**. Finalmente, se graficó la señal y la curva usando **pyplot**.

3.1.2. Experimento

distance	rssi
10	-52.55195
11	-54.77925
12	-56.68085
13	-58.339575
14	-59.810625
15	-61.132175
16	-62.33185
17	-63.43025
18	-64.443175
19	-65.383025
20	-66.2595
21	-67.080725
22	-67.85325
23	-68.5825
24	-69.273075
25	-69.928875

Cuadro 3.1: Extracto de captura simulada de perfilado (capture.csv)

Para poder saber la relación entre los RSSI recibidos y la distancia, es necesario perfilar nuestro dispositivo. Esto se debe a que distintos dispositivos Wi-Fi con distinto hardware tienen distintas características de transmisión y recepción, por ejemplo, por la antena misma que este posee. Al perfilar, entonces sabremos con qué intensidad recibe la señal transmitida por el dispositivo móvil de prueba y cómo decae esa señal a medida que este se aleja. Esto tiene como objetivo final averiguar los valores de **N** y **A** para completar nuestra fórmula de conversión de RSSI a distancia. El perfilado se realizó primero en un ambiente simulado para probar la eficacia de nuestros algoritmos y, finalmente, un experimento real. En este experimento en particular en NS3, se les aplicó a los nodos, dentro de una misma red **Ad-Hoc**, un modelo de movilidad donde un nodo **Sniffer** permanece quieto mientras que el otro, el nodo **Objetivo**, se aleja de este hasta **100** metros y su señal decae. Después de llevar a cabo la simulación y recopilar los datos resultantes en un archivo .csv, procedimos con el análisis de los mismos. Los datos recogidos representaban la relación entre la intensidad de la señal RSSI y la distancia entre los nodos, que fue crucial para luego perfilar nuestro dispositivo utilizando **lmfit**.

3.1.3. Conclusión

En la Figura 3.2, se puede apreciar un decaimiento de la señal RSSI del nodo **Objetivo** conforme este se aleja del nodo **Sniffer**. Se observa un rango efectivo de menos de **51** metros, con pérdidas de señal muy frecuentes superados los **50** metros. Se pudo observar una pequeña deriva entre la ubicación real y la predicha, esto se debe a que el *fitting* de la curva simulada no es perfecto y tuvo un pequeño margen de error. Ese error depende de la distancia de la medición, ya que hay partes de la curva que se ajustan mejor que otras. En este caso, muy cerca del transmisor (menos de 1 metro) y al borde de la pérdida de transmisión (más de 45 metros) es donde más difieren, como se puede observar en la figura 3.2. A pesar de ser un entorno simulado, estos resultados nos proporcionan una visión preliminar útil de

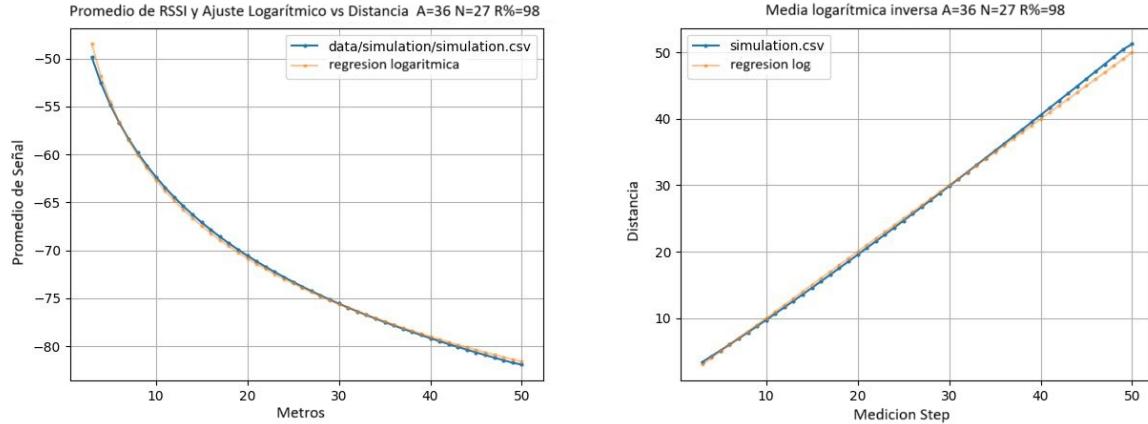


Figura 3.2: A la derecha, media de RSSI segregado por distancia de la medición vs. regresión logarítmica, simulación usando NS3. A la izquierda, inversa al logaritmo de la distancia y distancia predicha en base a \mathbf{A} y \mathbf{N} calculados para la media del RSSI.

cómo se comportaría nuestro sistema en la realidad, verificando que nuestros algoritmos funcionan y es posible ahora sí llevarlo a una prueba de campo. Los próximos pasos implicarán la realización de pruebas en entornos reales. Además, este experimento nos proporcionó valores de \mathbf{N} y \mathbf{A} , necesarios para nuestra fórmula de conversión de RSSI a distancia que se utilizarán luego en el experimento de multilateración simulado.

3.2. Multilateración

3.2.1. Implementación

Similar al experimento simulado de perfilado, se utilizó NS3, esta vez con más nodos y haciendo un recorrido en dos dimensiones. Se utilizó el siguiente script para ejecutar la simulación **2DTrilateration.cc**. Además, para identificar inequívocamente una captura de un paquete emitido puntual, se utilizó *millis* para segregar paquetes al momento de realizar la multilateración. De esta manera, si dos paquetes (dos entradas del archivo *csv*) potencialmente distintos tienen el mismo valor en la columna *millis*, se consideran el mismo paquete en un momento dado y se calculan en la misma ecuación de multilateración. El resultado de la ejecución es otro archivo *csv* con las siguientes columnas: **millis**, **x**, **y**, **rssi**, **target_x**,

millis	node	x	y	rssi	distance	target _x	target _y
9	2	12.5	0	-63.5639	12.492	12.492	12.492
9	4	25	12.5	-63.5649	12.508	12.492	12.492
9	5	0	12.5	-63.5639	12.492	12.492	12.492
9	7	12.5	25	-63.5649	12.508	12.492	12.492
9	1	0	0	-68.0793	17.6664	12.492	12.492
9	3	25	0	-68.0799	17.6777	12.492	12.492
9	6	0	25	-68.0799	17.6777	12.492	12.492
9	8	25	25	-68.0804	17.689	12.492	12.492

Cuadro 3.2: Extracto de captura simulada que luego se usará como *input* para multilateración. (Simulation.csv)

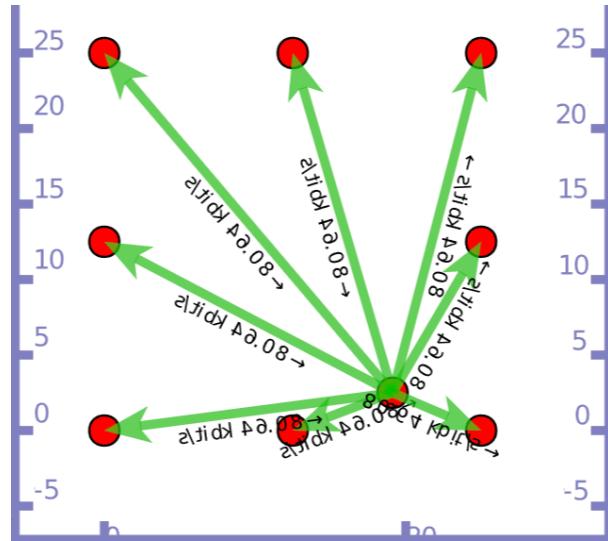


Figura 3.3: Visualización de la simulación de NS3. El nodo objetivo realiza un recorrido mientras los otros ocho nodos *Sniffer* captan su señal.

target_y. **Millis** representa los milisegundos de la simulación, esto se usará luego para agrupar por ese criterio a la hora de hacer la multilateración. **X** e **Y** representan la ubicación del nodo que capturó el **rssi** de esa medición. Finalmente, **target_x** y **target_y** son la ubicación real del nodo **Objetivo** cuando la medición fue realizada, esto lo sabemos con certeza dado que el entorno es simulado. Este archivo luego es procesado por nuestro algoritmo de multilateración, utilizando los valores de **A** y **N** hallados anteriormente para el perfilado simulado. Nuestro algoritmo de multilateración primero agrupa por paquetes que se hayan capturado al mismo momento, es decir, que los milisegundos (**Millis**) de la simulación sean los mismos. Luego solo se le aplicará multilateración y, de manera subsecuente, se agregarán al recorrido aquellos grupos de paquetes que al menos hayan sido capturados por 4 nodos o más. Esto se debe a que nuestro algoritmo de cuadrados mínimos necesita como mínimo al menos 4 nodos distintos para producir una predicción. Finalmente, se graficó el recorrido en 2d usando la biblioteca *pyplot*.

3.2.2. Experimento

Inicialmente, se planteó un experimento con 4 nodos formando un cuadrado de 100x100 metros. Esto no produjo buenos resultados, ya que como el límite de recepción es alrededor de 50 metros, por momentos el nodo objetivo dejaba de estar al alcance de al menos 4 nodos y, entonces, la predicción fallaba. Esto se hacía evidente, produciendo recorridos que discrepanse enormemente del recorrido real. Al hacer más chico el cuadrado de 50x50 metros, la predicción se acerca mucho más a la realidad (Figura izquierda 3.4). Finalmente, se diseñó un segundo recorrido de 25x25 metros, con 8 nodos (figura derecha 3.4), mediante el cual se llegó al mejor resultado. Usando **A = 36** y **N = 27** obtenidos durante el experimento de perfilado, se logró buena precisión a la hora de calcular el recorrido hecho por el nodo **Objetivo**.

