PENGEMBANGAN INDOOR POSITIONING SYSTEM MENGGUNAKAN METODE FINGERPRINTING DENGAN SINTETIS DATA DAN KNN



Disusun Oleh:

N a m a : Yuanda Hanif Hisyam
NIM : 20523134

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2025

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGEMBANGAN INDOOR POSITIONING SYSTEM MENGGUNAKAN METODE FINGERPRINTING DENGAN SINTETIS DATA DAN KNN

TUGAS AKHIR

Disusun Oleh:

N a m a : Yuanda Hanif Hisyam
NIM : 20523134

Yogyakarta, 29 Juli 2025

Pembimbing 1	Pembimbing 2
(Fayruz Rahma, S.T., M.Eng.)	(Kurniawan Dwi Irianto, S.T., M.Sc.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGEMBANGAN INDOOR POSITIONING SYSTEM MENGGUNAKAN METODE FINGERPRINTING DENGAN SINTETIS DATA DAN KNN

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia
Yogyakarta, 29 Juli 2025

Tim Penguji

Kurniawan Dwi Irianto, S.T., M.Sc.

Anggota 1

Moh. Idris, S.Kom., M.Kom.

Anggota 2

Izzati Muhimmah, S.T., M.Sc., Ph.D.

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Ir. Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Yuanda Hanif Hisyam
NIM : 20523134

Tugas akhir dengan judul:

PENGEMBANGAN INDOOR POSITIONING SYSTEM MENGGUNAKAN METODE FINGERPRINTING DENGAN SINTETIS DATA DAN KNN

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 29 Juli 2025



Yuanda Hanif Hisyam

HALAMAN PERSEMBAHAN

Segala puji bagi Allah Subhanahu wa Ta'ala. yang telah memberikan saya kesempatan menempuh studi Informatika di Universitas Islam Indonesia. Saya juga mengucapkan terima kasih kepada kedua orang tua saya yang telah memberikan kesempatan untuk menempuh studi di Universitas Islam Indonesia. Terakhir, tak lupa saya ucapkan terima kasih kepada seluruh dosen dan teman-teman di Universitas Islam Indonesia yang telah memberikan saya inspirasi sehingga bisa menjadi pribadi yang lebih baik ke depannya.

HALAMAN MOTO

“I cry, I pray, mon dieu”

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah Swt. atas segala limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan penyusunan laporan Tugas Akhir ini. Penulisan laporan ini turut menjadi wadah untuk mencurahkan berbagai perasaan, mulai dari suka cita hingga tantangan yang dialami sepanjang prosesnya. Dalam proses penyelesaian tugas akhir ini, penulis memperoleh banyak dukungan dan bantuan dari berbagai pihak. Oleh karena itu, pada kesempatan ini, penulis ingin menyampaikan rasa terima kasih dan penghargaan setinggi-tingginya kepada:

1. Allah Swt., atas rahmat dan petunjuk-Nya dalam proses penyusunan Tugas Akhir ini.
2. Kedua orang tua tercinta dan adik tersayang, atas doa, cinta, dukungan moral dan material, serta motivasi yang tidak pernah henti.
3. Ibu Fayruz Rahma, S.T., M.Eng. dan Bapak Kurniawan Dwi Irianto, S.T., M.Sc. selaku dosen pembimbing yang telah membantu selama tugas akhir.
4. Teman-teman yang senantiasa memberikan dukungan serta menemani saya dalam pengerjaan tugas akhir ini.

Semoga tugas akhir ini dapat bermanfaat bagi orang lain. Penulis ucapkan mohon maaf atas segala kekurangan dan terima kasih.

Yogyakarta, 29 Juli 2025



(Yuanda Hanif Hisyam)

SARI

Indoor Positioning System (IPS) berbasis Wi-Fi *fingerprinting* merupakan solusi populer untuk navigasi di lingkungan dalam ruangan, tetapi akurasinya sangat bergantung pada ketersediaan data yang padat. Proses pengumpulan data secara manual pada fase *offline* seringkali padat karya dan memakan waktu sehingga menghasilkan dataset yang langka dan menurunkan kinerja algoritma penentuan posisi. Penelitian ini mengembangkan sistem IPS menggunakan metode Wi-Fi *fingerprinting* dan mencoba mengatasi masalah kelangkaan data dengan mengevaluasi efektivitas augmentasi dataset menggunakan data sintetis untuk meningkatkan akurasi IPS. Metodologi penelitian ini melibatkan pembuatan sistem yang akan digunakan untuk pengambilan data, pembuatan data sintetis menggunakan model generatif Multi-Layer Perceptron (MLP) yang dilatih secara terbalik untuk memprediksi nilai Received Signal Strength Indicator (RSSI) dari input koordinat, dan model prediksi lokasi menggunakan k-Nearest Neighbor (KNN). Model prediksi dibuat menggunakan algoritma KNN dalam tiga skenario: model dasar, model dengan penyaringan sinyal Kalman Filter, dan model yang dilatih dengan gabungan data asli dan data sintetis. Hasil pengujian menunjukkan bahwa model yang diperkaya dengan data sintetis mencapai kinerja terbaik pada dataset yang belum pernah ditemui selama pelatihan, dengan kesalahan selisih prediksi terendah sebesar 184,97 cm, mengungguli model yang dilatih menggunakan dataset mentah 190,34 cm (peningkatan 13,3%) dan model dengan Kalman Filter 186,67 cm (peningkatan 11,3 %). Temuan dari penelitian menunjukkan augmentasi dataset menggunakan data sintetis dapat menyamai bahkan meningkatkan akurasi dan kemampuan generalisasi dari sistem penentuan posisi dalam ruangan (IPS) berbasis Wi-Fi fingerprinting KNN.

Kata kunci: Indoor Positioning System, KNN, Multi-Layer Perceptron, Kalman Filter

GLOSARIUM

Access Point (AP) Perangkat keras jaringan nirkabel yang memancarkan sinyal WI-FI.

Augmentasi Data Teknik untuk meningkatkan jumlah dan variasi data pelatihan dengan cara membuat data baru secara artifisial.

Data Sintetis Data yang dibuat secara artifisial oleh model algoritmik dan bukan dikumpulkan dari pengukuran di dunia nyata.

Indoor Positioning System (IPS) Sistem teknologi yang digunakan untuk menentukan lokasi objek atau orang di dalam ruangan.

Kalman Filter Algoritma rekursif yang digunakan untuk mengurangi *noise* (derau) dari serangkaian data pengukuran dari waktu ke waktu.

k-Nearest Neighbor (KNN) Algoritma pembelajaran mesin yang digunakan untuk klasifikasi dan regresi.

Metode Fingerprinting Pendekatan dalam IPS di mana sebuah peta sinyal (radio map) dibuat dengan merekam pola kekuatan sinyal unik (sidik jari RSSI) di banyak titik referensi yang diketahui.

RSSI (Received Signal Strength Indicator) Ukuran kekuatan sinyal yang diterima oleh perangkat dari sebuah access point atau pemancar nirkabel.

SSID (Service Set Identifier) Nama unik yang mengidentifikasi sebuah jaringan nirkabel (Wi-Fi)

Titik Referensi (Reference Point - RP) Lokasi fisik di dalam ruangan dengan koordinat (x, y) yang telah diketahui secara pasti.

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI viii
GLOSARIUM ix
DAFTAR ISI x
DAFTAR TABEL xii
DAFTAR GAMBAR xiii
BAB I PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Rumusan Masalah 3
1.3. Batasan Masalah 3
1.4. Tujuan Penelitian 4
1.5. Manfaat Penelitian 4
1.6. Metodologi Penelitian 4
1.6.1. Pengembangan sistem 4
1.6.2. Pembuatan data sintetis 4
1.6.3. Pembuatan model prediksi 4
1.7. Sistematika Penulisan 5
1.7.1. BAB I PENDAHULUAN 5
1.7.2. BAB II KAJIAN PUSTAKA 5
1.7.3. BAB III METODOLOGI 5
1.7.4. BAB IV HASIL DAN PEMBAHASAN 5
1.7.5. BAB V KESIMPULAN DAN SARAN 5
BAB II KAJIAN PUSTAKA 6
2.1. Penelitian Terdahulu 6
2.2. Sistem Penentuan Posisi dalam Ruangan (Indoor Positioning System - IPS) 7
2.3. Received Signal Strength Indicator (RSSI) 8
2.4. Kalman Filter 8
2.5. Multi-Layer Perceptron 9
2.6. Arsitektur MLP 10
2.6.1. Fungsi Aktivasi ReLU (Rectified Linear Unit) 11
2.6.2. Optimizer Adam (Adaptive Moment Estimation) 12
2.7. k-Nearest Neighbor (KNN) 12
2.8. Metrik Evaluasi Model Prediksi 12
2.8.1. Mean Euclidean Error 13
2.8.2. Median Euclidean Error 13
2.8.3. Standard Deviation of Error 13
2.8.4. Maximum Error 13
BAB III METODE PENELITIAN 15

3.1. Alur Pengerjaan Tugas Akhir 15
3.2. Pengembangan Sistem Untuk Fase Offline 16
3.2.1. Teknologi yang digunakan 16
3.2.2. Alur Sistem 17
3.3. Membuat Data Sintetis 20
3.3.1. Pengumpulan Data 21
3.3.2. Pemrosesan Data 25
3.3.3. Membuat Data Sintetis 26
3.4. Pengembangan Model Prediksi 27
BAB IV HASIL DAN PEMBAHASAN 29
4.1. Pengembangan Sistem Untuk Fase Offline 29
4.1.1. Memilih Batch Scan 29
4.1.2. Memilih Koordinat pada Map 31
4.1.3. Melakukan Scanning 32
4.1.4. Export Data 34
4.2. Membuat Data Sintetis 35
4.2.1. Pemrosesan Data 36
4.2.2. Membuat Data Sintetis 41
4.3. Pengembangan Model Prediksi 46
4.3.1. Skenario A (Data original) 46
4.3.2. Skenario B (*Signal Filter*) 46
4.3.3. Skenario C (Data set Sintetis) 47
4.4. Diskusi 51
BAB V KESIMPULAN DAN SARAN 52
5.1. Kesimpulan 52
5.2. Saran 53
DAFTAR PUSTAKA 54
LAMPIRAN 58

DAFTAR TABEL

Tabel 2.1 Kajian Pustaka Penelitian Terdahulu 6
Tabel 3.1 Spesifikasi Perangkat Keras Laptop 23
Tabel 3.2 Spesifikasi Perangkat AP yang digunakan 24
Tabel 4.1 Hasil Pengumpulan Data 35
Tabel 4.2 Hasil penyaringan Data 36
Tabel 4.3 Hasil Sintetis Data 44
Tabel 4.4 Komparasi Eror 48
Tabel 4.5 Komparasi Eror prediksi pada perangkat Asus 49
Tabel 4.6 Komparasi Eror prediksi pada perangkat Macbook 50

DAFTAR GAMBAR

Gambar 2.1Arsitektur model 10
Gambar 2.2 Grafik fungsi aktifasi ReLU 11
Gambar 2.3 Ilustrasi algoritma KNN 12
Gambar 3.1 Alur Pengerjaan Tugas Akhir 16
Gambar 3.2 Alur Sistem 17
Gambar 3.3 Contoh format SVG. 19
Gambar 3.4 Keluaran program iw 20
Gambar 3.5 Alur Pembuatan Data Sintetis 21
Gambar 3.6 Map Laboratorium Siber 21
Gambar 3.7 Laboratorium Siber 22
Gambar 3.8 Posisi AP Akses Poin 22
Gambar 3.9 Setup pemindaian RSSI 23
Gambar 3.10 Map titik referensi 24
Gambar 3.11 Pengambilan data menggunakan perangkat Asus zenbook 14 28
Gambar 3.12 Pengambilan data menggunakan perangkat Macbook Pro 2017 28
Gambar 4.1 Struktur basis data. 29
Gambar 4.2 Tampilan *Dropdown* Kelompok pemindaian. 30
Gambar 4.3 Dialog tambah kelompok pemindaian 30
Gambar 4.4 Antarmuka map 31
Gambar 4.5 Map dalam format JSON 32
Gambar 4.6 Konversi meter ke sekala dalam SVG 32
Gambar 4.7 Progres Pemindaian 33
Gambar 4.8 Kode Untuk Melakukan Pemindaian 33
Gambar 4.9 Kode Untuk Export Data 34
Gambar 4.10 Berkas CSV 35
Gambar 4.11 Distribusi orisinal RSSI RuijieAP1 37
Gambar 4.12 Distribusi orisinal RSSI RuijieAP2 37
Gambar 4.13 Distribusi orisinal RSSI RuijieAP3 38
Gambar 4.14 Distribusi RSSI RuijieAP1 Setelah Penyeimbangan 38
Gambar 4.15 Distribusi RSSI RuijieAP2 Setelah Penyeimbangan 39
Gambar 4.16 Distribusi RSSI RuijieAP2 Setelah Penyeimbangan 39
Gambar 4.17 Sinyal Sebelum dan Sesudah Kalman Filter 40
Gambar 4.18 Hasil Normalisasi Koordinat 41
Gambar 4.19 Kode untuk pelatihan model 42
Gambar 4.20 Hasil pelatihan model 42
Gambar 4.21 Kode untuk membuat data sintetis 44
Gambar 4.22 Ploting nilai RSSI RuijieAP1 44
Gambar 4.23 Ploting nilai RSSI RuijieAP2 45
Gambar 4.24 Ploting nilai RSSI RuijieAP3 45
Gambar 4.25 Hasil Pelatihan KNN 46
Gambar 4.26 Hasil Pelatihan KNN + Kalman Filter 47
Gambar 4.27 Hasil Pelatihan KNN + Data Sintetis 48
Gambar 4.28 Visualisasi hasil Skenario A 49
Gambar 4.29 Visualisasi hasil Skenario B 49

Gambar 4.30 Visualisasi hasil Skenario C.....49
Gambar 4.31 Visualisasi *tracking* menggunakan model Skenario A pada perangkat Asus....50
Gambar 4.32 Visualisasi *tracking* menggunakan model Skenario B pada perangkat Asus....50
Gambar 4.33 Visualisasi *tracking* menggunakan model Skenario C pada perangkat Asus....50
Gambar 4.34 Visualisasi tracking menggunakan model Skenario A pada perangkat Macbook 51
Gambar 4.35 Visualisasi tracking menggunakan model Skenario B pada perangkat Macbook 51
Gambar 4.36 Visualisasi tracking menggunakan model Skenario C pada perangkat Macbook 51

BAB I
PENDAHULUAN

1.1. Latar Belakang

Pada era digital ini, perkembangan teknologi telah memberikan dampak signifikan terhadap berbagai sektor kehidupan. Salah satu bidang yang sangat terbantu oleh perkembangan ini adalah navigasi. Salah satu yang sering digunakan adalah Sistem Pemosisian Global *(Global Positioning System* - GPS) yang menyediakan informasi posisi dan waktu yang tepat pada navigasi luar ruangan (Sohan et al., 2019). Namun, karena menggunakan sinyal yang didapatkan dari satelit, GPS kurang dapat diandalkan untuk melakukan navigasi dalam ruangan. Sinyal yang diperoleh terganggu oleh struktur bangunan yang menyebabkan konektivitas tidak dapat diandalkan dan akurasi berkurang (Vaccari, Coruzzolo, Lolli, & Sellitto, 2024). Hal ini sangat berpengaruh dalam berbagai aspek kehidupan, terutama melihat konteks transformasi digital yang sedang terjadi (Goecks, Habekost, Coruzzolo, & Sellitto, 2024). Sistem Penentuan Posisi dalam Ruangan (*Indoor Positioning System* - IPS) digunakan untuk melakukan navigasi layanan lokasi di dalam ruangan untuk menemukan barang, orang, periklanan, dan pelacakan aset seperti perangkat *Internet of Things* (IoT) di dalam gedung atau ruang (Firmansyah, 2024).

IPS memiliki berbagai penerapan praktis dalam kehidupan sehari-hari. Sebagai contoh, dalam perawatan kesehatan, IPS dapat digunakan untuk membantu keluarga dalam memonitor aktivitas dari lansia yang memiliki kendala kesehatan dan memerlukan perawatan serta perhatian lebih (Willyanto & Santoso, 2025). Kendaraan Berpemandu Otomatis (Automated Guided Vehicle - AGV) adalah area lain di mana sistem navigasi memainkan peran penting. Dengan menyediakan data lokasi yang tepat, IPS dapat memfasilitasi pengoperasian kendaraan berpemandu otomatis yang lancar, memastikan kendaraan tersebut melakukan navigasi secara efisien melalui gudang atau fasilitas produksi secara *real-time* (Vaccari et al., 2024).

Pada lingkungan perbelanjaan besar, IPS dapat digunakan untuk membantu pengunjung menemukan toko atau produk dengan mudah, serta untuk manajemen keamanan dalam melacak keberadaan barang, staf, dan pengunjung secara *real-time*. Pada tempat rekreasi seperti museum, IPS dapat digunakan sebagai sistem yang interaktif bagi pengunjung untuk pengunjung dapat mendapatkan informasi yang sesuai dengan tempat di mana dia berada. Dengan kemampuan untuk menyediakan data lokasi yang akurat dan tepat waktu, IPS menjadi

alat yang sangat berharga dalam memperkuat keamanan, efisiensi, dan kenyamanan dalam berbagai aspek kehidupan sehari-hari (Firmansyah, 2024).

Tidak seperti satelit, sistem pemosisian dalam ruangan belum memilki standardisasi karena luasnya area penerapan dan persyaratan yang berbeda-beda yang timbul dari lingkungan penerapan yang bervariasi. Oleh karena itu, solusi pemosisian dalam ruangan cenderung sangat *deterministik*, yaitu dirancang dan disesuaikan untuk area/lingkungan tertentu (Philippopoulos et al., 2025). Terdapat berbagai teknik yang digunakan untuk memprediksi titik lokasi, salah satunya adalah teknik Wi-Fi *fingerprinting* (Firmansyah, Rahma, Irianto, & Shiddiqi, 2024).

Metode Wi-Fi *fingerprinting* terdiri dari dua fase utama, fase *offline* dan fase *online*. Pada fase *offline*, sebuah basis data atau *fingerprint* map dibangun dengan mengumpulkan data kekuatan sinyal yang diterima (R*eceived Signal Strength Indicator* - RSSI) dari berbagai *access point* (AP) di titik-titik referensi yang koordinatnya telah diketahui. Kemudian, pada fase online, pengguna mengukur nilai RSSI di lokasi mereka yang tidak diketahui. Lalu, sistem akan membandingkan data tersebut dengan *fingerprint* map yang ada untuk memperkirakan posisi pengguna (Ren, Wang, Liu, & Lv, 2023).

Meskipun metode Wi-Fi *fingerprinting* menjanjikan, implementasinya di dunia nyata menghadapi tantangan signifikan (Homayounvala, Nabati, Shahbazian, Ghorashi, & Moghtadaiee, 2019). Salah satu kendala utama adalah proses pengumpulan data pada fase *offline* yang memerlukan banyak waktu dean tenaga. Untuk membangun *fingerprint* map yang akurat dan komprehensif, diperlukan pengambilan sampel data di banyak titik referensi yang sering kali tidak praktis untuk lingkungan yang luas atau dinamis (Lu et al., 2023). Keterbatasan jumlah data referensi ini dapat secara langsung menurunkan akurasi algoritma penentuan posisi, seperti pada model K-Nearest Neighbor (KNN) yang kinerjanya sangat bergantung pada kepadatan dan kualitas data latih.

KNN adalah algoritma pembelajaran mesin yang digunakan untuk klasifikasi dan regresi. KNN adalah salah satu algoritma penentuan posisi yang banyak digunakan dalam metode *fingerprint* (Shang & Wang, 2022) (Firmansyah, 2024). KNN adalah algoritma pencocokan pola yang paling sederhana, tetapi efektif digunakan pada fase *online* dalam metode Wi-Fi *fingerprinting* (Zhang, Li, & Li, 2025).

Untuk membantu mengatasi permasalahan tersebut, salah satu yang dapat dilakukan adalah menambah jumlah data titik referensi. Penambahan data pada titik referensi dapat dilakukan menggunakan sintetis data. Sintetis data atau augmentasi data adalah teknik untuk menghasilkan data buatan secara artifisial yang meniru karakteristik data asli. Dalam konteks

IPS, teknik ini menjadi solusi yang menjanjikan untuk mengatasi masalah jumlah data pada titik referensi tanpa perlu melakukan survei secara manual. Pada penelitian ini, sintetis data dilakukan dengan pendekatan regresi terbalik menggunakan model *Multi-Layer Perceptron* (MLP) .

Multi-Layer Perceptron adalah salah satu bentuk arsitektur *Artificial Neural Network* (ANN). Kemampuan utama MLP terletak pada kemampuannya untuk mempelajari dan memodelkan hubungan yang kompleks dan non-linear antara data masukan ke dalam keluaran, menjadikannya alat yang sangat kuat untuk berbagai tugas termasuk klasifikasi dan regresi.

