

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Localization adalah suatu mekanisme untuk menemukan dan menentukan lokasi dari suatu objek. Secara umum, penentuan posisi atau lokasi dibagi menjadi dua, yakni *outdoor* dan *indoor localization*. Penentuan lokasi di luar ruangan atau *outdoor* menggunakan teknologi *Global Positioning System* (GPS). Sedangkan penentuan lokasi di dalam ruangan atau *indoor* tidak bisa menggunakan GPS karena pelemahan sinyal GPS saat memasuki gedung sehingga proses penentuan lokasi menjadi tidak memungkinkan.(Brena *et al.*, 2017). Oleh karena itu, dalam penentuan lokasi dalam ruangan atau gedung digunakan teknologi nirkabel seperti WiFi dan Bluetooth.(Zafari, Gkelias and Leung, 2019).

Indoor Localization saat ini dapat menggunakan 2 metode, yakni metode propagasi sinyal dan *fingerprinting*. Metode propagasi sinyal memperoleh koordinat dari objek berdasarkan jarak antara target dan perangkat yang berfungsi sebagai *reference points*. Kelemahan metode ini adalah rentan terhadap lingkungan yang kompleks (tembok, pintu, jendela, orang berjalan), sehingga menyebabkan gangguan sinyal yang mempengaruhi koordinat yang diperoleh. Akibatnya, koordinat yang diperoleh tidak sesuai dengan koordinat sebenarnya. Salah satu metode untuk mengatasi kelemahan metode propagasi sinyal pada struktur lingkungan yang kompleks adalah metode *fingerprinting*. Metode *fingerprinting* adalah teknik penentuan lokasi dengan mengklasifikasikan pola sinyal yang diperoleh berdasarkan *data training* yang telah didapat sebelumnya.(Subedi dan Pyun, 2017). Metode *fingerprinting* dibagi menjadi 2 tahap, yakni tahap *offline* dan tahap *online*. Tahap *offline* adalah tahap pembentukan *data training*, yakni data yang terdiri dari kumpulan pola sinyal yang diberi label sesuai dengan nama lokasi dimana pola sinyal tersebut diperoleh. *Data training* ini nantinya akan dipakai untuk melatih model yang digunakan untuk mengklasifikasikan pola sinyal. Tahap *online* adalah tahap penentuan lokasi dengan cara mengklasifikasikan pola sinyal yang diperoleh berdasarkan model yang telah dilatih dengan *data training* sebelumnya sehingga memperoleh hasil klasifikasi berupa nama lokasi. (Subedi dan Pyun, 2017). Keuntungan dari metode *fingerprinting* adalah dapat menghindari efek samping dari gangguan sinyal yang dapat mengakibatkan kesalahan klasifikasi. Keuntungan lainnya adalah tidak perlu mengetahui posisi dari *reference points* sehingga praktis untuk digunakan. (Jiang *et al.*, 2015)

Saat ini, teknologi yang paling banyak digunakan dengan metode *fingerprinting* adalah WiFi dan *Bluetooth Low Energy* (BLE). Penggunaan WiFi dengan metode *fingerprinting* menghasilkan akurasi penentuan lokasi mencapai 85,9%.(Jiang *et al.*, 2015). Sayangnya, durasi *advertisement SSID* dari Wifi yang lambat, yakni sekitar 100ms tiap *access point* menyebabkan satu periode scanning membutuhkan waktu beberapa detik. Hal ini tentunya dapat mengakibatkan kesalahan dalam penentuan lokasi saat perangkat bergerak. (Faragher dan Harle,

2015). Teknologi lain yang dapat digunakan untuk *Indoor Localization* adalah *Bluetooth Low Energy* (BLE). Menurut Faragher dan Harle (2015), *advertisement packet* dari BLE memiliki durasi lebih cepat dari WiFi, sehingga BLE dapat digunakan sebagai pengganti alternatif dari WiFi. Keunggulan lain dari BLE adalah ukuran perangkat yang kecil dibandingkan dengan perangkat WiFi, sumber energi menggunakan baterai yang dapat bertahan berbulan-bulan bahkan bertahun-tahun, harganya murah, didesain untuk komunikasi “*machine-to-machine*” dan mudah untuk digunakan.

Penggunaan teknologi BLE dan metode *fingerprinting* saat ini menggunakan konsep *Device Based Localization* (DBL), yakni sebuah konsep dimana perangkat yang dilacak atau dicari lokasinya secara aktif melakukan *scanning* terhadap *reference points* sekaligus melakukan proses klasifikasi pola sinyal menjadi nama lokasi. (Zafari, Gkelias and Leung, 2019). Konsep ini tidak dapat digunakan pada kondisi dimana objek yang dilacak tidak dibolehkan atau tidak memungkinkan membawa perangkat yang dapat melakukan pemindaian aktif (contoh perangkat: *smartphone*) seperti pada rumah sakit jiwa, penjara, dsb. Untuk itulah diperlukan konsep lain yang dapat mengatasi permasalahan tersebut.

Berdasarkan permasalahan di atas, pada penelitian ini penulis menawarkan solusi untuk mengatasi permasalahan pada konsep DBL, yakni dengan mengimplementasikan *indoor localization* menggunakan metode *fingerprinting* dengan teknologi *Bluetooth Low Energy* berbasis *Monitor Based Localization* (MBL). Konsep MBL adalah konsep dimana perangkat yang dilacak hanya memancarkan sinyal secara simultan, sedangkan komponen lain yang akan melakukan proses penentuan lokasi. Konsep MBL bekerja dengan menggunakan setidaknya 3 komponen. Komponen pertama sebagai perangkat yang dilacak, penulis menggunakan *passive tag* yang akan memancarkan sinyal bluetooth secara terus menerus. Komponen yang kedua adalah perangkat yang berperan sebagai *reference points* yang akan menangkap sinyal bluetooth yang dipancarkan oleh *passive tag* kemudian mengirimkan data RSSI-nya menuju server. Komponen ketiga adalah server yang akan menerima data RSSI dari *reference points* untuk diklasifikasikan menjadi lokasi ruangan berdasarkan *data training* yang telah dikumpulkan sebelumnya.

1.2 Identifikasi Masalah

Permasalahan yang muncul karena menggunakan konsep DBL adalah tidak bisa diimplementasikan pada kondisi dimana objek yang akan ditentukan lokasinya tidak memungkinkan untuk membawa perangkat yang dapat melakukan klasifikasi pola sinyal menjadi lokasi contohnya seperti *smartphone*. Contoh kasus pada rumah sakit jiwa, dimana pasien tidak diperbolehkan membawa perangkat semacam *smartphone* atau pada penjara dimana para tahanan tidak diperbolehkan membawa peralatan elektronik. Berdasarkan permasalahan ini diperlukan suatu konsep yang dapat mengatasi kelemahan ini.

Salah satu konsep yang dapat digunakan adalah konsep *Monitor Based Localization*. Konsep ini bekerja dengan mengklasifikasikan pola sinyal yang

diperoleh dari suatu *passive tag* menjadi nama lokasi. Untuk mengetahui apakah konsep ini layak diimplementasikan pada kehidupan nyata, maka perlu dilakukan penelitian untuk mengukur akurasi sistem *indoor localization* yang menggunakan konsep MBL. Untuk mengukur akurasi, pada penelitian ini penulis akan menggunakan konsep MBL yang terdiri dari 3 komponen seperti yang dijelaskan di atas. Implementasi sistem akan dibagi menjadi 2 tahap, yakni tahap *offline* dan tahap *online*. Pada tahap *offline*, pola sinyal pada masing-masing lokasi atau ruangan akan direkam dan disimpan menjadi *data training* pada server. Mekanisme kerjanya adalah pertama, *passive tag* dinyalakan pada salah satu ruangan, kemudian *reference points* akan menangkap sinyal yang dipancarkan oleh *passive tag*. Selanjutnya, *reference points* akan mengirimkan RSSI dari *passive tag* menuju server. Kemudian pada server, data RSSI yang diterima dari *reference points* akan disimpan dan diberi label nama ruangan. Proses ini diulang sampai pola sinyal dari semua ruang selesai direkam, disimpan, dan diberi label nama ruangan.

Tahap selanjutnya adalah tahap *online*. Pada tahap ini dilakukan proses penentuan lokasi berdasarkan *data training* telah dikumpulkan sebelumnya. Mekanisme kerja tahap *online* adalah pertama, *passive tag* dinyalakan pada salah satu ruang secara acak. Kemudian, *reference points* akan menangkap sinyal yang dipancarkan oleh *passive tag* dan mengirimkan data RSSI-nya menuju server. Pada sisi server, data RSSI yang dikirimkan *reference points* akan diklasifikasikan menjadi nama ruangan menggunakan algoritma KNN berdasarkan *data training* yang telah dikumpulkan sebelumnya. Sebagai catatan, algoritma KNN disini hanya digunakan sebagai *tools* untuk klasifikasi.

1.3 Rumusan Masalah

1. Bagaimana implementasi penentuan lokasi dalam gedung menggunakan metode *fingerprinting* dengan Bluetooth Low Energy?
2. Bagaimana implementasi *Monitor Based Localization* dalam penentuan lokasi dalam gedung?
3. Bagaimana tingkat akurasi dari implementasi pelacakan lokasi dalam gedung menggunakan Bluetooth Low Energy?

1.4 Tujuan

Tujuan dari dilakukannya penelitian ini adalah untuk merancang dan mengimplementasikan *indoor localization* menggunakan metode *fingerprinting* dengan teknologi *Bluetooth Low Energy* (BLE). Konsep penentuan lokasi yang digunakan adalah konsep *Monitor Based Localization* (MBL). Konsep MBL ini ialah terdapat suatu perangkat yang menentukan lokasi dari perangkat lain sehingga dapat digunakan pada kondisi seperti pada rumah sakit jiwa dan penjara.

1.5 Manfaat

Manfaat dilakukannya penelitian ini adalah :

1. Sebagai sebuah solusi alternatif dalam penentuan lokasi dalam gedung pada kondisi seperti rumah sakit jiwa, penjara, dsb. Karena pada kondisi seperti contoh yang disebutkan diatas, tidak memungkinkan objek yang ditentukan lokasinya untuk membawa perangkat semacam *smartphone*
2. Manfaat lain penelitian ini adalah sebagai referensi bagi peneliti yang akan melakukan penelitian dengan tema *indoor localization*.

1.6 Batasan Masalah

Berdasarkan latar belakang diatas didapatkan batasan masalah sebagai berikut :

1. Perangkat pelacakan berbasis *Bluetooth Low Energy* (BLE)
2. Hasil *output* penentuan lokasi hanya berupa nama lokasi, bukan koordinat
3. Mekanisme yang digunakan adalah *Monitor Based Localization*
4. Perangkat yang digunakan pada pengguna yang dilacak adalah *passive tag*

1.7 Sistematika Pembahasan

Sistematika susunan laporan penelitian ini adalah sebagai berikut :

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang penelitian *indoor localization* beserta identifikasi masalah, rumusan masalah, tujuan dilakukannya penelitian, manfaat dari penelitian ini, dan sistematika pembahasan yang menerangkan tentang masing-masing bab pada skripsi ini.

BAB II LANDASAN KEPUSTAKAAN

Bab ini menjelaskan tentang teori-teori pemecahan masalah yang digunakan sebagai pendukung segala sesuatu yang berhubungan dengan topik penelitian ini. Pada bab ini juga menyajikan tentang beberapa penelitian sebelumnya tentang *indoor localization* yang digunakan sebagai acuan dalam penelitian ini.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan mengenai rancangan sistem dan juga alur yang akan menunjang keberhasilan penelitian ini dan agar dapat diimplementasikan di dalam sistem dengan mengacu pada teori-teori pendukung dan metode yang telah dijabarkan pada bab 2.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan menjelaskan secara teknis tentang implementasi dari penelitian ini. Pada bab ini juga akan dijelaskan teknis dan hasil dari pengujian sesuai dengan parameter dan skenario pengujian yang telah dirancang sebelumnya. Hasil implementasi dan pengujian akan dibahas dan dianalisa pada bab 5

BAB V HASIL DAN PEMBAHASAN

Bab ini akan membahas dan menganalisa hasil dari implementasi dan pengujian sistem pada bab 4. Hasil analisa akan digunakan untuk membentuk kesimpulan pada bab 6.

BAB VI PENUTUP

Bab ini menjelaskan tentang kesimpulan yang diambil berdasarkan tahapan - tahapan yang sudah dilakukan mulai dari perancangan, implementasi, pengujian. Pada kesimpulan juga menjawab pertanyaan - pertanyaan pada rumusan masalah dan menyebutkan saran untuk penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Commented [AB1]: typo

Pada Bab Ldanasan Kepustakaan ini terdapat kajian pustaka yang menjelaskan tentang penelitian sebelumnya yang memiliki hubungan dengan penelitian yang penulis lakukan. Dasar teori pada bab ini akan menjelaskan teori-teori yang digunakan dalam penelitian yang penulis lakukan.

2.1 Kajian Pustaka

Terdapat beberapa penelitian yang telah dilakukan sebelumnya yang berhubungan dengan penelitian yang penulis lakukan. Pada Tabel 2.1 di bawah ini menjelaskan tentang penelitian yang telah dilakukan sebelumnya dan penelitian yang dilakukan oleh penulis.

Tabel 2.1 Kajian Pustaka

| No. | Judul | Sitasi | Penelitian | Penelitian Penulis |
|-----|--|-----------------------------------|---|--|
| 1. | Evolution of Indoor Positioning Technologies: A Survey | (Brena <i>et al.</i> , 2017) | <i>Survey Paper</i> mengenai berbagai teknologi, metode, dan algoritma yang digunakan untuk Sistem <i>Indoor Positioning/Localization</i> | Implementasi <i>Indoor Localization</i> Menggunakan Metode <i>Fingerprinting</i> Berjenis <i>Monitor Based Localization</i> (MBL) Dengan Teknologi <i>Bluetooth Low Energy</i> (BLE) |
| 2. | A Survey of Indoor Localization Systems and Technologies | (Zafari, Gkelias dan Leung, 2019) | <i>Survey Paper</i> mengenai berbagai teknologi, metode, dan algoritma yang digunakan untuk Sistem <i>Indoor Positioning/Localization</i> | Implementasi <i>Indoor Localization</i> Menggunakan Metode <i>Fingerprinting</i> Berjenis <i>Monitor Based Localization</i> (MBL) Dengan Teknologi <i>Bluetooth Low Energy</i> (BLE) |
| 3. | Indoor mobile localization based on Wi-Fi fingerprint's important access point | (Jiang <i>et al.</i> , 2015) | Implementasi <i>Indoor localization</i> menggunakan metode <i>Fingerprinting</i> dengan teknologi WiFi | Implementasi Indoor Localization menggunakan metode <i>Fingerprinting</i> dengan teknologi BLE |

Tabel 2.1 Kajian Pustaka (Lanjutan)

| No. | Judul | Sitasi | Penelitian | Penelitian Penulis |
|-----|---|----------------------------|---|--|
| 4. | Location Fingerprinting With Bluetooth Low Energy Beacons | (Faragher dan Harle, 2015) | Implementasi <i>Indoor Localization</i> dengan metode <i>fingerprinting</i> menggunakan BLE. Mekanisme yang digunakan adalah <i>Device Based Localization</i> | Implementasi <i>Indoor Localization</i> dengan metode <i>fingerprinting</i> menggunakan BLE. Mekanisme yang digunakan adalah <i>Monitor Based Localization</i> |

Pada penelitian ini, penulis menggunakan *survey paper* yang ditulis oleh Ramon F. Brena, Juan Pablo García-Vázquez, Carlos E. Galván-Tejada, David Muñoz-Rodríguez, Cesar Vargas-Rosales, dan James Fangmeyer Jr (2017), "Evolution of Indoor Positioning Technologies: A Survey" sebagai acuan dasar dalam hal penentuan teknologi, metode, dan algoritma pada sistem *indoor localization* yang akan penulis lakukan. Pada jurnal ini dijelaskan mengenai beberapa aspek pada *indoor localization* seperti fitur, tantangan, dan peluang yang muncul. Pada jurnal ini juga disajikan secara komprehensif mengenai literatur tentang *indoor localization/positioning* yang bertujuan untuk memberikan perspektif tentang evolusi dalam teknologi penentuan lokasi/posisi dalam ruangan.

Jurnal ini menyebutkan beberapa metode yang digunakan dalam penentuan lokasi dalam ruangan seperti *Multilateration* dimana metode ini menggunakan pengukuran dari kombinasi jarak dari objek yang akan ditentukan posisinya berdasarkan beberapa perangkat *reference point*. Beberapa teknik yang dapat digunakan untuk memperoleh jarak antara objek dan *reference point* adalah *Time of Arrival* (ToA) yang menghitung jarak *transmitter* dan *receiver* menggunakan waktu tempuh sinyal, *Time of Difference Arrival* (TDoA) yang menggunakan perbedaan waktu tempuh antar *receiver*, *Angle of Arrival* (AoA) yang memanfaatkan sudut dari datangnya sinyal, dan *Received Signal Strength* (RSS) yang menggunakan kuat sinyal antara *transmitter* dan *receiver* untuk menentukan jarak.

Metode lain yang dapat digunakan adalah *fingerprinting*. Metode ini menentukan lokasi dari objek berdasarkan *data training* yang telah dikumpulkan dan dilatih sebelumnya. Algoritma yang sering dipakai dalam metode *fingerprinting* adalah algoritma K-Nearest Neighbour (KNN). Biasanya, algoritma KNN dipadukan dengan algoritma lain seperti *The Kalman Filter* untuk lebih memperhalus hasil dalam penentuan lokasi.

Terlepas dari metode yang digunakan, biasanya teknologi penentuan lokasi dalam ruangan menghadapi beberapa tantangan seperti *Signal Propagation*.