3.3. Conclusión

Se logró establecer un límite práctico en la cantidad mínima de nodos necesarios para determinar con precisión un recorrido durante una simulación, identificándose que con menos de cuatro nodos la predicción es inviable. Este requerimiento se explica por el uso de una ecuación tridimensional (involucrando las coordenadas *x*, *y* y el radio *r*), a diferencia de los cálculos bidimensionales comúnmente usados cuando

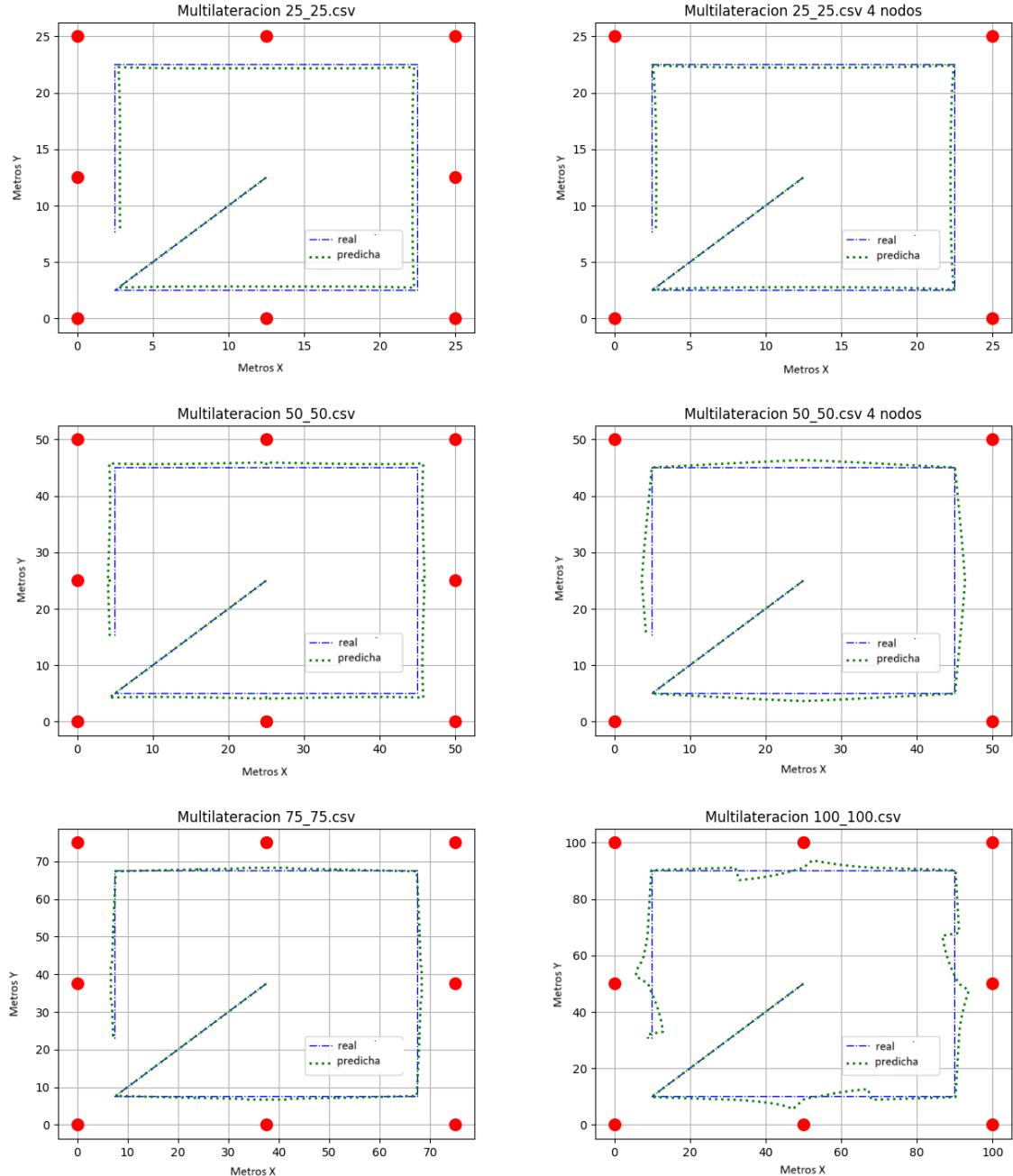


Figura 3.4: Multilateración simulada. Se toman los datos de la simulación en NS3 más el N y A hallados y podemos conseguir ubicar a los nodos con los paquetes capturados. Se simularon distintos tamaños de grilla, desde 25x25 hasta 100x100. Cabe destacar que la precisión aumenta según la cantidad de nodos en el experimento y cuántos nodos reciben señal simultáneamente del objetivo. Esto último está relacionado al tamaño de grilla, ya que cuanto más lejos, menos nodos pueden recibir la señal.

se realiza la técnica de trilateración, lo cual añade la precisión del radio a la medición pero exige un nodo adicional. Este dato de precisión en la predicción (radio en metros de la predicción) es útil para poder descartar datos con mucho ruido. Adicionalmente, se observó que en escenarios con cuadrículas mayores a 50x50, la eficacia de la predicción disminuye progresivamente, atribuible a la reducción en el número de nodos receptores a medida que la distancia aumenta. Otro factor que incide en la pérdida de precisión es la creciente discrepancia entre las distancias predichas por el algoritmo durante la fase de perfilado y las distancias reales, particularmente al aproximarse a los 50 metros, punto en el que la capacidad de recepción del nodo transmisor se ve comprometida y la discrepancia aumenta, como se puede ver en la figura 3.2. Los experimentos de simulación, no obstante, arrojaron resultados prometedores, facilitando el desarrollo de técnicas y herramientas y permitiendo iterar rápidamente el proceso de desarrollo para encontrar soluciones en un entorno controlado. El perfilado proporcionó valores específicos para las variables **A** y **N**, cuya aplicación en experimentos de multilateración simulados confirmó la viabilidad del algoritmo. Esta fase de simulación culminó con buenos resultados, posibilitando avanzar hacia pruebas de campo por parte del equipo de investigación.

Capítulo 4

Experimentos de campo

4.1. Perfilado preliminar con “steps”

4.1.1. Implementación

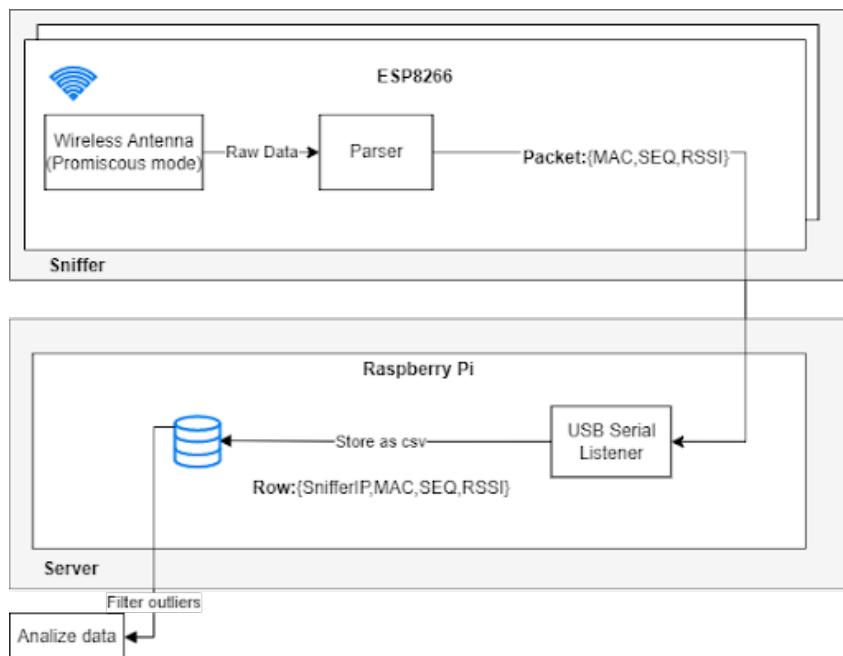


Figura 4.1: Flujo de datos en mediciones de perfilado. El Arduino transfiere los datos mientras llegan directamente a la RPI mediante USB. Estos se agrupan por distancia (ingresada por el usuario) gracias al script de captura.

Antes de avanzar con los experimentos de campo hacia la etapa de multilateración, se quiso confirmar primero que la relación RSSI a distancia que informaban los ESP8266 se acercaba a la realidad (es decir que se aproxime a lo que debería ser el ideal, o el modelo de *Path Loss* antes visto). Similar a lo realizado en el experimento simulado pero con más nodos **Sniffer** para garantizar precisión.

Si bien se eligió utilizar una infraestructura *wireless* más adelante como se describe en la sección 4.2, primero se realizó un experimento preliminar cableado conectando tres ESP8266 mediante USB a una computadora *Raspberry Pi* para tomar múltiples muestras a una distancia conocida y poder poner



Figura 4.2: Experimento de campo de perfilado. Para que todos los dispositivos tengan sus antenas orientadas en el mismo ángulo hacia la misma ubicación durante todo el experimento se utilizó un soporte de metal con monturas hechas con impresora 3D.

un corte entre muestra y muestra a medida se recibíamos. De esta manera poder etiquetar múltiples muestras a 2 metros, a 5, a 10, etc.

Para recibir esta información se desarrolló un script que toma **M** muestras y luego se detiene a esperar a que el usuario le indique la nueva distancia en la que se realizará la próxima medición. De esta manera se obtiene **M** muestras agrupadas por la distancia, a esto le llamamos “*step*”.

Además se decidió hacer el experimento de perfilado en un ambiente controlado (en el campo de deportes de la UNGS con muy poca interferencia en la banda 2.4 GHz). Se garantizó la precisión de los datos mediante la toma de 200 muestras por cada “*step*” de medición. Y se realizaron 21 mediciones por cada dispositivo a 4.5 metros de distancia entre cada una. Dos de los dispositivos finalizaron el experimento, pero uno sufrió un desperfecto y dejó de recibir.

Como transmisor se utilizó un dispositivo móvil en modo AP para poder obtener una visión precisa del comportamiento de la señal.

Para procesar los datos de RSSI, primero se realizó un filtrado *Kalman* de la captura entera por dispositivo, como se puede observar en la figura 2.6. Como último paso, calculamos la media por cada “*step*” de distancia para filtrar las anomalías de los datos. Finalmente se ejecutó una regresión logarítmica usando cuadrados mínimos, el resultado nos da un **A** y un **N** que podría ser utilizado en el paso de multilateración.

4.1.2. Experimento

La figura 4.4 muestra los resultados obtenidos de este experimento de campo. En el eje X se puede observar la distancia real desde el dispositivo ESP8266 hasta el dispositivo móvil transmisor en metros, mientras que en el eje Y se representa el RSSI medido. Cada punto en el gráfico representa una medición individual del RSSI en una determinada distancia.