Penelitian ini bertujuan untuk melakukan pengembangan sistem IPS menggunakan metode Wi-Fi fingerprinting dan mengatasi kelangkaan data menggunakan sintetis data. Pada penelitian ini akan dibuat sistem yang digunakan untuk melakukan survei titik referensi, serta penambahan *dataset* menggunakan data sintetis menggunakan MLP dan evaluasi bagaimana sintetis data berpengaruh terhadap kinerja sistem dalam menentukan lokasi di lingkungan dalam ruangan menggunakan algoritma KNN.

1.2. Rumusan Masalah

- Bagaimana mengembangkan sebuah sistem penentuan Sistem Penentuan Posisi dalam Ruangan menggunakan metode Wi-Fi *fingerprinting*?
- Bagaimana mengembangkan dataset sintetis untuk menambah variasi data pada dataset Wi-Fi *fingerprinting*?
- Bagaimana dampak *dataset* sintetis terhadap performa model pada metode Wi-Fi *fingerprinting*?

1.3. Batasan Masalah

Batasan masalah yang akan dibahas pada penelitian ini adalah sebagai berikut:

- Penelitian ini akan membatasi diri pada pengembangan Sistem Penentuan Posisi dalam Ruangan yang menggunakan metode Wi-Fi *fingerprinting*.
- Penelitian akan berfokus pada pengembangan model dan penambahan *dataset* sintetis pada fase *offline* dalam metode Wi-Fi *fingerprinting*.
- Pengujian dilakukan dengan hanya pada dua perangkat laptop dan tiga AP (*Access Point*).
- Sistem yang dikembangkan hanya dapat digunakan untuk memantau perangkat yang di dalamnya terdapat sistem tersebut dan tidak bisa memantau perangkat lainnya

1.4. Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk melakukan pengembangan Sistem Penentuan Posisi dalam Ruangan dan melakukan eksplorasi augmentasi *dataset* sintetis menggunakan MLP. Sistem akan dapat digunakan untuk menjalankan fase *offline* pengumpulan data dengan antarmuka yang mudah digunakan. Selain itu, penelitian ini bertujuan untuk mengkaji dampak data sintetis pada metode Wi-Fi fingerprinting.

1.5. Manfaat Penelitian

Manfaat penelitian ini didefinisikan sebagai berikut:

- Memberikan kontribusi pada pengembangan teknologi *Indoor Positioning System* terutama pada fase *offline* dalam metode Wi-Fi *fingerprinting*.
- Memberikan dasar bagi pengembangan lebih lanjut untuk fase *online* pada *Indoor Positioning System*.
- Memberikan hasil mengenai pengujian data sintetis pada fase *offline* metode Wi-Fi *fingerprinting* pada *Indoor Positioning System*.

1.6. Metodologi Penelitian

Dalam penelitian, metode yang akan digunakan adalah sebagai berikut:

1.6.1. Pengembangan sistem

Pada tahap ini dikembangkan sebuah sistem IPS untuk membantu melakukan pengambilan data padat tahap *offline*. Sistem akan dibuat untuk mengakomodasi satu skenario pengambilan data.

1.6.2. Pembuatan data sintetis

Setelah pengembangan sistem, tahap selanjutnya adalah membuat data sintetis. Pada tahap ini dilakukan pengumpulan *dataset* yang merupakan tahapan awal fase *offline*. Selanjutnya dilakukan pemrosesan data, dan terakhir dibangun sebuah model untuk membuat data sintetis menggunakan MLP.

1.6.3. Pembuatan model prediksi

Tahap terakhir adalah pembuatan model untuk prediksi lokasi menggunakan KNN, data yang telah dikumpulkan sebelumnya akan digunakan untuk menjalankan tiga skenario model prediksi: (A) model prediksi menggunakan data mentah dengan KNN, (B) Model prediksi menggunakan data mentah dengan pemrosesan kalman filter + KNN, (C) Model prediksi menggunakan data sintetis dengan KNN. Hasil dari ketiga skenario akan dibahas pada bab ini.

1.7. Sistematika Penulisan

Struktur penulisan laporan Tugas Akhir dijelaskan pada pembahasan di bawah ini:

1.7.1. BAB I PENDAHULUAN

Bab Pendahuluan dijelaskan gambaran umum tentang topik yang dibahas. Bagian pendahuluan memuat Latar Belakang, Rumusan Masalah, Batasan Masalah, Tujuan Penelitian, Manfaat Penelitian, Metodologi Penelitian, dan Sistematika Penulisan.

1.7.2. BAB II KAJIAN PUSTAKA

Bab Kajian Pustaka mengulas teori-teori, konsep, dan penelitian sebelumnya yang relevan dengan topik penelitian ini.

1.7.3. BAB III METODOLOGI

Bab Metodologi menjelaskan tahapan-tahapan yang dilakukan dalam penelitian, mulai dari pengembangan sistem, pengumpulan *dataset*, teknik yang digunakan untuk pemrosesan data, teknik dalam pembuatan data sintetis, serta pembuatan model prediksi. Tahapan-tahapan yang dilakukan dalam penelitian dijelaskan secara sistematis sesuai dengan alur penelitian yang dilakukan.

1.7.4. BAB IV HASIL DAN PEMBAHASAN

Bab Hasil dan Pembahasan menyajikan hasil yang diperoleh dari penerapan metodologi penelitian. Hasil penelitian akan disajikan dalam bentuk gambaran alur sistem, gambar kode, tabel, foto ataupun bentuk lainnya yang diuraikan sesuai dengan tahapan penelitian.

1.7.5. BAB V KESIMPULAN DAN SARAN

Bab Kesimpulan dan Saran berisi rangkuman dan kesimpulan dari hasil penelitian yang dilakukan. Bab ini juga berisi saran serta wawasan berdasarkan temuan-temuan untuk pengembangan lebih lanjut yang berkaitan dengan topik bahasan pada penelitian ini.

BAB II
KAJIAN PUSTAKA

2.1. Penelitian Terdahulu

Bagian ini akan membahas penelitian yang sudah ada dan berkaitan dengan topik yang sedang dibicarakan. Tinjauan ini memberikan gambaran umum mengenai berbagai penelitian yang relevan, agar dapat menjelaskan posisi penelitian yang dilakukan dengan lebih jelas.

Tabel 2.1 Kajian Pustaka Penelitian Terdahulu

Penulis	Metode	Kekurangan	Hasil
(Liu, de Lacerda, & Fiorina, 2021)	K-Nearest Neighbor (KNN), weighted K-nearest neighbor (WKNN), XGBoost, dan WKNN + Improved XGBoost	Pengumpulan data fase offline memakan banyak tenaga.	Gabungan WKNN dan XGBoost menunjukkan penurunan eror sebanyak 1.22 m, sebanyak 20.26–45.58% dibandingkan performa individu algoritma KNN, WKNN, dan XGBoost.
(Naderi Salim, Alizadeh, Chamankar, & Schuh, 2023)	Nearest neighbor (NN), WKNN, maximum filter, mean (average) filter, dan limited filter	Data tidak terlalu banyak.	WKNN unggul dibandingkan NN dengan RMSE 0.6735m dan 2.1354m.
(Chen et al., 2023)	Random Forest, menambah Time of Flight (TOF), Gaussian filter dan median filter.	Menggunakan perangkat keras khusus.	Penambahan nilai TOF pada pengukuran membantu menambah akurasi sebanyak 25–31% dengan rata-rata eror 1m.
(Yudha, Hasbi, & Sukarna, 2018)	KNN	Tidak menerapkan *preprocessing.*	Akurasi tinggi untuk melakukan prediksi ruang sebesar 96,71% dengan *error rate* sebesar 3,29%.
(K. Wang et al., 2019)	KNN, SVM, DBSCAN, DBSCAN-KRF, Random Forest.	Pengujian dilakukan pada ruang kecil dan objek yang tidak bergerak.	Gabungan antara KNN, DBSCAN, dan Random Forest menunjukkan peningkatan performa dibanding penggunaan algoritma secara individu.

Pada beberapa penelitian sebelumnya, seperti yang dilakukan oleh (Liu et al., 2021), berfokus kepada implementasi WKNN dan XGBoost menunjukkan penurunan eror sebanyak 1.22 m, sebanyak 20.26–45.58% dibandingkan performa individu algoritma KNN, WKNN, dan XGBoost. Dikutip dari akhir penelitian (Liu et al., 2021), pengumpulan data memakan

banyak tenaga karena banyaknya titik referensi yang harus diukur. Pada penelitian oleh (Naderi Salim et al., 2023) algoritma WKNN unggul dibandingkan NN dengan RMSE 0.6735m dan 2.1354m, tetapi data yang dikumpulkan terbilang tidak terlalu banyak jika dibanding dengan penelitian lain. Terakhir, pada empat dari lima penelitian yang dikutip, pembandingan akurasi dilakukan pada algoritma KNN.

Penelitian ini mencoba untuk mengurangi tenaga yang dialokasikan untuk melakukan pengumpulan data pada fase *offline*. Salah satu solusi yang diajukan adalah dengan memanfaatkan pembelajaran mesin untuk mengisi celah pada Wi-Fi *fingerprinting* map yang sudah dibuat.

2.2. Sistem Penentuan Posisi dalam Ruangan (Indoor Positioning System - IPS)

Indoor Positioning System (IPS) adalah sebuah sistem yang dirancang untuk menentukan lokasi objek atau individu di dalam lingkungan tertutup, seperti gedung perkantoran, pusat perbelanjaan, atau bandara (Chen et al., 2023). Terdapat berbagai teknologi yang dapat dimanfaatkan untuk IPS, termasuk WI-FI, Bluetooth Low Energy (BLE), Ultra-Wideband (UWB), dan sensor internal perangkat seluler. Di antara teknologi tersebut, IPS berbasis WI-FI menjadi pilihan populer karena memanfaatkan infrastruktur jaringan nirkabel yang sudah ada secara luas sehingga tidak memerlukan biaya pemasangan perangkat keras tambahan (Firmansyah et al., 2024).

Sejauh ini, hampir semua IPS berbasis WI-FI mengandalkan kekuatan sinyal yang diterima (*Received Signal Strength* - RSS) (Shang & Wang, 2022). Secara umum, sistem ini dapat diklasifikasikan ke dalam dua kategori: (1) pemosisian dalam ruangan berdasarkan pengukuran jarak (metode berbasis jarak) dan (2) pemosisian dalam ruangan berdasarkan Wi-Fi *fingerprinting*.

Metode Wi-Fi *fingerprinting* adalah salah satu pendekatan yang paling umum digunakan dalam IPS. Metode ini bekerja dengan mengenali pola unik dari kekuatan sinyal di setiap lokasi. Prosesnya dibagi menjadi dua fase utama (W. Wang & Liu, 2025).

Fase *Offline* (Fase Pelatihan): Pada fase ini, dilakukan survei lokasi untuk membangun sebuah basis data yang disebut *fingerprint* map. Pengumpulan data kekuatan sinyal yang diterima RSSI dari berbagai AP di sejumlah titik referensi yang koordinatnya telah diketahui. Setiap titik referensi akan memiliki 'sidik jari' unik yang terdiri dari vektor nilai RSSI dari AP yang terdeteksi, yang kemudian disimpan bersama dengan koordinatnya di dalam basis data.

Fase Online (Fase Penentuan Posisi): Ketika seorang pengguna ingin mengetahui lokasinya, perangkatnya akan memindai sinyal WI-FI di sekitarnya dan membentuk sebuah

vektor RSSI *real-time*. Vektor ini kemudian dikirim ke sistem, yang akan membandingkannya dengan semua data yang tersimpan pada basis data. Dengan menggunakan algoritma pencocokan pola, sistem akan mencari sidik jari yang paling mirip dan mengestimasi lokasi pengguna berdasarkan koordinat dari sidik jari tersebut.

Namun, pada metode Wi-Fi *fingerprinting* pengumpulan data pada fase *offline* memakan banyak tenaga. Pembuatan peta jaringan yang harus dilakukan secara manual pada tiap titik referensi yang akan mempengaruhi akurasi dari fase *online*. Pada penelitian ini, diajukan metode pembuatan data sintetis untuk membantu mengurangi beban pada tahap pengumpulan data.

2.3. Received Signal Strength Indicator (RSSI)

Received Signal Strength Indicator (RSSI) adalah sebuah metrik yang merepresentasikan level atau kekuatan daya dari sinyal frekuensi radio (RF) yang diterima oleh sebuah perangkat. Dalam konteks teknologi nirkabel seperti Wi-Fi (IEEE 802.11), RSSI mengindikasikan seberapa kuat sinyal yang dipancarkan oleh sebuah AP diterima oleh perangkat klien, seperti ponsel pintar atau laptop. Nilai RSSI diukur dalam satuan desibel-miliwatt (dBm) dan biasanya direpresentasikan sebagai nilai negatif, di mana nilai yang lebih mendekati nol mengindikasikan sinyal yang lebih kuat.

Keunggulan pemanfaatan RSSI untuk sistem penentuan posisi dalam ruangan terletak pada kemampuannya untuk menggunakan infrastruktur Wi-Fi yang sudah ada secara luas, yang secara langsung menjadikan implementasinya sangat murah dan sederhana tanpa memerlukan perangkat keras tambahan.

Namun, keunggulan ini berhadapan dengan kelemahan mendasar dari sifat sinyal yang sangat tidak stabil. Nilai RSSI sangat rentan terhadap fluktuasi akibat efek pantulan dan halangan (multipath fading), membuat kinerjanya sensitif terhadap perubahan dinamis lingkungan seperti pergerakan orang atau perabotan. Lebih lanjut, akurasinya sangat bergantung pada kepadatan dan penempatan Akses Poin di sebuah area, ditambah dengan masalah inkonsistensi pembacaan sinyal antar jenis perangkat pengguna yang berbeda.

2.4. Kalman Filter

Algoritma filter Kalman, juga dikenal sebagai *Linear Quadratic Estimation* (LQE), adalah suatu algoritma yang menghasilkan estimasi yang cenderung lebih akurat dengan menggunakan hasil pengamatan atau pengukuran selama periode waktu tertentu. Tujuan dari filter Kalman adalah untuk mengurangi variasi dalam kesalahan estimasi. Oleh karena itu,

algoritma filter Kalman adalah salah satu pendekatan terbaik untuk penentuan posisi (Aziz, Jamil, & Din, 2019).

Kalman Filter memiliki dua proses utama, yaitu prediksi dan koreksi (Firmansyah, 2024). Proses prediksi adalah fungsi untuk menghitung nilai estimasi dari suatu *state* atau keadaan berdasarkan nilai-nilai sebelumnya. Koreksi berfungsi untuk memperbaiki nilai estimasi tersebut agar menghasilkan nilai prediksi yang lebih akurat.

Seperti yang terlihat pada (2.1), (2.2), (2.3), (2.4), (2.5) terdapat beberapa istilah dalam perhitungan Kalman Filter. X adalah matriks keadaan (*state matrix*). P adalah matriks kovarians proses (p*rocess covariance matrix*). A dan B adalah matriks adaptasi yang mengonversi keadaan menjadi input keadaan proses. U adalah matriks variabel kontrol. Q adalah matriks kovarians noise proses. W adalah matriks noise keadaan yang diprediksi. K adalah gain. R adalah matriks kovarians pengukuran (*noise sensor*).

Persamaan untuk tahap prediksi.

$$X_{k-1} = AX_{k-1} + BU_k + W_k \quad (2.1)$$

$$P_k = P_{k-1} + Q_k \quad (2.2)$$

Persamaan untuk tahap koreksi.

$$K = P_k(P_k + R) - 1 \quad (2.3)$$

$$X_k = X_k + K(Y_k - X_k) \quad (2.4)$$

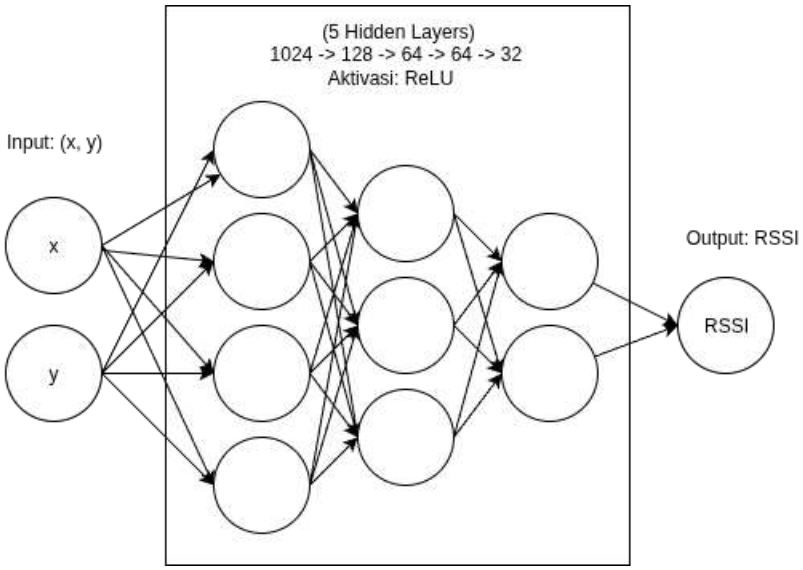
$$P_k = (1 - K)P_k \quad (2.5)$$

2.5. Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) adalah salah satu bentuk arsitektur Jaringan Saraf Tiruan (JST) atau *Artificial Neural Network* (ANN). MLP termasuk dalam kategori jaringan *feedforward*, yang berarti informasi bergerak dalam satu arah, dari lapisan masukan (*input*

layer) menuju lapisan keluaran (*output* layer) melalui satu atau lebih lapisan tersembunyi (*hidden layers*). Kemampuan utama MLP terletak pada kemampuannya untuk mempelajari dan memodelkan hubungan yang kompleks dan non-linear antara data masukan dan keluaran, menjadikannya alat yang sangat kuat untuk berbagai tugas termasuk klasifikasi dan regresi.

2.6. Arsitektur MLP



Gambar 2.1Arsitektur model

Arsitektur model dapat dilihat pada Gambar 2.1 dan dijelaskan sebagai berikut:

a. **Lapisan Input**: Lapisan ini berfungsi sebagai pintu masuk data ke dalam jaringan. Jumlah neuron (unit) pada lapisan ini sesuai dengan jumlah fitur atau variabel independen dari *dataset*. Dalam konteks penelitian ini, lapisan masukan akan terdiri dari dua neuron yang merepresentasikan koordinat lokasi (x, y).
b. **Lapisan Tersembunyi**: Setiap neuron pada lapisan ini terhubung sepenuhnya dengan neuron pada lapisan sebelumnya. Lapisan inilah yang bertanggung jawab untuk melakukan sebagian besar proses komputasi, mengekstraksi fitur, dan mempelajari pola-pola rumit dari data. Sebuah MLP dapat memiliki satu atau lebih lapisan tersembunyi. Jumlah lapisan dan jumlah neuron di setiap lapisan merupakan *hyperparameter* yang dapat disesuaikan untuk mengoptimalkan kinerja model.
c. **Lapisan Output**: Lapisan ini menghasilkan hasil akhir dari prediksi model. Jumlah neuron pada lapisan keluaran bergantung pada tugas yang diemban. Untuk tugas regresi seperti pada penelitian ini, di mana tujuannya adalah memprediksi satu nilai kontinu. Neuron ini akan menghasilkan nilai prediksi RSSI.

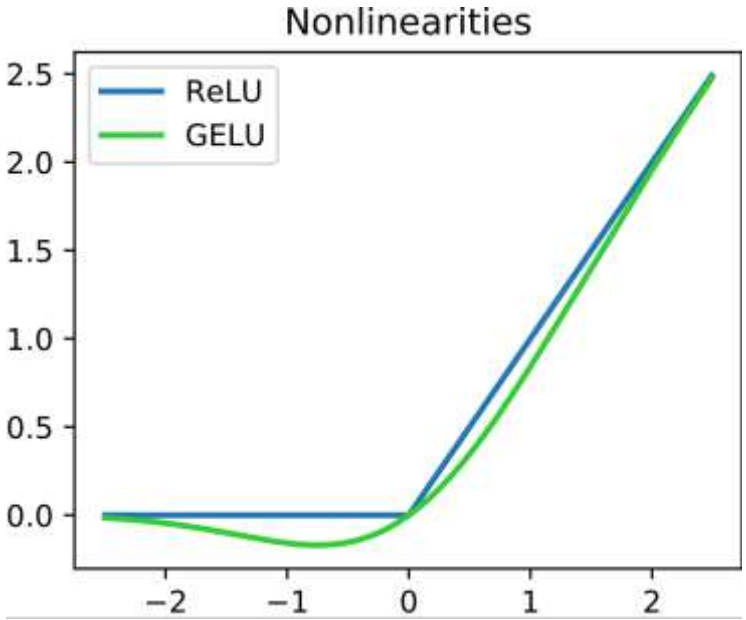
Setiap neuron di dalam lapisan tersembunyi dan lapisan keluaran melakukan dua operasi matematis utama: penjumlahan terbobot (*weighted sum*) dari masukan yang diterimanya, diikuti oleh penerapan fungsi aktivasi non-linear ReLU (R*ectified Linear Unit*) yang memungkinkan MLP untuk mempelajari hubungan non-linear yang tidak dapat dimodelkan oleh regresi linear biasa.