Beberapa sistem *indoor localization* menggunakan teknik *Signal Propagation*, dimana setiap sinyal yang ditransmisikan melakukan propagasi, maka semakin lemah sinyal yang ditransmisikan karena semakin menjauh dari transmitter. Hal ini juga diperparah dengan adanya gangguan-gangguan yang semakin melemahkan sinyal tersebut sehingga semakin sulit dalam melakukan pengukuran untuk memperoleh lokasi dari objek.

Tantangan lain adalah dalam hal *Multipath Environment*. Tantangan yang muncul adalah apabila sinyal yang ditransmisikan bercampur dengan pantulannya sendiri sehingga menyebabkan teracak dan sulit untuk dikenali. Tantangan adalah dalam hal *Line of Sight*, yakni masalah yang muncul adalah apabila teknologi atau sistem penentuan lokasi mensyaratkan tidak ada halangan antara *transmitter* dan *receiver* atau yang biasa disebut *Line of Sight* (LOS). Maka, jika terdapat penghalang diantara *transmitter* dan *receiver* proses penentuan lokasi menjadi lebih susah untuk dilakukan. Tantangan selanjutnya adalah dalam hal *Synchronization*. Pada beberapa teknik yang digunakan dalam sistem *indoor localization/positioning* diperlukan akurasi yang tinggi dalam hal waktu, contohnya pada ToA dan TDoA.

Teknologi yang digunakan pada sistem *indoor localization/positioning* adalah *Radio Frequency Signals* (RF). Contoh teknologi yang menggunakan RF adalah Wi-Fi dan Bluetooth. Kemudian teknologi lain adalah 'cahaya', yang dikategorikan menjadi 2 bagian, yakni '*visible light*' dan '*infrared light*'. Selanjutnya adalah teknologi suara, baik yang '*audible*' maupun yang ultrasonic. Teknologi lain adalah Medan Magnet, baik medan magnet alami bumi maupun medan magnet buatan.

Pada jurnal ini juga disertakan perbandingan dari penggunaan berbagai macam metode dan teknologi. Pada jurnal ini dijelaskan secara detail hasil dari perbandingan tersebut, seperti teknologi apa yang digunakan, dan metode atau teknik yang digunakan, layout dari ruangan atau lokasi yang dipakai, cakupan sistem, biaya yang digunakan, hingga akurasi yang diperoleh sistem dalam penentuan lokasi/posisi.

Penulis juga menggunakan jurnal yang ditulis oleh Faheem Zafari, Athanasios Gkelias, dan Kin K. Lung (2019) berjudul "A Survey of Indoor Localization Systems and Technologies" sebagai dasar acuan dalam mereview teknik dan teknologi yang digunakan dalam *indoor localization*. Pada jurnal ini dijelaskan mengenai teknik dan teknologi apa saja yang dapat digunakan dalam *indoor localization*, dan juga dijelaskan kaitan antara *indoor localization* dengan *Internet of Things*.

Pada jurnal ini dijelaskan bahwa ada beberapa teknik yang dapat digunakan dalam *indoor localization*. Teknik tersebut antara lain, a) *Received Signal Strength Indicator* (RSSI) atau dalam bahasa lain adalah kuat sinyal, b) *Channel State Information* (CSI), c) *Angle of Arrival* (AoA), d) *Time of Flight* (ToF), e) *Time Difference of Arrival*, f) *Return Time of Arrival* (RToF), g) *Phase of Arrival* (PoA) dan h) *Fingerprinting/Scene Analysis*, pada teknik *fingerprinting* biasanya digunakan algoritma untuk menentukan hasil klasifikasi seperti, *Probabilistic Method*,

Artificial Neural Networks (ANN), k-Nearest Neighbor (kNN), dan Support Vector Machine (SVM).

Pada jurnal ini juga dijelaskan mengenai teknologi yang digunakan dalam *indoor localization*. Teknologi yang dapat digunakan adalah WiFi, Bluetooth, Zigbee, *Radio Frequency Identification Device (RFID)*, *Ultra Wideband (UWB)*, *Visible Light Communication (VLC)*, Ultrasound, dan *Acoustic Signal*. Selanjutnya, dijelaskan juga mengenai perbandingan dari penggunaan teknologi *indoor localization* tersebut. Perbandingan dilakukan dengan melihat dari berbagai macam aspek, seperti jangkauan maksimal, *Maximum Throughput*, Tenaga yang dikonsumsi, keuntungan dan kekurangan.

Jurnal ini juga menjelaskan mengenai kaitan antara sistem *indoor localization* dan *Internet of Things (IoT)*. IoT terdiri dari 3 komponen dasar, yakni a) *Sensing/Data Collection*, dimana pada komponen ini bertindak sebagai pengumpul data seperti data suhu, kelembapan, dsb. b) *Data Communication*, dimana data yang telah dikumpulkan pada sensor harus ditransmisikan pada entitas pusat seperti server, *cluster head*, dsb, yang mana hal ini dapat dilakukan menggunakan teknologi kabel atau berbasis *wireless*. c) *Data Processing*, setelah data dari sensor dikirimkan menuju server, data perlu diproses untuk selanjutnya ditampilkan pada *user*.

Tantangan sistem *indoor localization* pada lingkungan IoT adalah : a) Privasi dan Keamanan, yakni bagaimana memberikan keamanan pada lokasi serta privasi dari user dari pihak atau entitas yang tidak berwenang. b) Heterogenitas dan diversitas, yakni bagaimana memadukan berbagai macam perangkat yang berbeda jenis dalam hal semisal teknologi komunikasi yang digunakan (contoh : SigFox, LoRa, Wifi, BLE, dsb) agar dapat terhubung dan bisa bekerja secara bersamaan. c) Manajemen Jaringan dan *Scalability*, yakni bagaimana manajemen jaringan saat ada sinyal/paket tambahan yang digunakan untuk *localization* dapat menghasilkan gangguan dalam komunikasi serta mengurangi efisiensi pada jaringan IoT dan jaringan konvensional yang beroperasi dalam frekuensi yang sama. d) Yang terakhir adalah tantangan dalam perangkat IoT yang digunakan, dimana biasanya perangkat-perangkat IoT tersebut memiliki harga yang murah sehingga hanya dapat melakukan tugas-tugas yang terbatas seperti mengumpulkan data (*sensing*) atau hanya sebagai *data transmission* saja, sehingga hal ini perlu diperhatikan dalam pengimplementasiannya.

Pada jurnal ini, dijelaskan juga evaluasi untuk sistem *indoor localization*. Adapun aspek-aspek yang dievaluasi pada sistem *indoor localization* adalah 1) *Availability*, yakni kesiapan dan ketersediaan sistem pada perangkat user. Hal ini adalah salah satu aspek penting pada sistem *indoor localization*. 2) *Cost/Biaya*, dimana biaya dari sistem *indoor localization* haruslah terjangkau oleh user, karena sebagian user bisa jadi adalah pelaku bisnis kecil yang tidak mampu membeli apabila harga yang terlalu mahal. 3) Efisiensi Energi, hal ini juga hal yang penting dalam sistem *indoor localization* karena tujuan utama dari *localization* adalah untuk mengembangkan pelayanan kepada user, sehingga sistem yang menghabiskan energi yang banyak besar kemungkinan tidak akan digunakan. 4)

Jangkauan Penerimaan, yakni jangkauan dari sistem harus mencakup bangunan semisal perkantoran, rumah sakit, mall, dsb. 5) Akurasi, hal ini juga salah satu aspek yang penting, karena sistem *indoor localization* memiliki tujuan untuk menentukan lokasi dari objek sehingga objek atau user dapat mengetahui lokasi dimana dia berada, maka jika terlalu banyak kesalahan dalam menentukan posisi atau lokasi, tujuan dari menentukan lokasi akan sia-sia dan membuat sistem menjadi tidak berguna. 6) *Latency/Delay*, hal ini berkaitan dengan seberapa cepat sistem dalam menentukan lokasi dari objek atau user. Sistem *Indoor localization* idealnya dapat melakukan penentuan lokasi secara *real-time* dimana objek dapat langsung diketahui lokasinya tanpa harus menunggu lama. 7) *Scalability*, sistem *indoor localization* idealnya dapat melakukan penentuan lokasi secara bersamaan kepada sejumlah besar user dan dalam cakupan yang luas.

Penelitian selanjutnya adalah tulisan dari Pei Jiang, Yunzhou Zhang, Wenyan Fu, Huiyu Liu, dan Xiaolin Su (2015) dengan judul “Indoor Mobile Localization Based on Wi-Fi Fingerprint’s Important Access Point”. Penelitian ini menerapkan metode *fingerprinting* menggunakan teknologi WiFi. Pembentukan *fingerprint* pada penelitian ini berdasarkan pola sinyal yang terbentuk dari *important access point* (IAP), yakni beberapa WiFi *access point* yang memiliki kuat sinyal yang tinggi pada lokasi implementasi yang telah ditentukan. Kemudian pada tahap *localization*, *fingerprint* yang dipilih dengan IAP yang sama pada *fingerprint* pada basis data, selanjutnya jarak dan pengulangan *access point* dari *fingerprint* digunakan untuk menghitung tingkat kesamaan. Lokasi dari *fingerprint* yang memiliki kesamaan dengan perkiraan *fingerprint* dapat dianggap sebagai perkiraan lokasi.

Kontribusi utama dari penelitian ini adalah : 1) penelitian ini menggunakan basis data *offline* dengan informasi yang sederhana. Hanya informasi dari AP yang digunakan dalam *fingerprint* yang disimpan. 2) Algoritma yang digunakan dalam mencocokkan *fingerprint* diajukan berdasarkan AP yang penting agar dapat secara efektif mempersempit ruang lingkup pengambilan basis data, sehingga dapat mengurangi beban komputasi dan menghilangkan interferensi untuk meningkatkan akurasi dalam penentuan lokasi. 3) Pada penelitian ini mengusulkan algoritma untuk mencocokkan *fingerprint* dengan mempertimbangkan jarak dan tingkat kecocokkan dari beberapa AP. 4) Tidak perlu untuk mengkonfigurasi ulang atau membuat ulang lokasi implementasi.

Penelitian ini pada dasarnya dibagi menjadi 2 tahap, yakni tahap pembentukan basis data *fingerprint* dan tahap *online positioning*. Pada tahap pembentukan basis data, sinyal *wireless* pada tiap ruangan akan direkam. Untuk mendeskripsikan karakteristik dari kuat sinyal WiFi dengan cara yang lebih baik, perlu untuk mengatur titik pengambilan *sample* pada tiap lokasi yang berbeda dalam ruangan. Tiap titik haruslah diambil *sample* beberapa kali. Kemudian, *data sample* yang telah diperoleh akan difilter menggunakan algoritma *filtering*. Pada *fingerprint* tiap posisi diurutkan berdasarkan kekuatan sinyalnya, sehingga dapat diperoleh informasi penting dari *fingerprint* tiap *access point*.

Pada tahap *online positioning*, digunakan *handphone* untuk mendeteksi kekuatan sinyal di sekitar AP dan mengumpulkan *fingerprint* lokasi. Kemudian, dapat dicari *fingerprint* yang memiliki AP yang konsisten dengan *fingerprint* yang berada pada basis data. Kemudian, operasi untuk mencocokkan antara hasil pencarian dengan *fingerprint* target dapat dilakukan untuk memperoleh estimasi lokasi dari *fingerprint* target.

Hasil dari penelitian ini cukup akurat. Pada penelitian ini diperoleh akurasi dari penentuan lokasi menggunakan algoritma IAP-FP mencapai 85,9%. Penelitian lain yang menggunakan algoritma Nearest Neighbour hanya mencapai 68,6%, sedangkan menggunakan algoritma *hybrid networks* mencapai 72,3%. Kemudian jika menggunakan algoritma milik Marques, Meneses dan Moreira, (2012) hanya mencapai 74,1%. Alasan utama sehingga akurasi dari algoritma IAP-FP dapat lebih tinggi dari algoritma lain adalah karena algoritma ini secara efektif mengurangi cakupan dari pencocokan *fingerprint*. Proses ini tidak hanya mengurangi beban komputasi, tapi juga membuang beberapa *fingerprint* yang memiliki tingkat kecocokan yang tinggi namun tidak dalam satu ruang kelas yang sama dengan *fingerprint* yang akan ditentukan lokasinya.

Kesimpulan yang diperoleh dari penelitian ini adalah untuk lingkungan yang terdapat sinyal WiFi, dapat menggunakan algoritma lokalisasi *fingerprint* berbasis Wi-Fi berdasarkan *Important Access Point* (IAP). Dengan menggunakan algoritma ini, lokasi dari ponsel dapat diperkirakan secara akurat. Dengan jarak *fingerprint* dan pencocokan Wi-Fi AP, algoritme ini dapat secara efektif mengurangi rentang pencocokan *fingerprint*. Akhir kata, hal ini dapat memberikan estimasi yang akurat tanpa perlu mengetahui struktur bangunan dan informasi distribusi AP. Oleh karena itu, hal ini memiliki nilai praktis yang tinggi untuk lokalisasi pribadi di dalam ruangan.

Penelitian selanjutnya yang penulis gunakan adalah karya Ramsey Faragher dan Robert Harle (2015) dengan judul : "Location Fingerprinting With Bluetooth Low Energy Beacons". Pada penelitian ini, sistem *indoor localization* menggunakan teknologi BLE dengan metode *fingerprinting*. Motivasi dari penelitian ini menggunakan BLE karena ada beberapa kelemahan yang dimiliki oleh teknologi WiFi.

Kelemahan dari teknologi WiFi adalah : 1) Meningkatnya durasi pemindaian WiFi secara pasif, dimana perangkat diharuskan menunggu *broadcast SSID* dari tiap-tiap *access point* sehingga menyebabkan keterbatasan dalam *update rate*. 2) Laporan dari *broadcast WiFi access point* dilaporkan dalam sebuah laporan tunggal, lamanya durasi dari pemindaian tidak hanya menyebabkan keterbatasan dalam *update rate*, namun juga menyebabkan *fingerprints* menjadi tidak akurat saat user berpindah tempat. 3) Pemindaian WiFi secara aktif secara terus-menerus oleh perangkat dapat menyebabkan penurunan kecepatan lalu lintas jaringan serta dapat mengurangi privasi. 4) Tidak semua platform *mobile* dapat mengakses data pemindaian WiFi seperti Apple's iOS misalnya. 5) Spesifikasi sinyal WiFi tidak menggunakan satuan unit tertentu untuk kekuatan sinyal, yang mana hal ini menjadi sulit apabila *positioning* menggunakan konsep lintas perangkat.

Sementara itu, BLE tidak menemui kendala seperti di atas : paket BLE diterima hampir seketika dan satuan yang dilaporkan menggunakan satuan unit dBm. Kemudian BLE juga memiliki beberapa keuntungan, diantaranya adalah : 1) Penggunaan daya oleh BLE lebih sedikit daripada menggunakan WiFi. 2) BLE lebih mudah untuk digunakan (terutama jika menggunakan daya baterai) ketimbang WiFi dan tidak dibatasi cakupan komunikasi yang seragam.

Penelitian ini mengevaluasi sistem *fingerprint* BLE saja yang mengasumsikan suar BLE statis didistribusikan ke seluruh lingkungan dan membandingkannya dengan sistem WiFi. Penelitian menggunakan tingkat *advertising rate* yang tinggi (50 Hz) dan kekuatan transmisi yang tinggi. Selanjutnya dengan menggunakan *post-processing* dengan paket yang telah di-drop dan data mentah yang dilemahkan secara manual, penelitian ini dapat menyelidiki laju dan daya transmisi yang dibutuhkan untuk penentuan posisi yang baik.

Kontribusi dalam penelitian akan dilakukan adalah 1) Percobaan pertama dalam BLE *positioning* akan menggunakan *fingerprinting*. 2) Studi yang terperinci dalam paramter kunci untuk *indoor positioning* yang akurat menggunakan sinyal radio BLE. 3) Menemukan bahwa penggunaan tiga saluran pita frekuensi yang lebar namun sempit pada saat *advertising* dapat menyebabkan variasi RSS yang parah jika nomor saluran BLE tidak dilaporkan ke sistem (ini dapat dilihat sebagai kelemahan dalam spesifikasi BLE saat ini). 4) Pengujian skema mitigasi untuk menghindari masalah tersebut. 5) Validasi experimental yang terperinci dan 6) Serangkaian rekomendasi untuk *developer* sistem *indoor positioning/localization* berbasis BLE.

Penelitian ini tidak menggunakan *fingerprint* BLE yang memudar (*faded*). Fading yang tajam (cepat) akan secara alami dihilangkan pada tahap pembuatan peta (tahap *offline*) melalui penerapan regresi. Hal ini membuat peta hanya menunjukkan data yang secara dominan mewakili *free space loss* dan hilangnya bayangan (*shadow*) dari objek. Namun, masalah tetap ada selama fase *online* (pencocokan peta) di mana satu pembacaan mungkin terjadi *sharp fade*.

Penelitian ini membentuk konstruksi *fingerprint* BLE dengan mempertimbangkan bahwa *advertisement* BLE yang cepat (hampir seketika) sehingga harus membentuk *fingerprint* RSS berdasarkan pada sisi waktu pada pengukuran BLE. Pilihan untuk lebar waktu yang digunakan, tergantung pada banyak faktor, termasuk :

- 1) Tingkat perpindahan. Jika *receiver* bergerak ketika tahap pengumpulan *fingerprint*, maka *fingerprint* akan dibentuk dari pengukuran pada posisi spasial yang berbeda. Tingkat perpindahan menentukan batas atas maksimum panjang jendela waktu yang dapat diterima. Apabila jendela waktu terlalu panjang, maka batas spasial *fingerprint* yang besar akan membatasi keberhasilan dalam pencocokan.
- 2) Kehadiran yang cepat pudar. Jendela waktu yang sangat pendek membatasi jangkauan spasial *fingerprint* tetapi meningkatkan risiko

pengukuran *fingerprint* dilakukan dalam *fast fade null* pada setidaknya satu saluran.