4.1.3. Conclusión

Puede observarse que a medida que la distancia aumenta, el RSSI se ajusta a una curva según lo predicho por el modelo, lo que se alinea con lo que esperaríamos de la pérdida de señal en función de la distancia en un entorno con mínima interferencia. Sin embargo, también se pueden identificar algunas variaciones y anomalías en los datos, posiblemente debido a factores ambientales no controlados o a



Figura 4.3: Experimento de perfilado en el campo de deportes de la UNGS

variaciones inherentes en la recepción de la señal. Finalmente, realizamos una regresión logarítmica para obtener una representación ajustada de la relación entre la distancia y el RSSI. El resultado de esta regresión nos proporcionó los parámetros **A** y **N** para cada dispositivo, al igual que en el experimento simulado.

4.2. Infraestructura de pruebas de campo *wireless*

Se plantearon distintas alternativas de *hardware* para poder cubrir el área más grande posible al menor costo. Se pensó en la utilización de *routers* comerciales como los *Ubiquiti* usando un software de código abierto como *OpenWRT*, también la posibilidad de usar *Raspberry Pi* o similares e incluso microcontroladores con Wi-Fi compatibles con Arduino como el ESP8266 y el ESP32.

Cualquiera de estas soluciones trae consigo un problema ineludible: Los adaptadores Wi-Fi hasta donde hemos podido investigar no pueden operar en modo promiscuo en simultáneo mientras que opera en otro modo, por ejemplo, en modo *Access Point* o modo cliente mientras se encuentra en modo promiscuo. Se planteó la posibilidad de usar dispositivos que cuentan con dos adaptadores de Wi-Fi, uno para capturar paquetes *Probe Request* y otro para evacuarlos hacia una base de datos centralizada para que luego puedan ser analizados pero no se encontró ninguno de bajo costo y de uso masivo.

Por esto fue elegida la alternativa de utilizar muchos dispositivos ESP8266 como *Sniffer* y un *Raspberry Pi 3* en modo *Access Point* para recibir los datos. Para facilitar las mediciones a modo de soporte aseguramos cada *Sniffer* a un tubo de PVC con una base de cemento. Los ESP8266 fueron adheridos

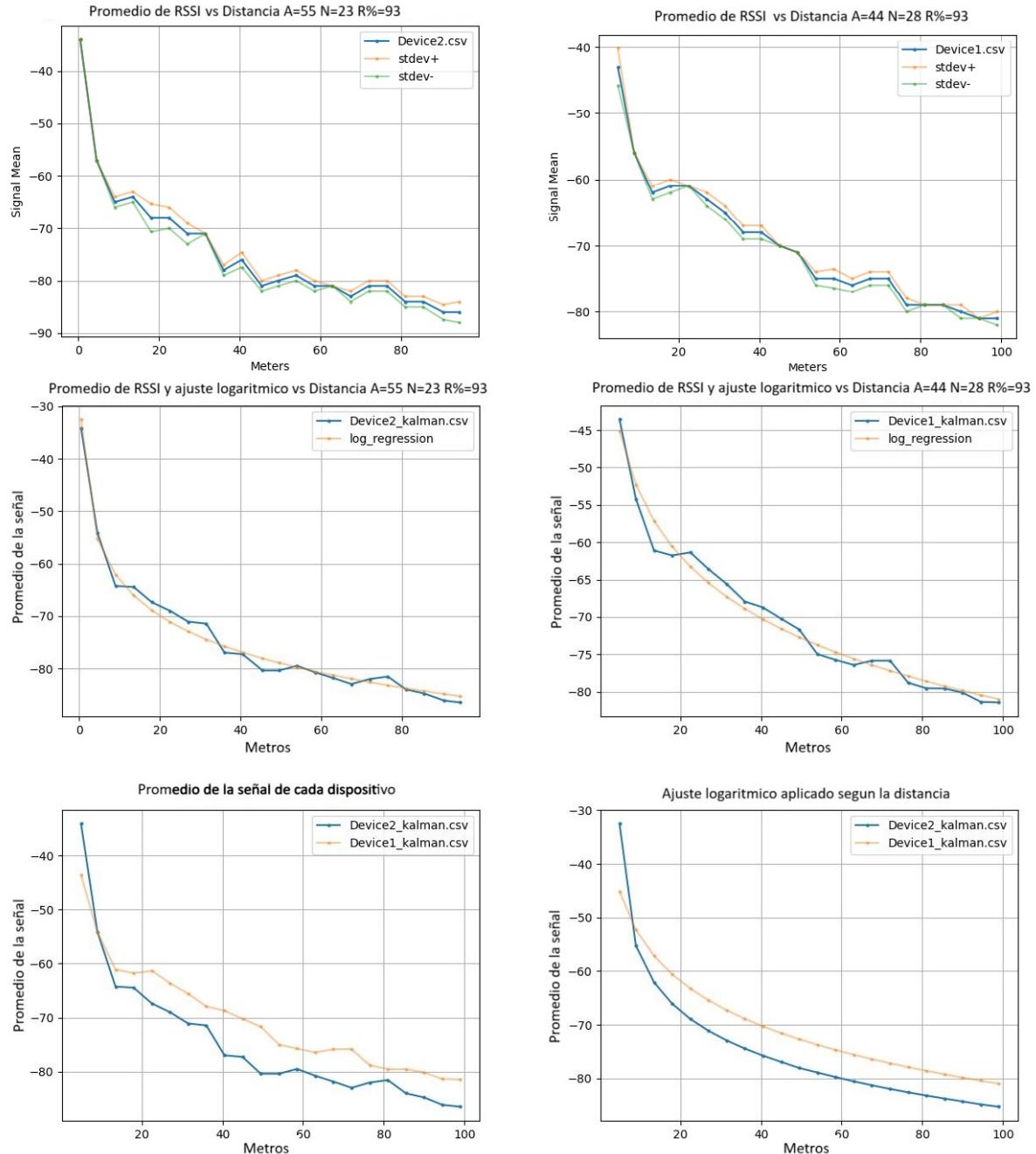


Figura 4.4: Media de RSSI segregado por distancia de la medición vs regresión logarítmica, prueba de campo con “steps”, con ESP8266 en campo de deportes UNGS. Se hallaron valores del dispositivo 1 para $A=37$, $N=21$ y $R(\text{residual})=89$ Para el dispositivo 2 $A=49$, $N=18$ y $R(\text{residual})=88$

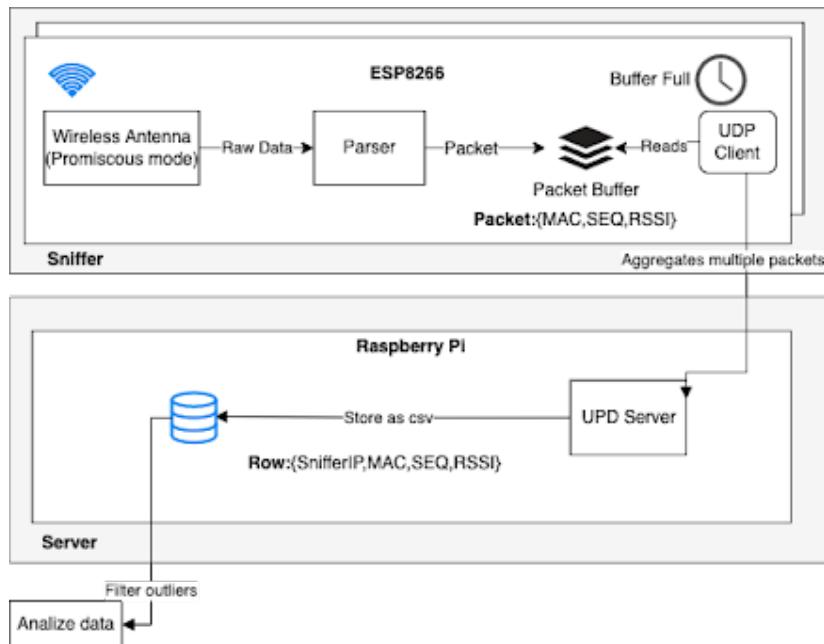


Figura 4.5: Flujo de datos en mediciones de multilateración. Los Arduinos recolectan los datos hasta agotar el espacio en memoria o llegar a un límite establecido. Finalmente descargan toda la información en una RPI de manera inalámbrica.

con *velcro* al tubo y se le agregó una batería de LI-ION conectada a los pines VIN y GND del dispositivo como se puede observar en la imagen 4.9.

El ESP8266 fue programado utilizando Arduino y Platformio para el sensor y Python para el server que recibe los datos. Se configuró el dispositivo en modo promiscuo y se procedió a capturar paquetes y descomponer los *Beacon Frames* en sus parámetros utilizando el *SDK* del ESP8266. De dicho *frame*, parte del 802.11, se extrae puntualmente el RSSI, MAC y el número de secuencia del paquete.

Se configuró el ESP8266 para conectarse a una red Wi-Fi emitida por una *Raspberry Pi* (configurada en modo *AP* para facilitarnos la infraestructura de red) y enviar a la *Raspberry Pi* vía UDP los paquetes agregados de a grupos de **N** paquetes donde **N** es el máximo número que se podía enviar sin llenar el *frame* de UDP. De esta manera optimizamos el tiempo de envío ya que múltiples nodos enviando a la vez de a un paquete pequeño por vez vía Wi-Fi tendrían a sobrecargar a la red y a la RPI que necesita procesar y escribir los datos en una tarjeta SD. La *Raspberry Pi* contenía un script que recibe los datos UDP de cada nodo, y guarda en un solo archivo CSV la siguiente información: **ip_sniffer**, **rssi**, **num_secuencia**. Luego ese archivo se descarga manualmente conectándose por SSH a la RPI.

El número de secuencia se emplea como un identificador único para cada paquete, permitiendo su agrupación durante la realización de experimentos de multilateración o perfilado, similar a cómo se agruparon los paquetes por milisegundos de captura en los experimentos simulados. Este método elimina la necesidad de mantener relojes sincronizados entre los nodos, como sería necesario si se usaran los milisegundos de captura como criterio de agrupación. Sincronizar relojes en un entorno de infraestructura *wireless* representaría un desafío considerable que se decidió evitar. De esta manera, al analizar un paquete, podemos identificarlo con certeza y comparar de manera inequívoca los RSSI recibidos y la diferencia entre estos para cada *Sniffer*.

Finalmente, una vez capturadas suficientes tuplas de (**número_secuencia**, **rssi**) descartando las que no sean de la MAC de nuestro *Objetivo*, los dispositivos cortan la recepción antes de llenar la limitada memoria del dispositivo, este proceso suele durar 2 minutos aunque varía según la tasa de transmisión

de paquetes del *Objetivo*. Luego los ESP8266 proceden a agregar las tuplas capturadas y enviarlas de a grupos mediante UDP hacia la RPI. Esta última recopila los datos y les agrega la IP de cada *Sniffer* para poder identificar de dónde vino la información para luego ser procesada. Finalmente se persiste en un archivo CSV.