2.6.1. Fungsi Aktivasi ReLU (Rectified Linear Unit)

Fungsi aktivasi adalah komponen krusial dalam arsitektur Jaringan Saraf Tiruan, termasuk MLP. Peran utamanya adalah untuk memperkenalkan non-linear ke dalam model, yang memungkinkan jaringan untuk mempelajari hubungan yang kompleks antara *input* dan *output*. Tanpa fungsi aktivasi non-linear, sebuah MLP dengan banyak lapisan pun hanya akan setara dengan model regresi linear tunggal.

Fungsi ReLU didefinisikan secara sederhana. Untuk setiap input yang diberikan (dilambangkan dengan z, yaitu hasil dari penjumlahan terbobot pada sebuah neuron), fungsi ReLU akan menghasilkan input itu sendiri jika nilainya positif, dan akan menghasilkan nol jika nilainya negatif.

$$f(z) = \begin{Bmatrix} z \; if \; z > 0 \\ z \; if \; z \leq 0 \end{Bmatrix} \quad (2.6)$$



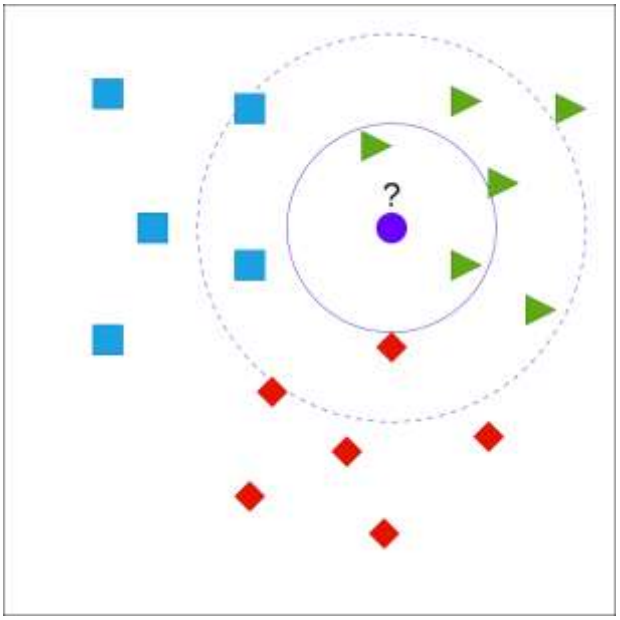
Gambar 2.2 Grafik fungsi aktifasi ReLU
Sumber: (“Rectifier (Neural Networks),” 2025)

2.6.2. Optimizer Adam (Adaptive Moment Estimation)

Adam adalah optimizer yang bekerja berdasarkan metode *gradient descent*, tetapi dengan beberapa penyempurnaan cerdas. 2.5.2. Optimizer Adam tidak hanya menggunakan satu *learning rate* (laju pembelajaran) yang statis untuk seluruh bobot, melainkan menghitung *learning rate* yang adaptif untuk setiap parameter (bobot) secara individual.

2.7. k-Nearest Neighbor (KNN)

K-Nearest Neighbors (KNN) merupakan salah satu algoritma pembelajaran mesin yang banyak digunakan dalam berbagai aplikasi klasifikasi dan regresi. KNN bekerja berdasarkan prinsip kesamaan atau kedekatan, di mana suatu data baru akan diklasifikasikan berdasarkan kelas dari tetangga terdekatnya di ruang fitur. Proses ini dimulai dengan menentukan nilai K, yaitu jumlah tetangga terdekat yang akan dipertimbangkan dalam pengambilan keputusan. Setelah menentukan K, algoritma menghitung jarak antara data baru dan seluruh data yang sudah ada, menggunakan berbagai metrik jarak seperti *Euclidean*, *Manhattan*, atau *Minkowski*. Selanjutnya, data baru akan dimasukkan ke dalam kelas yang paling umum di antara K tetangga terdekatnya seperti ilustrasi pada Gambar 2.3.



Gambar 2.3 Ilustrasi algoritma KNN

2.8. Metrik Evaluasi Model Prediksi

Untuk melakukan evaluasi kinerja dari skenario pemodelan secara kuantitatif, serangkaian metrik statistik digunakan. Metrik-metrik ini dirancang untuk mengukur berbagai aspek dari akurasi dan keandalan sistem penentuan posisi (Kia, Laura, & Talvitie, 2022). Eror

posisi dihitung sebagai jarak *Euclidean*, yang digunakan untuk menghitung jarak antara dua titik dalam bidang dua dimensi. Dengan mempertimbangkan titik antara koordinat yang diprediksi oleh model (x_{pred}, y_{pred}) dan koordinat sebenarnya (x_{true}, y_{true}) dengan rumus (2.6):

$$Error = \sqrt{\left(x_{pred} - x_{true}\right)^2 - \left(y_{pred} - y_{true}\right)^2} \quad (2.6)$$

Matriks evaluasi yang digunakan adalah sebagai berikut:

2.8.1. Mean Euclidean Error

Metrik ini mengukur rata-rata dari seluruh eror posisi yang dihitung pada dataset uji. Ini adalah indikator utama untuk akurasi sistem secara keseluruhan dan memberikan gambaran umum tentang seberapa jauh, rata-rata, prediksi model menyimpang dari lokasi sebenarnya. Nilai yang lebih rendah menunjukkan performa yang lebih baik.

2.8.2. Median Euclidean Error

Metrik ini adalah nilai tengah dari seluruh posisi eror yang telah diurutkan dari yang terkecil hingga terbesar. Berbeda dengan rata-rata eror, median tidak terlalu terpengaruh oleh beberapa prediksi dengan eror yang sangat besar (*outlier*). Nilai ini memberikan indikasi yang lebih baik mengenai kinerja tipikal dari model, merepresentasikan akurasi untuk mayoritas kasus pengujian.

2.8.3. Standard Deviation of Error

Standar deviasi mengukur tingkat sebaran atau variasi dari eror posisi terhadap nilai rata-ratanya. Metrik ini sangat penting untuk menilai konsistensi model. Nilai standar deviasi yang rendah menunjukkan bahwa kinerja model stabil, di mana besar eror pada setiap prediksi cenderung serupa. Sebaliknya, nilai yang tinggi mengindikasikan bahwa akurasi model tidak dapat diprediksi.

2.8.4. Maximum Error

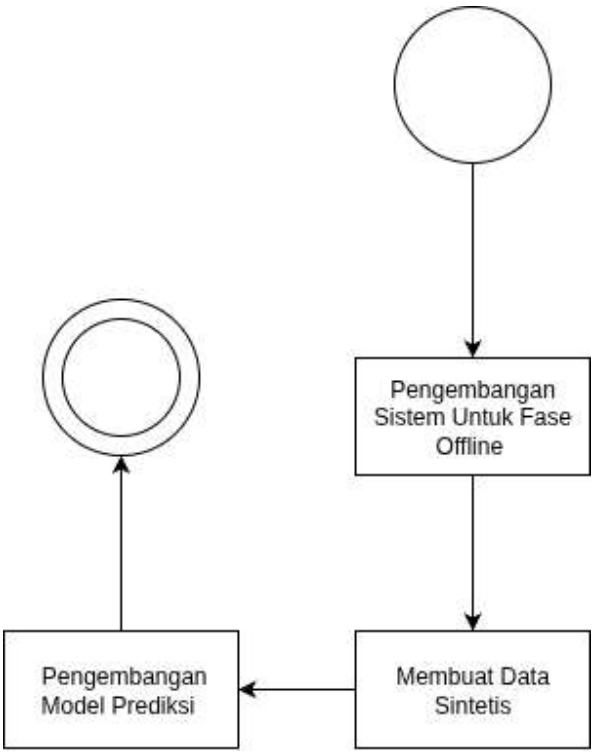
Metrik ini mengidentifikasi nilai kesalahan terbesar yang dihasilkan oleh model pada seluruh *dataset*. 2.7.4. *Maximum Error* penting untuk dipahami karena menunjukkan skenario kinerja terburuk (*worst-case skenario*) dari sistem. Dalam aplikasi praktis, mengetahui batas atas dari ketidakakuratan sistem sering kali sama pentingnya dengan mengetahui akurasi rata-ratanya.

BAB III
METODE PENELITIAN

Penelitian ini berfokus pada pengembangan sistem untuk melakukan pengumpulan data pada fase *offline* metode Wi-Fi *fingerprinting*, penambahan sintetis data menggunakan MLP, dan pengembangan model prediksi menggunakan KNN. Untuk itu, metodologi penelitian ini dirancang dengan urutan yang terstruktur untuk memastikan keberhasilan dari setiap tahapan, mulai dari perancangan sistem hingga evaluasi akhir. Penjelasan ini mencakup beberapa bagian utama yang terdiri dari alur pengerjaan tugas akhir, alur eksperimen, dan alur kerja sistem.

3.1. Alur Pengerjaan Tugas Akhir

Penelitian ini mengikuti alur proses yang digambarkan dalam diagram alur pada Gambar 3.1, yang terdiri dari tiga tahapan utama. Tahap pertama adalah pengembangan sistem untuk fase *offline*, pada tahap ini dilakukan pengembangan sistem yang nantinya akan digunakan pada tahap *offline* pada metode Wi-Fi *fingerprinting*. Tahap kedua adalah membuat data sintetis, pada tahap ini akan dilakukan pengumpulan data, pengolahan data, pembuatan model menggunakan MLP, serta pembuatan data sintetis. Tahap ketiga adalah pengembangan model prediksi, model KNN akan digunakan untuk membuat tiga model prediksi dengan dataset yang berbeda, yaitu: (1) menggunakan data mentah, (2) mengolah data mentah menggunakan kalman filter, (3) menggunakan data sintetis.



Gambar 3.1 Alur Pengerjaan Tugas Akhir

3.2. Pengembangan Sistem Untuk Fase Offline

Untuk melakukan metode Wi-Fi *fingerprinting* pada tahap *offline*, diperlukan sebuah sistem untuk melakukan pengumpulan sinyal guna membangun *fingerprint* map. Untuk mencapai tujuan tersebut, sebuah sistem sederhana dikembangkan.

3.2.1. Teknologi yang digunakan

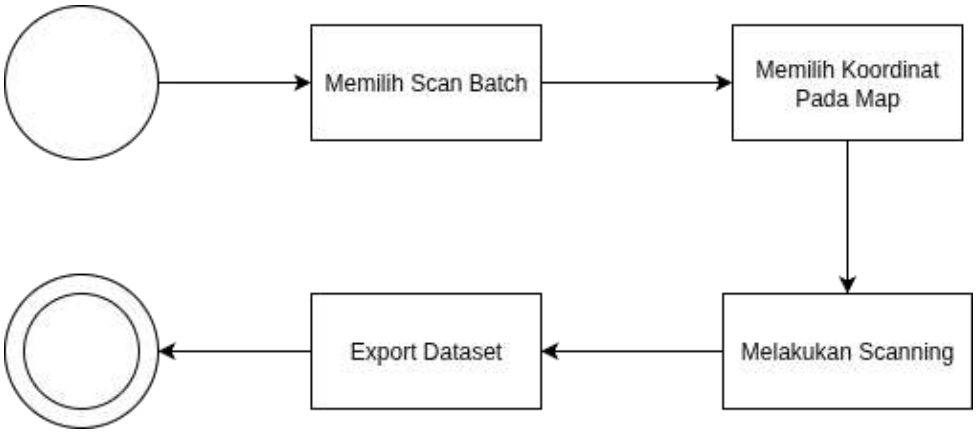
Untuk mendukung proses implementasi yang efisien dan terstruktur, berbagai perangkat lunak dan teknologi pendukung digunakan dalam pengembangan sistem ini:

- Tauri : adalah *framework* untuk membangun aplikasi desktop *multiplatform* dengan memanfaatkan *webview*.
- Rust : Rust adalah bahasa pemrograman *general-purpose* yang menekankan pada performa, *type safety*, dan *concurrency*.
- React : Pustaka Javascript untuk membangun antarmuka.
- Scalable Vector Graphics (SVG) : Format grafik vektor berbasis XML untuk mendefinisikan grafik dua dimensi, yang mendukung kontrol dan animasi.
- SQLite : SQLite adalah pustaka bahasa C yang mengimplementasikan mesin basis data SQL yang berukuran kecil, cepat, *self-contained*, reliabilitas tinggi, dan memiliki fitur lengkap.

- iw : adalah alat bantu konfigurasi berbasis CLI untuk perangkat nirkabel berbasis nl80211 pada sistem operasi Linux.

3.2.2. Alur Sistem

Untuk mempermudah pengumpulan data, alur atau tahapan sistem perlu didefinisikan dengan jelas. Diagram alur kerja pada Gambar 3.2 memberikan gambaran mengenai langkah-langkah yang dilakukan dalam sistem ini. Setiap tahap dalam alur kerja memiliki peran yang penting untuk memastikan hasil estimasi posisi yang akurat dan dapat diandalkan.



Gambar 3.2 Alur Sistem

a. Memilih Batch Scan

Tahap awal untuk melakukan pengumpulan data adalah dengan membuat atau memilih kelompok (*batch*) dari pemindaian yang akan dilakukan. Tujuan adalah untuk dapat melakukan dan mengategorikan banyak pemindaian lebih mudah. Disediakan antarmuka *dropdown* dan juga dialog untuk membuat kelompok pemindaian baru.

b. Memilih Koordinat Pada Map

Setelah menentukan kelompok pemindaian, tahap selanjutnya adalah menentukan di mana posisi koordinat perangkat akan ditempatkan pada map yang sudah dibuat. Untuk mempermudah tahap ini, maka dikembangkan antarmuka map dengan rasio yang sama persis dengan lokasi pengambilan data. Map dibuat menggunakan komponen elemen *Scalable Vector Graphics* (SVG) yang umum pada ada pada peramban. Penggunaan SVG dilakukan untuk mengakomodasi perubahan map agar lebih mudah dan untuk kontrol yang lebih mudah karena SVG adalah elemen pada peramban yang dapat dimanipulasi, contoh kode SVG ada pada Gambar 3.3.

```
<svg width="800" height="1200" style="display: block; touch-action: none;">
 <rect fill="white" x="0" y="0" width="800" height="1200" style="pointer-events: none;"/>
 <g transform="matrix(0.8,0,0,0.8,0,0)">
  <rect fill="#fff" x="0" y="0" width="1000" height="1200"/>
  <g>
   <defs>
    <mask id="mask1">
     <path type="path" id="room" d="M 80 80 h 765 v 1149 h -443 v -543 h -322 v -606 z" fill="white" stroke="black"/>
    </mask>
   </defs>
   <path type="path" id="room" d="M 80 80 h 765 v 1149 h -443 v -543 h -322 v -606 z" fill="none" stroke="black"/>
   <g mask="url(#mask1)">
    <path d=" M 80 80 v 1149  M 120 80 v 1149  M 160 80 v 1149  M 200 80 v 1149  M 240 80 v 1149  M 280 80 v 1149  M 320 80 v 1149  M 360 80 v 1149  M 400 80 v 1149  M 440 80 v 1149  M 480 80 v 1149  M 520 80 v 1149  M 560 80 v 1149  M 600 80 v 1149  M 640 80 v 1149  M 680 80 v 1149  M 720 80 v 1149  M 760 80 v 1149  M 800 80 v 1149  M 840 80 v 1149  M 880 80 v 1149  M 920 80 v 1149  M 960 80 v 1149  M 1000 80 v 1149  M 1040 80 v 1149 " stroke="red" stroke-width="0.5"/>
    <path d=" M 80 80 h 1000  M 80 120 h 1000  M 80 160 h 1000  M 80 200 h 1000  M 80 240 h 1000  M 80 280 h 1000  M 80 320 h 1000  M 80 360 h 1000  M 80 400 h 1000  M 80 440 h 1000  M 80 480 h 1000  M 80 520 h 1000  M 80 560 h 1000  M 80 600 h 1000  M 80 640 h 1000  M 80 680 h 1000  M 80 720 h 1000  M 80 760 h 1000  M 80 800 h 1000  M 80 840 h 1000  M 80 880 h 1000  M 80 920 h 1000  M 80 960 h 1000  M 80 1000 h 1000  M 80 1040 h 1000  M 80 1080 h 1000  M 80 1120 h 1000  M 80 1160 h 1000  M 80 1200 h 1000 " stroke="red" stroke-width="0.5"/>
   </g>
   <circle class="origin-center duration-200" cx="479.7075" cy="1189.215" r="5" fill="red"/>
   <circle class="origin-center duration-200" cx="559.9999999999999" cy="1189.9999999999998" r="5" fill="red"/>
   <circle class="origin-center duration-200" cx="639.9999999999999" cy="1189.9999999999998" r="5" fill="red"/>
```

```
</g>
 </g>
</svg>
```

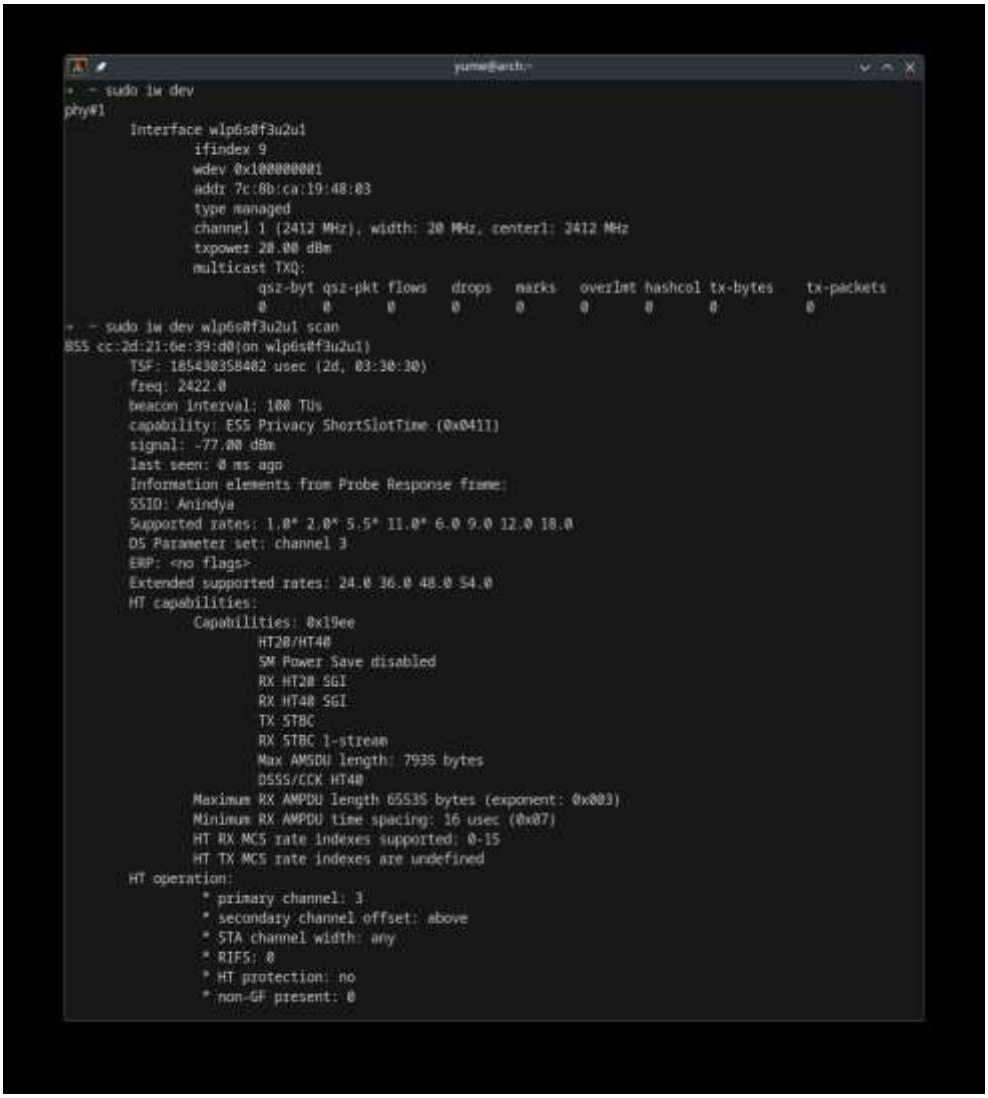
Gambar 3.3 Contoh format SVG.