- 3) Tingkat Suar. Penelitian ini menetapkan *receive rate* sebagai tingkat di mana *advertisement* BLE dilaporkan dalam aplikasi. Nilai ini adalah batas yang lebih rendah pada tingkat suar karena paket *advertisement* tertentu dapat di-drop atau dilewatkan. Jelas, *receive rate* yang lebih rendah memerlukan jendela *fingerprint* yang lebih lama untuk membangun dimensi *fingerprint* yang sama.

Penelitian ini menggunakan testbed dengan tingkat akurasi yang tinggi, maka konstruksi *map* dibentuk menggunakan data yang dikumpulkan saat eksperimen. Kemudian digunakan proses regresi Gaussian untuk membentuk kekuatan sinyal *map* pada tiap sumber. Penelitian ini menggunakan basis data yang terpisah untuk kumpulan parameter konstruksi *fingerprint*. Kemudian untuk memposisikan perangkat, penelitian ini menggunakan Bayesian *estimator*. Seluruh area yang digunakan akan dibagi menjadi *grid cells* dengan sisi sepanjang 1 m.

Eksperimen dilakukan menggunakan 4 eksperimen berjalan. Pada masing-masing eksperimen berjalan memakan waktu kurang lebih 5 – 18 menit dan mencakup semua area test bed. Hasil dari penelitian ini menunjukkan bahwa peningkatan yang signifikan dalam *positioning* dibanding WiFi bahkan menggunakan penyebaran suar BLE yang jarang-jarang. Penelitian ini dapat mengimplementasikan *indoor positioning* dengan akurasi kurang dari 2,6 m menggunakan distribusi suar yang rapat, dan akurasi < 4,8 m dengan distribusi suar yang kurang rapat (jarang-jarang) pada lingkungan dimana WiFi hanya dapat memperoleh akurasi < 8,5 m.

2.2 Sistem Pelacakan Dalam Gedung (Indoor Localization)

Sistem pelacakan lokasi dalam ruangan (*Indoor Localization*) adalah sebuah sistem atau layanan untuk penentuan lokasi seseorang atau benda menggunakan sebuah koordinat relatif pada sebuah ruangan atau gedung.(Chan dan Sohn, 2012). Secara umum, komponen dari sistem *indoor localization* terdiri dari beberapa *anchor node* yang dapat berfungsi sebagai pemancar sinyal atau dapat juga sebagai penangkap sinyal dari perangkat yang dilacak. Perangkat yang dilacak berfungsi sebagai perangkat yang akan dicari lokasi/posisi-nya sekaligus juga sebagai perangkat yang menginterpretasikan pola sinyal yang didapat menjadi nama lokasi atau koordinat. Pada beberapa kasus, terdapat komponen tambahan sebagai perangkat yang menginterpretasikan pola sinyal menjadi nama lokasi atau posisi.

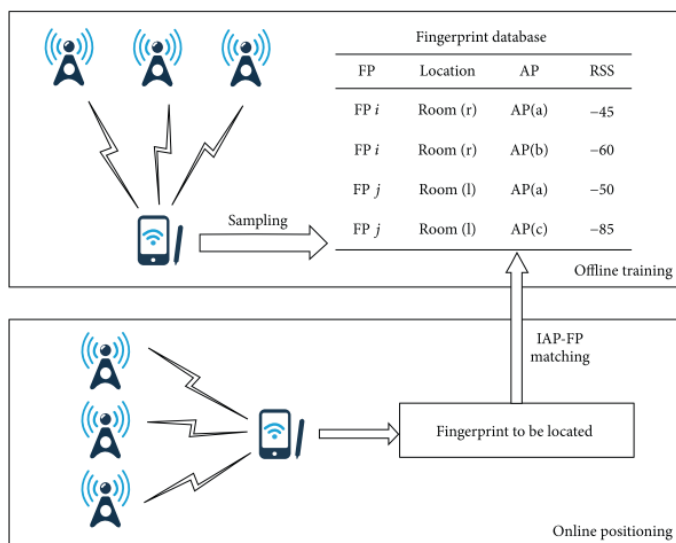
Beberapa jenis teknologi sinyal yang dapat digunakan untuk *indoor localization* adalah WiFi, Bluetooth, RFID, Ultra Wideband (UWB), ZigBee dan lain sebagainya. Jenis teknologi sinyal yang digunakan tergantung pada kebutuhan implementasi. Kemudian, metode yang digunakan untuk interpretasi sinyal menjadi nama lokasi ada beberapa macam seperti : *Angle of Arrival* (AOA), *Time of Flight* (ToF), *Fingerprinting*, *Time Difference of Arrival* (TDoA), *Return Time of Flight* (RTof),

Phase of Arrival (PoA). (Zafari, Gkelias dan Leung, 2017). Hasil dari interpretasi pola sinyal dapat berupa nama lokasi, koordinat, atau dapat juga berbentuk visual 2 dimensi bahkan 3 dimensi. (Li, Yang dan Zhou, 2008).

2.3 Fingerprinting

Fingerprinting adalah sebuah metode untuk menginterpretasikan sinyal yang didapat dari beberapa *reference points* menjadi sebuah nama lokasi. Metode ini bekerja dengan mencocokkan pengukuran sinyal yang diperoleh dari beberapa *reference points* dengan data sinyal yang terdapat pada database. (Jiang *et al.*, 2015).

Secara umum, *Fingerprinting* bekerja dalam 2 tahap, yakni tahap *offline* dan tahap *online*. Tahap *offline* adalah tahap pengumpulan data pola sinyal. Pada tahap ini karakteristik sinyal pada masing-masing ruangan diukur dan disimpan pada database bersamaan dengan nama ruangan yang diukur. Data pola sinyal dan nama ruangan yang disimpan disebut dengan *reference points* (RPs). Tahap kedua adalah tahap *online*, yakni tahap pelacakan atau tahap klasifikasi. Pada tahap ini, data sinyal yang diperoleh dari *reference points* dicocokkan dengan *Reference Points* (RPs) dengan algoritma tertentu. *Reference Points* yang memiliki kecocokkan dengan pengukuran pada tahap online adalah hasil dari pelacakan. (Zafari, Gkelias dan Leung, 2017).



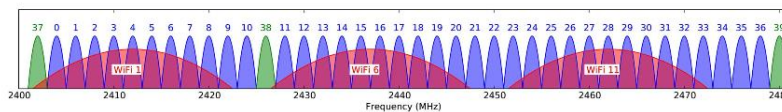
Gambar 2.1 Arsitektur metode *fingerprinting*
(Sumber : Jiang *et al.*, 2015)

2.4 Bluetooth Low Energy (BLE)

Bluetooth adalah teknologi komunikasi nirkabel yang menggunakan informasi yang tertanam secara digital pada sinyal frekuensi radio. Awalnya dimaksudkan untuk pertukaran data dalam jarak pendek, hal ini didefinisikan oleh standar IEEE 802.15.1. Tujuan utama dari teknologi ini adalah untuk memfasilitasi komunikasi antara perangkat bergerak dan perangkat diam atau dua perangkat seluler, untuk menghilangkan kabel dan konektor antar perangkat (misal, dalam penggunaan headphone nirkabel), dan untuk memfasilitasi sinkronisasi data antara perangkat pribadi. (Chatschik, 2001)

Teknologi Bluetooth telah dipertimbangkan untuk sistem posisi dalam ruangan sebagai pesaing Wi-Fi, khususnya sejak ditemukan *Bluetooth Low Energy* (BLE), karena *availability*-nya (didukung oleh sebagian besar smartphone modern), biaya rendah, dan mengonsumsi daya yang sangat rendah, yang memungkinkan alat tetap bekerja menggunakan baterai selama beberapa bulan atau bahkan bertahun-tahun. (Faragher dan Harle, 2015).

BLE menggunakan 40 *channels*, masing-masing selebar 2 MHz, mencakup pita radio 2,4 GHz juga digunakan oleh WiFi (Faragher dan Harle, 2015). Gambar menjelaskan tentang channel yang digunakan BLE.



Gambar 2.2 BLE channels

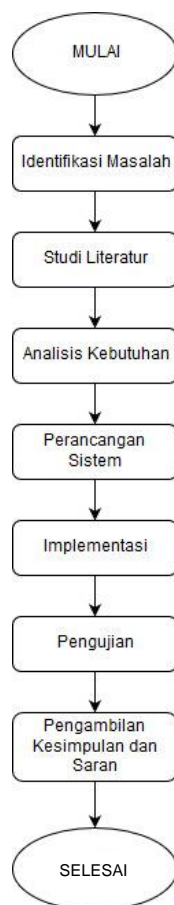
(Sumber : Faragher dan Harle, 2015)

Protokol BLE menggunakan pesan berdurasi singkat untuk menghemat penggunaan baterai. (Heydon dan Hunn, 2012). Pesan ini bisa berupa data atau *advertisement*. Pesan yang lain adalah pesan *broadcast* yang digunakan untuk menemukan (*discovery*) perangkat BLE lain. Kekuatan sinyal yang berasal dari pesan *advertisement* pada BLE dapat digunakan pada *fingerprinting* untuk membentuk pola sinyal. (Faragher dan Harle, 2015)

BAB 3 METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan mengenai langkah-langkah apa saja yang akan dilakukan untuk menyelesaikan masalah pada penelitian ini. Langkah yang akan disusun untuk menyelesaikan masalah pada penelitian ini meliputi : Identifikasi masalah, studi literatur, analisis kebutuhan, perancangan sistem, implementasi, pengujian dan pengambilan kesimpulan dan saran.

Dibawah ini adalah susunan metodologi yang akan dilakukan dalam penelitian ini.



Gambar 3.1 Metodologi Penelitian

3.1 Identifikasi Masalah

Identifikasi masalah merupakan tahapan untuk mengetahui detail permasalahan sesuai dengan topik penelitian yang diambil. Pada kegiatan ini, penulis melakukan identifikasi permasalahan dengan melakukan studi literatur. Tujuan dari identifikasi masalah adalah mengetahui detail permasalahan yang ada pada topik penelitian yang diambil sehingga menghasilkan kebutuhan dan solusi untuk mengatasi permasalahan tersebut.

Pada penelitian ini, penulis mengidentifikasi permasalahan pada penelitian dengan tema *indoor localization*. Penulis mencari permasalahan dengan mempertimbangkan penelitian yang sudah ada dengan kondisi saat ini. Tujuannya adalah untuk menemukan permasalahan pada *indoor localization* yang dapat diselesaikan secara komputasi serta sebagai acuan penulis untuk melakukan penelitian.

3.2 Studi Literatur

Studi literatur merupakan kegiatan yang dilakukan untuk menentukan objek penelitian yang sesuai dengan topik yang diambil. Dalam pembahasan studi literatur penelitian ini, penulis melakukan studi literatur agar dapat mempelajari teori-teori pendukung tentang *indoor localization*. Tujuannya adalah memperoleh dasar acuan dalam melakukan penelitian ini serta mendapat solusi atas permasalahan yang berkaitan dengan penelitian tentang *indoor localization*.

Pada studi literatur ini, penulis mengkaji penelitian dengan tema *indoor localization*. Pertama, penulis mengkaji metode dan teknologi yang saat ini digunakan dalam *indoor localization*. Kajian ini berfungsi untuk mengetahui metode dan teknologi apa saja yang digunakan pada *indoor localization* serta mengetahui kelebihan dan kelemahan dari masing-masing metode dan teknologi yang digunakan. Hal ini dilakukan sebagai acuan dalam memutuskan metode dan teknologi apa yang akan digunakan agar tujuan penelitian dapat tercapai.

3.3 Analisis Kebutuhan

Analisis kebutuhan memiliki tujuan untuk memahami kebutuhan dan fungsi dari sistem yang akan diimplementasikan. Analisis kebutuhan didasarkan pada hasil kajian pada studi literatur. Hasil dari analisis kebutuhan ini merupakan landasan untuk melakukan perancangan sistem.

Sistem pelacakan dalam gedung berbasis BLE menggunakan metode *fingerprinting* dibagi menjadi 2 tahap. Tahap pertama adalah tahap *offline*, yakni melakukan pengumpulan data pola sinyal tiap ruangan sehingga membentuk sebuah *data training*. Tahap kedua adalah tahap *online*, yakni penentuan lokasi. Dibawah ini akan dijelaskan mengenai kebutuhan dan perancangan dari sistem yang akan diimplementasikan.

Commented [AB2]: apakah tidak ada kebutuhan fungsional mas?

Pada implementasi *indoor localization* ini dibutuhkan 3 komponen. Komponen pertama adalah perangkat *passive tag* yang berfungsi memancarkan sinyal bluetooth secara terus menerus. Komponen ini sekaligus menjadi komponen yang akan ditentukan lokasinya. Komponen kedua adalah *reference points* yang berfungsi sebagai penangkap sinyal bluetooth yang dipancarkan *passive tag* dan mengirimkan data RSSI-nya menuju server. *Reference points* juga mengirimkan MAC address mereka bersamaan dengan data RSSI dari *passive tag*. Agar *reference points* dapat mengirimkan dua data yang berbeda tipe secara bersamaan, maka digunakan format data JSON dalam pengiriman data menuju server. Komponen ketiga adalah server yang berfungsi sebagai penyimpan data pola sinyal yang diterima dari beberapa *reference points* (*data training*) dan juga sebagai perangkat yang memproses pola sinyal yang diperoleh menjadi nama lokasi.

Implementasi sistem *indoor localization* membutuhkan lokasi di dalam gedung. Di dalam gedung tersebut akan dipilih beberapa ruangan untuk digunakan sebagai implementasi sistem. Jumlah ruangan yang digunakan harus lebih dari 1, karena penelitian ini menghasilkan output berupa nama lokasi bukan koordinat. Pada penelitian ini, penulis menggunakan 3 ruangan untuk implementasi dikarenakan keterbatasan sumber daya perangkat yang digunakan.

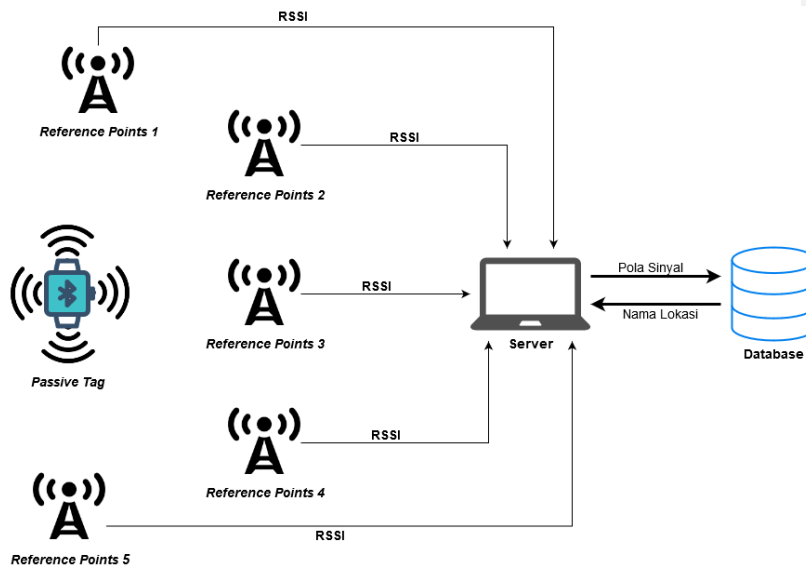
Pada Tabel 3.1 dibawah ini adalah rangkuman dari komponen yang dibutuhkan dalam penelitian ini.

Tabel 3.1 Kebutuhan Komponen Sistem

| Komponen | Fungsi | Jumlah |
|-------------------------|---|--------|
| <i>Passive Tag</i> | Perangkat yang akan ditentukan lokasinya. Perangkat ini memancarkan sinyal bluetooth secara terus-menerus | 1 buah |
| <i>Reference points</i> | Perangkat yang akan menangkap sinyal bluetooth yang dipancarkan <i>passive tag</i> dan mengirimkan data RSSI <i>passive tag</i> menuju server bersamaan dengan MAC address dari <i>reference points</i> tersebut menggunakan tipe data JSON | 5 buah |
| Server | Pada tahap <i>offline</i> metode <i>fingerprinting</i> , perangkat ini berfungsi menerima data JSON yang berisi data RSSI dari <i>passive tag</i> dan MAC address dari <i>reference points</i> yang mengirimnya. Kemudian data RSSI akan disimpan menjadi <i>data training</i> . Pada tahap <i>online</i> , perangkat ini berfungsi mengubah pola sinyal yang dikirim beberapa <i>reference points</i> menjadi nama lokasi. | 1 buah |

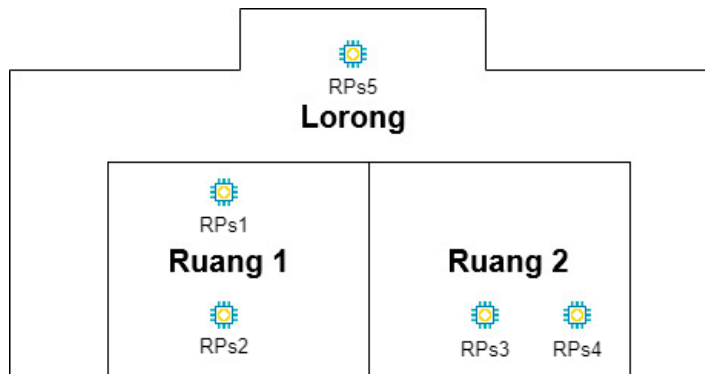
3.4 Perancangan Sistem

Dalam tahap perancangan, peneliti melakukan perancangan yang meliputi arsitektur sistem dan alur sistem yang akan diimplementasikan. Arsitektur sistem dirancang berdasarkan pada komponen yang akan digunakan dan alur kerja dari sistem yang akan diimplementasikan. Pada Gambar 3.2 di bawah ini menjelaskan mengenai arsitektur sistem yang akan diimplementasikan.