Como nodo *Objetivo* se utilizó un teléfono celular en modo *Access Point* para incrementar la tasa de transmisión de paquetes. De esta manera nos limitamos a capturar los paquetes de tipo *Beacon Frame* que emite el teléfono usando su MAC como filtro. Consideramos que es un buen equivalente a una situación donde un transeúnte cuenta con un teléfono sin estar en modo *Access Point* pero emitiendo *Probe Request*. Esto, con la salvedad de la Mac Address Randomization y la escasa frecuencia de emisión de los *Probe Request*, lo que no lo haría beneficioso para este experimento.

4.3. Interferencias y Rango Usable para Multilateración con ESP8266

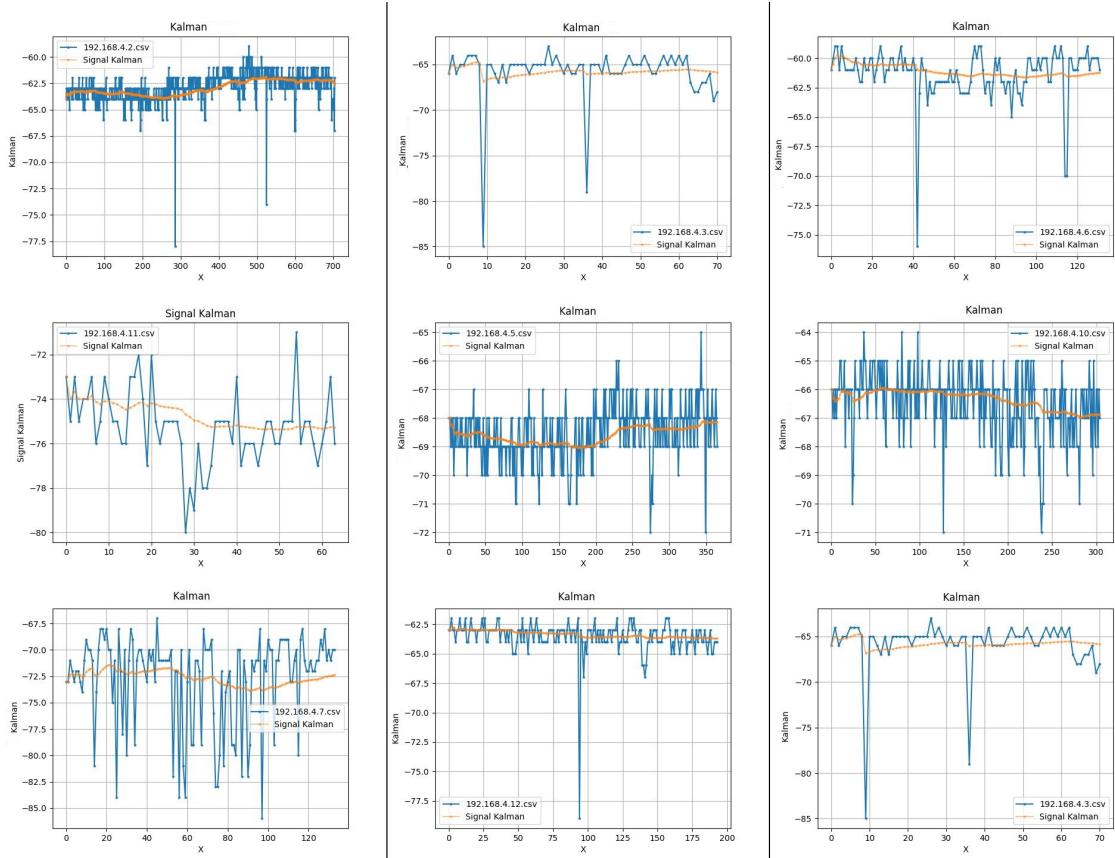


Figura 4.6: Experimento estático, misma disposición de nodos que el de multilateración. Se pueden observar grandes variaciones por interferencia del ambiente en el experimento sin movimiento. Los filtros de Kalman ayudan a mejorar esto.

4.3.1. Experimento estático

Utilizando la técnica de captura *stepless* ya descrita se colocó un teléfono móvil a 15 metros de los nodos, los cuales se dispusieron en línea recta uno al lado del otro al igual que en el experimento

de perfilado. Luego, sin mover el móvil, se procedió a capturar paquetes hasta completar la medición. Como se puede observar en la figura 4.6 algunos nodos capturaron más datos que otros. Esto se debe a que durante la medición algunos nodos experimentaron fallos intermitentes en la alimentación, lo que provocó interrupciones en la recepción de señal. Durante la medición, algunos nodos experimentaron fallos intermitentes en la alimentación debido a un suministro de voltaje inadecuado. Esto ocurrió porque los nodos fueron alimentados directamente desde una batería de iones de litio con un voltaje nominal de 3.7V y un máximo de 4.2V, lo cual es insuficiente para el regulador AMS1117-3.3 integrado en la placa. Este regulador requiere un mínimo de 4.5V para operar correctamente; al no recibir un voltaje adecuado, la tensión de salida fluctuó por debajo de los 3.3V requeridos por el ESP8266, provocando estados inestables en su ejecución. Como resultado, algunos nodos dejaron de recibir paquetes al quedar en un estado de bloqueo (*hang state*) o reiniciarse inesperadamente. Este problema afectó la consistencia de la captura de datos, generando discrepancias en la cantidad de paquetes registrados por cada nodo.

Otros experimentaron grandes interferencias, variando así por varios metros una potencial predicción de distancia.

En todas las gráficas se puede ver cómo el filtro de Kalman ayuda a reducir el ruido y se adapta a los cambios en el perfil de recepción de la señal que persisten en el tiempo. Dicho filtro resultará una herramienta fundamental para mejorar la precisión en los experimentos subsiguientes.

4.3.2. Precisión en el perfilado

Luego de realizar varios experimentos de perfilado quedó en evidencia el problema que surge del modelo de *Path Loss*: A medida que la distancia aumenta, la precisión baja considerablemente. Esto se puede observar en la figura 4.7, donde en la gráfica de la izquierda podemos observar cómo dada una posible interferencia temporal en la medición, la media de RSSI recibida a 12.5 metros es igual a la predicha por el ajuste logarítmico a 19 metros, en total un error de 6.5 metros. Lo que es más aún, la media de RSSI a 12.5 metros es igual a la media recibida a aproximadamente 25 metros, en total un error de 12.5 metros. Claramente, incluso a cortas distancias de menos de 20 metros se pueden ver distorsiones enormes en la distancia predicha. Continuando con este análisis, si consideramos la gráfica derecha en la misma figura, la cota inferior del desvío estándar (*stdev-*) a 27 metros es igual a la cota superior del desvío a 51 metros, en total, 24 metros de error entre ambos desvíos.

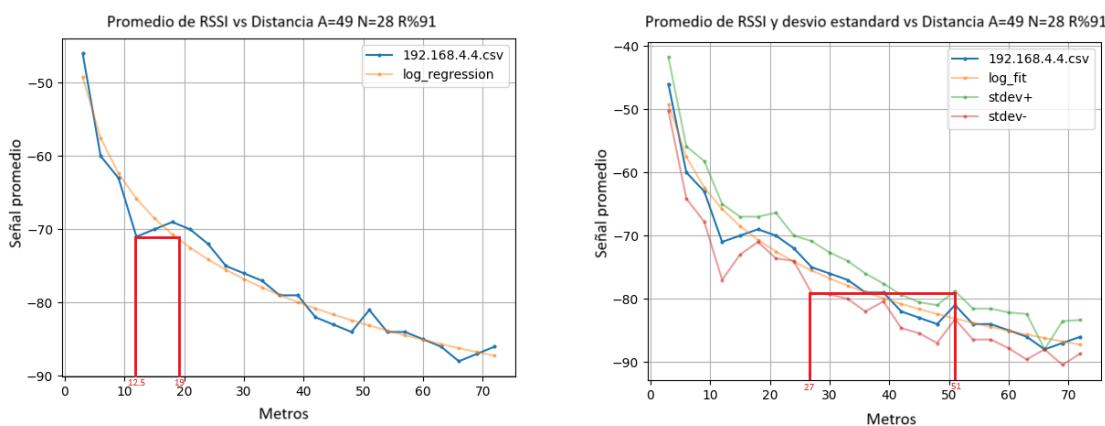


Figura 4.7: Variaciones en la media y los desvíos estándar demuestran la poca precisión de estos dispositivos en distancias de más de 10 metros.

Por estos motivos se determinó que el rango usable efectivo de los dispositivos ESP8266, para experimentos de multilateración enfocados en identificar patrones de movimiento, es de hasta 10 metros. Incluso si se toma el modelo de *Path Loss* sin interferencia como referencia, como podemos ver en la

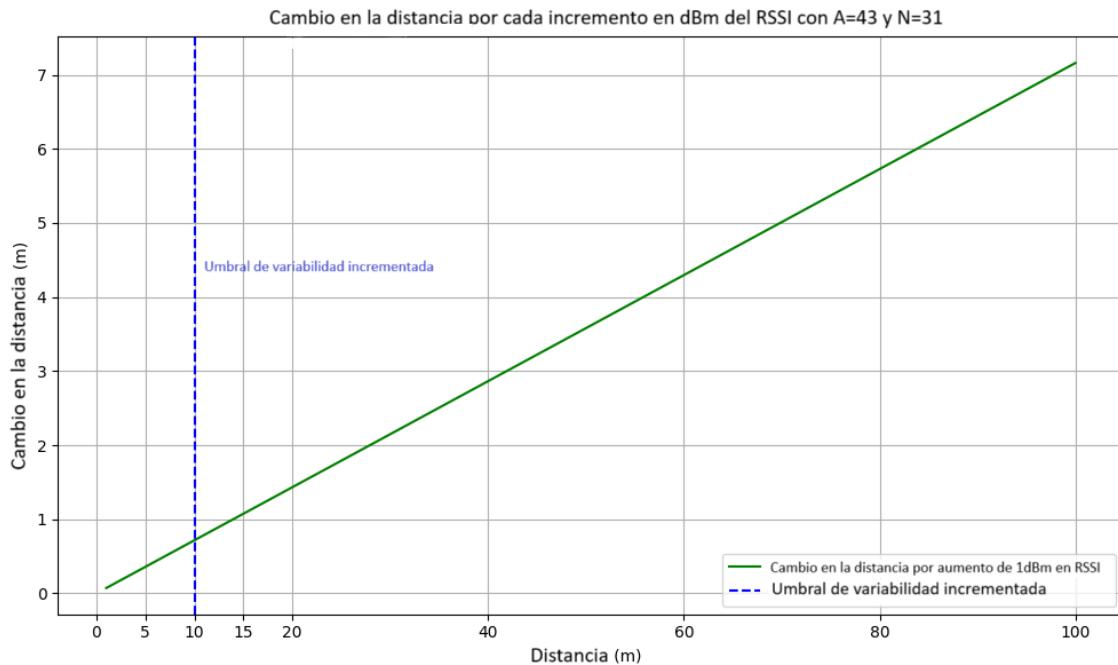


Figura 4.8: Variación de la distancia estimada por cada incremento de 1 dBm en RSSI, demostrando la precisión dentro de un rango de 10 metros y la creciente inexactitud más allá de este límite.

figura 4.8, podemos observar que a partir de los 10 metros, por cada 1 dBm de variación en la intensidad de la señal RSSI se traducirá en cambios de aproximadamente 1 metro en la distancia estimada.