Pada antarmuka ini pengguna dapat menentukan lokasi di mana perangkat akan melakukan pemindaian dengan menambahkan titik pada map, pengguna juga dapat menghapus titik yang sudah ditambahkan. Titik koordinat yang ada pada map terkorelasi pada koordinat di dunia nyata dengan sekala 1px berbanding 1cm.

c. Melakukan Scanning

Setelah menentukan koordinat perangkat, sistem dapat memulai melakukan pemindaian. Pada antarmuka akan terdapat bilah progres untuk mengetahui berapa banyak pemindaian yang sudah dilakukan dan berapa lama pemindaian akan berlangsung. Untuk penelitian ini jumlah pemindaian diatur untuk melakukan 300 iterasi pemindaian pada satu kali sesi pemindaian. Pemindaian dilakukan menggunakan perangkat lunak iw, contoh *output* program ada pada Gambar 3.4.

Pada kondisi ideal, semua RSSI yang dibutuhkan dapat tertangkap. Namun, terkadang terdapat halangan dari lingkungan pengujian, atau gangguan selama pengujian yang menyebabkan RSSI yang diharapkan tidak dapat tertangkap. Hal ini dapat mengganggu proses pengumpulan data, dan hal ini juga terjadi pada proses pengambilan data yang dilakukan pada eksperimen ini. Terdapat beberapa titik referensi yang memiliki jumlah nilai RSSI yang kurang dari 300, hal ini akan dibahas lebih lanjut pada proses pengolahan data.



Gambar 3.4 Keluaran program iw

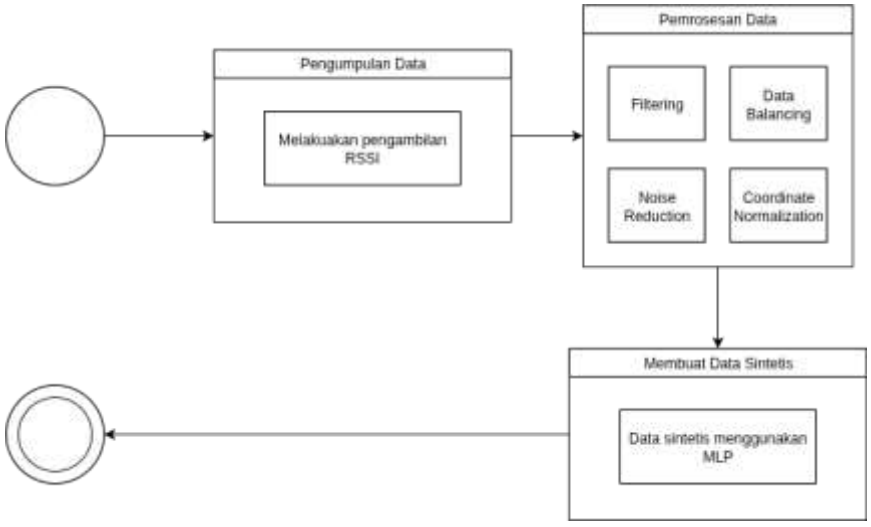
d. Export Data

Tahap akhir setelah menyelesaikan satu kelompok pemindaian adalah melakukan ekspor data. Basis data yang digunakan oleh sistem menggunakan SQLite, tetapi untuk mempermudah pengolahan lebih lanjut, data dapat di ekspor ke dalam format lain seperti CSV.

3.3. Membuat Data Sintetis

Pada proses pembuatan data sintetis, terdapat tiga tahapan yang dilakukan, seperti pada Gambar 3.5. Tahap pertama adalah pengambilan *dataset* RSSI untuk membuat *fingerprint* map, *dataset* dikumpulkan pada lingkungan dalam ruangan yang terkontrol untuk memastikan konsistensi data. Tahap kedua adalah pemrosesan data, setelah *dataset* dikumpulkan, data harus diproses terlebih dahulu. Tahapan pemrosesan data terdiri dari *Filtering* data, Penyeimbangan

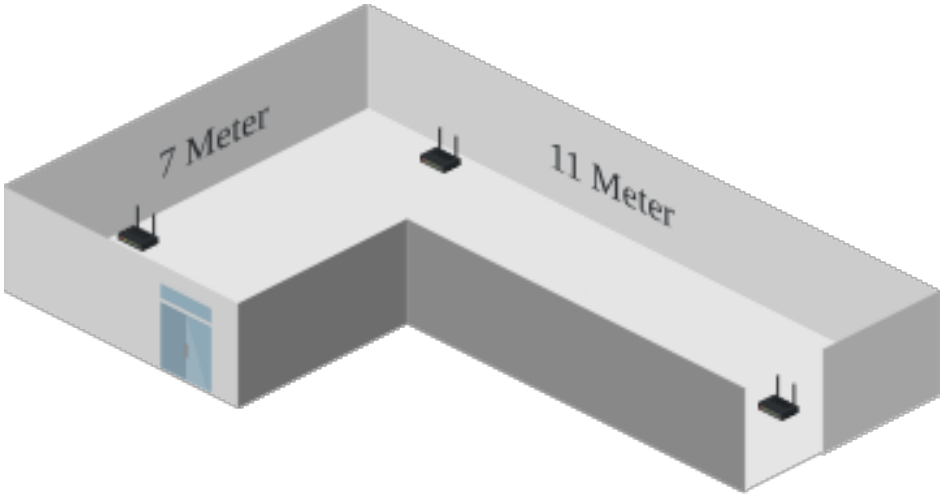
data, Menghilangkan *noise* pada RSSI, dan normalisasi koordinat. Tahap terakhir adalah membuat data sintetis menggunakan model MLP.



Gambar 3.5 Alur Pembuatan Data Sintetis

3.3.1. Pengumpulan Data

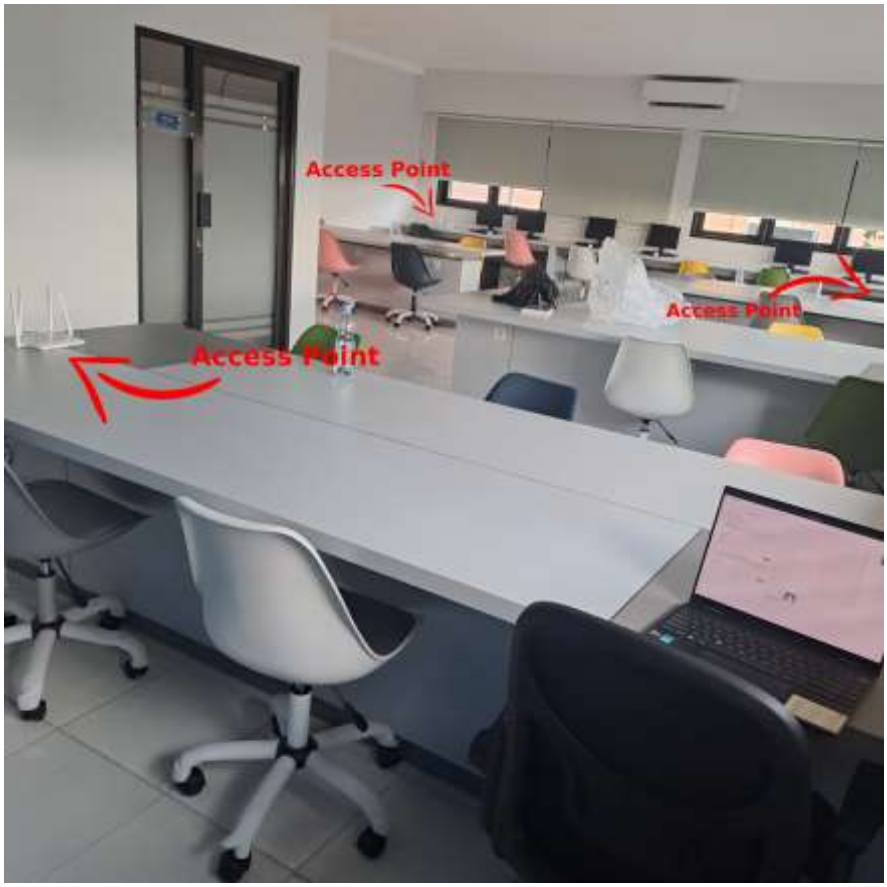
a. Lokasi dan Peralatan



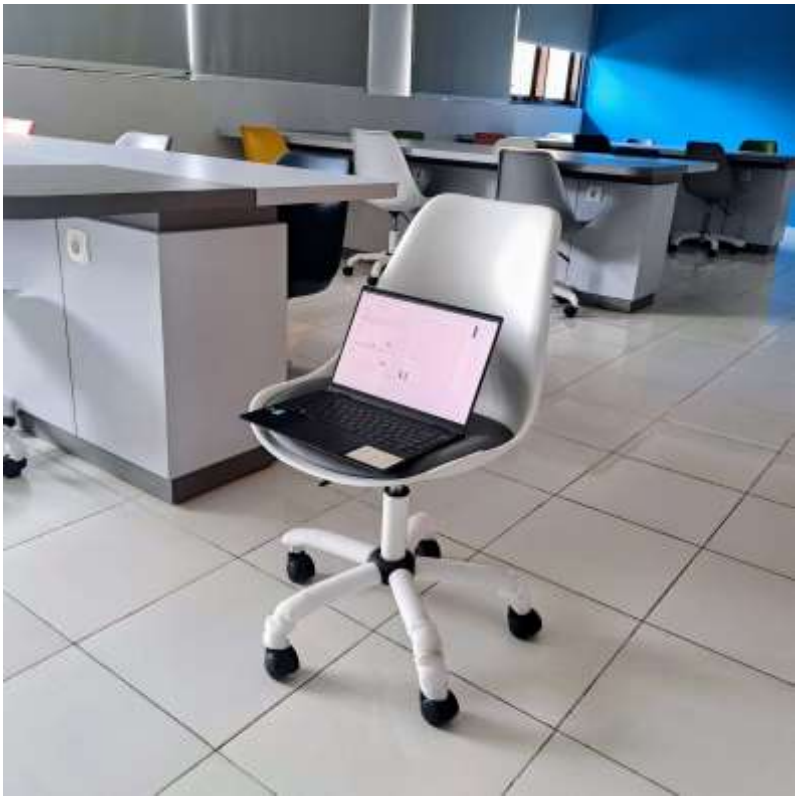
Gambar 3.6 Map Laboratorium Siber



Gambar 3.7 Laboratorium Siber



Gambar 3.8 Posisi AP Akses Poin



Gambar 3.9 Setup pemindaian RSSI

Pengambilan data dilakukan di area Laboratorium Siber, Fakultas Teknologi Industri, Universitas Islam Indonesia, dengan dimensi ruangan sekitar 11 x 7 meter, seperti diilustrasikan pada Gambar 3.6 dan foto pada Gambar 3.7. Digunakan tiga akses poin yang ditempatkan di tiga sudut ruangan, seperti pada Gambar 3.8. Perangkat yang digunakan untuk pengambilan sinyal adalah laptop Asus Zenbook 14 un5401q dengan *network interface* Intel® WI-FI 6 AX200 menggunakan sistem operasi Linux, spesifikasi lengkap seperti pada Tabel 3.1dan Gambar 3.9. Tiga AP yang digunakan adalah Ruijie RG-EW1200 dengan Standar WI-FI 5 (802.11ac) dan penguat sinyal 2.4 GHz serta 5 GHz, spesifikasi lengkap seperti pada Tabel 3.2.

Tabel 3.1 Spesifikasi Perangkat Keras Laptop

Komponen	**Spesifikasi**
Jenis perangkat	Laptop
Central Processing Unit (CPU)	AMD Ryzen 5 4600H with Radeon Graphics 6 Cores (12 CPUs), ~3.0GHz
Random Access Memory (RAM)	16GB DDR4 3200MHz

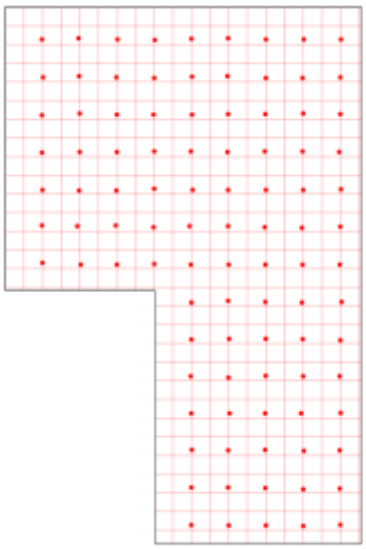
Graphics Processing Unit (GPU)	Rodeon Vega 8
Penyimpanan	SSD 512GB
Sistem Operasi	Kubuntu 24.4 LTS

Tabel 3.2 Spesifikasi Perangkat AP yang digunakan

Komponen	Spesifikasi
Nama	Ruijie RG-EW1200
Standar Wi-Fi	WI-FI 5
Lebar Kanal	2.4 GHz: Otomatis/20/40 MHz MHz5 GHz: Otomatis/20/40/80 MHz
Penguatan Antena	2.4 GHz: 5 dBi 5GHz: 5dBi

b. Dataset

Dataset utama dikumpulkan dengan mengambil data sinyal pada 98 titik referensi (RP). Titik-titik ini ditempatkan secara teratur membentuk *grid* dengan jarak 80 cm antar titik yang berdekatan, seperti pada Gambar 3.10.



Gambar 3.10 Map titik referensi

c. Pengumpulan data

Untuk setiap titik referensi, dilakukan 300 kali pemindaian dengan durasi sekitar 3 menit untuk menangkap nilai RSSI dan data disimpan ke dalam basis data SQLite.

3.3.2. Pemrosesan Data

Sebelum data digunakan untuk pemodelan, dilakukan serangkaian langkah pemrosesan untuk memastikan kualitas dan konsistensi data.

a. Data Filtering

Dataset mentah yang berisi pembacaan RSSI, koordinat (x, y), dan alamat MAC dimuat dari berkas CSV. Data kemudian disaring untuk mengisolasi pembacaan dari tiga AP dengan *Service Set Identifier* (SSID) spesifik yang menjadi fokus penelitian: RuijieAP1, RuijieAP2, dan RuijieAP3.

b. Data Balancing

Pada skenario dunia nyata, pada tahap pemindaian tidak akan berjalan secara lancar, akan terdapat gangguan baik dari faktor lingkungan maupun factor yang lain sehinggan tiap koordinat dapat memiliki jumlah sampel yang berbeda. Proses ini memastikan setiap koordinat memiliki representasi data yang setara. Prosedur yang digunakan untuk melakukan penyeimbangan data adalah sebagai berikut:

- **Downsampling**: Untuk koordinat dengan jumlah sampel di atas rata-rata *global*, *subset* acak dipilih untuk mengurangi jumlahnya hingga setara dengan rata-rata.
- **Upsampling**: Untuk koordinat dengan jumlah sampel di bawah rata-rata, data yang ada ditambah dengan sampel baru yang nilainya merupakan rata-rata dari pengukuran yang ada di lokasi tersebut.

c. Noise Reduction

Pada kasus dunia nyata, akan terjadi fluktuasi sinyal pada AP (Firmansyah, 2024). Untuk menghaluskan data RSSI dan mengurangi fluktuasi pengukuran yang acak pada pengukuran data, Kalman Filter diterapkan pada setiap urutan data RSSI per koordinat yang telah diseimbangkan untuk menghasilkan estimasi kekuatan sinyal yang lebih stabil dan andal (Firmansyah et al., 2024). Untuk menjamin bahwa estimasi kestabilan sinyal yang dihasilkan oleh Kalman Filter diterapkan pada jumlah data yang tidak seimbang pada semua *reference point*, maka *Noise Reduction* dilakukan setelah dilakukan *data balancing*.

d. Coordinate Normalization

Sistem koordinat (x, y) dinormalisasi dengan menggeser titik pusat area pengujian ke titik asal (0,0). Praktik ini merupakan langkah prapemrosesan standar dalam *machine learning*, terutama untuk model berbasis jaringan saraf (Goodfellow, Bengio, & Courville, 2016). Tujuannya adalah untuk mempercepat dan menstabilkan proses pelatihan model generatif (MLP). Dengan memastikan data *input* terpusat di sekitar nol dan memiliki skala yang seragam, permukaan fungsi kerugian (*loss landscape*) menjadi lebih simetris sehingga memungkinkan algoritma optimasi seperti Adam untuk menemukan solusi optimal dengan lebih efisien. Selain itu, normalisasi memastikan bahwa semua dimensi *input* (sumbu x dan y) memberikan kontribusi yang seimbang selama proses pembelajaran.

3.3.3. Membuat Data Sintetis

Model data sintetis dibuat menggunakan pendekatan regresi terbalik (*inverse regression*). Model *Multi-Layer Perceptron* (MLP) dilatih secara terpisah untuk RuijieAP1, RuijieAP2, dan RuijieAP3. Model generatif dilatih untuk memprediksi nilai RSSI berdasarkan input koordinat x dan y. Setiap model MLP dibangun menggunakan API Keras dari TensorFlow dengan arsitektur *deep feedforward* yang dirancang untuk menangkap hubungan *non-linear* antara lokasi dan kekuatan sinyal.

- **Lapisan Input**: Menerima vektor 2-dimensi yang merepresentasikan koordinat (x, y) yang telah dinormalisasi.
- **Lapisan Tersembunyi**: Terdiri dari lima lapisan yang disusun dalam struktur corong (f*unnel structure*) terdiri dari lapisan sebanyak (1024, 128, 64, 64, 32 neuron). Semua lapisan tersembunyi menggunakan fungsi aktivasi ReLU (Rectified Linear Unit) untuk memungkinkan model mempelajari pola spasial yang kompleks.
- **Lapisan Output**: Menghasilkan satu nilai skalar yang merepresentasikan prediksi RSSI. Lapisan ini menggunakan fungsi aktivasi kustom (lambda x: -tf.nn.relu(x) - 1e-6) yang dirancang khusus untuk memastikan nilai keluaran selalu negatif sesuai dengan nilai RSSI yang diukur dalam dBm dan selalu negatif.

Keluaran model digunakan untuk mengisi nilai RSSI pada tiap AP untuk koordinat dengan interval 10 cm.

3.4. Pengembangan Model Prediksi

Untuk melakukan prediksi Lokasi dan mengukur pengaruh sintetis data, tiga skenario pemodelan dievaluasi menggunakan algoritma KNN:

- Skenario A (Data *original*): Model KNN dilatih hanya menggunakan data set latih *original* (98 titik).
- Skenario B (Data *original* dan *signal filtering*): Model KNN dilatih menggunakan data original yang diolah lebih lanjut menggunakan *signal filtering* yaitu kalman filter.
- Skenario C (Dengan Sintetis): Model KNN dilatih menggunakan data set sintetik yang dihasilkan oleh model MLP.

Untuk menentukan konfigurasi model KNN yang paling optimal untuk setiap skenario (dasar, dengan Kalman Filter, dan data sintetis), dilakukan proses *hyperparameter tuning*. Proses ini bertujuan untuk menemukan kombinasi parameter yang menghasilkan eror prediksi lokasi terendah. Untuk tugas ini, digunakan pustaka GridSearchCV.

Pencarian parameter dilakukan pada variabel n_neighbors (jumlah tetangga) dari 1 hingga 31, matriks evaluasi menggunakan fungsi *Mean Euclidean Error* untuk mencari jarak eror terendah, dan validasi silang yang digunakan pada tiap model adalah lima.

Untuk melakukan simulasi tahap *online* pada metode Wi-Fi fingerprinting, model akan diuji menggunakan dataset yang berbeda dari dataset pelatihan dan validasi. Dataset akan diambil menggunakan dua perangkat yang berbeda yaitu Asus Zenbook 14 dengan *network interface* Wi-Fi 5 dan Macbook Pro 2017 dengan *network interface* Wi-Fi 4. Skenario pengujian adalah pengambilan data akan dilakukan pada titik acak di dalam ruangan menggunakan kedua perangkat dengan iterasi sebanyak 300 pada tiap pengambilan sinyal. *Setup* pengambilan data dapat dilihat pada Gambar 3.11 dan Gambar 3.12.



Gambar 3.11 Pengambilan data menggunakan perangkat Asus zenbook 14



Gambar 3.12 Pengambilan data menggunakan perangkat Macbook Pro 2017

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas hasil dari penelitian yang telah dilakukan serta menjawab rumusan masalah yang telah dirumuskan sebelumnya. Pembahasan ini mencakup beberapa aspek penting dalam pengembangan dan evaluasi sistem IPS. Fokus utama penelitian ini adalah membangun sistem untuk melakukan pengumpulan data pada fase *offline*, serta untuk mengurangi tenaga yang diperlukan untuk melakukan pengumpulan data pada fase *offline* menggunakan data sintetis.

