

**PENGEMBANGAN ALGORITMA KOMUNIKASI  
ANTAR-UNMANNED AERIAL VEHICLE  
BERBASIS *PAINLESSMESH* PADA  
MIKROKONTROLER ESP32**

***DEVELOPMENT OF AN INTER-UNMANNED  
AERIAL VEHICLE COMMUNICATIONS  
ALGORITHM BASED ON *PAINLESSMESH* USING  
AN ESP32 MICROCONTROLLER***

**TUGAS AKHIR**

Disusun sebagai syarat mata kuliah Tugas Akhir  
Program Studi S1 Teknik Elektro

Disusun oleh:  
**MUHAMMAD ADEEL MAHDI SUIYANTO**  
**1102183191**



**Universitas  
Telkom**

**FAKULTAS TEKNIK ELEKTRO  
UNIVERSITAS TELKOM  
BANDUNG  
2022**

**LEMBAR PENGESAHAN**

**TUGAS AKHIR**

**PENGEMBANGAN ALGORITMA KOMUNIKASI  
ANTAR-UNMANNED AERIAL VEHICLE  
BERBASIS *PAINLESSMESH* PADA  
MIKROKONTROLER ESP32**

***DEVELOPMENT OF AN INTER-UNMANNED  
AERIAL VEHICLE COMMUNICATIONS  
ALGORITHM BASED ON *PAINLESSMESH* USING  
AN ESP32 MICROCONTROLLER***

**Telah disetujui dan disahkan sebagai Buku Tugas Akhir**

**Program Studi S1 Teknik Elektro**

**Fakultas Teknik Elektro**

**Telkom University**

**Disusun oleh:**

**MUHAMMAD ADEEL MAHDI SUIYANTO**

**1102183191**

**Bandung, x Agustus 2022**

Pembimbing 1	Pembimbing 2
Dr. Eng. Willy Anugrah Cahyadi, S.T, M.T.	Ir. Uke Kurniawan Usman,M.T.

## LEMBAR PERNYATAAN ORISINALITAS

Nama : Muhammad Adeel Mahdi Suviyanto  
NIM : 1102183191  
Alamat : Taman Alfa Indah D4/11, Joglo, Kembangan, Jakarta Barat  
No Tlp/HP : 021 73443989 / 087777882699  
Email : adeelsuviyanto@student.telkomuniversity.ac.id

Menyatakan bahwa Tugas Akhir ini merupakan karya orisinal saya sendiri dengan judul:

**Pengembangan Algoritma Komunikasi Antar-Unmanned Aerial Vehicle Berbasis *PainlessMesh* Pada Mikrokontroler ESP32**

***Development of an Inter-Unmanned Aerial Vehicle Communications Algorithm based on *PainlessMesh* using an ESP32 Microcontroller***

Atas pernyataan ini, saya siap menanggung risiko/sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap kejujuran akademik atau etika keilmuan dalam karya ini, atau ditemukan bukti yang menunjukkan ketidak aslian karya ini.



Bandung, 25 Agustus 2022



Muhammad Adeel Mahdi Suviyanto  
1102183191

## ABSTRAK

Salah satu faktor penentu kesuksesan dalam operasi *Search and Rescue* (SAR) pada bencana alam adalah kecepatan dalam menemukan lokasi dan posisi korban serta pengiriman logistik bantuan penunjang hidup bagi korban tersebut. Penggunaan *Unmanned Aerial Vehicle* dapat mendukung operasi SAR, dan untuk meningkatkan koordinasi antar UAV dalam sistem maka dikembangkan sebuah algoritma komunikasi antar-UAV. Model komunikasi yang dikembangkan berupa *Flying Ad-hoc Network* (FANET), dan teknologi komunikasi yang dipilih berupa Wi-Fi menggunakan mikrokontroler ESP32 dan *library PainlessMesh*. Kinerja jaringan diuji dari nilai *throughput*, *packet loss*, dan *round-trip delay*. Penelitian ini menghasilkan sebuah jaringan mesh dengan jarak antar drone hingga 30 meter, mampu mengirimkan data lokasi dari drone *sender* ke drone *receiver* dengan nilai *packet loss* maksimum sebesar 89 persen, *throughput* jaringan maksimum sebesar 813,173 B/s, dan *round-trip delay* maksimum sebesar 1536 ms. Analisis regresi linear menunjukkan korelasi antara *throughput*, *round-trip delay*, dan *packet loss* terhadap *signal strength* antar node, dimana nilai *signal strength* yang semakin mengecil menghasilkan kinerja jaringan yang menurun.

**Kata kunci:** PainlessMesh, ESP32, *Unmanned Aerial Vehicles*, *Drone*, Komunikasi Jaringan

## ABSTRACT

One of the defining factors in deciding the success of a Search and Rescue (SAR) operation during a natural disaster is in how quickly victims can be found and delivering life-sustaining logistics for said victim. The usage of Unmanned Aerial Vehicles can support SAR operations, and to improve the coordination between UAVs, an inter-UAV communications algorithm is developed. The type of communication system developed is a Flying Ad-hoc Network (FANET), and the communications technology chosen to build the FANET is Wi-Fi using ESP32 microcontrollers and the PainlessMesh library. This research built a mesh network of ESP32 nodes, with each drones being placed 30 meters apart, in which it is able to send GPS location data from the sender drone to the receiver drone with a maximum packet loss of 89 percent, maximum throughput of 813,713 B/s, and maximum round-trip delay of 1536 ms. Based on linear regression analysis, there is a correlation between throughput, round-trip delay, and packet loss values to the signal strength between nodes, with a lower signal strength between nodes resulting in lowered network performance.