Este rango se ha demostrado adecuado para experimentos donde se buscaba determinar patrones de movimiento específicos, como patrones en forma de estrella o cuadrados, dentro de áreas confinadas. La elección de limitar las pruebas a una cuadrícula de 10x10 metros surgió tras experimentos multilaterales fallidos en áreas más extensas, como cuadrículas de 25x25 metros, donde el aumento de la imprecisión en las estimaciones de distancia comprometía la fiabilidad de los resultados.

Sin embargo, es importante destacar que el rango usable de 10 metros identificado en nuestra investigación es específico, primero, para el perfil de recepción de estos dispositivos y, segundo, puntualmente para la detección de patrones de movimiento detallados mediante multilateración. Para otros objetivos, como detección de presencia en áreas más amplias, el rango usable podría extenderse aún más. Y si consideramos el uso de antenas de mayor ganancia o la implementación de una mayor cantidad de dispositivos para cubrir un área mayor, este rango podría ser aún mejor.

4.4. Perfilado sin *steps*

A pesar de los resultados prometedores obtenidos en los experimentos preliminares de perfilado, se hizo evidente que el perfilado es específico para cada sensor en un ambiente determinado. Esto se debe a que la curva de atenuación de la señal, que describe cómo disminuye la intensidad de la señal a medida que el transmisor se aleja, depende en gran medida del ruido ambiental del lugar donde se realice el experimento y de las características propias de cada dispositivo.

Con el objetivo de evaluar la capacidad para predecir movimientos de un individuo en un ambiente urbano, y ante las limitaciones de acceso al campo de deportes de la UNGS, se optó por realizar experimentos en el campus de la UNGS, un lugar de alta concurrencia y con potencial interferencia de señal



Figura 4.9: A la izquierda: Nodos *Sniffer* construidos con PVC y una base de cemento. El ESP8266 fue adherido con velcro al tubo junto con una batería de Li-ION. A la derecha, un acercamiento a uno de los nodos notando donde se conectó el ESP8266 a la batería utilizando los terminales 5V y GND.

debido a la presencia de edificios. Del cual podíamos disponer en cualquier momento y así realizar más experimentos.

Se decidió además probar el método de captura *wireless* descrito en la sección 4.2 en un escenario de perfilado conocido antes de avanzar a la etapa de multilateración, con el fin de integrar todos los componentes de este trabajo. Este método no solo facilita la realización del experimento de perfilado, sino que también se asemeja más al escenario final de multilateración que se empleará posteriormente.

El término *stepless* hace referencia a un enfoque de perfilado que no requiere de los pasos discretos predefinidos en términos de distancia conocida durante la captura de datos. En lugar de establecer intervalos fijos de distancia para la toma de mediciones, este método permite una captura continua a medida que el **Objetivo** se mueve a través del espacio. No obstante, para poder utilizar las ya mencionadas herramientas de perfilado, todavía se generan "pasos" virtuales basados en el tiempo y la secuencia de los paquetes capturados, lo que permite estimar la distancia recorrida. Estos pasos generados no corresponden a agrupaciones discretas con distancias conocidas, sino que son el resultado de una estimación que utiliza la secuencia de paquetes, la velocidad de movimiento y la frecuencia de transmisión del dispositivo móvil, facilitando así un análisis detallado sin la necesidad de mediciones de distancia fijas y conocidas previamente.

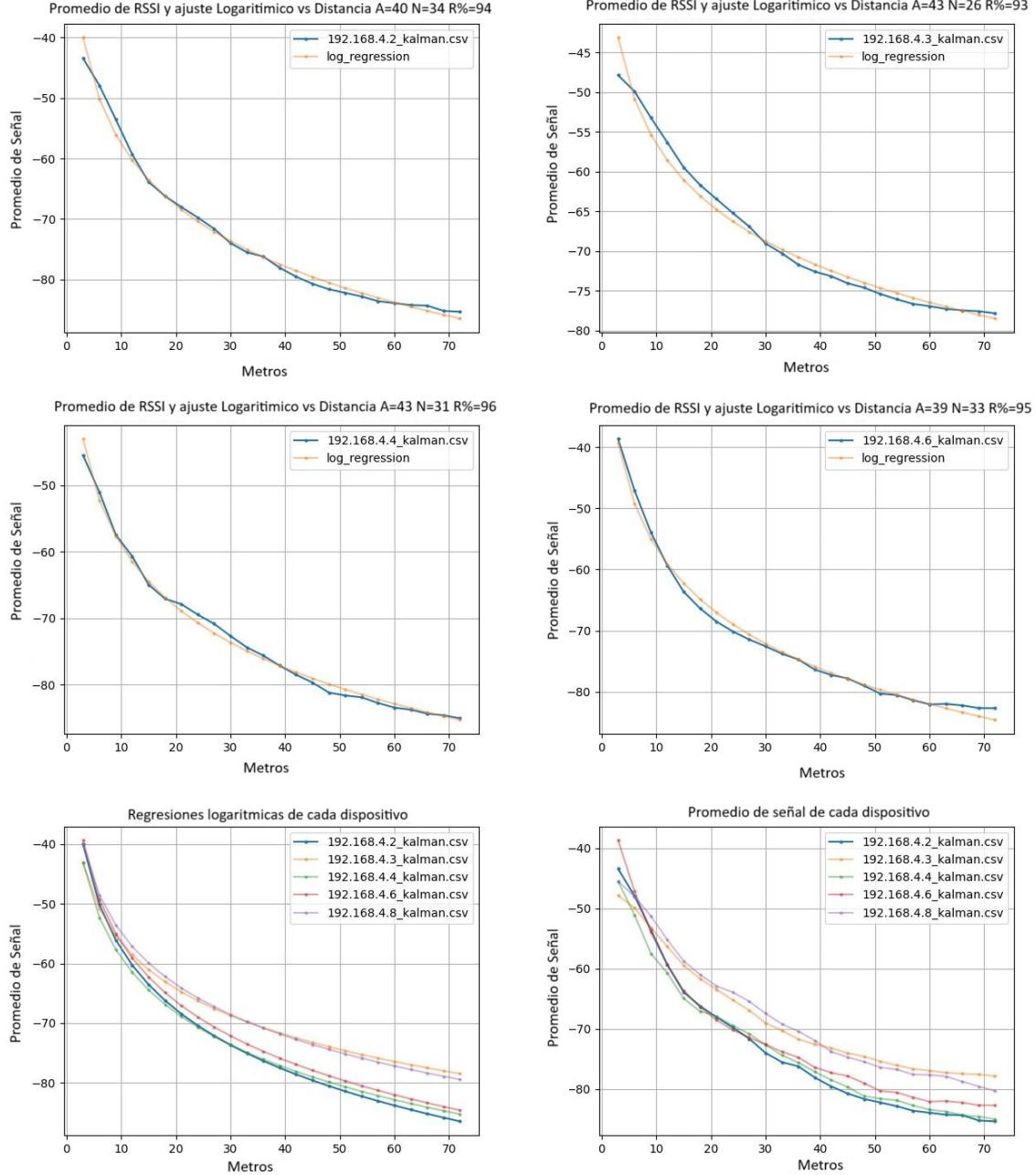
4.4.1. Experimento

A diferencia del experimento preliminar de perfilado realizado en el campo de deportes, este experimento se llevó a cabo en el campus de la UNGS.

Se dispusieron múltiples nodos en línea perpendicular al recorrido, formando una T con este, y se procedió a alejarse hasta alcanzar los 75 metros. Una vez completado el recorrido, se esperó a que los nodos finalizaran la captura y descarga de datos.

El método de captura *wireless* empleado consiste en caminar a velocidad constante desde el punto donde se ubican los sensores, tomando nota del tiempo transcurrido hasta alcanzar una distancia determinada (por ejemplo, 30 metros en 30 segundos), mientras los sensores capturan los paquetes. Una vez finalizado el experimento, los datos se almacenan en la RPI.

Dado que el dispositivo móvil utilizado (Samsung S20) emite paquetes cada 150 ms y conocemos el número de secuencia de cada paquete, es posible estimar la distancia a la que fue capturado y agruparlos en *steps* discretos de distancia. Aunque se pierde precisión en la distancia exacta de captura, este agrupamiento y el cálculo del promedio mitigan el potencial error.

Figura 4.10: Resultados de perfilado *stepless*.

4.4.2. Conclusiones

El perfilado sin *steps* permitió realizar experimentos de manera más rápida, facilitando la obtención de los valores de N y A en diferentes ambientes y para múltiples nodos simultáneamente, sin necesidad de conexión por cable o de tomar notas manuales sobre las distancias, reduciendo el tiempo de medición de 1 hora a 15 minutos. Además, se posiciona como una opción ideal para el despliegue rápido de nodos en un nuevo ambiente. Como se muestra en la figura 4.10, se logró determinar los valores de N y A para cada sensor.

No obstante, algunos sensores presentaron interferencias significativas o su recepción se detuvo abruptamente, un problema recurrente a lo largo de este trabajo. La variabilidad observada en un experimento estático sin movimiento sugiere posibles defectos en los nodos o interferencias en el área de experimentación que afectaron a algunos nodos más que a otros, como se ilustra en la figura ??.