4.1. Pengembangan Sistem Untuk Fase Offline

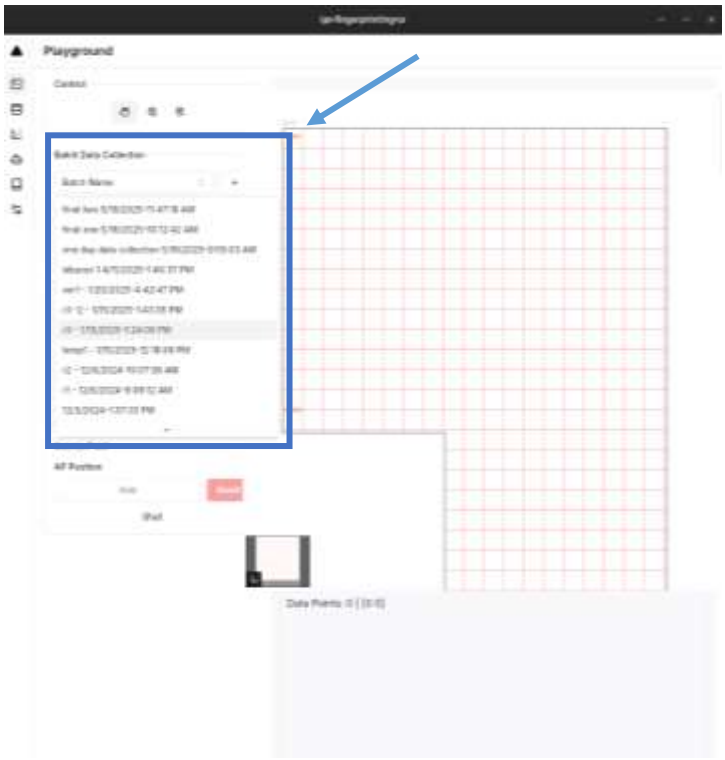
Pada subbab ini, akan dibahas secara detail hasil dari pengembangan alur kerja untuk melakukan pengumpulan data pada fase *offline*. Pembahasan akan berisi struktur kode yang diterapkan pada sistem ini. Penjelasan akan dibuat secara terstruktur berdasarkan jalannya alur sistem.

4.1.1. Memilih Batch Scan

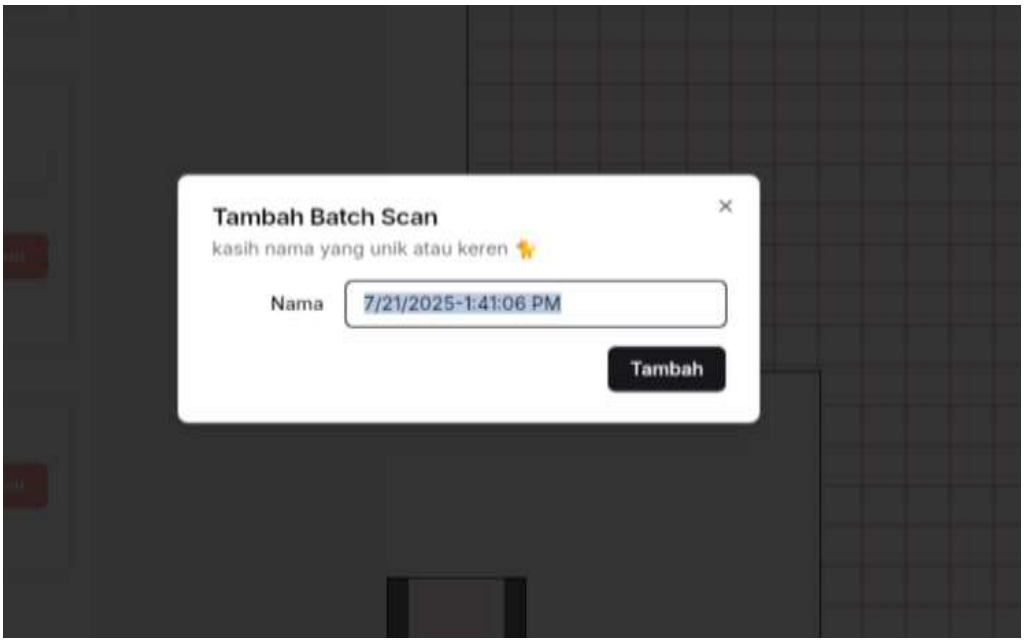
Langkah pertama yang dilakukan pada pemindaian adalah memilih atau membuat kelompok pemindaian. Hal ini dilakukan untuk mempermudah pengelolaan banyak sesi pemindaian. Struktur basis data yang digunakan untuk proses ini sangat sederhana, seperti pada **Error! Reference source not found.**.

Gambar 4.1 Struktur basis data.

Tiap satu nilai RSSI hasil pemindaian akan berkorelasi pada satu kelompok pemindaian. Antarmuka untuk memilih kelompok pemindaian dapat dilihat pada Gambar 4.2. Antarmuka untuk menambah kelompok pemindaian ada pada Gambar 4.3.



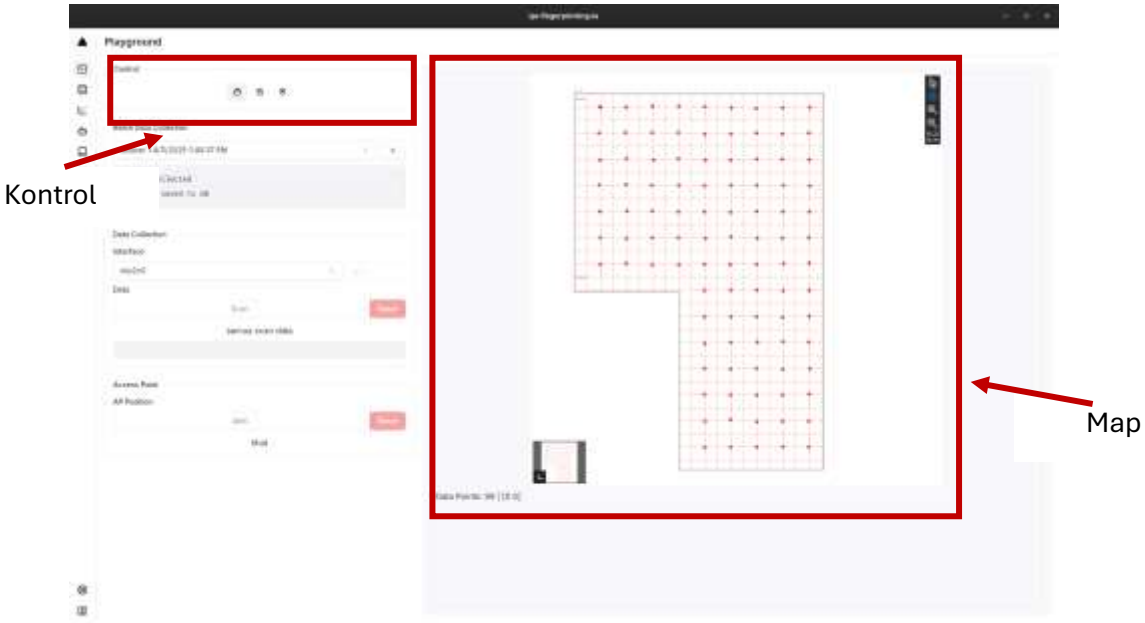
Gambar 4.2 Tampilan *Dropdown* Kelompok pemindaian.



Gambar 4.3 Dialog tambah kelompok pemindaian

4.1.2. Memilih Koordinat pada Map

Langkah berikutnya setelah menentukan kelompok pemindaian adalah menentukan koordinat pemindaian pada map. Tahap ini dilakukan dengan menambahkan titik pada antarmuka map, seperti pada Gambar 4.4.



Gambar 4.4 Antarmuka map

Pada antarmuka ini, map dibuat menggunakan SVG agar perubahan pada map dapat dilakukan dengan mudah dan interaksi pada map dapat dikelola dengan mudah. Kode program yang digunakan untuk membuat map terdapat pada Gambar 4.5. Data untuk map disimpan dalam format JSON, kemudian data diproses untuk ditampilkan dan di manipulasi sesuai skala dunia nyata oleh kode program pada Gambar 4.6. Data SVG disimpan ke dalam format JSON untuk mempermudah konversi sekala sebelum data ditampilkan.

```
{
  "zeroCoordinateReference": { "x": 80, "y": 80 },
  "room": {
    "width": 7.65,
    "height": 11.49
  },
  "elements": [
    {
      "type": "path",
      "id": "room",
      "d": [
        "M 80 80",
        "h meters(7.65)",
        "v meters(11.49)",
        "h meters(-4.43)",
```

```
        "v meters(-5.43)",
        "h meters(-3.22)",
        "v meters(-6.06)",
        "z"
      ],
      "fill": "none",
      "stroke": "black"
    }
  ],
  "apPoints": [
    {
      "id": "AP1",
      "x": "meters(0.1)",
      "y": "meters(0.15)"
    },
    {
      "id": "AP2",
      "x": "meters(0.1)",
      "y": "meters(5.60)"
    }
  ]
}
```

Gambar 4.5 Map dalam format JSON

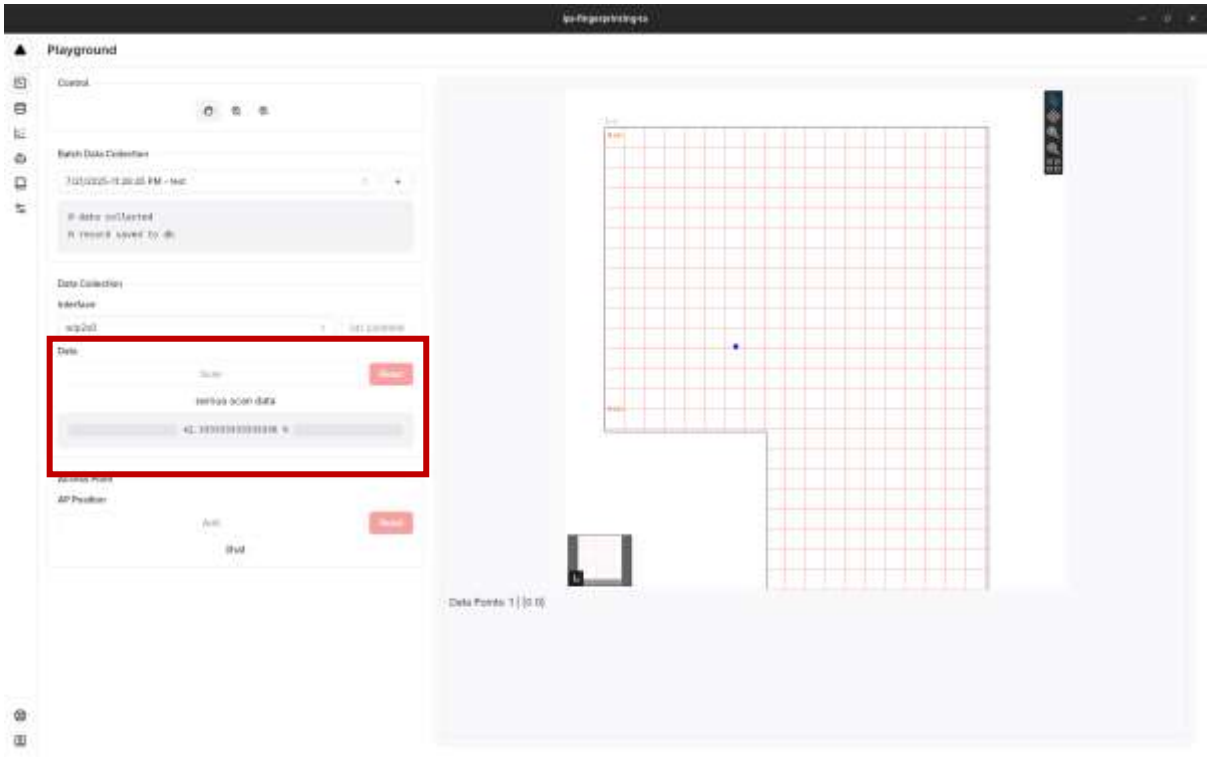
```
const METERS_TO_SVG_UNITS = 100; // 1 meter = 100 SVG units
const metersToSvgUnits = (meters: number) => {return meters *
METERS_TO_SVG_UNITS;};
```

Gambar 4.6 Konversi meter ke sekala dalam SVG

Berkas JSON pada Gambar 4.5 akan dimuat oleh sistem kemudian dilakukan *parsing* pada objek `elements` dan `apPoints` untuk mencari nilai yang memiliki format "meters(angka)" menggunakan REGEX berikut `/meters\(([^)]+)\)/`. Apabila ditemukan maka nilai pada JSON akan ditimpa dengan hasil konversi ke sekala SVG ke dunia nyata menggunakan fungsi pada Gambar 4.6 Konversi meter ke sekala dalam SVG.

4.1.3. Melakukan Scanning

Setelah menentukan koordinat pada map dan menempatkan perangkat sesuai dengan koordinat di dunia nyata. Selanjutnya tahap pemindaian dapat dilakukan dengan menekan tombol *scan* pada antarmuka, selanjutnya progres pemindaian akan tampil pada antarmuka, seperti pada Gambar 4.7.



Gambar 4.7 Progres Pemindaian

Pada tahap ini pemindaian RSSI dilakukan setiap ~0.5 detik untuk hasil yang konsisten. Kode yang dijalankan pada proses ini ada pada Gambar 4.8. Sistem akan memanggil *shell script* untuk mendapatkan semua nilai RSSI menggunakan perangkat lunak iw. Langkah ini dilakukan untuk menghindari terlalu banyak lapisan abstraksi pemanggilan kode yang dapat mempengaruhi durasi pengambilan nilai.

```
#!/bin/bash

home=$HOME
profile=$PROFILE

interface=$1

sudo iw dev $interface scan
OUTPUT=$?

exit $OUTPUT
```

Gambar 4.8 Kode Untuk Melakukan Pemindaian

Sistem akan memanggil kode pada Gambar 4.8 dan menunggu keluaran dari kode. Selanjutnya sistem akan memanggil kembali *script* menggunakan perulangan selama jumlah pemindaian kurang dari 300. Secara ideal sistem akan melakukan pemindaian setiap 0.5 detik dengan hasil RSSI dapat berbeda-beda.

.

4.1.4. Export Data

Untuk menyiapkan tahap berikutnya, data hasil pemindaian yang ada pada basis data dipindahkan ke format CSV. Hal ini dilakukan untuk mempermudah pengolahan data karena data akan mudah untuk di ubah dengan teks editor dibandingkan harus menggunakan bahasa *query* basis data. Untuk melakukan *export* data ke dalam CSV menggunakan kode pada Gambar 4.9.

```
import csv
import sqlite3

conn     =     sqlite3.connect('/home/yoru/.config/com.ips-fingerprinting-
ta.app/test.db')

cursor = conn.cursor()

with open('./test_points/nama_file.csv', 'w', newline='') as csvfile:

# to select all column we will use
statement = '''SELECT * FROM fingerprint_data WHERE survey_batch_id = 7'''
cursor.execute(statement)

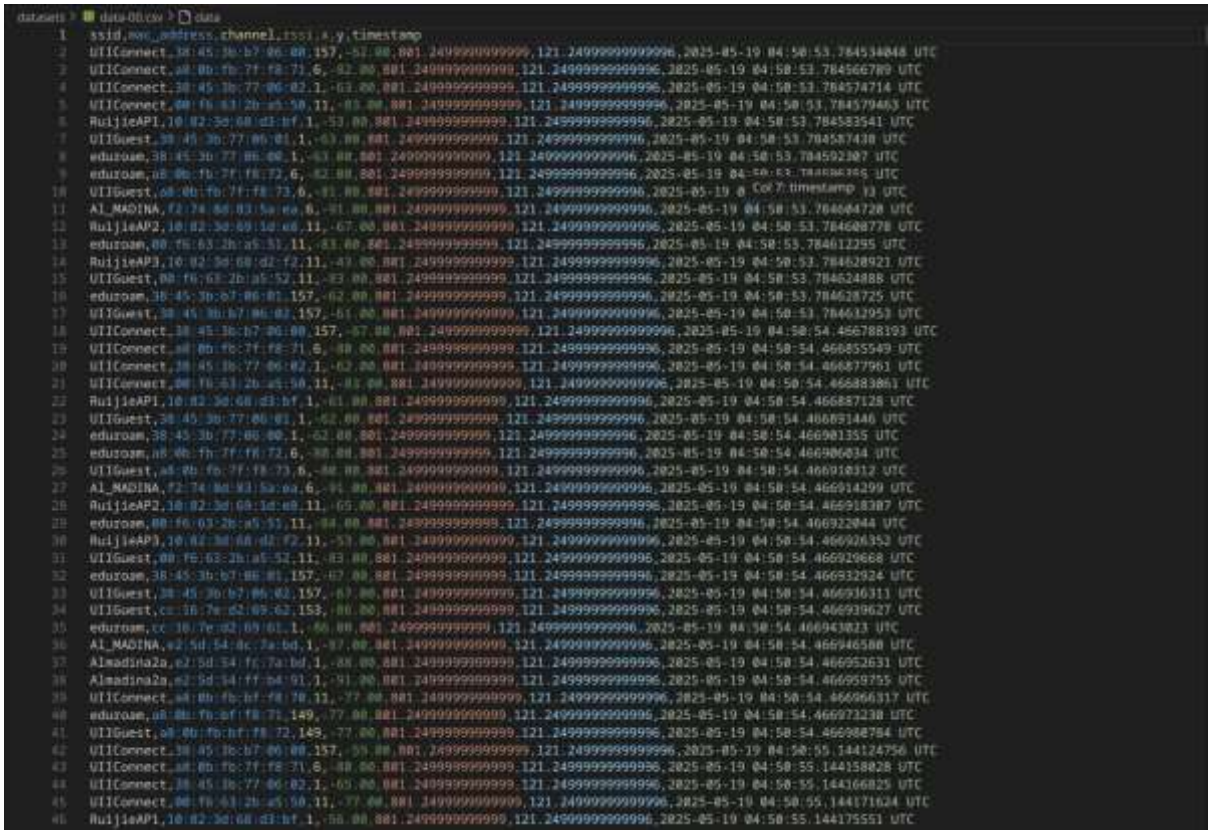
output = cursor.fetchall()
fieldnames = ['rssi', 'ssid', 'mac_address', 'channel', 'x', 'y']
writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
writer.writeheader()

for row in output:
  rssi = row[2]
  channel = row[3]
  ssid = row[4]
  mac_address = row[5]
  x = row[6]
  y = row[7]
  writer.writerow({'rssi': rssi, 'ssid': ssid, 'mac_address': mac_address,
'channel': channel, 'x': x, 'y': y})
  print(row)
  conn.commit()

# Close the connection
conn.close()
```

Gambar 4.9 Kode Untuk Export Data

Dengan menggunakan pustaka *sqlite3* untuk melakukan koneksi ke basis data dan menjalankan *query* “SELECT * FROM fingerprint_data WHERE survey_batch_id = ?” untuk mengambil data pada kelompok pemindaian tertentu. Selanjutnya data ditulis ke dalam berkas menggunakan pustaka CSV yang akan menghasilkan berkas seperti pada Gambar 4.10.



Gambar 4.10 Berkas CSV

4.2. Membuat Data Sintetis

Tahapan berikutnya setelah mengembangkan sistem untuk melakukan pengambilan data untuk fase *offline* adalah membuat data sintetis. Lingkup uji pengambilan data serta cara pengambilan data telah dijelaskan pada sebelumnya. Hasil dari pengumpulan data tersebut ada pada Tabel 4.1. Terdapat lebih dari satu juta RSSI yang terkumpul dari 48 Akses Poin yang tertangkap saat melakukan pemindaian pada 9 titik referensi.

Tabel 4.1 Hasil Pengumpulan Data

Label	**Nilai**
Total Akses Poin	48
Total RSSI	1.156.570
Total Titik Referensi	98

4.2.1. Pemrosesan Data

a. Data Filtering

Pada tahapan ini akan dilakukan penyaringan data, karena tidak semua Akses Poin akan digunakan. Pada kasus dunia nyata, akses poin sering saling tumpang tindih dalam satu bangunan. Namun pada konteks IPS, tidak semua akses poin dapat digunakan. Beberapa akses poin mungkin terlalu lemah atau tidak akan selamanya ada di lingkungan tersebut. Maka tahap pertama pada pemrosesan data adalah melakukan filter akses poin yang akan digunakan.

Pada penelitian ini hanya akan digunakan tiga akses poin dengan SSID RuijieAP1, RuijieAP2, dan RuijieAP3. Setelah dilakukan filter SSID data set berkurang secara drastis menjadi seratus ribu RSSI, seperti pada Tabel 4.2.