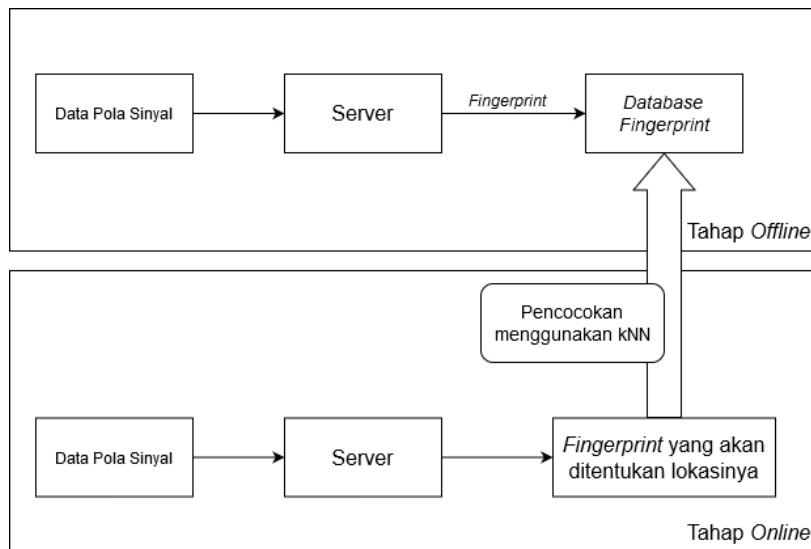
Gambar 3.2 Arsitektur Sistem Indoor Localization

Pada Gambar 3.2 di atas menjelaskan bahwa pada sistem *indoor localization* ini, peneliti menggunakan 3 komponen, yakni *passive tag*, *Reference Points*, dan *Server*. *Passive tag* berfungsi untuk memancarkan sinyal bluetooth secara terus menerus, kemudian *Reference Points* akan menangkap sinyal bluetooth tersebut dan mengirimkannya menuju server. Kemudian, server mengklasifikasikan pola sinyal bluetooth yang dikirimkan oleh beberapa *Reference Points* menjadi nama lokasi. Pada Gambar 3.3 di bawah ini, merupakan denah implementasi sistem dan penempatan dari *reference points* pada masing-masing lokasi.



Gambar 3.3 Denah Implementasi dan Penempatan *Reference Points*

Lokasi untuk implementasi dibagi menjadi 3. Lokasi pertama dan kedua berupa ruangan, sedangkan lokasi ketiga adalah lorong. Lokasi implementasi berada pada Gedung F lantai 9, Fakultas Ilmu Komputer, Universitas Brawijaya. *Reference points* akan ditempatkan pada ketiga lokasi tersebut. Pada Gambar 3.3 diatas, 2 *reference points* diletakkan pada Ruang 1 dan Ruang 2, sedangkan pada Lorong ditempatkan 1 buah *reference points*.



Gambar 3.4 Arsitektur *fingerprinting*

Untuk alur sistem akan dibagi 2 tahap berdasarkan pada mekanisme kerja metode *fingerprinting*. Seperti yang disajikan pada Gambar 3.4, tahap pertama disebut sebagai tahap *offline* adalah tahapan untuk membuat *data training*. Tahap ini berfungsi untuk merekam pola sinyal dari sinyal bluetooth yang dipancarkan

Commented [MH3]: Gambar disederhanakan

Commented [AB4]: Bisakah tahapan ini direpresentasikan juga dalam bentuk diagram?

passive tag pada masing-masing ruang. Pola sinyal yang diterima disimpan pada server dan diberi label nama ruangan dimana pola sinyal tersebut diukur.

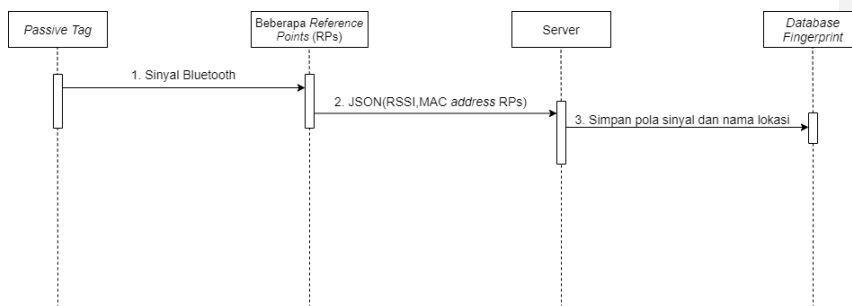
Tahap kedua adalah tahap *online*, dimana pada tahap ini *passive tag* akan ditentukan lokasinya. Penentuan lokasi dari *passive tag* tersebut menggunakan algoritma KNN berdasarkan pada *data training* yang telah dibuat. Hasil dari penentuan lokasi berupa nama ruangan dimana *passive tag* tersebut berada. Sebagai tambahan, penentuan lokasi dilakukan secara *real-time* dan hasil dari penentuan lokasi akan dicatat pada sebuah *file log*.

3.4.1 Perancangan Tahap Offline

Pada tahap ini dilakukan pengumpulan pola sinyal bluetooth dari masing-masing lokasi sehingga membentuk suatu *data training*. *Data training* ini nantinya akan digunakan sebagai referensi pada saat penentuan lokasi. Pada sub bab ini akan dijelaskan mengenai perancangan skema dan alur dari implementasi tahap *offline*. Kemudian akan dijelaskan juga mengenai perancangan format *data training* yang akan digunakan.

3.4.1.1 Perancangan Skema dan Alur Tahap Offline

Pada sub bab ini akan dijelaskan skema dan alur dari tahap *offline*. Tahap *offline* dimulai dengan *Passive tag* yang sudah menyala dan akan memancarkan sinyal bluetooth. Beberapa *reference points* akan menangkap sinyal bluetooth yang dipancarkan *passive tag*, kemudian akan diukur RSSI dari *passive tag* berdasarkan kekuatan sinyal *passive tag* terhadap *reference points*. Selanjutnya, *reference points* akan mengirimkan data RSSI *passive tag* dan juga MAC address dari *reference points* menuju server. Untuk bisa mengirim 2 tipe data yang berbeda secara bersamaan, digunakan tipe data JSON. Data JSON yang dikirim oleh beberapa *reference points* akan diterima oleh server. Oleh server data JSON dari beberapa *reference points* akan dikumpulkan. Data RSSI pada data JSON tadi akan diurutkan berdasarkan urutan *reference points*, kemudian disimpan dengan memberi label nama lokasi dan waktu ke database. Pada Gambar 3.4 dibawah ini merupakan skema implementasi tahap *offline*.

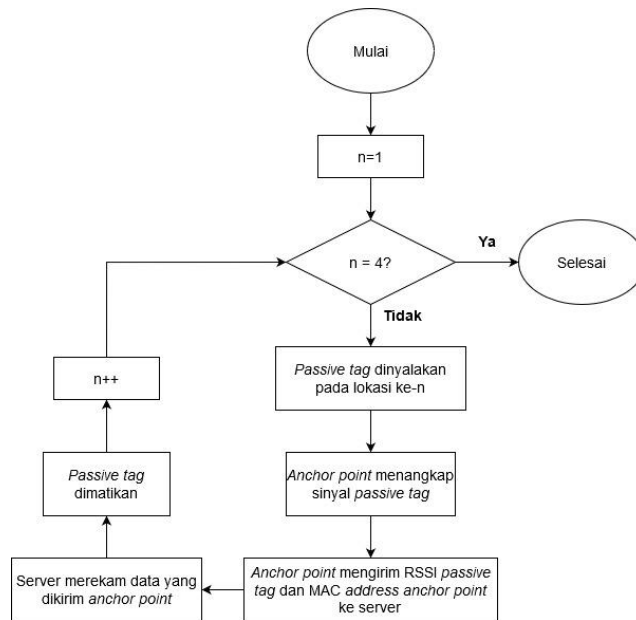


Gambar 3.5 Skema Implementasi Tahap Offline

Commented [AB5]: bisakah ditampilkan denah ruangnya juga?

- Untuk arsitektur keseluruhan sistem pakai komponen
- Denah untuk menunjukkan placement
- Untuk menggambarkan langkah sebaiknya pakai sequence diagram (offline dan online)

Kemudian untuk alur pengambilan pola sinyal pada masing-masing lokasi tergambar pada diagram alir dibawah ini.



Gambar 3.6 Alur Implementasi Tahap Offline

1. Pertama, pada lokasi ke-1 *passive tag* dinyalakan. Pada tahap ini *passive tag* akan dinyalakan selama ± 3 menit
2. Kemudian, *reference points* menangkap sinyal bluetooth yang dipancarkan oleh *passive tag*
3. Selanjutnya, *reference points* akan mengirimkan data RSSI dari *passive tag* dan MAC address *reference points* menuju server dalam format data JSON
4. Server menerima data JSON yang dikirim *reference points* dan menuliskan RSSI *passive tag* pada file csv sesuai urutan dari *reference points* (format urutan pada Tabel 4.2). Label nama lokasi dan waktu juga dituliskan bersamaan dengan penulisan RSSI.
5. Langkah terakhir, *passive tag* dimatikan sebelum pindah ke lokasi berikutnya. Hal ini untuk menghindari kesalahan penulisan RSSI pada saat berpindah lokasi
6. Langkah 1-5 diulang sampai ketiga ruangan selesai direkam pola sinyal-nya.

3.4.1.2 Perancangan Format Data Training

Pada bagian ini, akan dijelaskan format dari *data training* yang akan dibuat. *Data training* yang dibuat terdiri dari :

1. Label nama lokasi
2. Label waktu pengambilan pola sinyal dalam format HH:MM:SS
3. RSSI dari *passive tag* terhadap *reference points* (RPs)

Tabel 4.2 dibawah ini merupakan format dari *data training* yang akan dibuat.

Tabel 3.2 Format Data Training

| Nama Ruangan | Waktu | RSSI RPs 1 | RSSI RPs 2 | RSSI RPs 3 | RSSI RPs 4 | RSSI RPs 5 |
|--------------|-------|------------|------------|------------|------------|------------|
| Ruang 1 | | | | | | |
| | | | | | | |
| Ruang 2 | | | | | | |
| | | | | | | |
| Lorong | | | | | | |

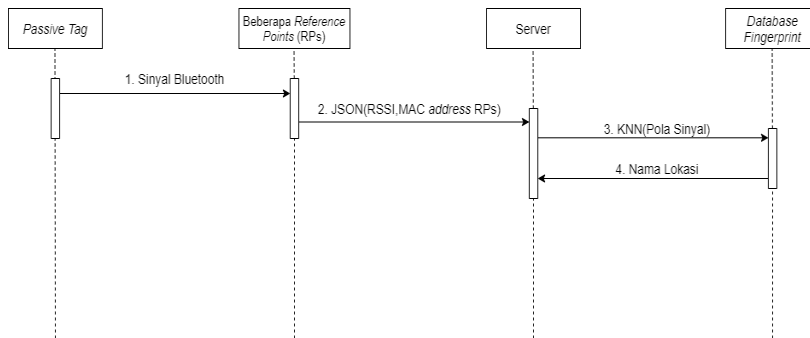
Data training ini nantinya akan disimpan dalam sebuah file dengan ekstensi (.csv) pada server. Format csv digunakan untuk memudahkan dalam menggunakan *data training* pada saat proses penentuan lokasi.

3.4.2 Perancangan Tahap *Online*

Pada tahap *online*, dilakukan mekanisme penentuan lokasi berdasarkan *data training* yang telah dibuat. Perancangan tahap *online* dibagi menjadi beberapa bagian. Bagian pertama adalah perancangan skema tahap *online*, yang akan menjelaskan tentang mekanisme penentuan lokasi dari perangkat *passive tag*. Bagian kedua adalah perancangan alur implementasi tahap *online*.

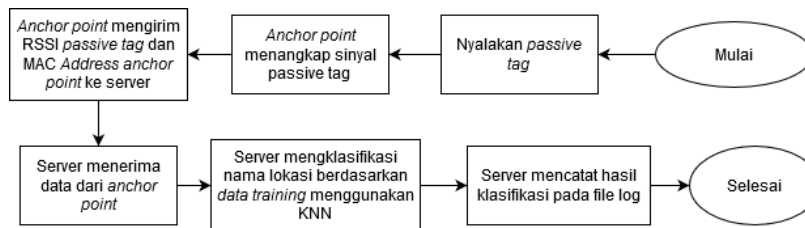
3.4.2.1 Perancangan Skema dan Alur Tahap *Online*

Pada bagian ini akan dijelaskan mengenai gambaran implementasi dari tahap *online*. Pertama, *passive tag* akan dinyalakan pada salah satu lokasi. Untuk memudahkan, lokasi pertama yang digunakan adalah Ruang 1. Kemudian *reference points* akan menangkap sinyal yang dipancarkan oleh *passive tag* dan mengirimkan data RSSI beserta MAC address dari *reference points* menuju server. Oleh server, data yang diterima dari beberapa *reference points* akan ditentukan lokasi nya berdasarkan *data training* menggunakan algoritma KNN. Nama lokasi yang telah ditentukan akan ditampilkan oleh server. Pada Gambar 3.6 dibawah ini, merupakan skema dari implementasi tahap *online*.



Gambar 3.7 Skema Implementasi Tahap *Online*

Kemudian untuk alur dari implementasi tahap online tergambar pada Gambar 3.6 dibawah ini.



Gambar 3.8 Alur Tahap *Online*

1. Pertama, *passive tag* dinyalakan pada salah satu lokasi
2. Saat *passive tag* dinyalakan, *reference points* akan secara otomatis menangkap sinyal bluetooth yang dipancarkan *passive tag*
3. Kemudian *reference points* akan mengirimkan RSSI dari *passive tag* dan MAC address dari *reference point* menuju server
4. Server menerima data yang dikirimkan *reference points* kemudian mengklasifikasikan nama lokasi dari pola sinyal yang dikirimkan oleh beberapa *reference points* dengan menggunakan algoritma KNN.
5. Hasil klasifikasi akan ditampilkan pada layar monitor dan akan dicatat oleh server pada sebuah file log.

3.4.2.2 Perancangan Format File Log Tahap *Online*

Pada sub bab ini akan dijelaskan format yang digunakan pada *file log* tahap *online*. *File* ini nantinya akan disimpan pada server dengan format csv. *File log* yang akan dibuat akan terdiri dari :

1. Nama lokasi hasil klasifikasi
2. Waktu penentuan lokasi (klasifikasi) dengan format HH:MM:SS

Pada Tabel dibawah ini merupakan format dari *file log* yang akan digunakan.

Tabel 3.3 Format File Log

| Hasil klasifikasi (lokasi) | Waktu |
|----------------------------|----------|
| Ruang 1 | 00:00:00 |
| Ruang 1 | 00:00:01 |
| | |
| Ruang 2 | 00:03:00 |

3.5 Implementasi

Tahapan implementasi merupakan kegiatan pengimplementasian sistem yang telah dirancang dan disusun sebelumnya. Implementasi dilakukan dengan menggunakan perangkat-perangkat yang telah disebutkan sebelumnya pada analisis kebutuhan. Implementasi akan dibagi menjadi 2 tahap, tahap *offline* dan *online*. Pada tahap implementasi, seluruh perangkat yang digunakan akan diberikan kode sumber untuk dieksekusi. Kemudian perangkat *reference points* akan diletakkan pada masing-masing ruang. Jumlah perangkat *reference points* yang digunakan sebanyak 5 perangkat karena keterbatasan jumlah perangkat. Dengan jumlah sekian, maka pada 2 ruangan terdapat masing-masing 2 perangkat *reference points* dan 1 ruangan diletakkan 1 perangkat *reference points*.

Pada sisi server diberikan kode sumber untuk mengeksekusi perintah sesuai dengan perancangan yang telah dibuat. Pada penelitian ini penulis menggunakan bahasa pemrograman python untuk implementasi kode sumber pada server. Setelah instalasi python, langkah selanjutnya adalah menginstall *library* yang dibutuhkan agar server dapat berjalan sesuai dengan fungsinya. Pada penelitian ini penulis menggunakan beberapa *library*. Salah satu *library* yang penting adalah *scikit learn*. *Library* ini berfungsi untuk mengolah berbagai data yang dibutuhkan pada penelitian ini seperti membuat *data training*, klasifikasi lokasi berdasarkan *data training*, dan menghitung hasil akurasi pengujian.

3.6 Pengujian

Pada tahap ini, sistem telah berhasil dirancang dan implementasikan lalu penulis akan melakukan pengujian pada sistem yang dibangun. Pengujian dilakukan untuk mengetahui bahwa sistem telah dapat berjalan sesuai dengan spesifikasi, kebutuhan, dan tujuannya. Parameter yang digunakan dalam pengujian ini adalah akurasi, untuk mengetahui seberapa akurat sistem yang telah diimplementasikan dan juga faktor yang mungkin dapat menyebabkan tingkat akurasi tidak maksimal.

Pengujian dengan menggunakan parameter akurasi dibagi menjadi 2 tahap, tahap pertama adalah pengujian untuk memperoleh akurasi kesalahan penentuan

Commented [AB6]: 1. Parameter pengujian (mengukur apa, pakai metric apa? -> akurasi, latency)
2. Skenario. Bagaimana skenario pengujiannya?

Taruh di sub bab implementasi pengujian di bab Implementasi
3. Bagaimana anda melakukan pengujian?

Hasil dan pembahasan
4. Hasil pengujian dan analisis

lokasi secara umum. Kedua, adalah pengujian untuk memperoleh akurasi kesalahan pada tiap sub-lokasi pada masing-masing lokasi implementasi. Implementasi pengujian dilakukan pada kondisi lingkungan yang sebenarnya untuk mengetahui kinerja dari sistem apabila diimplementasikan pada kondisi sebenarnya. Hasil dari pengujian ini kemudian akan dianalisis pada bab 6.