4.5. Multilateración

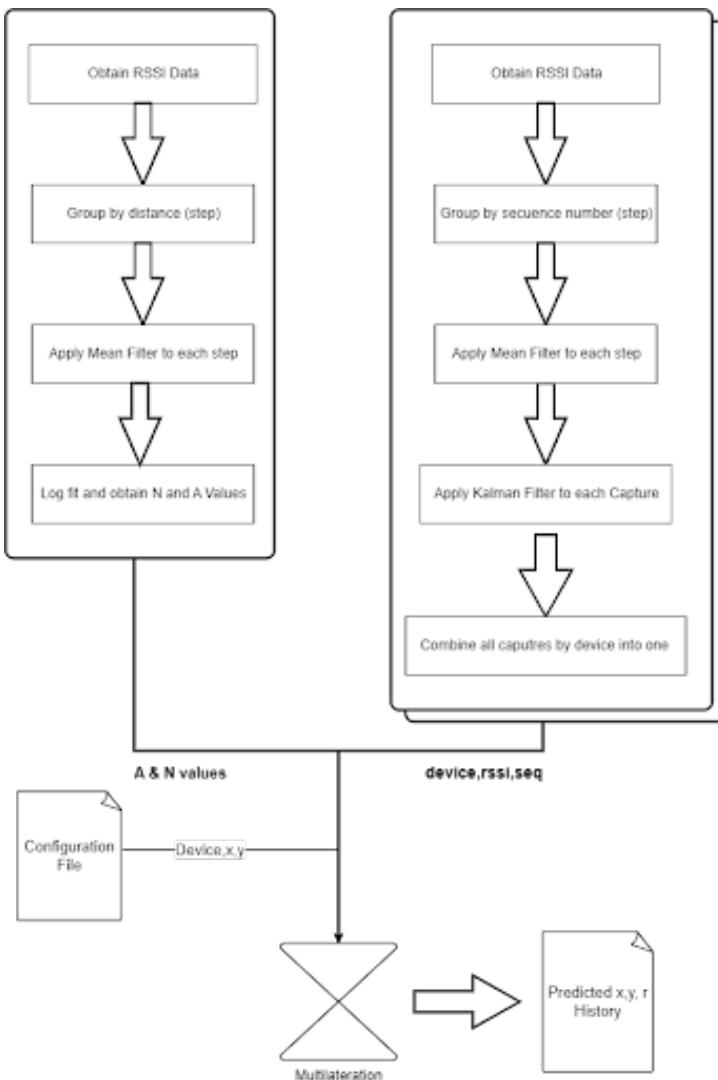


Figura 4.11: Flujo de datos en el experimento de multilateración.

4.5.1. Infraestructura

De manera similar al experimento de perfilado, se implementó una red de nodos **Sniffer** en el campus de la UNGS, siguiendo un esquema predeterminado. En este caso, se colocaron ocho nodos formando un cuadrado de 10 x 10 metros. Estos nodos se dedicaron a monitorear las señales Wi-Fi, capturando la intensidad de la señal RSSI emitida por el nodo **Objetivo**, que se desplazaba siguiendo una ruta



Figura 4.12: Experimento de perfilado en el campus de la UNGS

preestablecida dentro del alcance de los nodos **Sniffer**. Durante el experimento, el nodo **Objetivo** mantuvo una velocidad constante.

Los nodos **Sniffer** registraron la RSSI de las señales emitidas por el nodo **Objetivo**, junto con el número de secuencia de cada señal, transmitiendo estos datos a una *Raspberry Pi* mediante Wi-Fi, una vez completado el experimento y llenados sus buffers para el posterior análisis de los datos.

Con los valores de **A** y **N** derivados de los experimentos de perfilado previos y utilizando la fórmula que convierte el valor de RSSI en distancia, se estimó la separación entre cada nodo **Sniffer** y el nodo **Objetivo** para cada paquete con el mismo número de secuencia. Se realizaron un mínimo de cinco mediciones de distancia con distintos nodos para cada punto en el tiempo para garantizar una mayor precisión.

Si bien durante la etapa de perfilado, se calculó un **A** y un **N** distintos para cada nodo. Tal como se discutirá más adelante, las interferencias fueron un factor muy importante durante las pruebas en el campus de la UNGS. Muchos nodos arrojaron valores muy distintos simplemente por los grandes cambios en la señal, incluso estando quietos, como se aprecia más adelante en la figura 4.6. Por este motivo, se eligieron los valores de **A** y de **N** del experimento de perfilado que mejor ajuste tuvo, es decir que tuvo menor residual. En este caso **A=43** y **N=31**.

4.5.2. Experimento

Se llevaron a cabo múltiples recorridos en distintas ejecuciones de este experimento. Si bien durante el experimento se utilizaron 8 nodos, en la figura 4.13 solo se representaron con círculos rojos 6 nodos. En la figura 4.14 se puede ver que solo hay representados 5 nodos. Esto se debe a que algunos nodos presentaron altas interferencias y sus datos fueron descartados. También se debe a que algunos nodos no recibieron señal parcial o totalmente en algunos o todos los experimentos. Estos fenómenos serán explicados en más detalle en el capítulo 5. La multilateración se llevó a cabo comparando las posiciones estimadas del nodo **Objetivo** con su trayectoria real, con el fin de evaluar la precisión de este método. Aunque los resultados mostraron cierto nivel de error en la estimación de la ubicación, estos se discutirán

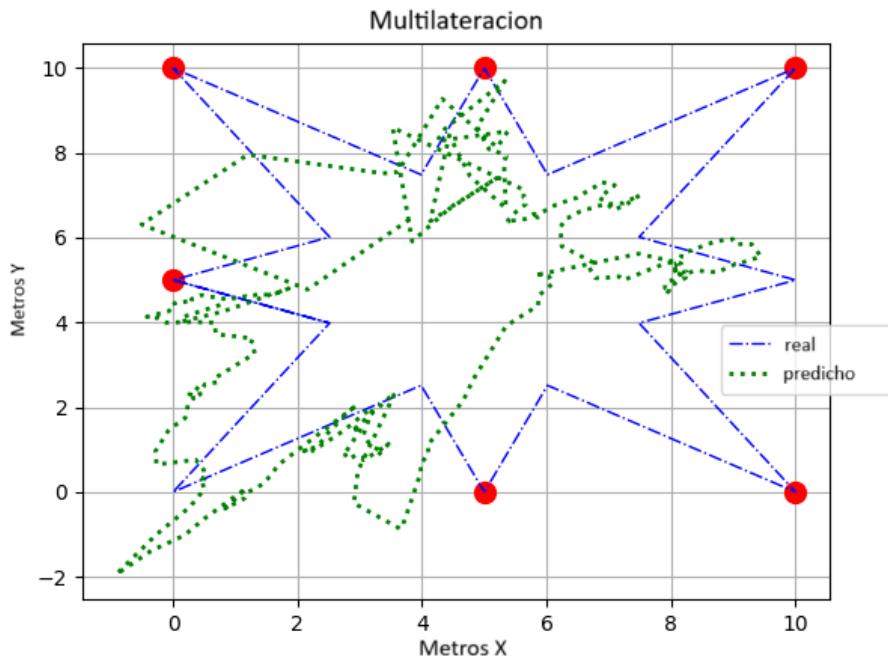


Figura 4.13: Configuración del experimento de multilateración en tiempo real en el campo de deportes de la UNGS.

en detalle en la sección siguiente.

Las estimaciones de la ubicación obtenidas mediante la multilateración, representadas en las figuras 4.13 y 4.14, se compararon con la trayectoria real del nodo **Objetivo** para evaluar la precisión del método. En los gráficos, los nodos **Sniffer** están ubicados en posiciones fijas, mientras que el nodo **Objetivo** sigue una ruta predeterminada.

Cada punto en los gráficos simboliza una estimación de la posición del nodo **Objetivo**, calculada a partir de las mediciones de RSSI recogidas por varios nodos **Sniffer**, el número de secuencia de los paquetes, y los parámetros **A** y **N** obtenidos de los experimentos de perfilado.

La línea azul ilustra la trayectoria real seguida por el nodo **Objetivo**, mientras que la línea verde muestra las estimaciones de su posición según la multilateración. Se puede observar que las estimaciones recrean levemente la trayectoria original, aunque presentan grandes deformaciones.

4.5.3. Conclusión

A pesar de las discrepancias y deformaciones observadas en las estimaciones de la ubicación del nodo **Objetivo**, el recorrido estimado mantiene cierta semejanza con la trayectoria original. Esto indica que, aunque existen errores significativos en la precisión de la localización, el enfoque de multilateración empleado logra capturar la esencia del movimiento del nodo **Objetivo** dentro del área de cobertura de los nodos **Sniffer**. Sin embargo, se identificaron zonas donde las estimaciones se alejaban significativamente de la trayectoria real, atribuibles a diversos factores como la interferencia inalámbrica, movimientos no intencionados del nodo **Objetivo**, y errores en la medición de RSSI.

Dentro de los resultados obtenidos, las discrepancias constantes en las estimaciones de ubicación, manifestadas como desplazamientos constantes o rotaciones en el gráfico, pueden deberse a múltiples factores relacionados con el entorno y la metodología de medición. Un desplazamiento constante en las estimaciones indica un sesgo sistemático en la medición del RSSI, posiblemente a causa de diferencias en

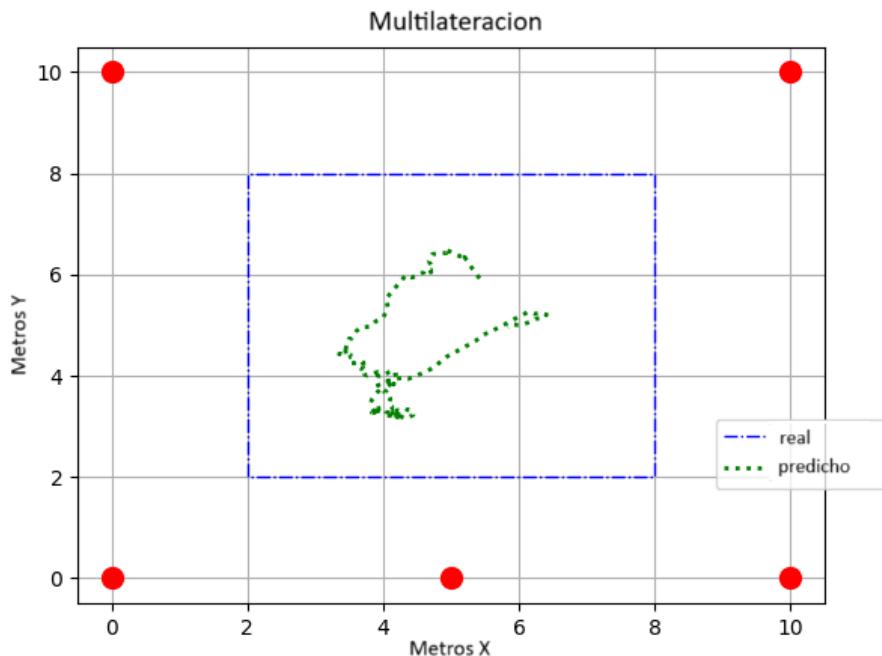


Figura 4.14: Resultados de la multilateración en tiempo real, mostrando la trayectoria estimada del nodo **Objetivo** en comparación con su trayectoria real.

la ganancia de las antenas de los nodos **Sniffer** o que la antena del nodo **Objetivo** no fuera precisamente omnidireccional (como no suelen serlo en teléfonos móviles).