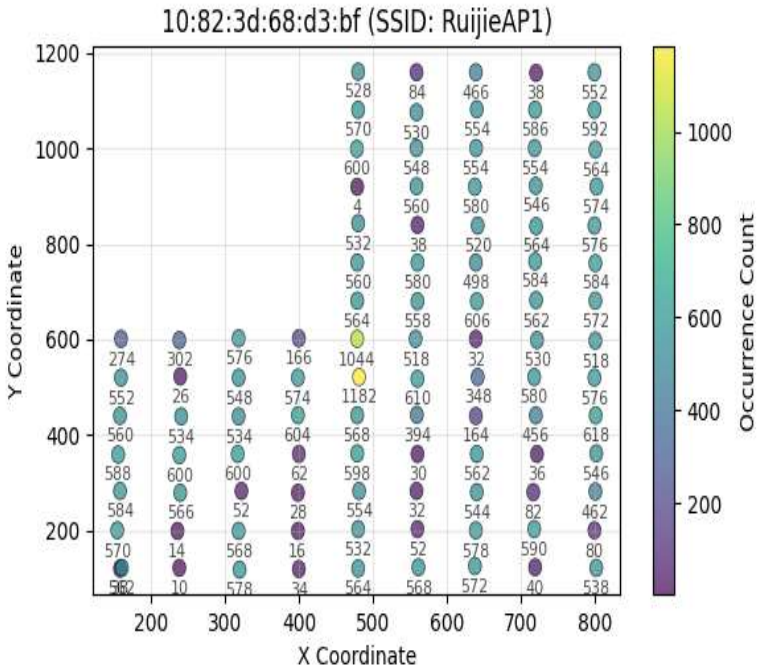
Tabel 4.2 Hasil penyaringan Data

Label	**Nilai**
Total Akses Poin	3
Total RSSI	131.682
Total Titik Referensi	98

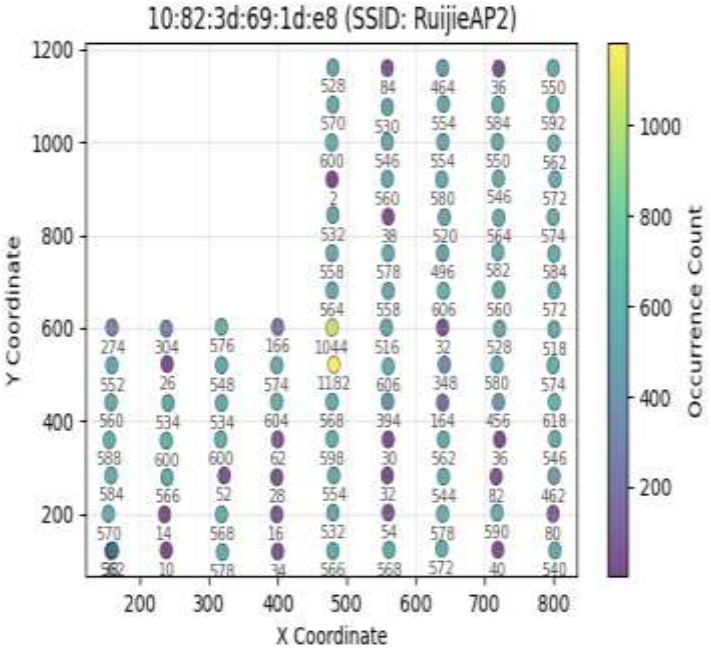
b. Data Balancing

Setelah dilakukan penyaringan data, data set masih harus diproses lebih lanjut dengan melakukan penyeimbangan data. Pada kasus dunia nyata, kesalahan pengumpulan data dapat terjadi baik itu dikarenakan oleh perubahan lingkungan, kesalahan perangkat, hingga kesalahan manusia. Hal ini dapat menyebabkan jumlah sampel data pada tiap titik referensi berbeda-beda dan perbedaan ini bisa sangat ekstrem terjadi. Salah satu permasalahan yang ditemu pada penelitian ini adalah terdapat titik referensi yang memiliki jumlah RSSI yang sangat tinggi dengan jumlah 3546 sedangkan ada satu titik dengan jumlah RSSI hanya sebanyak 8, dapat dilihat pada Lampiran A. Selain itu, Penyeimbangan data dilakukan agar model data sintetis tidak bias terhadap titik referensi yang memiliki lebih banyak data.

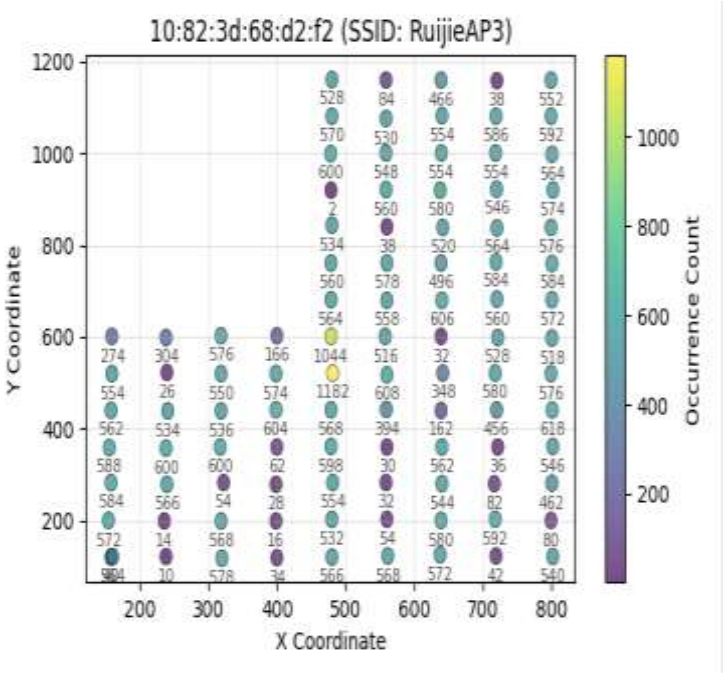
Penjelasan mekanisme data *balancing* dapat dilihat pada sebelumnya, kode yang digunakan dapat dilihat Lampiran B. Distribusi Jumlah RSSI awal sebelum dilakukan penyeimbangan data dapat dilihat pada Gambar 4.11, Gambar 4.12, dan Gambar 4.13. Hasil penyeimbangan data dapat dilihat pada gambar Gambar 4.14, Gambar 4.15, dan Gambar 4.16.



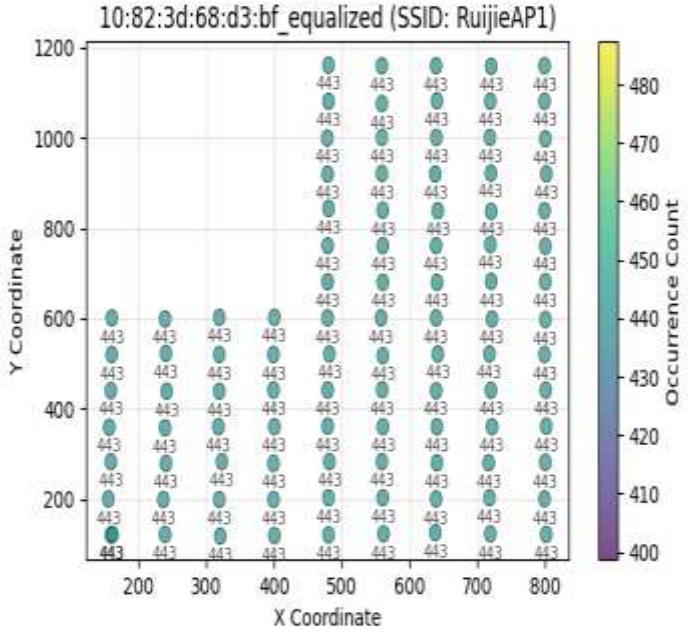
Gambar 4.11 Distribusi orisinal RSSI RuijieAP1



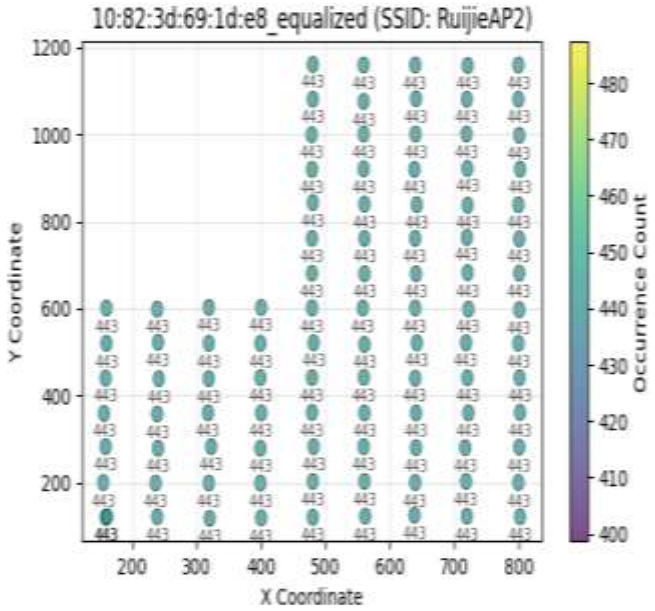
Gambar 4.12 Distribusi orisinal RSSI RuijieAP2



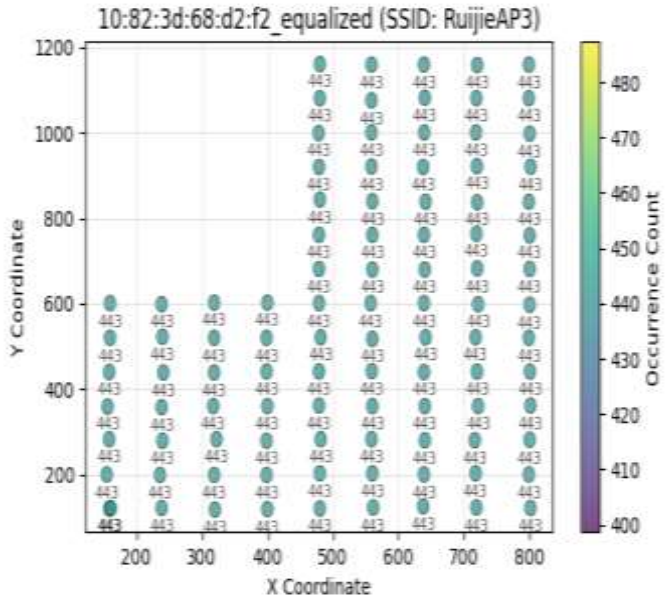
Gambar 4.13 Distribusi orisinal RSSI RuijieAP3



Gambar 4.14 Distribusi RSSI RuijieAP1 Setelah Penyeimbangan



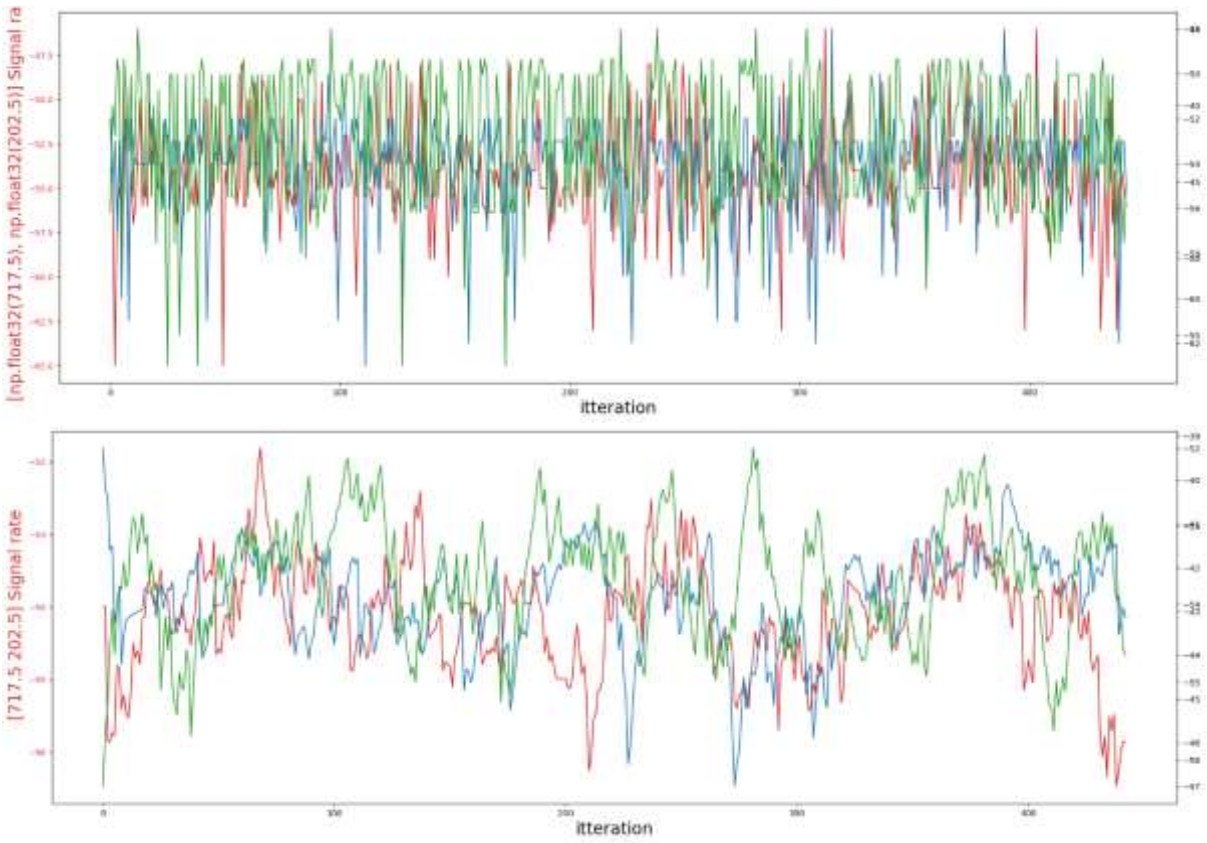
Gambar 4.15 Distribusi RSSI RuijieAP2 Setelah Penyeimbangan



Gambar 4.16 Distribusi RSSI RuijieAP2 Setelah Penyeimbangan

c. Noice Reduction

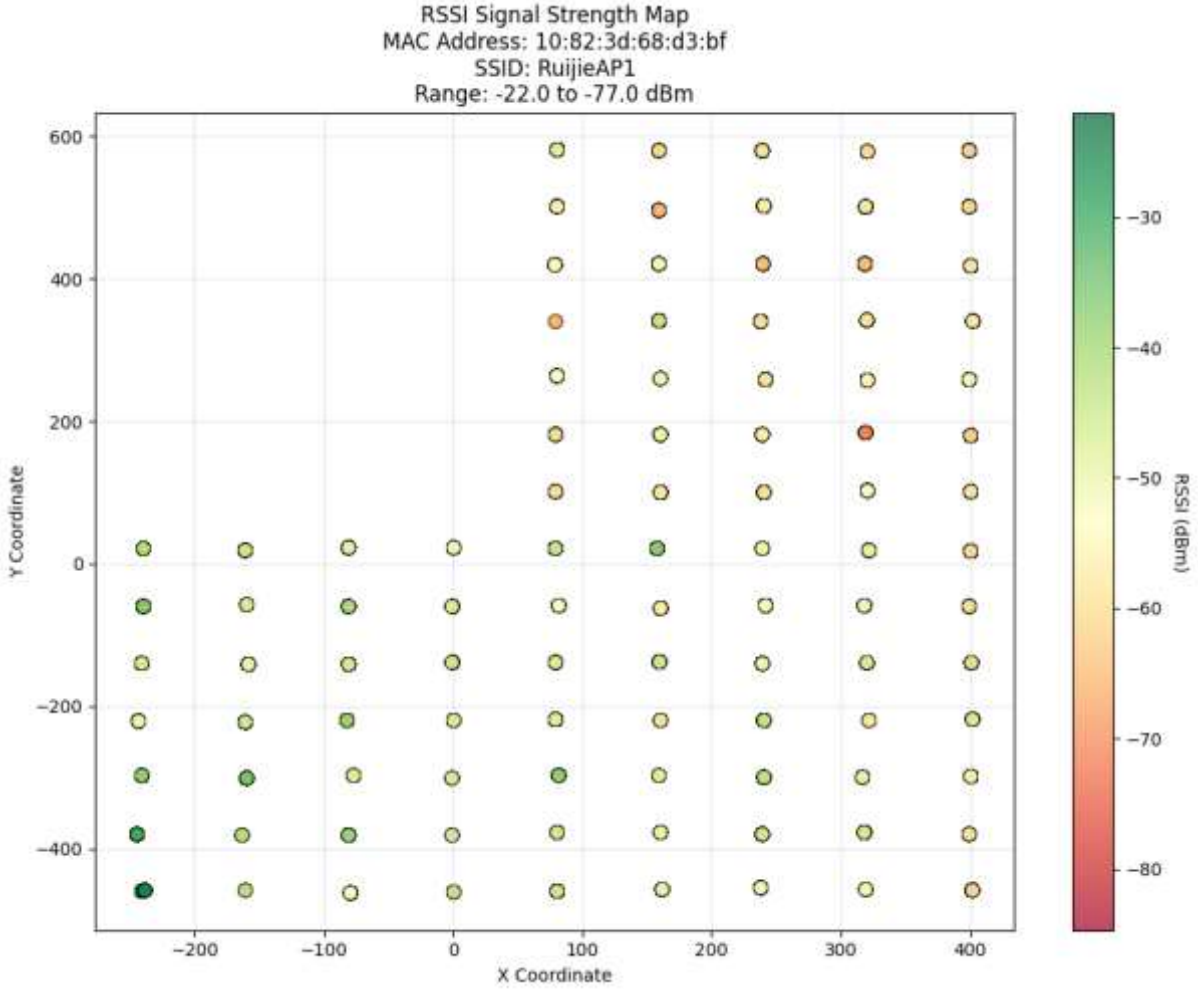
Fluktuasi RSSI adalah hal yang wajar pada pengukuran sinyal nirkabel. Kalman Filter digunakan untuk membuat fluktuasi sinyal menjadi lebih stabil. Contoh perbedaan sinyal sebelum dan sesudah diproses menggunakan kalman filter dapat dilihat pada Gambar 4.17.



Gambar 4.17 Sinyal Sebelum dan Sesudah Kalman Filter

d. Coordinate Normalization

Pada percobaan penelitian ini, titik koordinat berpengaruh terhadap kinerja model untuk membuat data sintetis. Pada Koordinat (x, y) yang dimulai dari (0, 0) dan berakhir dengan (800, 1200) model memiliki bias untuk menghasilkan keluaran secara linear. Pada penelitian ini, melakukan normalisasi koordinat dengan menggeser titik tengah koordinat ke (0, 0) memberikan hasil yang lebih baik. Hasil dari normalisasi koordinat dapat dilihat pada Gambar 4.18.



Gambar 4.18 Hasil Normalisasi Koordinat

4.2.2. Membuat Data Sintetis

a. Membuat Model

Proses pembuatan data sintetis diawali dengan melakukan pelatihan model regresi terbalik menggunakan MLP. Model akan diberikan masukan berupa koordinat (x, y) dan akan melakukan prediksi nilai RSSI pada koordinat tersebut. Kode untuk melakukan pelatihan model ada pada Gambar 4.19.

```
AP_IDX = 0
X = np.array(halfed_filtered_training_data[AP_IDX].x, dtype=np.float32)
Y = np.array(halfed_filtered_training_data[AP_IDX].y, dtype=np.float32)

X_train, X_test, Y_train, Y_test =
train_test_split_custom(halfed_training_data[AP_IDX].x,
halfed_training_data[AP_IDX].y, train_ratio=0.85)

model_synth = Sequential([
```

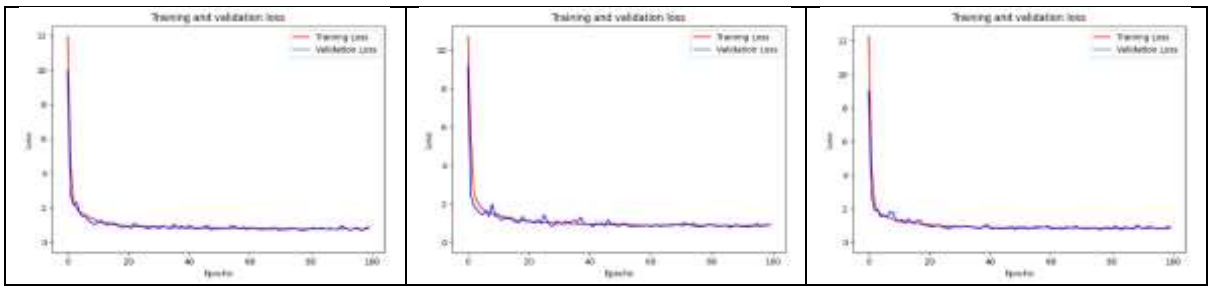
```
    Input(shape=(X[1].shape)),
    Dense(1024, activation='relu'),
    Dense(128, activation='relu'),
    Dense(64, activation='relu'),
    Dense(64, activation='relu'),
    Dense(32, activation='relu'),
    Dense(1, activation=lambda x: -tf.nn.relu(x) - 1e-6)
])

model_synth.compile(loss='mean_absolute_error', optimizer='adam',
metrics=['mae'])

history = model_synth.fit(x=X,y=Y,epochs=100,validation_data
(X_test,Y_test))
```

Gambar 4.19 Kode untuk pelatihan model

Penjelasan lebih lanjut terkait arsitektur model dapat dilihat pada penjelasan di bab sebelumnya. Rasio data set pelatihan dan pengujian adalah 85% berbanding 15%, hasil pelatihan untuk tiap AP dapat dilihat pada Gambar 4.20.