3.6.1 Perancangan Pengujian Akurasi Kesalahan Sistem

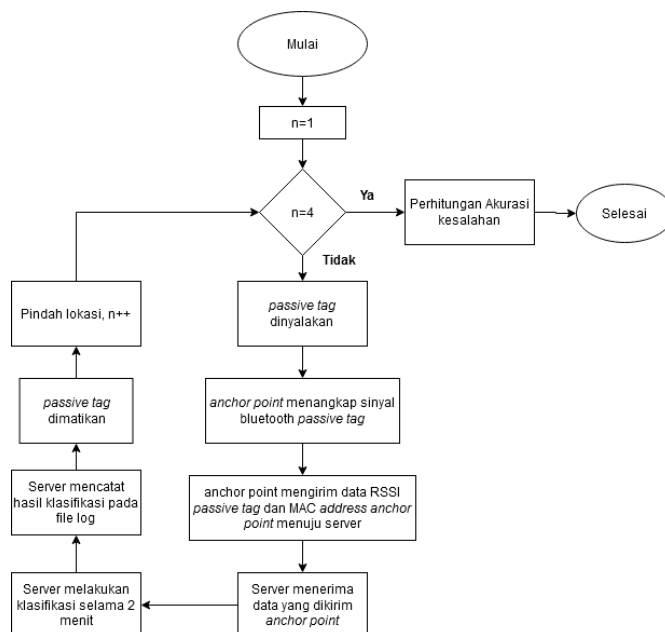
Pada sub bab ini akan dijelaskan mengenai pengujian untuk memperoleh akurasi kesalahan dalam penentuan lokasi secara umum. Pada pengujian ini digunakan persamaan dibawah ini untuk menghitung akurasi kesalahan pada penentuan lokasi

$$Akurasi\ Kesalahan = \frac{Jumlah\ Kesalahan\ Prediksi}{Total\ Prediksi} \times 100\ \% \quad (3.1)$$

Hasil perhitungan diatas digunakan untuk menunjukkan akurasi dari sistem penentuan lokasi yang diimplementasikan. Rumus diatas juga digunakan pada pengujian untuk menghitung akurasi kesalahan tiap sub-lokasi dalam ruang

3.6.1.1 Alur Pengujian Akurasi Kesalahan Sistem

Pada sub bab ini akan dijelaskan alur dari pelaksanaan pengujian untuk memperoleh akurasi kesalahan penentuan lokasi secara umum. Alur pengujian ini tergambar pada Gambar dibawah ini.



Gambar 3.9 Alur Pengujian Akurasi Kesalahan Sistem

1. Pada ruang ke-n, *passive tag* dinyalakan
2. *Reference points* akan menangkap sinyal bluetooth dari *passive tag*
3. *Reference points* akan mengirim RSSI dari *passive tag* dan MAC address *reference points* menuju server menggunakan format data JSON
4. Server menerima data yang dikirim *reference points*
5. Server melakukan klasifikasi selama ± 2 menit dan mencatat hasil klasifikasi pada *file log*
6. *Passive tag* dimatikan
7. Proses nomor 1-6 diulang sampai semua lokasi selesai dilakukan pengambilan data.

Setelah hasil klasifikasi selesai diambil pada semua lokasi, langkah selanjutnya adalah menghitung akurasi kesalahan dengan menggunakan persamaan 3.1. Persamaan ini akan diimplementasikan pada sebuah kode sumber yang akan menghitung secara otomatis berdasarkan data pada *file log* yang telah dikumpulkan.

3.6.1.2 Format File Log Pengujian Akurasi Kesalahan Sistem

Pada sub bab ini akan dijelaskan mengenai format pada *file log* yang akan digunakan untuk perhitungan akurasi kesalahan dalam penentuan lokasi. *File log* ini terdiri dari :

1. Hasil klasifikasi (nama lokasi)
2. Waktu klasifikasi (HH:MM:SS)
3. Nama lokasi sebenarnya
4. Kecocokan, maksudnya adalah apabila hasil klasifikasi sama dengan lokasi sebenarnya, maka hasil pencocokan = cocok (match). Sebaliknya, jika hasil klasifikasi tidak sama dengan lokasi sebenarnya, maka hasil kecocokan = tidak cocok (mismatch)

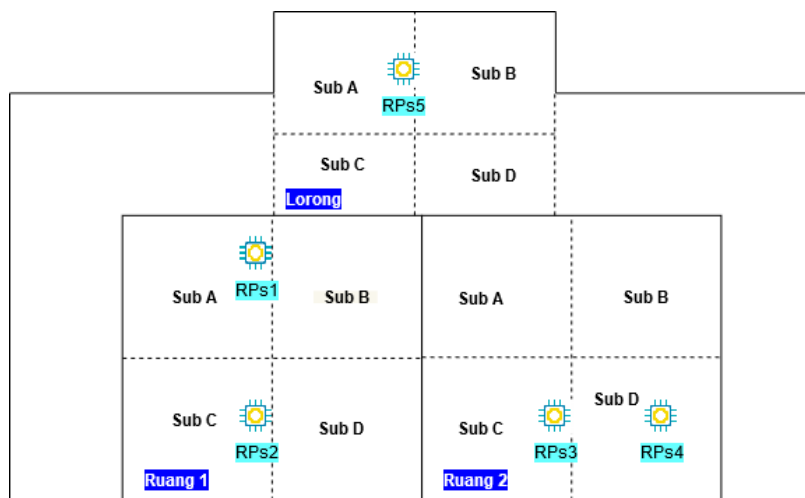
Tabel dibawah ini merupakan format dalam *file log* untuk pengujian ini.

Tabel 3.4 Format *file log* Pengujian Akurasi Kesalahan Sistem

| Hasil klasifikasi | Waktu | Lokasi sebenarnya | Kecocokan |
|-------------------|----------|-------------------|-----------|
| Ruang 1 | 00:00:01 | Ruang 1 | Match |
| Lorong | 00:00:02 | Ruang 1 | Mismatch |
| Ruang 1 | 00:00:03 | Ruang 1 | Match |

3.6.2 Perancangan Pengujian Akurasi Kesalahan Sistem Pada Tiap Sub-Lokasi

Pada sub bab ini akan dijelaskan secara detail mengenai implementasi dari pengujian akurasi kesalahan tiap sub lokasi pada ruangan. Tujuannya adalah untuk mengetahui pada sub lokasi mana pada ruangan yang memiliki tingkat akurasi kesalahan yang tinggi, sehingga nantinya dapat dijadikan objek permasalahan pada penelitian mendatang. Pada tiap lokasi implementasi akan dibagi menjadi 4 sub lokasi seperti Gambar 3.8 dibawah ini.

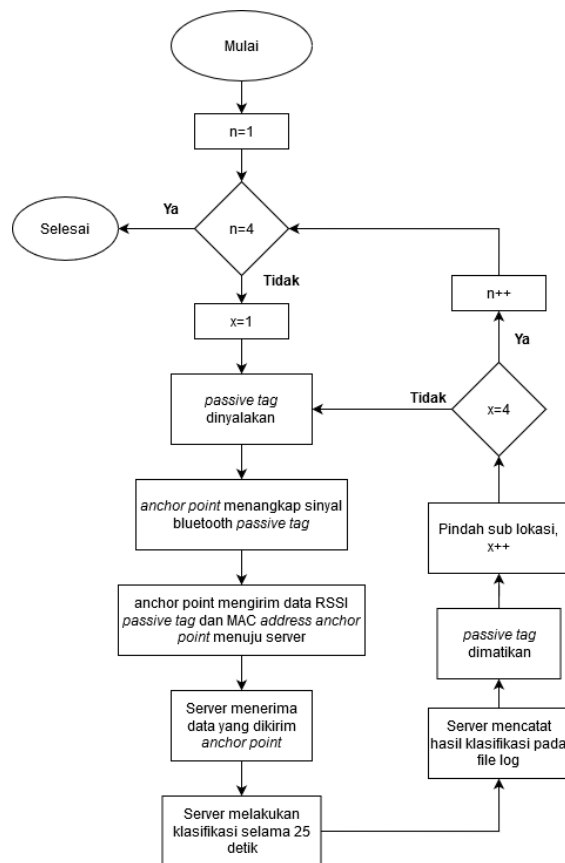


Gambar 3.10 Pembagian Sub Lokasi

Sub bab ini terdiri dari 2 bagian, bagian pertama untuk menjelaskan alur dari pengujian dan bagian kedua adalah format pada file log yang digunakan untuk mencatat hasil klasifikasi pada pengujian ini. *File log* ini nantinya akan diproses untuk mengetahui akurasi tiap sub lokasi pada suatu ruangan.

3.6.2.1 Alur Pengujian Akurasi Kesalahan Tiap Sub-Lokasi

Pada sub bab ini akan dijelaskan mengenai alur dari pengujian untuk menghitung akurasi kesalahan pada tiap sub lokasi dalam ruangan. Pada Gambar 3.9 dibawah merupakan alur dari pengujian yang akan dilakukan.



Gambar 3.11 Alur Pengujian Akurasi Kesalahan Sub Lokasi

1. Variabel n mewakili ruangan dan variabel x mewakili sub lokasi
2. Pada ruang ke-n dan sub lokasi ke-x *passive tag* dinyalakan
3. *Reference points* akan menangkap sinyal bluetooth dari *passive tag*
4. *Reference points* akan mengirim RSSI dari *passive tag* dan MAC address *reference points* menuju server menggunakan format data JSON
5. Server menerima data yang dikirim *reference points*
6. Server melakukan klasifikasi selama ± 25 detik pada sub lokasi x dan mencatat hasil klasifikasi pada *file log*
7. *Passive tag* dimatikan
8. Apabila x bukan sub lokasi terakhir maka pindah pada sub lokasi lain.
9. Apabila x adalah sub lokasi terakhir, maka pindah lokasi

10. Proses pada nomor 2-9 diulang sampai semua sub lokasi pada masing-masing ruangan tercatat hasil klasifikasinya pada *file log*

3.6.2.2 Format File Log Pengujian Akurasi Kesalahan Tiap Sub-Lokasi

Pada bagian ini akan dijelaskan format dari *file log* yang digunakan pada pengujian ini. *File log* yang digunakan terdiri dari :

1. Hasil klasifikasi
2. Waktu klasifikasi (HH:MM:SS)
3. Nama Ruangan sebenarnya
4. Sub lokasi pada ruangan
5. Kecocokan, maksudnya adalah apabila hasil klasifikasi sama dengan lokasi sebenarnya, maka hasil pencocokan = cocok (match). Sebaliknya, jika hasil klasifikasi tidak sama dengan lokasi sebenarnya, maka hasil kecocokan = tidak cocok (mismatch)

Tabel dibawah ini menunjukkan format dari *file log* yang digunakan.

Tabel 3.5 Format *file log* pengujian akurasi kesalahan pada tiap sub lokasi

| Hasil klasifikasi | Waktu | Lokasi sebenarnya | Sub lokasi | Kecocokan |
|-------------------|----------|-------------------|------------|-----------|
| Ruang 1 | 00:00:01 | Ruang 1 | A | Match |
| Lorong | 00:00:02 | Ruang 1 | B | Mismatch |
| Ruang 1 | 00:00:03 | Ruang 1 | C | Match |
| Ruang 1 | 00:00:03 | Ruang 1 | D | Match |

Setelah pengumpulan data pada *file log* selesai dikumpulkan, langkah selanjutnya adalah menghitung akurasi kesalahan pada tiap lokasi sehingga diketahui pada sub lokasi mana di suatu ruangan yang memiliki tingkat akurasi kesalahan paling tinggi. Pada Tabel 3.6 dibawah ini merupakan format untuk hasil perhitungan akurasi kesalahan pada tiap sub lokasi.

Tabel 3.6 Tabel akurasi kesalahan tiap sub-lokasi dalam ruang

| No | Ruang | Sub-Lokasi | Akurasi kesalahan |
|----|---------|--------------|-------------------|
| 1 | Ruang 1 | Sub Lokasi A | |
| | | Sub Lokasi B | |
| | | Sub Lokasi C | |
| | | Sub Lokasi D | |

Tabel 3.6 Tabel akurasi kesalahan tiap sub-lokasi dalam ruang (lanjutan)

| No | Ruang | Sub-Lokasi | Akurasi kesalahan |
|----|---------|--------------|-------------------|
| 2 | Ruang 2 | Sub Lokasi A | |
| | | Sub Lokasi B | |
| | | Sub Lokasi C | |
| | | Sub Lokasi D | |
| 3 | Lorong | Sub Lokasi A | |
| | | Sub Lokasi B | |
| | | Sub Lokasi C | |
| | | Sub Lokasi D | |

Pada Tabel 3.6 disajikan data berupa nama ruang, sub lokasi dan akurasi kesalahan. Tiap-tiap ruang yang digunakan untuk implementasi akan dibagi menjadi 4 bagian, kemudian akurasi kesalahannya dihitung menggunakan Persamaan 3.1.

3.7 Analisis Hasil Pengujian

Pada bab ini akan dianalisis hasil dari pengujian yang telah dilakukan. Tujuan dari analisis ini adalah untuk menemukan alasan kenapa dapat terjadi kesalahan dalam penentuan lokasi. Analisis dilakukan pada hasil pengujian akurasi sistem pada tiap sub lokasi. Pada hasil pengujian akan dilihat pada sub lokasi mana yang memiliki tingkat akurasi kesalahan yang paling tinggi. Sub lokasi yang memiliki tingkat akurasi kesalahan paling tinggi akan dianalisis untuk memperoleh jawaban kenapa dapat terjadi kesalahan dalam penentuan lokasi.

3.8 Kesimpulan dan Saran

Kesimpulan merupakan hasil akhir dari setiap langkah-langkah yang dilewati pada penelitian ini yang akan menjawab rumusan masalah yang telah disebutkan terlebih dahulu pada awal penelitian. Penulisan saran dibutuhkan untuk mengoreksi kesalahan-kesalahan yang ada pada penelitian ini. Diharapkan dengan adanya penulisan saran, peneliti yang akan melakukan implementasi dengan topik yang sama di masa yang akan datang dapat memperbaiki kesalahan yang ada pada penelitian ini.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Pada bab 4 berisi tentang penjelasan mengenai teknis implementasi dan pengujian sistem berdasarkan perancangan dari bab 3.4. Pada bab ini juga akan ditampilkan hasil dari implementasi dan pengujian. Bab ini dibagi menjadi 4 bagian, pertama akan menjelaskan mengenai implementasi tahap *Offline*, kedua akan menjelaskan mengenai implementasi tahap *Online*, ketiga akan menjelaskan mengenai implementasi pengujian akurasi kesalahan sistem, dan keempat akan menjelaskan mengenai pengujian akurasi kesalahan sistem pada tiap sub lokasi ruangan.

Pada tahap implementasi, perangkat yang digunakan untuk memenuhi kebutuhan komponen adalah sebagai berikut.

Tabel 4.1 Daftar Perangkat Dan Fungsinya

| No. | Nama Perangkat | Fungsi | Keterangan |
|-----|----------------|------------------------|---|
| 1. | iTAG | <i>Passive Tag</i> | Perangkat iTAG memiliki fitur untuk memancarkan sinyal bluetooth secara terus menerus, sehingga dapat digunakan sebagai <i>passive tag</i> . |
| 2. | ESP32 | <i>Reference Point</i> | Perangkat ESP32 memiliki fitur BLE dan WiFi. Fitur BLE digunakan untuk menangkap sinyal dari <i>passive tag</i> . Fitur WiFi digunakan untuk mengirim data RSSI menuju server |
| 3. | Laptop | <i>Server</i> | Proses penulisan <i>data training</i> hingga proses klasifikasi dilakukan pada server. |

4.1 Implementasi Tahap *Offline*

Tahap *Offline* diperlukan untuk membentuk *data training*. *Data training* ini nantinya akan digunakan pada tahap *online* untuk klasifikasi nama lokasi sebagaimana pada rancangan tahap *online*. Tahap *Offline* dimulai dengan dengan menyalakan *passive tag* yang memancarkan sinyal bluetooth secara terus-menerus. Kemudian beberapa *reference points* akan menangkap sinyal dari *passive tag* dan mengirimkan data RSSI *passive tag* dan MAC address *reference points* menuju server dengan menggunakan format JSON. Pada *pseudocode* dibawah ini menjelaskan tentang cara kerja dari *reference points*.

Tabel 4.2 Pseudocode Reference Point

| | |
|--|---|
| KAMUS DATA | |
| Address : String | // MAC address passive tag |
| RSSI : integer | // Untuk menyimpan rssi passive tag |
| MAC : String | // Untuk menyimpan MAC address ESP32 |
| json : JSON | // Untuk menyimpan rssi passive tag dan MAC Address ESP32 |
| BEGIN | |
| Mulai pemindaian | |
| IF hasil_pemindaian == Address | |
| Stop pemindaian | |
| RSSI \leftarrow get_passive_tag_rssi | |
| MAC \leftarrow get_ESP32_MAC_address | |
| json \leftarrow mac : MAC, rssi : RSSI | |
| Send json to Server | |
| END | |

Mekanisme dari cara kerja *reference points* berdasarkan *pseudocode* di atas adalah pertama *reference points* akan mencari *passive tag* berdasarkan dari MAC address *passive tag*, kemudian setelah perangkat ditemukan, *reference points* akan menyimpan data RSSI pada variabel dengan nama RSSI. Setelah itu, *reference points* akan membuat variabel dengan format JSON. Pada variabel JSON tersebut diberi nilai dengan menggunakan variabel RSSI milik *passive tag* dan nilai dari MAC address *reference points* dengan format

```
{
  "mac" = MAC;
  "rssi" = RSSI;
}
```

Apabila *passive tag* tidak terdeteksi atau nilai RSSI kurang dari -120dBm, maka nilai RSSI yang dikirim ke server akan default bernilai -120dBm. Hal ini dilakukan agar nilai RSSI yang terekam dalam *data training* menjadi teratur dan memudahkan dalam proses klasifikasi pada tahap *online*. Proses selanjutnya setelah perangkat *references points* mengirim data RSSI *passive tag* dan MAC address dari *reference points* adalah server menerima data JSON yang dikirim oleh perangkat *reference points*. Data JSON tersebut kemudian akan diproses dan disimpan pada sebuah file dengan format csv dengan format seperti pada perancangan sub bab 3.4.1.2. Pada *pseudocode* dibawah ini akan dijelaskan mengenai proses server menerima data JSON dan menuliskannya pada sebuah file.