Por otro lado, las rotaciones observadas en la predicción del recorrido, podrían deberse a personas en el camino de la señal o errores en la ubicación de los nodos, por ejemplo, un velcro flojo y un nodo que apunta ligeramente hacia un lado o al otro. Factores adicionales, como la reflexión de señales en superficies cercanas o la difracción a través de obstáculos también pueden contribuir a los errores y deformaciones registrados.

Capítulo 5

Discusiones

Los experimentos realizados, tanto en entornos simulados como reales, han mostrado resultados alentadores en la estimación de ubicación, mostrando resultados prometedores en la estimación de ubicación. Al mismo tiempo, se han detectado ciertos errores y limitaciones que afectan la precisión del sistema. En este capítulo analizaremos en detalle los errores identificados y propondremos mejoras que permitan optimizar aún más el desempeño de un sistema similar.

5.1. Análisis de Errores y Mejoras

5.1.1. Interferencia Ambiental y Variabilidad de la Señal

Uno de los principales desafíos encontrados durante los experimentos fue la interferencia ambiental, que afecta significativamente la calidad y la consistencia de las señales Wi-Fi. Como se muestra en la Figura 4.6, las variaciones en la intensidad de la señal RSSI debido a la interferencia del ambiente pueden introducir errores significativos en la estimación de la distancia y, por ende, en la localización del dispositivo.

Otro impedimento fue la inconsistencia en la recepción de paquetes, atribuible a inestabilidades en el suministro de energía de los dispositivos utilizados. Como se puede ver también en la Figura 4.6, se observó una marcada diferencia en el número de muestras recibidas: mientras algunos dispositivos capturaron hasta 700 muestras, otros apenas 70. Dado que no hubo movimiento en el experimento, estas discrepancias sugieren que, además de la interferencia ambiental, la variabilidad en la alimentación al usar batería de *LI-ION* sin adaptar el voltaje a 5V continuo pudo haber afectado el desempeño, generando fallos a nivel de hardware.

Se propone como trabajo futuro: Primero revisar la consistencia en la recepción mejorando la alimentación del hardware agregándole un módulo *Step-Up* de 3V-4V a 5V. Para mitigar el impacto de la interferencia ambiental, se recomienda implementar algoritmos de filtrado más avanzados, como filtros de partículas o técnicas de aprendizaje automático, que puedan identificar y compensar las fluctuaciones anómalas en la intensidad de la señal en tiempo real. Además, el uso de modelos de propagación de señales más complejos que tengan en cuenta las características específicas del entorno urbano puede ayudar a mejorar la precisión de las estimaciones.

5.1.2. Desafíos en la Multilateración y la Estimación de Distancias

La multilateración, aunque efectiva en teoría, se enfrenta a desafíos prácticos significativos cuando se implementa en entornos urbanos densamente poblados. La precisión de la multilateración depende de la exactitud de las estimaciones de distancia basadas en el RSSI, que pueden verse comprometidas por múltiples factores, incluyendo la orientación de los dispositivos y la presencia de obstáculos físicos.

Se propone como trabajo futuro: Una posible mejora sería incorporar información adicional en el proceso de perfilado, mediante la adición de un módulo GPS. Se podrían, de esta manera, incorporar coordenadas exactas de los nodos durante la etapa de perfilado y la orientación exacta del dispositivo pasaría a ser irrelevante si se utiliza una antena omnidireccional.

Otra mejora podría ser realizar perfilado individual por cada nodo y aplicar los valores encontrados en el proceso de multilateración. En vez del perfilado general realizado donde se eligieron los valores con el menor residual y se aplicaron a todos los nodos.

Finalmente, la calibración periódica de los nodos **Sniffer** utilizándose entre sí mismos como puntos conocidos permitiría el ajuste dinámico de los parámetros del modelo de propagación de señales contribuyendo así también a una mayor precisión.

5.1.3. Limitaciones del Hardware y la Infraestructura

El rendimiento y las capacidades de los nodos **Sniffer** basados en ESP8266, aunque impresionantes dada su baja coste y accesibilidad, imponen limitaciones en términos de procesamiento y alcance de la señal. Estas limitaciones pueden restringir la escala y la resolución de los experimentos de localización.

Desde que el grupo de investigación comenzó a utilizar los ESP8266, surgieron múltiples problemas. El más grave de ellos fue la confiabilidad en la recepción de señal. Hubieron experimentos enteros que tuvieron que ser descartados debido a la recepción de la señal con grandes errores. Y muchos donde uno o varios de los nodos por momentos dejaron de recibir señal. Si bien algunos de estos problemas fueron identificados como falta de energía por el error al conectar el suministro de 5V como se describió anteriormente, también se lo atribuimos al software. El ESP-SDK, que si bien nos facilitó poner dichos dispositivos en modo promiscuo, dificultó el desarrollo del trabajo. Durante los primeros experimentos los datos de dichas pruebas tuvieron que ser completamente descartados por haber usado como base para nuestro software ejemplos de código del SDK disponible en internet que emitía mediciones completamente erradas de RSSI. Luego se investigó un poco más y se utilizó otro software y fue resuelto. El software utilizado se encuentra en el último capítulo de este trabajo.

Los problemas observados en los experimentos se deben, en parte también, a la forma en que se instalaron los nodos. En la configuración actual, los nodos se apoyaron sobre un tubo de PVC, lo que implicó que quedaran orientados en direcciones desconocidas y utilizaban las antenas que vienen impresas en el circuito del dispositivo que sabemos que tienen un patrón direccional sesgado en lugar de antenas omnidireccionales [23]. Además, la ubicación GPS de los nodos fue estimada, lo que impidió conocer con exactitud su posición y altura. En un escenario ideal, al emplear antenas omnidireccionales la orientación de los nodos no afectaría la captación de señales y si se dispone de datos precisos de GPS, junto con una altura definida (por ejemplo, al estar instalados en postes de luz), permitiría obtener mediciones más estables y precisas.

Además La recepción de paquetes se vio limitada en su duración ya que la memoria de los dispositivos se llena muy rápido y al no tener múltiples interfaces, una para recibir y otra para transmitir los paquetes. Sólo era posible recibir y enviar por turnos debido a que en modo promiscuo sólo es posible recibir en esa interfaz.

Se propone como trabajo futuro: la exploración de hardware alternativo que incorpore múltiples antenas, una para la recolección de datos y otra para la evacuación, con capacidades superiores de procesamiento y una antena con mejor ganancia para mejorar la recepción de señales. Además, el desarrollo de una infraestructura de red robusta que permita manejar de manera eficiente la recopilación y el análisis de grandes volúmenes de datos de RSSI para ser enviados a un servidor centralizado, será fundamental para superar las limitaciones actuales. Estas mejoras permitirían que los nodos se mantuvieran más fijos y que las mediciones fueran más precisas, optimizando el desempeño en aplicaciones a gran escala.

5.2. Consideraciones Finales

Durante el análisis de errores se constató que, si bien la localización y seguimiento de dispositivos móviles en entornos urbanos presenta desafíos inherentes, los experimentos han permitido identificar

áreas claras de mejora. Las estrategias de optimización implementadas han contribuido a mejorar la consistencia en la recepción de señales y la precisión de las estimaciones, lo que refuerza la viabilidad del enfoque propuesto. Esta etapa ha proporcionado un marco de referencia sólido para futuras investigaciones, especialmente en el desarrollo de soluciones de bajo costo aplicables en el contexto de ciudades inteligentes.

Capítulo 6

Conclusiones y trabajo futuro

Este proyecto de tesis ha abordado el desafío de desarrollar un sistema de seguimiento móvil utilizando tecnología Wi-Fi, centrándose en las técnicas de multilateración y perfilado, con el objetivo de estimar de manera precisa la ubicación de dispositivos móviles en entornos urbanos. A lo largo de esta investigación, se ha demostrado la viabilidad de implementar una solución tecnológica de bajo costo que puede ser integrada dentro del concepto de ciudades inteligentes, ofreciendo aplicaciones potenciales en *Smart Parking* y el monitoreo del flujo peatonal y vehicular.

6.1. Herramientas de Software Desarrolladas

Durante la elaboración de este trabajo, se desarrollaron múltiples herramientas de software para llevar a cabo las tareas de perfilado y multilateración, así como también el firmware de los ESP8266, script de la Raspberry PI y herramientas para gráficas.

A continuación se listarán los repositorios donde se contiene el código usado en este trabajo para cada una de dichas herramientas.

- **Easy-Trilateration:** Se desarrolló una biblioteca de Python publicada en **PyPI** (Python Package Index), basada en otros trabajos [14], para realizar multilateración con su correspondiente herramienta gráfica para facilitar su uso.
- **RSSI-Filter-Profiling:** Herramienta que permite tomar múltiples capturas de perfilado, aplicar filtros como Kalman, mediana y promedio, realizar ajustes logarítmicos usando mínimos cuadrados y luego producir múltiples gráficas, incluyendo inversa y desviación estándar, comparando múltiples capturas de distintos dispositivos a la vez. También posee un combinador de gráficas, lo que permite tomar múltiples imágenes y convertirlas en una sola.
- **ESP8266-Sniffer-Node:** Firmware para ESP8266 capaz de alternar entre el modo de captura de *Probe Request* y el modo de subida de datos.
- **NS3-RSSI-Trilateration:** Software que permite realizar perfilado y multilateración simulada usando NS3. Combina tanto **RSSI-Filter-Profiling** como **Easy-Trilateration** como submódulos de Git. Funciona aplicando un perfil de movilidad a un nodo, generando una captura simulada y luego graficando la reducción en la señal a medida que este se aleja, hallando los valores de *A* y *N*. Posteriormente, permite simular un escenario de multilateración utilizando la herramienta mencionada. La idea de este software es ejecutar ambas simulaciones de manera sencilla.