Gambar 4.20 Hasil pelatihan model

Pada *validation loss*, yang mengukur kinerja model pada data yang tidak terlihat selama pelatihan menunjukkan penurunan pada tahap pelatihan. Ketiga model memiliki kemiripan penurunan *validation loss* yang aman tinggi di beberapa *epoch* awal. Model RuijieAP1 mengalami penurunan dari 9.1808 ke 0.8966 dan stabil ada pada angka tersebut. Begitu juga dengan model RuijieAP2 mengalami penurunan dari 9.1848 ke 0.8172. Pada model RuijieAP3 mengalami penurunan dari 8.9969 ke 0.9142.

b. Membuat Data Sintetis

Setelah model untuk ketiga akses poin selesai dibuat, selanjutnya adalah memanfaatkan model untuk membuat data sintetis. Tiap model akan digunakan untuk mengisi nilai RSSI dengan interval 10cm, kode ada pada Gambar 4.21.

```
ap_id = ['1', '2', '3']
for id in ap_id:
    # 1. SETUP: Define a checkpoint file and how often to save
    CHECKPOINT_FILE = f'synthetic_data_progress-{id}-10.pkl'
```

```
SAVE_INTERVAL = 1000 # Save progress every 1000 iterations

model_synth.load_weights(f'model_synth_best_AP{id}.keras')

# 2. LOAD/RESUME: Check if a checkpoint file exists
if os.path.exists(CHECKPOINT_FILE):
    print(f"Found checkpoint file. Resuming from 'CHECKPOINT_FILE}'...")
    with open(CHECKPOINT_FILE, 'rb') as f:
        test_synth_data = pickle.load(f)

    start_index = len(test_synth_data.y)
    print(f"Resuming prediction from iteration {start_index}.")

else:
    print(" No checkpoint file found. Starting a new generation...")
    test_synth_data = TrainingData(
          name= None,
          x=[],
          y=[],
          ssid=None
      )


    for xx in range(-400, 400 + 1, 10):
        for xy in range(-600, 600 + 1, 10):
            test_synth_data.x.append([float(xx), float(xy)])

 start_index = 0

# 3. GENERATION LOOP: The long-running process
# The loop now starts from 'start_index'
total_items = len(test_synth_data.x)
for i in range(start_index, total_items):

    # Get the current x value
    current_x = test_synth_data.x[i]

    # Make the prediction
    predicted_y = model_synth.predict(x=np.array([current_x]))

    # Append the result
    test_synth_data.y.append(predicted_y)

    if (i + 1) % 100 == 0: # Print progress every 100 steps
        print(f"Progress: {i + 1} / {total_items} predicted.")
        print(f"Coord: {current_x}")

    # 4. PERIODIC SAVE: Save progress at the specified interval
    if (i + 1) % SAVE_INTERVAL == 0:
        print(f"\n--- Saving progress at iteration {i + 1} ---\n")
        with open(CHECKPOINT_FILE, 'wb') as f:
            pickle.dump(test_synth_data, f)

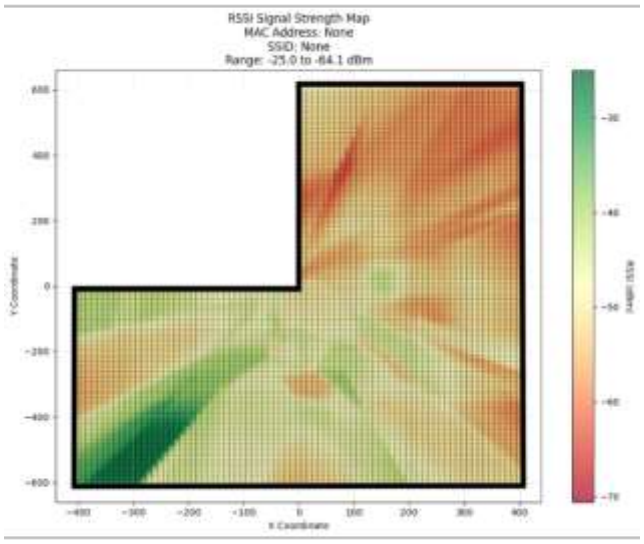
# 5. FINAL SAVE: Always save the completed data at the end
print(" Generation complete! Saving final data...")
with open(CHECKPOINT_FILE, 'wb') as f:
    pickle.dump(test_synth_data, f)
```

Gambar 4.21 Kode untuk membuat data sintetis

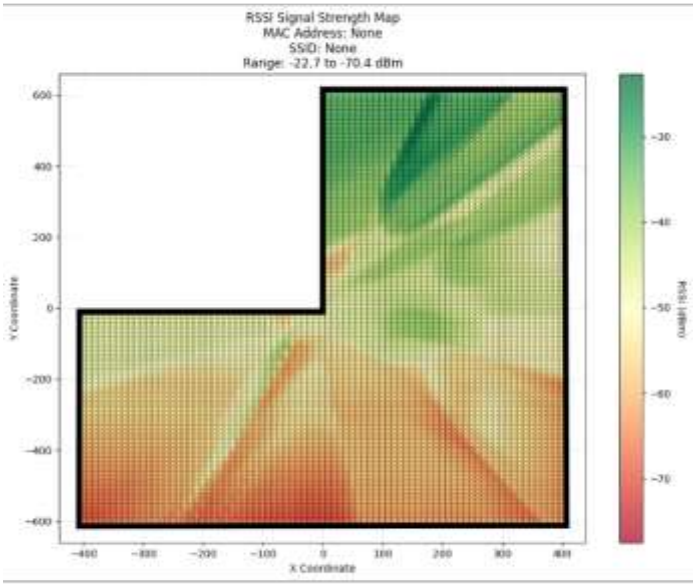
Tahap pembuatan data sintetis memakan waktu yang lumayan lama. Untuk menangani risiko gagal ditambah sistem *checkpoint* untuk melakukan penyimpanan progres dengan interval setiap membuat 1000 nilai RSSI. Total RSSI yang dihasilkan untuk ketiga AP adalah 29403 dengan 9801 titik referensi, seperti pada Tabel 4.3. Hasil dari data sintetis dapat dilihat pada Gambar 4.22 untuk RuijieAP1, Gambar 4.23 untuk RuijieAP2, Gambar 4.24 untuk RuijieAP3. Visualisasi menggambarkan distribusi dan nilai sinyal RSSI yang dihasilkan oleh model data sintetis (hijau mengindikasikan sinyal makin kuat dan merah mengindikasikan sinyal makin lemah) pada tiap titik referensi dengan interval 10cm.

Tabel 4.3 Hasil Sintetis Data

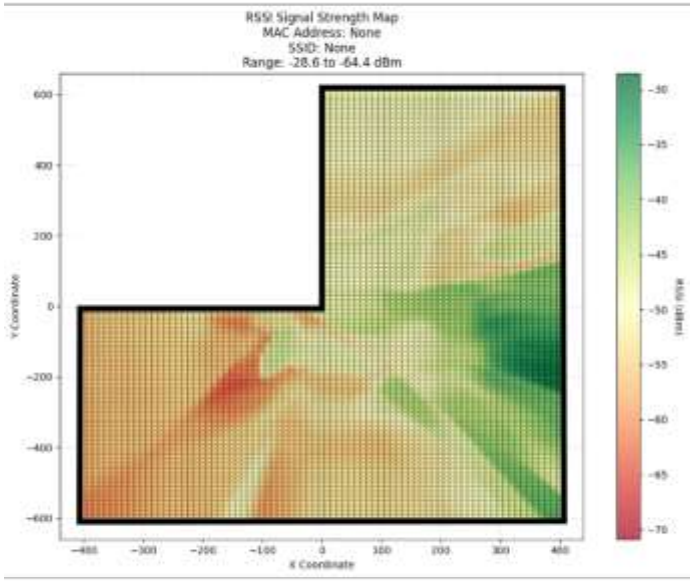
Label	**Nilai**
Total Akses Poin	3
Total RSSI	29.403
Total Titik Referensi	9.801



Gambar 4.22 Ploting nilai RSSI RuijieAP1



Gambar 4.23 Ploting nilai RSSI RuijieAP2



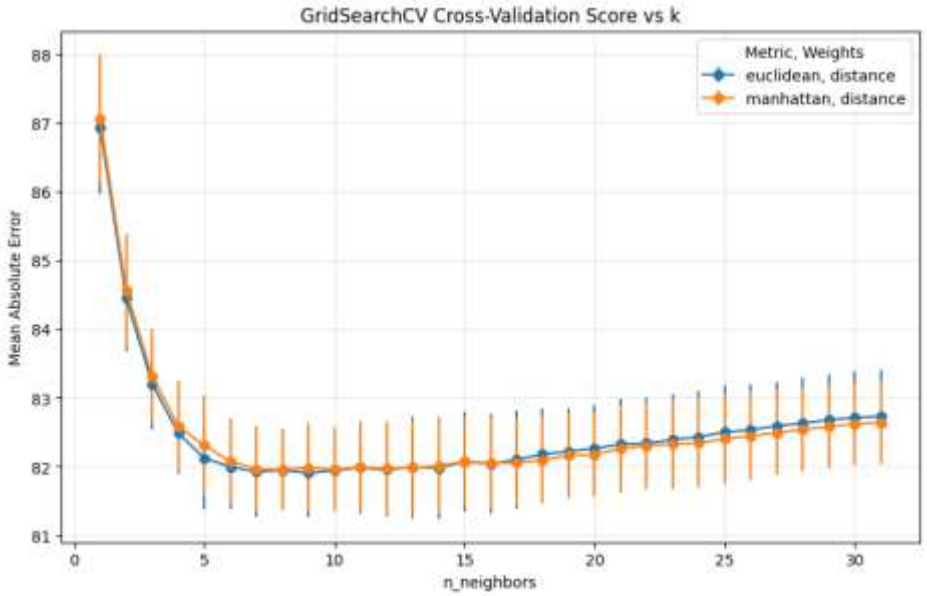
Gambar 4.24 Ploting nilai RSSI RuijieAP3

4.3. Pengembangan Model Prediksi

Pada tahap ini dikembangkan model prediksi yang nantinya dapat digunakan pada fase *online* pada sistem IPS dan juga sebagai evaluasi performa *dataset* sintetik. Seperti yang sudah dijelaskan di bab sebelumnya, *hyperparameter tunning* akan dijalakan untuk tiga skenario model KNN. Semua skenario dilakukan normalisasi koordinat dan pembagian data pelatihan dan validasi adalah 70% dan 30%.

4.3.1. Skenario A (Data original)

Pada skenario A data set yang digunakan adalah data set awal hasil pengumpulan data, tidak dilakukan pemrosesan data selain yang disebutkan sebelumnya. Jumlah sampel yang digunakan untuk pelatihan adalah 29892 dan validasi sebanyak 12812. Hasil yang diperoleh dari *hyperparameter tunning* ada pada Gambar 4.25 yang memvisualisasikan nilai eror (seberapa jauh titik koordinat prediksi meleset dari koordinat sebenarnya dalam satuan cm) pada sumbu x dan seberapa banyak tetangga (n) yang dibutuhkan untuk menghasilkan eror prediksi terendah pada sumbu y. Hasil pelatihan model skenario A menunjukkan performa terbaik ada pada pada n=9 dengan eror prediksi pada saat pelatihan sebesar 81,9 cm.

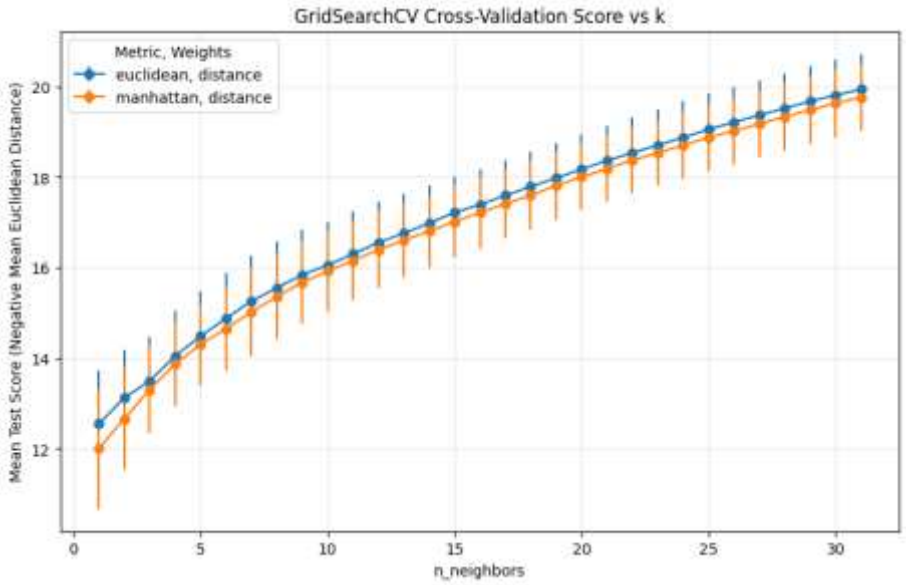


Gambar 4.25 Hasil Pelatihan KNN

4.3.2. Skenario B (*Signal Filter*)

Pada skenario B data set yang digunakan adalah data set awal hasil pengumpulan data dan dilakukan pemrosesan berupa *noice reduction* menggunakan kalman Filter. Jumlah sampel yang digunakan untuk pelatihan adalah 29892 dan validasi sebanyak 12812. Hasil yang

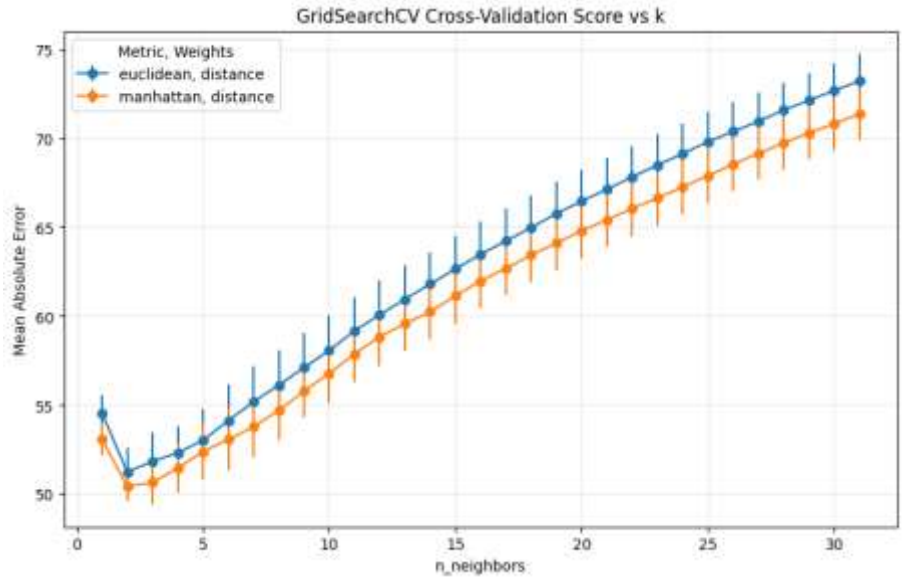
diperoleh dari *hyperparameter tunning* ada pada Gambar 4.26 yang memvisualisasikan nilai eror (seberapa jauh titik koordinat prediksi meleset dari koordinat sebenarnya dalam satuan cm) pada sumbu x dan seberapa banyak tetangga (n) yang dibutuhkan untuk menghasilkan eror prediksi terendah pada sumbu y. Hasil pelatihan model skenario B menunjukkan performa terbaik pada n=1 dengan eror prediksi pada saat pelatihan sebesar 12,0 cm.



Gambar 4.26 Hasil Pelatihan KNN + Kalman Filter

4.3.3. Skenario C (Data set Sintetis)

Pada skenario B data set yang digunakan adalah data set sintetis. Jumlah sampel yang digunakan untuk pelatihan adalah 5180 dan validasi sebanyak 2221. Hasil yang diperoleh dari *hyperparameter tunning* ada pada Gambar 4.27 yang memvisualisasikan nilai eror (seberapa jauh titik koordinat prediksi meleset dari koordinat sebenarnya dalam satuan cm) pada sumbu x dan seberapa banyak tetangga (n) yang dibutuhkan untuk menghasilkan eror prediksi terendah pada sumbu y. Hasil pelatihan model skenario C menunjukkan performa terbaik pada n=2 dengan eror prediksi pada saat pelatihan sebesar 50,42 cm.



Gambar 4.27 Hasil Pelatihan KNN + Data Sintetis

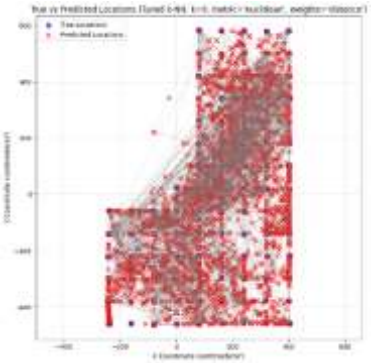
Tabel 4.4 Komparasi Eror

Metrik Evaluasi	Skenario A	Skenario B	Skenario C
Mean Euclidean Error (cm)	78,74	**11,02**	48,14
Median Euclidean Error (cm)	11,95	**0,00**	18,37
Standard Deviation of Error (cm)	123,97	**55,62**	74,91
Maximum Error (cm)	990,46	851,35	**789,03**
RMSE (cm)	103,85	**40,10**	62,96

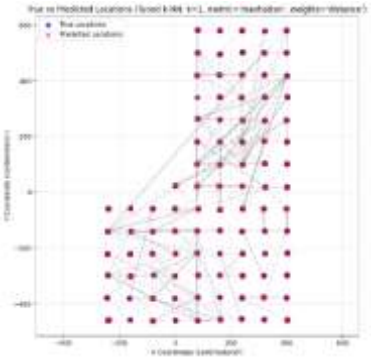
Berdasarkan Tabel 4.4, penerapan Kalman Filter memberikan peningkatan akurasi yang paling signifikan pada data set pelatihan. Rata-rata eror posisi turun drastis dari 78,74 cm pada skenario A menjadi hanya 11,02 cm pada skenario B. Nilai median eror 0,00 cm pada skenario B mengindikasikan akurasi yang sangat tinggi pada data pelatihan.

Namun performa pada *Median Euclidean Error* ini mengisyaratkan adanya potensi model yang terlalu kaku atau terspesialisasi. Hal ini terjadi karena Kalman Filter sangat efektif dalam menghaluskan dan mempelajari karakteristik derau (*noise*) spesifik dari data latih, visualisasi prediksi dapat dilihat pada Gambar 4.29, model prediksi tidak pernah menghasilkan prediksi di luar titik referensi yang ada di dataset. Akibatnya, model menjadi terlalu terspesialisasi pada data tersebut, tetapi ketika diuji pada data set lain dengan pola derau yang berbeda, kinerjanya menurun secara signifikan (seperti terlihat pada Tabel 4.5). Visualisasi

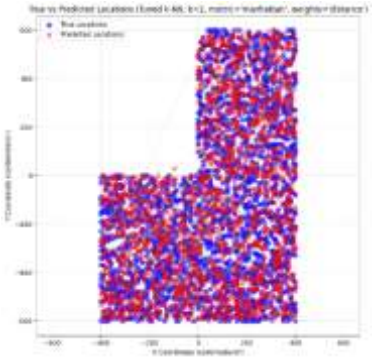
prediksi pada tahap pelatihan dari masing-masing hasil pada tabel 4.4 dapat dilihat pada Gambar 4.28 untuk skenario A, Gambar 4.29 untuk skenario B, dan Gambar 4.30 untuk skenario C. Visualisasi memberikan gambaran titik asli dan titik prediksi yang dilakukan oleh model, dihubungkan dengan garis.



Gambar 4.28 Visualisasi hasil Skenario A



Gambar 4.29 Visualisasi hasil Skenario B

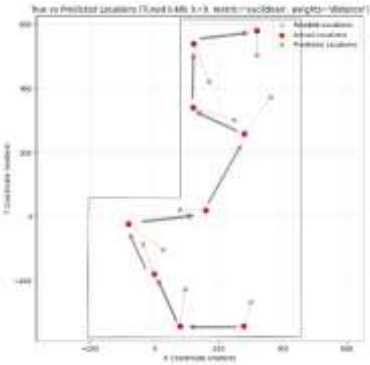


Gambar 4.30 Visualisasi hasil Skenario C

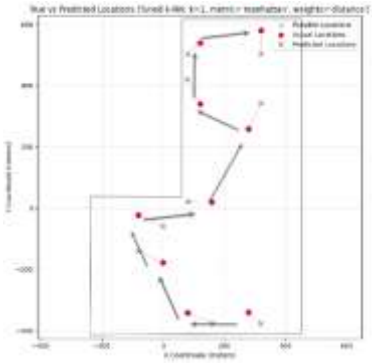
Terakhir pada tahap ini, model prediksi di uji menggunakan *dataset* baru dari dua perangkat yang berbeda. Pada perangkat Asus model yang menggunakan data sintetis menunjukkan kinerja terbaik di antara ketiganya, komparasi eror dapat dilihat pada Tabel 4.5. Model ini mencapai rata-rata eror posisi terendah (207,16 cm) dan RMSE terendah (164,97 cm). Visualisasi pelacakan prediksi dapat dilihat pada Gambar 4.31 untuk skenario A, Gambar 4.32 untuk skenario B, dan Gambar 4.33 untuk skenario C. Pada visualisasi terdapat alur *tracking* (arah panah), posisi sebenarnya perangkat pada dunia nyata (titik merah) dan hasil prediksi model (silang hijau) yang diindikasikan dengan garis abu-abu.