Tabel 4.3 Pseudocode server menerima data dari Reference Point

| | |
|--|---|
| KAMUS DATA | |
| mac1, mac2, mac3, mac4, mac5, : String | // MAC address reference point |
| rss : array | // Untuk menyimpan rssi <i>passive tag</i> yang dikirim dari masing-masing <i>reference point</i> |
| data : JSON | // Untuk menyimpan data yang dikirim <i>reference point</i> |
| waktu : time | // Mewakili waktu saat data pola sinyal diperoleh |
| BEGIN | |
| WHILE (incoming_connection) DO | |
| IF data['mac'] == mac1 | |
| rss[0] ← data['rssi'] | |
| ELSE IF data['mac'] == mac2 | |
| rss[1] ← data['rssi'] | |
| ELSE IF data['mac'] == mac3 | |
| rss[2] ← data['rssi'] | |
| ELSE IF data['mac'] == mac4 | |
| rss[3] ← data['rssi'] | |
| ELSE | |
| rss[4] ← data['rssi'] | |
| IF there is 'None' in rss | |
| PRINT "waiting another RSSI" | |
| ELSE | |
| IF all value in 'rssi' == -120 | |
| PRINT "All RSSI value -120" | |
| ELSE | |
| Write in file rssi_collected.csv → "nama lokasi", waktu, rss[0], rss[1], | |
| rss[2], rss[3], rss[4] | |
| PRINT "1 row inserted" | |
| END | |

Mekanisme kerja server berdasarkan *pseudocode* di atas adalah pertama deklarasi 5 variabel dengan nama mac1, mac2, mac3, mac4, dan mac5 untuk menyimpan nilai MAC address dari semua *reference point*. Kemudian server akan menyimpan data JSON yang dikirim *reference points* pada sebuah variabel dengan nama 'data'. Selanjutnya pada variabel 'data' dilakukan pengecekan apakah mengandung data MAC address dari mac1, mac2, mac3, mac4, atau mac5. Apabila mengandung salah satu MAC address dari salah satu variabel mac diatas, maka nilai RSSI pada variabel 'data' akan disimpan pada variabel array bernama rss[n], dengan n adalah urutan dari *reference point*. Proses ini kemudian akan berlangsung sampai semua variabel array 'rss[i]' terisi dengan data RSSI yang dikirim dari tiap *reference points*. Apabila pada variabel array 'rss[i]' masih terdapat anggota dengan nilai "None", maka server akan menampilkan tulisan "waiting another RSSI", jika tidak ada yang bernilai "None", maka selanjutnya

server mengecek apakah variabel array 'rssi[]' semua anggotanya bernilai -120, jika iya, maka server akan menampilkan tulisan "All RSSI value -120", jika tidak maka server akan menuliskan hasil pengumpulan rssi pada file dengan format "nama lokasi", waktu, rssi 1, rssi 2, rssi 3, rssi 4, rssi 5. Kemudian server akan menampilkan tulisan "1 row inserted". Proses ini diulang sampai semua ruangan berhasil direkam data pola sinyalnya.

4.2 Implementasi Tahap Online

Setelah *data training* berhasil diperoleh pada tahap *offline*, tahap selanjutnya adalah melakukan implementasi tahap *online* atau tahap untuk penentuan lokasi. Tahap ini berfokus pada penentuan lokasi pada *passive tag* sesuai dengan rancangan pada bab 3. Tahap *online* dimulai dengan menyalakan *passive tag* pada sembarang lokasi. Kemudian *reference points* akan menangkap sinyal dari *passive tag* dan mengirimkannya pada server beserta MAC address dari *reference point*. Mekanisme kerja pengiriman data RSSI dari *reference point* menuju server pada tahap *online* sama dengan pada tahap *offline*. Pada *pseudocode* di bawah ini akan menjelaskan mekanisme pengiriman data RSSI dari *reference point* menuju server.

Tabel 4.4 Pseudocode reference point mengirim data menuju server

| | |
|--------------------------------|---|
| KAMUS DATA | |
| Address : String | // MAC address passive tag |
| RSSI : integer | // Untuk menyimpan rssi passive tag |
| MAC : String | // Untuk menyimpan MAC address ESP32 |
| json : JSON | // Untuk menyimpan rssi passive tag dan MAC Address ESP32 |
| BEGIN | |
| Mulai pemindaian | |
| IF hasil_pemindaian == Address | |
| Stop pemindaian | |
| RSSI ← get_passive_tag_rssi | |
| MAC ← get_ESP32_MAC_address | |
| json ← mac : MAC, rssi : RSSI | |
| Send json to Server | |
| END | |

Mekanisme dari cara kerja *reference points* berdasarkan *pseudocode* di atas adalah pertama *reference points* akan mencari *passive tag* berdasarkan dari MAC address *passive tag*, kemudian setelah perangkat ditemukan, *reference points* akan menyimpan data RSSI pada variabel dengan nama RSSI. Setelah itu, *reference points* akan membuat variabel dengan format JSON. Pada variabel JSON tersebut diberi nilai dengan menggunakan variabel RSSI milik *passive tag* dan nilai dari MAC address *reference points* dengan format

```

{
  "mac" = MAC;
  "rssi" = RSSI;
}

```

Apabila *passive tag* tidak terdeteksi atau nilai RSSI kurang dari -120dBm, maka nilai RSSI yang dikirim ke server akan default bernilai -120dBm. Selanjutnya, server akan menerima data JSON yang dikirim oleh *reference point* dan melakukan klasifikasi untuk penentuan lokasi dari *passive tag*. Pada *pseudocode* dibawah ini akan menjelaskan mekanisme kerja dari server pada tahap *online* ini.

Tabel 4.5 Pseudocode server melakukan klasifikasi

| | |
|---|---|
| KAMUS DATA | |
| mac1, mac2, mac3, mac4, mac5, : String | // MAC address reference point |
| rssi : array | // Untuk menyimpan rssi <i>passive tag</i> yang dikirim dari masing-masing <i>reference point</i> |
| data : JSON | // Untuk menyimpan data yang dikirim <i>reference point</i> |
| waktu : time | // Mewakili waktu saat server melakukan klasifikasi |
| ds : array | // Untuk menyimpan kumpulan nilai rssi yang akan diklasifikasikan |
| dt : array | // Untuk menyimpan nilai data training |
| predictions : String | // Untuk menyimpan hasil prediksi/klasifikasi |
| BEGIN | |
| WHILE (incoming_connection) DO | |
| IF data['mac'] == mac1 | |
| rssi[0] ← data['rssi'] | |
| ELSE IF data['mac'] == mac2 | |
| rssi[1] ← data['rssi'] | |
| ELSE IF data['mac'] == mac3 | |
| rssi[2] ← data['rssi'] | |
| ELSE IF data['mac'] == mac4 | |
| rssi[3] ← data['rssi'] | |
| ELSE | |
| rssi[4] ← data['rssi'] | |
| IF there is no 'None' in rssi | |
| IF all value in 'rssi' == -120 | |
| PRINT "Beacon tidak terdeteksi" | |
| ELSE | |
| ds ← rssi | |
| dt ← rssi_collected.csv | |
| predictions ← PREDICT(ds,dt) | |
| PRINT "Pada Jam ", waktu, " Beacon berada di ", predictions | |
| Write in file real_time_result.csv → prediction, waktu | |
| END | |

Penjelasan dari *pseudocode* di atas adalah pertama server melakukan inisialisasi variabel array 'mac' dengan nilai adalah MAC *address* dari masing-masing *reference point*. Kemudian inisialisasi variabel array 'rssi' dengan nilai adalah data RSSI yang dikirim oleh *reference point*. Selanjutnya adalah server menerima data JSON yang dikirim oleh *reference point*. Data JSON yang diterima akan diinisialisasi pada variabel 'data'. Selanjutnya adalah server mengecek pada variabel 'data['mac']' memiliki nilai yang sama pada salah satu nilai pada variabel array 'mac'. Apabila ada yang sama, maka variabel array 'rssi' akan diberi nilai dengan data RSSI pada variabel 'data['rssi']'. Anggota variabel array 'rssi' diberi nilai sesuai dengan urutan MAC *address* dari *reference point*. Apabila semua nilai pada variabel array 'rssi[]' memiliki nilai -120, maka server menampilkan tulisan "iTAG tidak terdeteksi". Jika semua anggota variabel array 'rssi[]' terisi dengan data RSSI, selanjutnya server melakukan klasifikasi menggunakan algoritma KNN untuk menentukan lokasi dari *passive tag*. Proses klasifikasi dengan algoritma KNN berdasarkan pada *data training* yang telah diperoleh sebelumnya. Hasil klasifikasi ditampilkan pada layar beserta waktu klasifikasi. Setelah proses klasifikasi selesai, server mencatat hasil klasifikasi pada *file log* beserta waktu klasifikasi. Proses kerja server dalam melakukan klasifikasi berlangsung tiap detik atau *real-time*.

4.3 Implementasi Pengujian Akurasi Kesalahan Sistem

Tahap selanjutnya adalah menguji akurasi sistem yang telah diimplementasikan. Pada tahap ini pengujian dilakukan untuk menghitung akurasi kesalahan dari sistem penentuan lokasi ini. Pengujian ini berfokus pada akurasi kesalahan secara *general* atau secara umum. Tahap pengujian ini dimulai dengan mengumpulkan data hasil klasifikasi pada tiap lokasi. Kemudian setelah data hasil klasifikasi terkumpul, server melakukan perhitungan akurasi kesalahan menggunakan *script* yang telah disiapkan. Proses pengumpulan data hasil klasifikasi sama seperti pada tahap *online*, hanya saja terdapat perbedaan yang terletak pada alur pengambilan data dan format penulisan *file log* yang ditulis oleh server. Proses pengambilan data dilakukan sesuai alur pada bab 3, kemudian proses penentuan lokasi dan penulisan *file log* oleh server dijelaskan pada *pseudocode* dibawah ini.

Tabel 4.6 Proses mengumpulkan hasil klasifikasi

| KAMUS DATA | |
|--------------|---|
| mac : array | // Kumpulan MAC <i>address reference point</i> |
| rssi : array | // Untuk menyimpan rssi <i>passive tag</i> yang dikirim dari masing-masing <i>reference point</i> |
| data : JSON | // Untuk menyimpan data yang dikirim <i>reference point</i> |
| waktu : time | // Mewakili waktu saat server melakukan klasifikasi |
| ds : array | // Untuk menyimpan kumpulan nilai rssi yang akan diklasifikasikan |
| dt : array | // Untuk menyimpan nilai data training |

Tabel 4.6 Proses mengumpulkan hasil klasifikasi (lanjutan)

| | |
|------------------------|---|
| true_loc : String | // Untuk menyimpan data dari nama lokasi sebenarnya |
| predictions : String | // Untuk menyimpan hasil prediksi/klasifikasi |
| compatibility : String | // Untuk menyimpan nilai kecocokan antara hasil klasifikasi dengan nama lokasi sebenarnya |

```

BEGIN
WHILE (incoming_connection) DO
FOR (i in range LENGTH(mac))
IF data['mac'] == "locator"
true_loc ← data['true_loc']
BREAK
ELSE IF data['mac'] == mac[i]
rssi[i] ← data['rssi']
BREAK
END FOR
IF there is no 'None' in rssi
IF all value in 'rssi' == -120
PRINT "Beacon tidak terdeteksi"
ELSE
ds ← rssi
dt ← rssi_collected.csv
predictions ← PREDICT(ds,dt)
IF true_loc == predictions
compatibility ← "Match"
ELSE
compatibility ← "Mismatch"
PRINT "Pada Jam ", waktu, " Beacon berada di ", predictions
Write in file data_pengujian_1.csv → predictions, waktu, true_loc,
compatibility
END

```

Penjelasan pada *pseudocode* di atas adalah pertama, inisialisasi variabel array 'mac', 'rssi', dan variabel dengan nama 'true_loc'. Variabel 'true_loc' digunakan untuk menyimpan nama lokasi sebenarnya. Variabel 'true_loc' ini nantinya akan diberi nilai oleh entitas lain selain *reference point*. Penulis menggunakan perangkat lunak Postman untuk mengirim data lokasi sebenarnya yang nantinya akan disimpan pada variabel 'true_loc'. Selanjutnya, server menerima data JSON dan disimpan pada variabel 'data'. Apabila nilai dari 'data['mac']' adalah "locator" maka nilai dari 'data['true_loc']' akan disimpan pada variabel 'true_loc' sebagai data dari lokasi sebenarnya. Apabila nilai dari 'data['mac']' adalah data MAC address dari *reference point* maka nilai pada 'data['rssi']' akan disimpan pada variabel array 'rssi'. Urutan penulisan pada anggota variabel array 'rssi' sesuai dengan urutan anggota pada variabel array 'mac'.

Server selanjutnya mengecek apakah variabel array 'rssi[]' memiliki anggota yang bernilai 'None', jika ada, maka server tidak akan melakukan proses klasifikasi. Namun, jika tidak ada anggota dari variable array 'rssi[]' yang bernilai 'None', maka server akan mengecek apakah seluruh anggota variabel 'rssi[]' bernilai -120. Jika iya, maka server menampilkan tulisan "Beacon tidak terdeteksi". Namun, jika tidak semua anggota variabel array 'rssi[]' bernilai -120, maka server akan melakukan klasifikasi nama lokasi berdasarkan pola sinyal yang tersimpan pada variabel array 'rssi[]'. Apabila hasil klasifikasi sama dengan nama lokasi sebenarnya, maka inisialisasi variabel 'compatibility' dengan nilai "Match". Sebaliknya, jika hasil klasifikasi tidak sama dengan nama lokasi sebenarnya, maka inisialisasi variabel 'compatibility' dengan nilai "Mismatch". Selanjutnya, server menulis hasil klasifikasi pada *file log* sesuai dengan format pada perancangan untuk dihitung akurasi kesalahan pada langkah selanjutnya.

4.3.1 Perhitungan Akurasi Kesalahan Sistem

Tahap selanjutnya setelah memperoleh data hasil klasifikasi adalah menghitung akurasi kesalahan dari sistem. Perhitungan akurasi kesalahan sistem menggunakan persamaan 3.1 yang dituangkan pada sebuah *script*. *Script* ini bekerja dengan menginput *file log* dari hasil klasifikasi diatas, kemudian *script* akan melakukan perhitungan dan memberikan nilai output dari akurasi kesalahan sistem berupa angka dalam persen. Pada *pseudocode* dibawah ini akan menjelaskan mengenai mekanisme kerja untuk menghitung akurasi kesalahan sistem.

Tabel 4.7 Pseudocode server menghitung akurasi kesalahan sistem

| | |
|---|---|
| KAMUS DATA | |
| data : array | // untuk menyimpan data pengujian yang telah dikumpulkan sebelumnya |
| mismatch : integer | // untuk menyimpan jumlah kesalahan klasifikasi |
| percent : float | // untuk menyimpan hasil perhitungan akurasi kesalahan dalam persen |
| BEGIN | |
| data ← data_pengujian_1.csv | |
| mismatch ← 0 | |
| FOR (i in range length of data) | |
| IF data[i][2] == "Mismatch" | |
| mismatch ← mismatch + 1 | |
| END FOR | |
| percent ← mismatch / length of data * 100 | |
| print "Akurasi kesalahan sistem = ", percent, "%" | |
| END | |

Penjelasan *pseudocode* di atas adalah pertama inisialisasi variabel array 'data[]' yang berfungsi untuk menyimpan data pengujian yang telah dikumpulkan sebelumnya. Selanjutnya inisialisasi variabel 'mismatch[]' yang berfungsi untuk

menyimpan jumlah kesalahan klasifikasi. Kemudian inisialisasi variabel 'percent' untuk menyimpan hasil perhitungan akurasi kesalahan. Langkah selanjutnya adalah memasukkan data pengujian pada variabel 'data[]', lalu menghitung jumlah kesalahan klasifikasi dengan perulangan. Apabila nilai dalam variabel 'data[i][2]' adalah "Mismatch", maka nilai variabel 'mismatch' bertambah 1. Proses ini berulang sebanyak jumlah data pada data pengujian. Selanjutnya, variabel 'percent' diberi nilai dengan hasil dari variabel 'mismatch' dibagi dengan total jumlah data dikali dengan 100. Terakhir, ditampilkan pada layar hasil perhitungan dengan tulisan "Akurasi kesalahan sistem = ", percent, "%"

4.4 Implementasi Pengujian Akurasi Kesalahan Sistem Pada Tiap Sub Lokasi

Tahap selanjutnya adalah melakukan pengujian untuk mengukur akurasi kesalahan pada tiap sub-lokasi. Pengujian ini bertujuan untuk memetakan sub-lokasi mana yang memiliki tingkat akurasi kesalahan yang paling tinggi. Langkah pertama adalah mengambil data hasil klasifikasi pada tiap sub lokasi masing-masing ruangan. Data hasil klasifikasi akan ditulis pada *file log* untuk selanjutnya akan dilakukan perhitungan akurasi kesalahannya. Pada *pseudocode* di bawah ini akan menjelaskan mekanisme server dalam melakukan klasifikasi dan menyimpan datanya dalam *file log*.