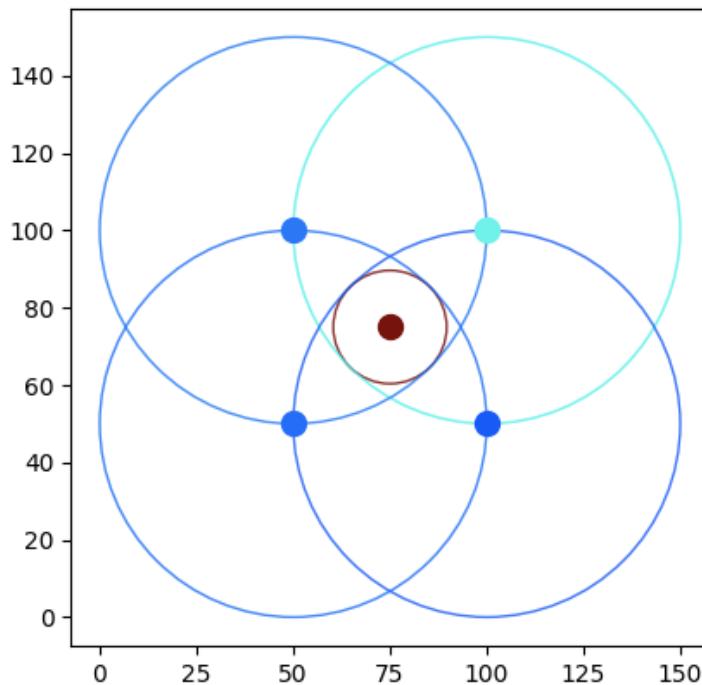


Figura 6.1: Herramienta gráfica de multilateración

6.2. Hallazgos Principales

Los experimentos realizados, tanto en entornos simulados como reales, han demostrado la capacidad de utilizar señales Wi-Fi para localizar dispositivos móviles de manera efectiva. Este enfoque presenta una alternativa viable y de bajo costo a los sistemas basados en hardware especializado, ofreciendo una nueva herramienta para el análisis del comportamiento peatonal y vehicular en las ciudades.

A través de la investigación, se ha identificado la importancia de una disposición y configuración óptima de los nodos **Sniffer** para maximizar tanto la cobertura como la precisión de las estimaciones de ubicación. La interacción entre la disposición de los nodos y las características específicas del entorno urbano, como la cantidad de edificios, transeúntes y por sobre todo las interferencias electromagnéticas. Han demostrado ser un factores crítico en la eficacia de la localización.

6.3. Limitaciones y Desafíos

El proyecto enfrentó varias limitaciones técnicas y metodológicas. La precisión de las estimaciones de ubicación fue impactada por la interferencia inalámbrica y la variabilidad en la recepción de los dispositivos. Las limitaciones inherentes del hardware elegido y la infraestructura disponible para sostenerlo. Además, la práctica de la randomización de direcciones MAC en dispositivos modernos representa un desafío significativo para el seguimiento continuo de dispositivos a lo largo del tiempo.

6.4. Recomendaciones para Futuras Investigaciones

Para continuar avanzando en esta área, se considera importante:

- Investigar en profundidad los efectos de la urbanización en la propagación de señales Wi-Fi y desarrollar técnicas de corrección o compensación que mejoren la precisión en la estimación de la ubicación [8].
- Explorar utilizar GPS, para poder perfilar con precisión los nodos y así mejorar las estimaciones de ubicación obtenidas a través de Wi-Fi.
- Desarrollar algoritmos de aprendizaje automático o alguna otra técnica para desenmascarar direcciones MAC que puedan identificar y seguir dispositivos a pesar de la randomización MAC [4].
- Realizar experimentos adicionales en una variedad de entornos urbanos y con diferentes configuraciones de hardware para validar la robustez y adaptabilidad de las soluciones propuestas [19].
- Probar dichas técnicas con vehículos e instalaciones de *Sniffers* sobre postes de luz para detectar vehículos estacionados[24].
- Explorar la posibilidad de detectar patrones de estacionamiento de automóviles, ciclistas que circulan por las calles y transeúntes en la vereda.

6.5. Comentarios Finales

La realización de esta tesis ha permitido explorar y expandir el conocimiento en el área de localización y seguimiento de dispositivos móviles en entornos urbanos, utilizando tecnología Wi-Fi con micro-controladores de bajo costo. A pesar de enfrentar desafíos significativos, los resultados obtenidos proporcionan una base interesante para futuras investigaciones.

Mirando hacia adelante, es claro que la adopción de soluciones tecnológicas de tipo *Smart Cities* jugará un papel fundamental en el diseño de las ciudades del futuro. Este trabajo contribuye a ese esfuerzo, sentando las bases para investigaciones futuras en el área de Smart Cities.

Bibliografía

- [1] IEEE 802.11-2016. Standard for information technology—telecommunications and information exchange between systems—local and metropolitan area networks—specific requirements—part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, December 2016. Superseded by IEEE 802.11-2020.
- [2] Lucas Addisi, Agustín Alexander, and Alexis Tcach. Wireless ad hoc sensor networks for city street light maintenance. In *48 Jornadas Argentinas de Informática e Investigación Operativa (JAIIO) - I Taller Argentino de Internet de las Cosas (TAIC 2019)*, pages 13–18, 2019.
- [3] Rafina Destiarti Ainul, Susilo Wibowo, Djuwari, and Martin Siswanto. An improved indoor rssi based positioning system using kalman filter and multiquad algorithm. In *2021 International Electronics Symposium (IES)*, pages 558–564, 2021.
- [4] Giovanni Baccichet, Corrado Innamorati, Alessandro E. C. Redondi, and Matteo Cesana. MAC Address De-Randomization Using Multi-Channel Sniffers and Two-Stage Clustering, 2024. Conference pre-print.
- [5] Suvankar Barai, Debajyoti Biswas, and Buddhadeb Sau. Estimate distance measurement using nodemcu esp8266 based on rssi technique. In *2017 IEEE Conference on Antenna Measurements Applications (CAMA)*. IEEE, 2017.
- [6] Wouter Bulten, Anne C. Van Rossum, and Willem F. G. Haselager. Human slam, indoor localisation of devices and users. In *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 211–222, 2016.
- [7] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34, 2000.
- [8] Jeongsik Choi, Yang-Seok Choi, and Shilpa Talwar. Unsupervised learning technique to obtain the coordinates of wi-fi access points. In *Proceedings of the 10th international conference on indoor positioning and indoor navigation (IPIN) 2019*, 2019.
- [9] J Etter-Olguín, C Duran-Faundez, J Rohten, R Seguel-Cardenas, and I Santana. Simulation of rssi-based positioning algorithms for wireless network using ns- 3. In *Proc. Conf. Sci. Int. (CCI)*, 2019.
- [10] Julien Freudiger. How talkative is your mobile device? an experimental study of wi-fi probe requests. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 1–6, 2015.
- [11] Xu Jiuqiang, Wei Liu, Fenggao Lang, Yuanyuan Zhang, and Chenglong Wang. Distance measurement model based on rssi in wsn. *Wireless Sensor Network*, 2:606–611, 01 2010.
- [12] Paul L. Jordan and Andrew Jon Sellers. Reliable presence detection through passive ieee 802.11 management frame sniffing. In *Reliable Presence Detection through Passive IEEE 802.11 Management Frame Sniffing*, 2015.

- [13] Thomas Lindner, Lothar Fritsch, Kilian Plank, and Kai Rannenberg. Exploitation of public and private wifi coverage for new business models. In Winfried Lamersdorf, Volker Tscharmer, and Stéphane Amarger, editors, *Building the E-Service Society*, pages 131–148, Boston, MA, 2004. Springer US.
- [14] madfolio. Least squares trilateration. <https://github.com/madfolio/Least-Squares-Trilateration>, 2018.
- [15] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. A study of mac address randomization in mobile devices and when it fails. *arXiv preprint arXiv:1703.02874*, 2017.
- [16] Robin Wentao Ouyang, Albert Kai-Sun Wong, and Chin-Tau Lea. Received signal strength-based wireless localization via semidefinite programming: Noncooperative and cooperative schemes. *IEEE Transactions on Vehicular Technology*, 59(3):1307–1318, 2010.
- [17] T.S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, 2nd edition, 2002.
- [18] Fengjun Shang, Wen Su, Qian Wang, Hongxia Gao, and Qiang Fu. A location estimation algorithm based on rssı vector similarity degree. *International Journal of Distributed Sensor Networks*, 10(8):371350, 2014.
- [19] Andrey Shestakov, Danila Doroshin, Dmitri Shmelkin, and Attila Kertesz-Farkas. *Lookup Lateration: Mapping of Received Signal Strength to Position for Geo-Localization in Outdoor Urban Areas: 7th International Conference, AIST 2018, Moscow, Russia, July 5–7, 2018, Revised Selected Papers*, pages 234–246. Springer, 07 2018.
- [20] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks*. Prentice Hall, 5th edition, 2011.
- [21] Athina Tsanousa, Vasileios-Rafail Xeferis, Dimos Ntioudis, Christos Chatzigeorgiou, Georgios M editskos, Stefanos Vrochidis, Charalampos Z Patrikakis, and Ioannis Kompatsiaris. Localization module for missing child scenario in iot safety domains. In *2021 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–5. IEEE, 2021.
- [22] Sifan Ye. Python adaptation of kalmanfilter. <https://github.com/wouterbulten/kalmanjs/tree/master/contrib/python>, 2018.
- [23] Yoppy, R Harry Arjadi, Henry Candra, Haryo Dwi Prananto, and Tyas Ari Wahyu Wijanarko. Rssi comparison of esp8266 modules. *2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS)*, pages 150–153, 2018.
- [24] Cheng Yuan, Li Fei, Chen Jianxin, and Ji Wei. A smart parking system using wifi and wireless sensor network. In *A smart parking system using WiFi and wireless sensor network*, pages 1–2, 05 2016.
- [25] Faheem Zafari, Athanasios Gkelias, and Kin K. Leung. A survey of indoor localization systems and technologies. *IEEE Communications Surveys Tutorials*, 21(3):2568–2599, 2019.