Tabel 4.5 Komparasi Eror prediksi pada perangkat Asus

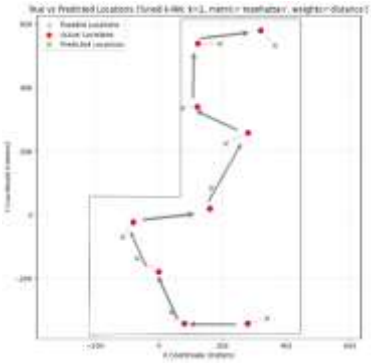
Metrik Evaluasi	Skenario A	Skenario B	Skenario C
Mean Euclidean Error (cm)	228,64	227,24	**207.16**
Median Euclidean Error (cm)	204,77	**201,32**	208,52
Standard Deviation of Error (cm)	139,35	132,90	**129,72**
Maximum Error (cm)	923,25	**850,86**	906,63
RMSE (cm)	190,34	186,76	**164,97**



Gambar 4.31 Visualisasi *tracking* menggunakan model Skenario A pada perangkat Asus



Gambar 4.32 Visualisasi *tracking* menggunakan model Skenario B pada perangkat Asus

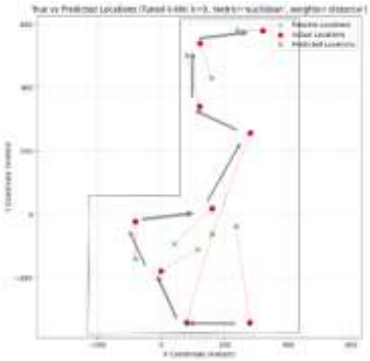


Gambar 4.33 Visualisasi *tracking* menggunakan model Skenario C pada perangkat Asus

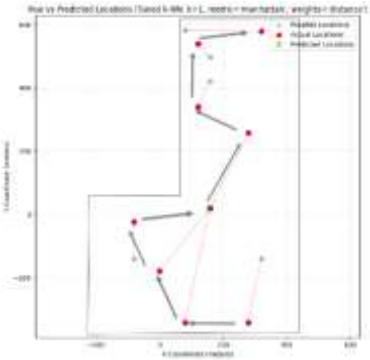
Pada perangkat Macbook, komparasi eror dapat dilihat pada Tabel 5.6 menunjukkan model yang menggunakan data sintetis kinerja memiliki terbaik di antara ketiganya. Model ini mencapai rata-rata eror posisi terendah (245,04 cm) dan RMSE terendah (233,83 cm). Visualisasi pelacakan prediksi dapat dilihat pada Gambar 4.31 untuk skenario A, Gambar 4.32 untuk skenario B, dan Gambar 4.33 untuk skenario C. Pada visualisasi terdapat alur tracking (arah panah), posisi sebenarnya perangkat pada dunia nyata (titik merah) dan hasil prediksi model (silang hijau) yang diindikasikan dengan garis abu-abu.

Tabel 4.6 Komparasi Eror prediksi pada perangkat Macbook

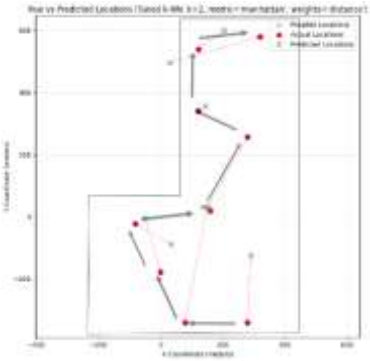
Metrik Evaluasi	Skenario A	Skenario B	Skenario C
Mean Euclidean Error (cm)	273.22	287.36	**245.04**
Median Euclidean Error (cm)	240.05	255.38	**216.31**
Standard Deviation of Error (cm)	**191.53**	237.95	222.06
Maximum Error (cm)	825.29	**823.99**	833.21
RMSE (cm)	241.94	263.81	**233.83**



Gambar 4.34 Visualisasi tracking menggunakan model Skenario A pada perangkat Macbook



Gambar 4.35 Visualisasi tracking menggunakan model Skenario B pada perangkat Macbook



Gambar 4.36 Visualisasi tracking menggunakan model Skenario C pada perangkat Macbook

4.4. Diskusi

Selain pengembangan IPS dan model prediksi data, pada penelitian ini terdapat beberapa temuan menarik yang dapat membuka peluang untuk pengembangan lebih lanjut dan penerapan teknologi ini dalam konteks IPS. Salah satu kendala yang dialami pada masa pengambilan data adalah waktu dan tenaga yang dibutuhkan tidak sedikit. Beberapa percobaan pengambilan data set mengalami gangguan karena pengambilan data terjadi pada hari yang berbeda, hal ini menyebabkan terjadinya *temporal drift* yang menyebabkan *state* sinyal mengalami penurunan atau perubahan sehingga sangat mempengaruhi kinerja model pada saat percobaan prediksi lokasi (Kayano et al., 2025).

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil yang diperoleh, pada perangkat Asus hasil pengujian model pertama, skenario B dengan model KNN dengan pemrosesan sinyal unggul dalam rata-rata eror sebesar 11,02 cm dibandingkan dengan skenario A dengan rata-rata error 78,74 cm. Sementara itu, penggunaan data sintetis juga menunjukkan perbaikan yang substansial dibandingkan skenario A. Rerata eror sebesar 48,14 cm, yang merupakan peningkatan akurasi sebesar 38,9% dari model skenario A. Meskipun tidak seakurat skenario B, model dengan data sintetis menunjukkan standar deviasi yang lebih rendah dibandingkan model dasar, mengindikasikan prediksi yang lebih konsisten. Selain itu, model ini mencatatkan error maksimum terendah, menunjukkan bahwa penambahan data sintetis dapat membantu mengurangi skenario prediksi terburuk.

Pada pengujian kedua menggunakan data baru dan perangkat baru, skenario C yang menggunakan data sintetis masih menunjukkan kinerja terbaik di antara ketiganya. Pada perangkat Asus, model ini mencapai rerata eror posisi terendah 207,16 cm dan RMSE terendah sebesar 184,97 cm. Peningkatan kinerja ini signifikan dibandingkan model KNN dasar degan RMSE 190,34 cm (peningkatan akurasi 13,3%) dan juga melampaui model dengan Kalman filter dengan RMSE 186,76 cm (peningkatan akurasi 11,6%). Pada perangkat Macbook, skenario C memiliki kinerja terbaik dengan RMSE 233,83 cm, peningkatan dari sekenario A dengan RMSE 241,94 cm (peningkatan akurasi 3,3%) dan Skenario B dengan RMSE 263,81 cm (peningkatan akurasi 11,3%). Temuan ini mengindikasikan bahwa meskipun Kalman filter sangat efektif dalam menghaluskan data yang memiliki karakteristik serupa dengan data latih, kemampuannya untuk melakukan generalisasi atau memahami bentuk data baru yang mungkin memiliki pola noise berbeda tidak sebaik model yang diperkaya dengan data sintetis.

Penelitian ini menyimpulkan bahwa augmentasi *dataset* menggunakan data sintetis dapat membantu meningkatkan akurasi dan kemampuan generalisasi dari sistem penentuan posisi dalam ruangan (IPS) berbasis Wi-Fi fingerprinting KNN. Pendekatan data sintetis dapat dipertimbangkan pada lingkup pengambilan data yang luas atau dengan sumber daya yang terbatas.

5.2. Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa rekomendasi perbaikan yang bisa diterapkan pada pengembangan selanjutnya. Rekomendasi tersebut meliputi:

- **Eksplorasi pada Lingkungan yang Lebih Kompleks**: Menguji metode ini pada dataset yang lebih besar dan beragam, mencakup area multilantai, lebih banyak penghalang, atau lingkungan dengan dinamika pergerakan manusia yang tinggi.
- **Evaluasi Metode Generatif Alternatif**: Membandingkan kinerja model MLP yang digunakan dengan teknik generatif lain yang lebih canggih, seperti *Generative Adversarial Networks* (GANs) atau *Variational Autoencoders* (VAEs), untuk melihat apakah metode tersebut dapat menghasilkan data sintetis yang lebih realistis.
- **Pengujian dengan Perangkat Heterogen**: Mengumpulkan data dari berbagai jenis perangkat seluler untuk melatih dan menguji model, tetapi lebih mencerminkan skenario penggunaan di dunia nyata di mana pengguna memiliki perangkat yang berbeda-beda.

DAFTAR PUSTAKA

Aziz, N. F. N. A., Jamil, N., & Din, M. M. (2019). An Improved Indoor Location Technique Using Combination of Kalman Filter and Centroid Positioning. *Journal of Communications*, 17–25. https://doi.org/10.12720/jcm.14.1.17-25

Chen, H., Yang, J., Hao, Z., Ga, M., Han, X., Zhang, X., & Chen, Z. (2023). Research on indoor positioning method based on LoRa-improved fingerprint localization algorithm. *Scientific Reports*, *13*(1), 13981. https://doi.org/10.1038/s41598-023-41250-x

Firmansyah, F. (2024). *Pengembangan Sistem Posisi Indoor Bergerak menggunakan Trilaterasi dan Filter Kalman* (Thesis, Universitas Islam Indonesia). Universitas Islam Indonesia. Retrieved from https://dspace.uii.ac.id/handle/123456789/51910

Firmansyah, F., Rahma, F., Irianto, K. D., & Shiddiqi, A. M. (2024). Indoor Positioning System: A Brief Review of Its Technologies and Signal-Filtering Techniques. *2024 International Conference on Smart Computing, IoT and Machine Learning (SIML)*, 1–7. Surakarta, Indonesia: IEEE. https://doi.org/10.1109/SIML61815.2024.10578093

Goecks, L. S., Habekost, A. F., Coruzzolo, A. M., & Sellitto, M. A. (2024). Industry 4.0 and Smart Systems in Manufacturing: Guidelines for the Implementation of a Smart Statistical Process Control. *Applied System Innovation*, *7*(2), 24. https://doi.org/10.3390/asi7020024

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge, Massachusetts: The MIT Press.

Homayounvala, E., Nabati, M., Shahbazian, R., Ghorashi, S. A., & Moghtadaiee, V. (2019). A novel smartphone application for indoor positioning of users based on machine learning. *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International*

Symposium on Wearable Computers, 430–437. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3341162.3349300

Kayano, K., Ohta, S., Nishio, T., Yoda, D., Taniguchi, K., & Nabetani, T. (2025). Experimental Evaluation of Long-Term Concept Drift and Its Mitigation in Wi-Fi CSI Sensing. *2025 IEEE 22nd Consumer Communications & Networking Conference (CCNC)*, 1–2. https://doi.org/10.1109/CCNC54725.2025.10975920

Kia, G., Laura, R., & Talvitie, J. (2022). Toward Accurate Indoor Positioning: An RSS-Based Fusion of UWB and Machine-Learning-Enhanced Wi-Fi. *Toward Accurate Indoor Positioning: An RSS-Based Fusion of UWB and Machine-Learning-Enhanced Wi-Fi*. https://doi.org/10.3390/s22093204

Liu, S., de Lacerda, R., & Fiorina, J. (2021). WKNN indoor Wi-Fi localization method using k-means clustering based radio mapping. *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*. Helsinki, Finland: IEEE. https://doi.org/10.1109/VTC2021-Spring51267.2021.9448961

Lu, H., Zhang, L., Chen, H., Zhang, S., Wang, S., Peng, H., & Zou, J. (2023). An Indoor Fingerprint Positioning Algorithm Based on WKNN and Improved XGBoost. *Sensors*, *23*(8), 3952. https://doi.org/10.3390/s23083952

Naderi Salim, S., Alizadeh, M. M., Chamankar, S., & Schuh, H. (2023). IMPROVED INDOOR POSITIONING USING FINGERPRINT TECHNIQUE AND WEIGHTED K-NEAREST NEIGHBOUR. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *X-4-W1-2022*, 575–580. https://doi.org/10.5194/isprs-annals-X-4-W1-2022-575-2023

Philippopoulos, P. I., Koutrakis, K. N., Tsafaras, E. D., Papadopoulou, E. G., Sigalas, D., Tselikas, N. D., … Vassilakis, C. (2025). Cost-Efficient RSSI-Based Indoor Proximity

Positioning, for Large/Complex Museum Exhibition Spaces. *Sensors*, *25*(9), 2713. https://doi.org/10.3390/s25092713

Rectifier (neural networks). (2025). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Rectifier_(neural_networks)&oldid=1301644984

Ren, Q., Wang, Y., Liu, S., & Lv, X. (2023). FSTNet: Learning spatial–temporal correlations from fingerprints for indoor positioning. *Ad Hoc Networks*, *149*, 103244. https://doi.org/10.1016/j.adhoc.2023.103244

Shang, S., & Wang, L. (2022). Overview of Wi-Fi fingerprinting-based indoor positioning. *IET Communications*, *16*(7), 725–733. https://doi.org/10.1049/cmu2.12386

Sohan, A. A., Ali, M., Fairooz, F., Rahman, A. I., Chakrabarty, A., & Kabir, Md. R. (2019). Indoor Positioning Techniques using RSSI from Wireless Devices. *2019 22nd International Conference on Computer and Information Technology (ICCIT)*, 1–6. Dhaka, Bangladesh: IEEE. https://doi.org/10.1109/iccit48885.2019.9038591

Vaccari, L., Coruzzolo, A. M., Lolli, F., & Sellitto, M. A. (2024). *Indoor Positioning Systems in Logistics: A Review*. https://doi.org/10.3390/logistics8040126

Wang, K., Yu, X., Xiong, Q., Zhu, Q., Lu, W., Huang, Y., & Zhao, L. (2019). Learning to Improve WLAN Indoor Positioning Accuracy Based on DBSCAN-KRF Algorithm From RSS Fingerprint Data. *IEEE Access*, *7*, 72308–72315. https://doi.org/10.1109/ACCESS.2019.2919329

Wang, W., & Liu, M. (2025). *Hyperbolic Spatial Covariance Modeling with Adaptive Signal Filtering for Robust Wi-Fi Indoor Positioning*. https://doi.org/10.3390/s25134125

Willyanto, A., & Santoso, P. (2025). SISTEM PELACAKAN POSISI DALAM RUANGAN. *Jurnal Teknik Elektro*, *18*(1), 1–7. https://doi.org/10.9744/jte.18.1.1-7

Yudha, D. P., Hasbi, B. I., & Sukarna, R. H. (2018). INDOOR POSITIONING SYSTEM BERDASARKAN FINGERPRINTING RECEIVED SIGNAL STRENGTH (RSS) WI-FI DENGAN ALGORITMA K-NEAREST NEIGHBOR (K-NN). *ILKOM Jurnal Ilmiah*, *10*(3), 274–283. https://doi.org/10.33096/ilkom.v10i3.364.274-283

Zhang, C., Li, Z., & Li, X. (2025, May 13). *Hybrid Wi-Fi/PDR Indoor Localization with Fingerprint Matching*. arXiv. https://doi.org/10.48550/arXiv.2505.08258

LAMPIRAN

LAMPIRAN A

Koordinat (x, y)	Jumlah RSSI
155, 200	1712
156, 359	1764
159, 120	116
159, 282	1752
159, 440	1682
160, 520	1658
160, 601	822
161, 121	1688
236, 199	42
239, 121	30
239, 358	1800
239, 599	910
240, 279	1698
240, 522	78
241, 439	1602

318, 360	1800
319, 199	1704
319, 439	1604
319, 520	1646
319, 602	1728
320, 118	1734
322, 282	158
399, 199	48
399, 279	84
399, 441	1812
399, 520	1722
400, 119	102
400, 360	186
400, 602	498
478, 1000	1800
479, 361	1794
479, 441	1704
479, 601	3132

479, 681	1692
479, 761	1678
479, 920	8
480, 120	1696
480, 202	1596
480, 844	1598
480, 1081	1710
480, 1161	1584
481, 282	1662
481, 521	3546
558, 601	1550
559, 282	96
559, 442	1182
559, 921	1680
559, 1001	1642
559, 1076	1590
559, 1160	252
560, 202	160

560, 360	90
560, 518	1824
560, 680	1674
560, 761	1736
560, 840	114
561, 122	1704
638, 125	1716
638, 920	1740
639, 200	1736
639, 440	490
639, 601	96
639, 761	1490
639, 1001	1662
639, 1160	1396
640, 280	1632
640, 360	1686
640, 680	1818
640, 1082	1662

641, 521	1044
641, 839	1560
716, 280	246
718, 202	1772
718, 521	1740
718, 1001	1658
719, 122	122
719, 764	1750
719, 1081	1756
720, 441	1368
720, 682	1682
720, 838	1692
720, 922	1638
720, 1159	112
721, 360	108
721, 599	1586
799, 200	240
799, 520	1726

799, 839	1726
799, 1081	1776
799, 1160	1654
800, 281	1386
800, 441	1854
800, 598	1554
800, 681	1716
800, 760	1752
800, 998	1690
801, 121	1618
801, 361	1638
801, 921	1720

Lampiran B

```
def equalize_coordinate_occurrences(training_data_list:
List[TrainingData], random_seed=None) -> List[TrainingData]: """ Equalize
the occurrences of coordinates within each TrainingData object
independently, using the global average occurrence count as the target.
    Parameters:
- training_data_list: List of TrainingData objects to equalize
- random_seed: Optional seed for reproducible random sampling

Returns:
- List of new TrainingData objects with equalized coordinate occurrences
"""

if random_seed is not None:
    random.seed(random_seed)
    np.random.seed(random_seed)

if not training_data_list:
    return []
```

```
# Step 1: Calculate global average occurrence count across ALL
TrainingData objects
all_coord_counts = []

for td in training_data_list:
    coord_counter = Counter(tuple(coord) for coord in td.x)
    all_coord_counts.extend(coord_counter.values())

global_average = sum(all_coord_counts) // len(all_coord_counts) if
all_coord_counts else 0

print(f"Global average occurrence count: {global_average}")
print(f"Total coordinate occurrences across all objects:
{all_coord_counts}")

# Step 2: Process each TrainingData object independently
new_training_data_list = []

for td_idx, td in enumerate(training_data_list):
    print(f"\nProcessing {td.name}:")

    # Group coordinates and their corresponding y values
    coord_groups = defaultdict(list)  # coord_tuple -> [(y_value,
original_index)]

    for i, (coord, y_val) in enumerate(zip(td.x, td.y)):
        coord_tuple = tuple(coord)
        coord_groups[coord_tuple].append((y_val, i))

    # Create new TrainingData object
    new_td = TrainingData(
        name=f"{td.name}_equalized",
        ssid=td.ssid,
        x=[],
        y=[]
    )

    # Process each coordinate group in this TrainingData object
    for coord_tuple, data_list in coord_groups.items():
        current_count = len(data_list)
        y_values = [item[0] for item in data_list]

        if current_count > global_average:
            # Randomly sample down to global average
            selected_data = random.sample(data_list, global_average)
            print(f"  Coordinate {coord_tuple}: Reduced from
{current_count} to {global_average}")

            # Add selected data to new object
            for y_val, _ in selected_data:
                new_td.x.append(list(coord_tuple))
                new_td.y.append(y_val)

        elif current_count < global_average:
            # Keep all existing data
            for y_val, _ in data_list:
```

```
                new_td.x.append(list(coord_tuple))
                new_td.y.append(y_val)

            # Add synthetic data with averaged y values
            avg_y = np.mean(y_values)
            needed = global_average - current_count

            for _ in range(needed):
                new_td.x.append(list(coord_tuple))
                new_td.y.append(avg_y)

            print(f"  Coordinate {coord_tuple}: Increased from
{current_count} to {global_average} (avg y: {avg_y:.2f})")

        else:
            # Already at target count
            for y_val, _ in data_list:
                new_td.x.append(list(coord_tuple))
                new_td.y.append(y_val)

            print(f"  Coordinate {coord_tuple}: Already at target count
({current_count})")

    new_training_data_list.append(new_td)

# Print summary
print("\nEqualization Summary:")
for i, (original, new) in enumerate(zip(training_data_list,
new_training_data_list)):
    print(f"{original.name}: {len(original.x)} -> {len(new.x)} data
points")

return new_training_data_list
```