Tabel 4.8 Pseudocode server mengumpulkan data klasifikasi

| | |
|--------------------------------|---|
| KAMUS DATA | |
| mac : array | // Kumpulan MAC <i>address reference point</i> |
| rss : array | // Untuk menyimpan rssi <i>passive tag</i> yang dikirim dari masing-masing <i>reference point</i> |
| data : JSON | // Untuk menyimpan data yang dikirim <i>reference point</i> |
| waktu : time | // Mewakili waktu saat server melakukan klasifikasi |
| x_ds : array | // Untuk menyimpan kumpulan nilai rssi yang akan diklasifikasikan |
| data_training : array | // Untuk menyimpan nilai data training |
| true_loc : String | // Untuk menyimpan data dari nama lokasi sebenarnya |
| sub_loc : String | // Untuk menyimpan nama sub lokasi |
| predictions : String | // Untuk menyimpan hasil prediksi/klasifikasi |
| compatibility : String | // Untuk menyimpan nilai kecocokan antara hasil klasifikasi dengan nama lokasi sebenarnya |
| BEGIN | |
| WHILE (incoming_connection) DO | |
| FOR (i in range LENGTH(mac)) | |
| IF data['mac'] == "locator" | |
| true_loc ← data['true_loc'] | |
| sub_loc ← data['sub_loc'] | |

Tabel 4.8 Pseudocode server mengumpulkan data klasifikasi (lanjutan)

```
BREAK
ELSE IF data['mac'] == mac[i]
  rssi[i] ← data['rssi']
  BREAK
END FOR
IF there is no 'None' in rssi
  IF all value in 'rssi' == -120
    PRINT "Beacon tidak terdeteksi"
  ELSE
    x_ds ← rssi
    data_training ← rssi_collected.csv
    predictions ← PREDICT(ds,dt)
    IF true_loc == predictions
      compatibility ← "Match"
    ELSE
      compatibility ← "Mismatch"
    PRINT "Pada Jam ", waktu, " Beacon berada di ", predictions
    Write in file data_pengujian_2.csv → predictions, waktu, true_loc,
      sub_loc, compatibility
  END
```

Penjelasan pada *pseudocode* di atas adalah pertama, inisialisasi variabel array 'mac', 'rssi', variabel dengan nama 'true_loc' dan 'sub_loc'. Variabel 'true_loc' digunakan untuk menyimpan nama lokasi sebenarnya, dan variabel 'sub_loc' digunakan untuk menyimpan data sub lokasi. Variabel 'true_loc' dan 'sub_loc' ini nantinya akan diberi nilai oleh entitas lain selain *reference point*. Penulis menggunakan perangkat lunak Postman untuk mengirim data lokasi sebenarnya dan data sub lokasi yang nantinya akan disimpan pada variabel 'true_loc' dan 'sub_loc'. Selanjutnya, server menerima data JSON dan disimpan pada variabel 'data'. Apabila nilai dari 'data['mac']' adalah "locator" maka nilai dari 'data['true_loc']' akan disimpan pada variabel 'true_loc' sebagai data dari lokasi sebenarnya dan nilai dari 'data['sub_loc']' akan disimpan pada variabel 'sub_loc'. Apabila nilai dari 'data['mac']' adalah data MAC *address* dari *reference point* maka nilai pada 'data['rssi']' akan disimpan pada variabel array 'rssi'. Urutan penulisan pada anggota variabel array 'rssi' sesuai dengan urutan anggota pada variabel array 'mac'.

Setelah variabel 'rssi' tidak ada yang bernilai 'None', maka server akan mengecek apakah semua anggota variabel array 'rssi[]' bernilai -120, jika iya maka server menampilkan tulisan "Beacon tidak terdeteksi". Namun, jika tidak maka server akan melakukan klasifikasi nama lokasi berdasarkan pola sinyal yang tersimpan pada variabel array 'rssi[]'. Apabila hasil klasifikasi sama dengan nama lokasi sebenarnya, maka inisialisasi variabel 'compatibility' dengan nilai "Match".

Sebaliknya, jika hasil klasifikasi tidak sama dengan nama lokasi sebenarnya, maka inisialisasi variabel 'compatibility' dengan nilai "Mismatch" . Selanjutnya, server menulis hasil klasifikasi pada *file log* sesuai dengan format pada perancangan untuk dihitung akurasi kesalahan tiap sub lokasi pada langkah selanjutnya.

4.4.1 Perhitungan Akurasi Kesalahan Sistem Pada Tiap Sub Lokasi

Langkah selanjutnya pada pengujian kedua ini adalah menghitung akurasi kesalahan pada tiap sub lokasi. Perhitungan akurasi dilakukan menggunakan *script* khusus yang dapat menghitung dan menampilkan hasil perhitungan dari tiap sub lokasi dalam bilangan persen. Pada *pseudocode* di bawah ini akan menjelaskan mengenai mekanisme kerja untuk menghitung akurasi kesalahan tiap lokasi.

Tabel 4.9 Perhitungan akurasi kesalahan tiap sub lokasi

| | |
|--|---|
| KAMUS DATA | |
| data : array | // untuk menyimpan data pengujian yang telah dikumpulkan sebelumnya |
| sub_loc : array | // untuk menyimpan nama sub lokasi |
| data_sub_loc_ruang1 : array | // untuk menyimpan data klasifikasi pada ruang 1 |
| data_sub_loc_ruang2 : array | // untuk menyimpan data klasifikasi pada ruang 2 |
| data_sub_loc_lorong : array | // untuk menyimpan data klasifikasi pada lorong |
| mismatch : integer | // untuk menyimpan jumlah kesalahan klasifikasi |
| percent : float | // untuk menyimpan hasil perhitungan akurasi kesalahan dalam persen |
| BEGIN | |
| data ← data_pengujian_2.csv | |
| mismatch ← 0 | |
| FOR i in range of data | |
| IF data[i][1] == "Ruang 1" | |
| FOR j in range(4) | |
| IF data[i][2] == sub_loc[j] | |
| APPEND(data_sub_loc_ruang1[j],data[i]) | |
| ELSE IF data[i][1] == "Ruang 2" | |
| APPEND(data_sub_loc_ruang2[j],data[i]) | |
| ELSE | |
| APPEND(data_sub_loc_lorong[j],data[i]) | |
| END FOR | |
| END FOR | |
| FUNCTION mismatch_calc(array) | |
| mismatch ← [0,0,0,0] | |
| FOR i in range array | |
| FOR j in range array[i] | |
| IF array[i][j][3] == "Mismatch" | |

Tabel 4.9 Perhitungan akurasi kesalahan tiap sub lokasi (lanjutan)

```
        mismatch[i] ← mismatch[i] + 1
    END FOR
END FOR
FOR i in range array
    Percent ← mismatch[i] / length of array * 100
    PRINT "Sub Lokasi ",array[i][0][2], "%"
END FOR
END FUNCTION
PRINT "Kesalahan Predeiksi Pada Ruang 1 adalah :"
```

mismatch_calc(data_sub_loc_ruang1)

PRINT "Kesalahan Predeiksi Pada Ruang 2 adalah :"

mismatch_calc(data_sub_loc_ruang2)

PRINT "Kesalahan Predeiksi Pada Lorong adalah :"

mismatch_calc(data_sub_loc_lorong)

END

Pertama, kita simpan data hasil klasifikasi pada variabel 'data'. Kemudian, kita inialisasi variabel array 'sub_loc'. Variabel array 'sub_loc' berfungsi untuk menyimpan data dari nama sub lokasi. Selanjutnya inialisai variabel array 'data_sub_loc_ruang1', 'data_sub_loc_ruang2', dan 'data_sub_loc_lorong'. Ketiga variabel array tersebut berfungsi untuk menyimpan hasil klasifikasi dari masing-masing sub lokasi pada masing-masing ruangan. Selanjutnya adalah memberi nilai pada variabel array 'data_sub_loc_ruang1', 'data_sub_loc_ruang2', dan 'data_sub_loc_lorong' menggunakan fungsi perulangan sehingga semua data hasil klasifikasi berhasil masuk pada masing-masing variabel array sesuai dengan nama lokasinya. Langkah selanjutnya adalah membuat fungsi untuk menghitung akurasi kesalahan. Fungsi ini dibuat berdasarkan pada persamaan 3.1 untuk menghitung akurasi kesalahan dan diberi nama 'mismatch_calc'. Langkah terakhir adalah menampilkan hasil dari perhitungan dengan memanggil fungsi 'mismatch_calc' dengan parameter adalah variabel array dari masing-masing lokasi. Hasil yang ditampilkan adalah akurasi kesalahan pada tiap sub lokasi pada masing-masing ruangan.

BAB 5 HASIL DAN PEMBAHASAN

Commented [AB7]: Skenario, parameter uji, cara menjalankan skenarionya dimana ya mas?

Pada bab ini akan dibahas mengenai hasil implementasi dan pengujian yang telah dilakukan pada bab 4. Bab ini akan dibagi menjadi 4 bagian, bagian pertama akan membahas hasil implementasi tahap offline, bagian kedua akan membahas hasil implementasi tahap *online*, bagian ketiga akan membahas hasil implementasi pengujian akurasi kesalahan sistem dan bagian keempat akan membahas hasil implementasi pengujian akurasi kesalahan sistem pada tiap sub lokasi.

5.1 Hasil Implementasi Tahap Offline

Hasil dari implementasi tahap *offline* adalah sebuah file *data training* dengan format csv yang berisi data nama lokasi beserta pola sinyal dari masing-masing lokasi tersebut. Pada tabel berikut ini akan ditampilkan isi dari file *data training* hasil implementasi pada tahap *offline*.

Tabel 5.1 Hasil Implementasi Tahap Offline

Commented [AB8]: Tidak perlu ditampilkan semua di sini. Cukup sampelnya. Data lengkap ada di Lampiran

| |
|--|
| Ruang 1,13:26:56,-75,-120,-120,-120 |
| Ruang 1,13:26:57,-83,-94,-120,-95,-120 |
| Ruang 1,13:26:58,-90,-89,-120,-120,-120 |
| |
| |
| Lorong,13:35:50,-120,-120,-120,-120,-89 |
| Lorong,13:35:50,-120,-87,-91,-120,-81 |
| Lorong,13:35:52,-120,-86,-120,-120,-89 |
| |
| |
| Ruang 2,13:40:03,-120,-120,-120,-120,-92 |
| Ruang 2,13:40:04,-120,-88,-93,-92,-91 |
| Ruang 2,13:40:05,-92,-98,-84,-87,-120 |

Penulis membatasi data yang ditampilkan pada tabel di atas karena banyaknya baris data yang terdapat pada file *data training*. Penulis melakukan pengambilan data dengan durasi ± 3 menit pada masing-masing lokasi. Pada tabel di atas dapat dilihat bahwa server menuliskan hasil pengambilan pola sinyal pada tiap baris dalam hitungan 1-2 detik. File ini kemudian akan dilatih untuk menjadi model dalam klasifikasi pada tahap *online*.

Pada data yang ditampilkan di atas, terlihat bahwa sistem dapat mengumpulkan pola sinyal dari *passive tag* pada tiap-tiap lokasi. Pola sinyal yang diperoleh dicatat bersama dengan waktu pengambilan pola sinyal, kemudian diberi label sesuai dengan nama lokasi. Kumpulan pola sinyal yang diperoleh tersebut kemudian disimpan dengan format file .csv. Berdasarkan penjelasan di atas, hasil implementasi tahap *offline* sudah sesuai dengan perancangan pada bab 3. Hasil implementasi tahap *offline* ini akan digunakan pada tahap *online* untuk menentukan lokasi.

5.2 Hasil Implementasi Tahap *Online*

Pada saat proses penentuan lokasi telah selesai dilakukan, hasil dari penentuan lokasi dicatat oleh server pada sebuah *file log*. Format *file log* yang dibuat oleh server sesuai dengan perancangan pada bab 3. Pada tabel dibawah ini merupakan isi dari *file log* hasil implementasi tahap *online*.

Tabel 5.2 Hasil Implementasi Tahap *Online*

| |
|------------------|
| Ruang 1,14:58:09 |
| Ruang 1,14:58:09 |
| Ruang 1,14:58:09 |
| |
| |
| Lorong,14:58:27 |
| Lorong,14:58:27 |
| Lorong,14:58:27 |

Commented [AB9]: Tidak perlu ditampilkan begini.

Kenapa tidak ditampilkan saja pengujian fungsional dari beberapa lokasi?

Pada Tabel 5.2 penulis tidak menampilkan seluruh hasil dari implementasi karena banyaknya baris data yang terekam. Pada tabel diatas dapat dilihat bahwa format penulisan file sudah sesuai dengan perancangan, yakni hasil klasifikasi dan waktu klasifikasi. Pada Gambar 5.1, Gambar 5.2, dan Gambar 5.3 berikut ini merupakan tangkapan layar dari implementasi tahap *online*.

```
PS D:\DATA KULIAH\SEMESTER VIII\skripsi-indoorlocalization\CODE\trial_env_2> py .\real_time_classifier_0.1.py
* Serving Flask app "real_time_classifier_0.1" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
Node 3 is Online
Node 1 is Online
Node 4 is Online
Node 5 is Online
Node 2 is Online
Beacon tidak terdeteksi
Beacon tidak terdeteksi
```

Gambar 5.1 Menjalankan Server

```
Beacon tidak terdeteksi
Beacon tidak terdeteksi
Beacon tidak terdeteksi
Pada Jam 14:58:09 beacon berada di Ruang 1
Pada Jam 14:58:09 beacon berada di Ruang 1
Pada Jam 14:58:09 beacon berada di Ruang 1
Pada Jam 14:58:10 beacon berada di Ruang 1
Pada Jam 14:58:10 beacon berada di Ruang 1
Pada Jam 14:58:10 beacon berada di Ruang 1
Pada Jam 14:58:10 beacon berada di Ruang 1
Pada Jam 14:58:10 beacon berada di Ruang 1
```

Gambar 5.2 Server menampilkan lokasi *passive tag* pada Ruang 1

```

Pada Jam 14:58:26 beacon berada di Ruang 1
Pada Jam 14:58:26 beacon berada di Ruang 1
Pada Jam 14:58:27 beacon berada di Lorong
Pada Jam 14:58:27 beacon berada di Lorong
Pada Jam 14:58:27 beacon berada di Lorong
Beacon tidak terdeteksi
Beacon tidak terdeteksi
Pada Jam 14:58:28 beacon berada di Lorong
Pada Jam 14:58:28 beacon berada di Lorong
Pada Jam 14:58:28 beacon berada di Lorong
Pada Jam 14:58:28 beacon berada di Lorong
Pada Jam 14:58:28 beacon berada di Lorong
Pada Jam 14:58:28 beacon berada di Lorong

```

Gambar 5.3 Server menampilkan lokasi *passive tag* pada lorong

```

Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2
Pada Jam 14:58:41 beacon berada di Ruang 2

```

Gambar 5.4 Server menampilkan lokasi *passive tag* pada Ruang 2

Pada Gambar 5.1 merupakan tampilan dari server saat pertama menjalankan server. Server menampilkan tulisan “Beacon tidak terdeteksi” karena *passive tag* belum dinyalakan. Kemudian pada Gambar 5.2 menampilkan saat *passive tag* dinyalakan, server menampilkan lokasi dari *passive tag* pada Ruang 1. Pada Gambar 5.3, lokasi dari *passive tag* berpindah menuju Lorong, sehingga server menampilkan lokasi dari *passive tag* pada lorong. Pada Gambar 5.4, *passive tag* berpindah menuju Ruang 2, sehingga server menampilkan lokasi *passive tag* pada Ruang 2.

Pada data di atas, terlihat bahwa sistem dapat melakukan penentuan lokasi dari *passive tag*. Lokasi yang telah didapatkan kemudian ditampilkan pada layar beserta waktu penentuan lokasi. Waktu penentuan lokasi yang ditampilkan memiliki selang waktu tidak sampai 1 detik. Hal ini mengindikasikan bahwa sistem dapat melakukan penentuan lokasi secara *real-time*. Pada data di atas juga ditampilkan apabila *passive tag* tidak terdeteksi, maka sistem akan menampilkan tulisan “Beacon tidak terdeteksi”. Berdasarkan data di atas, maka dapat disimpulkan bahwa implementasi tahap *online* bekerja sesuai dengan perancangan pada bab 3, sehingga dapat dikatakan bahwa tahap *online* berhasil dilakukan.

5.3 Hasil Implementasi Pengujian Akurasi Kesalahan Sistem

Implementasi pengambilan data hasil klasifikasi pada tahap ini menghasilkan sebuah *file log* yang berisi data hasil klasifikasi beserta nama lokasi sebenarnya. Data ini kemudian akan di hitung menggunakan persamaan 3.1 seperti pada bab

3. Berikut merupakan *file log* dari implementasi pengambilan data hasil klasifikasi yang telah dilakukan.

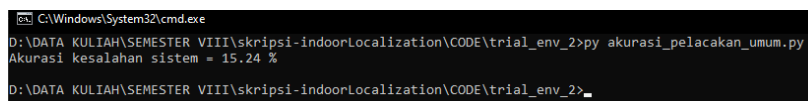
Tabel 5.3 Hasil pengambilan data pengujian 1

| |
|-----------------------------------|
| Ruang 1,15:31:46,Ruang 1,Match |
| Ruang 1,15:31:46,Ruang 1,Match |
| Ruang 1,15:31:46,Ruang 1,Match |
| |
| |
| Ruang 2,15:32:00,Ruang 1,Mismatch |
| Ruang 2,15:32:00,Ruang 1,Mismatch |
| Ruang 2,15:32:00,Ruang 1,Mismatch |

Pada tabel di atas disajikan hasil dari pengambilan data klasifikasi pada pengujian ini. Penulis hanya menampilkan beberapa baris saja, karena banyaknya jumlah data yang terekam. Pada tabel di atas terlihat bahwa sistem dapat melakukan penentuan lokasi dan mencatat hasil penentuan lokasi pada *file log*. Data yang dicatat memuat tentang hasil klasifikasi, waktu, nama lokasi sebenarnya, dan kecocokan. Pada tabel di atas sistem dapat menentukan kecocokan berdasarkan hasil klasifikasi dengan nama lokasi sebenarnya. Nilai kecocokan ditulis dengan 'Match' apabila hasil klasifikasi sama dengan nama lokasi sebenarnya. Sebaliknya, apabila hasil klasifikasi tidak sama dengan nama lokasi sebenarnya, maka nilai kecocokan ditulis 'Mismatch'. Nilai kecocokan ini akan digunakan untuk melakukan perhitungan akurasi kesalahan sistem.

5.3.1 Hasil Perhitungan Akurasi Kesalahan Sistem

Pada Gambar dibawah ini merupakan tangkapan layar dari eksekusi *script* untuk menghitung akurasi kesalahan sistem.



```
C:\Windows\System32\cmd.exe
D:\DATA KULIAH\SEMESTER VIII\skripsi-indoorLocalization\CODE\trial_env_2>py akurasi_pelacakan_umum.py
Akurasi kesalahan sistem = 15.24 %
D:\DATA KULIAH\SEMESTER VIII\skripsi-indoorLocalization\CODE\trial_env_2>
```

Gambar 5.5 Hasil Perhitungan Akurasi Kesalahan Sistem

Pada Gambar 4.1 diatas menunjukkan bahwa akurasi kesalahan sistem dalam penentuan lokasi sebesar 15,24%, yang berarti menunjukkan secara tidak langsung bahwa akurasi sistem untuk menentukan lokasi adalah $100\% - 15,24\% = 84,76\%$. Pada pengujian ini belum diketahui pada sub lokasi mana pada suatu ruangan yang memiliki tingkat presentase kesalahan akurasi paling tinggi. Untuk mengetahui pada sub lokasi mana yang memiliki presentase akurasi kesalahan paling tinggi akan dilakukan pada pengujian berikutnya. Berdasarkan pada data di atas, sistem dapat melakukan perhitungan akurasi kesalahan sistem. Hal ini mengindikasikan bahwa perhitungan akurasi kesalahan sistem telah sesuai dengan perancangan pada bab 3. Langkah selanjutnya adalah melakukan pengujian akurasi kesalahan sistem pada tiap sub lokasi.

5.4 Hasil Implementasi Pengujian Akurasi Kesalahan Sistem Pada Tiap Sub Lokasi

Implementasi pengambilan data hasil klasifikasi pada tahap ini menghasilkan sebuah *file log* yang berisi data hasil klasifikasi pada masing-masing sub lokasi. Data ini kemudian akan di hitung menggunakan *script* khusus yang akan menghitung akurasi kesalahan pada tiap sub lokasi. Berikut merupakan *file log* dari implementasi pengambilan data hasil klasifikasi yang telah dilakukan.

Tabel 5.4 Hasil pengambilan data klasifikasi tiap sub lokasi

| |
|-------------------------------------|
| Ruang 2,14:02:21,Ruang 1,A,Mismatch |
| Ruang 1,14:02:21,Ruang 1,A,Match |
| Ruang 1,14:02:21,Ruang 1,A,Match |
| |
| |
| Ruang 2,14:04:04,Ruang 1,B,Mismatch |
| Lorong,14:04:04,Ruang 1,B,Mismatch |
| Ruang 1,14:04:04,Ruang 1,B,Match |

Pada tabel di atas disajikan hasil dari pengambilan data klasifikasi pada pengujian ini. Penulis hanya menampilkan beberapa baris saja, karena banyaknya jumlah data yang terekam. Pada tabel di atas menunjukkan bahwa sistem dapat melakukan penentuan lokasi dan mencatat hasil penentuan lokasi bersama dengan waktu penentuan lokasi, nama lokasi sebenarnya, sub lokasi dan kecocokan. Nilai kecocokan ditentukan berdasarkan pada kecocokan antara hasil penentuan lokasi dengan nama lokasi sebenarnya. Pada tabel di atas menunjukkan bahwa implementasi pengujian yang dilakukan telah sesuai dengan perancangan pada bab 3. Langkah selanjutnya adalah mengukur akurasi kesalahan sistem pada tiap sub lokasi masing-masing ruangan.

5.4.1 Hasil perhitungan dan Analisa Akurasi Kesalahan Sistem Pada Tiap Sub Lokasi

Pada gambar dibawah ini adalah tangkapan layar dari eksekusi *script* untuk menghitung akurasi kesalahan pada tiap sub lokasi.

Commented [AB10]: Pisah sub bab Hasil Pengujian dan Analisis Hasil Pengujian


```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

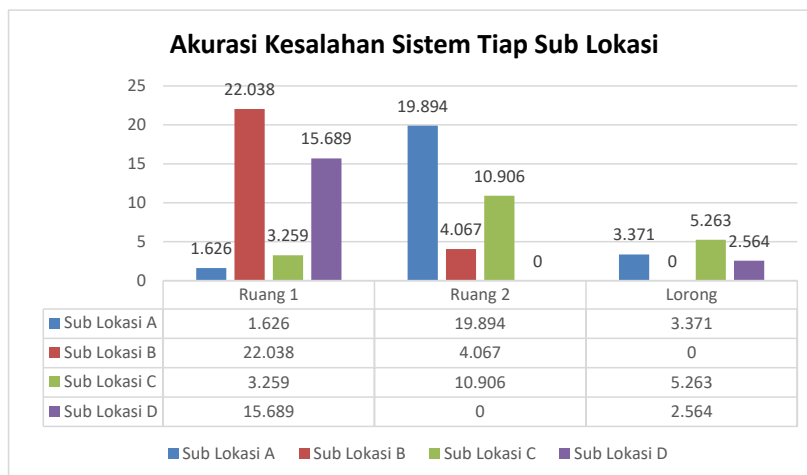
D:\DATA KULIAH\SEMESTER VIII\skripsi-indoorLocalization\CODE\trial_env_2>py akurasi_pengujian_2.py
Kesalahan Prediksi pada Ruang 1 adalah :
    Sub-lokasi A = 1.626 %
    Sub-lokasi B = 22.038 %
    Sub-lokasi C = 3.259 %
    Sub-lokasi D = 15.689 %
Kesalahan Prediksi pada Ruang 2 adalah :
    Sub-lokasi A = 19.894 %
    Sub-lokasi B = 4.067 %
    Sub-lokasi C = 10.906 %
    Sub-lokasi D = 0.000 %
Kesalahan Prediksi pada Lorong adalah :
    Sub-lokasi A = 3.371 %
    Sub-lokasi B = 0.000 %
    Sub-lokasi C = 5.263 %
    Sub-lokasi D = 2.564 %

D:\DATA KULIAH\SEMESTER VIII\skripsi-indoorLocalization\CODE\trial_env_2>

```

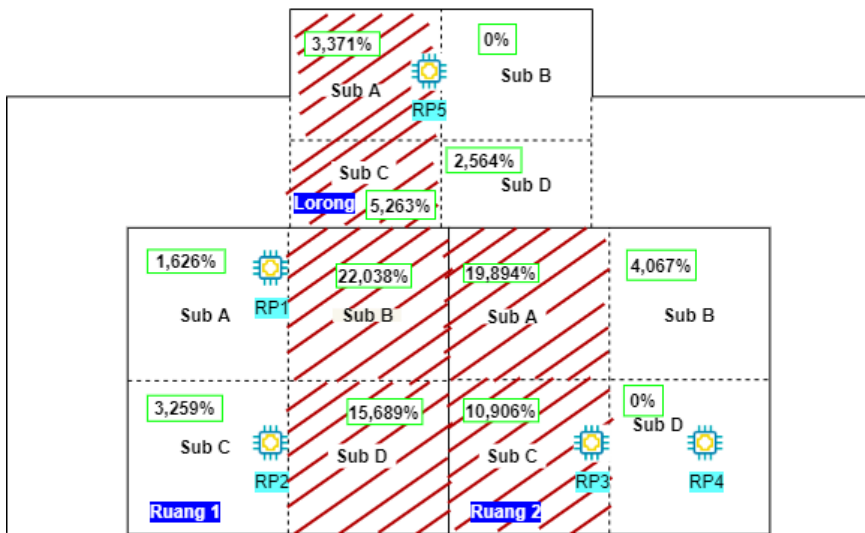
Gambar 5.6 Hasil Perhitungan akurasi sistem pada tiap sub lokasi

Pada Gambar 5.6 di atas dapat terlihat akurasi kesalahan sistem dari masing-masing sub lokasi pada tiap ruangan. Pada grafik di bawah ini disajikan akurasi kesalahan pada tiap sub lokasi.



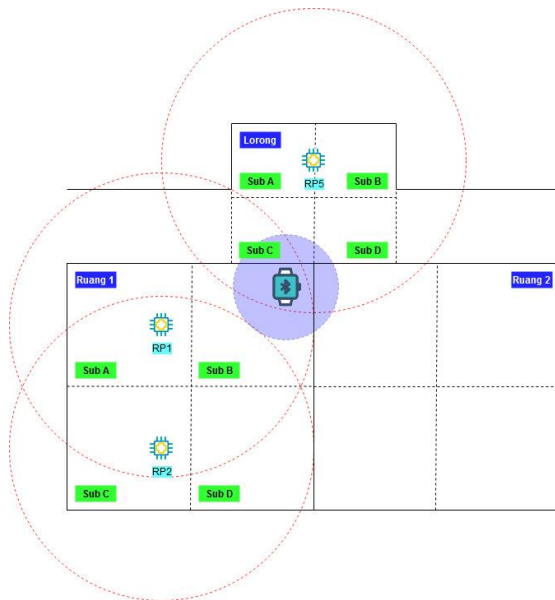
Gambar 5.7 Grafik Akurasi Kesalahan Tiap Sub-Lokasi

5.4.2 Analisa Hasil Perhitungan Akurasi Kesalahan Pada Tiap Sub-Lokasi

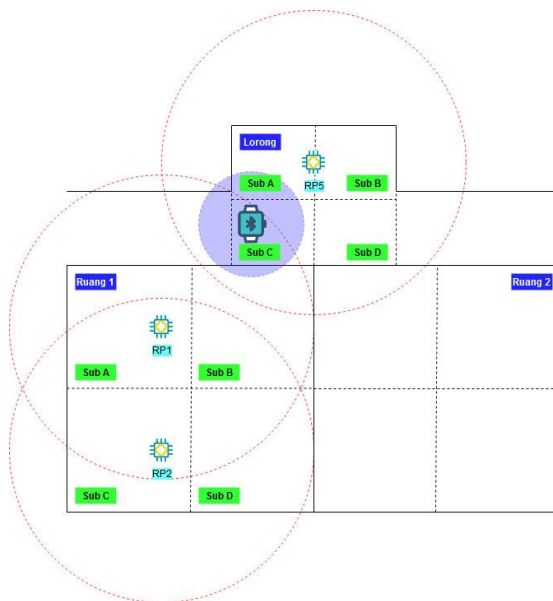


Gambar 5.8 Denah Sub Lokasi Beserta Akurasi Kesalahannya

Pada Gambar 5.7 di atas disajikan grafik dari akurasi kesalahan pada tiap sub lokasi. Sub lokasi yang berdekatan dengan sub lokasi dari lokasi lainnya memiliki tingkat akurasi kesalahan yang cukup tinggi. Pada Gambar 5.8 terlihat bahwa pada sub lokasi B dan D pada Ruang 1 memiliki akurasi kesalahan tertinggi pada ruangan tersebut. Pada Lorong, sub lokasi A dan C memiliki akurasi kesalahan tertinggi pada lokasi tersebut. Pada Ruang 2, sub lokasi A dan C memiliki akurasi kesalahan tertinggi pada lokasi tersebut. Gambar di atas menunjukkan bahwa pada sub-sub lokasi tersebut, terdapat kesalahan klasifikasi yang cukup tinggi. Hal ini dapat terjadi karena sub-sub lokasi tersebut memiliki pola sinyal/*fingerprint* yang cenderung sama dengan sub lokasi yang berdekatan pada ruang yang berbeda. Sebagai contoh, posisi *passive tag* berada pada sub lokasi B pada Ruang 1 namun sistem mendeteksi *passive tag* pada lorong. Hal ini bisa terjadi karena pola sinyal yang terdapat pada sub lokasi B Ruang 1 memiliki kemiripan dengan sub lokasi C Lorong.



Gambar 5.9 Jangkauan Sinyal saat *passive tag* di Ruang 1



Gambar 5.10 Jangkauan Sinyal saat *passive tag* di Lorong

Pada Gambar 5.8 dan Gambar 5.9 menunjukkan posisi dari *passive tag* yang berdekatan. Posisi *passive tag* pada saat berada pada sub lokasi B Ruang 1 dan sub lokasi C pada Lorong berada pada jangkauan terjauh dari *reference points* pada Ruang 1 dan juga berada posisi terjauh dari jangkauan sinyal *reference point* pada Lorong. Hal ini bisa saja menyebabkan pola sinyal yang diterima oleh server memiliki kemiripan karakteristik antara kedua lokasi sehingga dapat terjadi kesalahan saat melakukan klasifikasi. Untuk membuktikannya, penulis mencoba meninjau file data *training fingerprint* yang telah dikumpulkan pada tahap *offline*. Pada Tabel di bawah ini merupakan contoh dari *fingerprint* yang hampir sama.

Tabel 5.5 Fingerprint yang memiliki kemiripan

| Nama Lokasi | RP 1 | RP 2 | RP 3 | RP 4 | RP 5 | Baris Data ke- |
|-------------|------|------|------|------|------|----------------|
| Ruang 1 | -120 | -87 | -120 | -120 | -120 | 9 |
| | -120 | -91 | -120 | -120 | -120 | 67 |
| | -120 | -89 | -120 | -120 | -120 | 115 |
| Lorong | -120 | -90 | -120 | -120 | -120 | 255 |
| | -120 | -92 | -120 | -120 | -95 | 245 |
| | -120 | -94 | -120 | -120 | -120 | 319 |

Pada Tabel 5.5 di atas pada kolom pertama dari kiri terdapat nama lokasi, yakni lokasi dimana *fingerprint* tersebut direkam. Kemudian, pada kolom selanjutnya, yakni kolom RP 1 – RP 5 yang merupakan kolom untuk RSSI dari *passive tag* yang diperoleh dari masing-masing *reference point*. Pada kolom paling pojok kanan terdapat kolom yang menjelaskan pada baris data ke berapa pada *data training* data *fingerprint* yang ditampilkan di atas diperoleh.

Pada Tabel 5.5 tersebut dapat dilihat bahwa *fingerprint* pada Ruang 1 dan Lorong memiliki kemiripan. Bahkan RSSI dari *passive tag* yang diperoleh dari RP 1, dan RP 3 – RP 5 memiliki nilai yang sama persis (kecuali pada baris kedua lokasi Lorong). Sedangkan RSSI *passive tag* dari RP 2 hanya memiliki selisih nilai yang sedikit antar lokasi. Maka dapat disimpulkan bahwa kesalahan dalam penentuan lokasi dapat disebabkan karena adanya kemiripan *fingerprint* pada lokasi yang berbeda.

BAB 6 PENUTUP

6.1 Kesimpulan

Berdasarkan hasil dan analisa dari tahap perancangan, implementasi, dan pengujian yang telah dilakukan, maka untuk menjawab pertanyaan pada rumusan masalah, dapat disimpulkan bahwa :

1. Penentuan lokasi dalam gedung menggunakan metode *fingerprinting* dan Bluetooth Low Energy dapat diimplementasikan dengan menerapkan 2 tahap, yakni tahap *offline* dan tahap *online*. Pada tahap *offline* dilakukan pengumpulan data pola sinyal bluetooth pada masing-masing lokasi untuk membentuk sebuah *data training*. Kemudian, pada tahap *online* dilakukan penentuan lokasi dari perangkat bluetooth sesuai dengan perancangan yang telah dibuat.
2. Implementasi *Monitor Based Localization* pada penentuan lokasi dalam gedung dapat dilakukan dengan menggunakan 3 komponen. Komponen pertama adalah *passive tag* yang berfungsi sebagai perangkat yang akan ditentukan lokasinya. Perangkat ini hanya memancarkan sinyal bluetooth secara terus-menerus. Komponen kedua adalah *reference points* yang berfungsi untuk menangkap sinyal bluetooth yang dipancarkan oleh *passive tag* dan mengirimkannya menuju server. Komponen ketiga adalah server yang berfungsi untuk mengubah kumpulan pola sinyal yang dikirim oleh *reference points* menjadi sebuah nama lokasi. Perangkat ini juga berfungsi sebagai penyimpan *data training* yang berisi kumpulan pola sinyal dari masing-masing lokasi.
3. Tingkat akurasi dari sistem penentuan lokasi dalam gedung ini sebesar 84,76% dengan akurasi kesalahan sebesar 15,24%. Tingkat akurasi pada tiap sub lokasi dari suatu ruangan memiliki tingkat yang bervariasi. Tingkat akurasi kesalahan penentuan lokasi pada suatu sub lokasi yang berdekatan pada lokasi yang berbeda memiliki tingkat yang cukup tinggi dibanding pada sub lokasi yang lain. Hal ini dapat terjadi karena adanya kemiripan karakteristik pola sinyal bluetooth pada sub lokasi - sub lokasi tersebut.

6.2 Saran

Setelah menyelesaikan penelitian ini, ada beberapa saran yang dapat penulis sampaikan untuk mengembangkan penelitian dengan tema yang sama :

1. Implementasi penentuan lokasi ini dapat dikembangkan dengan menambah fitur untuk mencari perangkat berdasarkan inputan dari user.
2. Penentuan lokasi dalam gedung ini dapat diperbaiki pada sisi akurasi dengan mengembangkan teknik dan metode yang digunakan untuk menentukan lokasi saat posisi perangkat yang dilacak berada pada sub lokasi yang berdekatan dengan sub lokasi dari ruangan lain.