**Keywords:** PainlessMesh, ESP32, Unmanned Aerial Vehicles, Drone, Network Communications

## KATA PENGANTAR

Placeholder text for Abstract.

## **UCAPAN TERIMA KASIH**

Placeholder text for Abstract.

## DAFTAR ISI

<b>LEMBAR PENGESAHAN . . . . .</b>	<b>i</b>
<b>ABSTRAK . . . . .</b>	<b>iii</b>
<b>ABSTRACT . . . . .</b>	<b>iv</b>
<b>KATA PENGANTAR . . . . .</b>	<b>v</b>
<b>UCAPAN TERIMA KASIH . . . . .</b>	<b>vi</b>
<b>DAFTAR ISI . . . . .</b>	<b>vii</b>
<b>DAFTAR GAMBAR . . . . .</b>	<b>x</b>
<b>DAFTAR TABEL . . . . .</b>	<b>xiii</b>
<b>DAFTAR SINGKATAN . . . . .</b>	<b>xiv</b>
<b>I PENDAHULUAN . . . . .</b>	<b>1</b>
1.1 Latar Belakang Masalah . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan dan Manfaat . . . . .	2
1.4 Batasan Masalah . . . . .	3
1.5 Metode Penelitian . . . . .	4
1.6 Jadwal Pelaksanaan . . . . .	4
1.7 Sistematika Penulisan . . . . .	5
<b>II TINJAUAN PUSTAKA DAN KONSEP DASAR SISTEM . . . . .</b>	<b>7</b>
2.1 Desain Konsep Sistem . . . . .	7
2.2 Riset Terkait . . . . .	7
2.3 <i>Unmanned Aerial Vehicles (UAV)</i> . . . . .	10
2.4 <i>Quadcopter</i> . . . . .	10
2.5 Jaringan Nirkabel . . . . .	11
2.5.1 <i>Wireless Wide Area Network (WWAN)</i> . . . . .	11
2.5.2 <i>Wireless Local Area Network (WLAN)</i> . . . . .	11
2.5.3 <i>Wireless Personal Area Network (WPAN)</i> . . . . .	11
2.6 <i>Flying Ad-hoc Network</i> . . . . .	11
2.6.1 Bluetooth untuk penggunaan FANET . . . . .	11
2.6.2 Jaringan Seluler untuk penggunaan FANET . . . . .	12
2.6.3 Wi-Fi untuk penggunaan FANET . . . . .	12

2.7	Jaringan <i>mesh</i> . . . . .	12
2.8	ESP32 . . . . .	12
2.9	PainlessMesh . . . . .	13
2.9.1	Protokol PainlessMesh . . . . .	14
2.9.2	JavaScript Object Notation (JSON) . . . . .	14
2.10	<i>Global Positioning System</i> (GPS) . . . . .	15
2.11	GPS NMEA Data . . . . .	15
2.12	Pengukuran Kinerja Jaringan . . . . .	16
2.12.1	Round-trip delay . . . . .	16
2.12.2	Throughput . . . . .	16
2.12.3	Packet loss . . . . .	16
<b>III</b>	<b>PERANCANGAN SISTEM</b> . . . . .	<b>18</b>
3.1	Desain Sistem . . . . .	18
3.1.1	Prinsip Kerja Sistem . . . . .	18
3.1.2	Diagram Blok . . . . .	19
3.1.3	Fungsi dan Fitur . . . . .	20
3.2	Desain Perangkat Keras . . . . .	20
3.3	Desain Perangkat Lunak . . . . .	27
3.3.1	Sender Node . . . . .	27
3.3.2	Flying Receiver Node . . . . .	29
<b>IV</b>	<b>HASIL DAN ANALISIS</b> . . . . .	<b>32</b>
4.1	Pendahuluan . . . . .	32
4.1.1	Pelaksanaan Pengujian Sistem . . . . .	32
4.1.2	Kondisi Lingkungan Pengujian . . . . .	34
4.1.3	Desain Alat . . . . .	36
4.2	Skenario Pengujian . . . . .	39
4.2.1	Pengujian Tanpa Terbang . . . . .	39
4.2.2	Pengujian Terbang Satu Drone . . . . .	42
4.2.3	Pengujian Terbang Dua Drone . . . . .	44
4.3	Realisasi Algoritma Sistem . . . . .	45
4.4	Hasil dan Analisis Pengujian Skenario Komunikasi Dua Drone . . . . .	47
4.5	Hasil Pengujian . . . . .	50
4.5.1	Pengujian Tanpa Terbang . . . . .	50
4.5.2	Pengujian Terbang Satu Drone . . . . .	58
4.6	Analisis Hasil Pengujian Keseluruhan . . . . .	62

<b>V KESIMPULAN DAN SARAN . . . . .</b>	<b>65</b>
5.1 Kesimpulan . . . . .	65
5.2 Saran . . . . .	65

## DAFTAR GAMBAR

2.1	Desain konsep sistem . . . . .	7
2.2	Beberapa tipe-tipe UAV [13] . . . . .	10
2.3	Salah satu contoh <i>quadcopter</i> . . . . .	10
2.4	Bentuk topologi jaringan <i>mesh</i> penuh dan parsial . . . . .	13
2.5	ESP32 . . . . .	13
2.6	Topologi jaringan PainlessMesh [20], panah menunjukkan arah koneksi dari klien ke AP . . . . .	14
2.7	Setiap pesan pada jaringan PainlessMesh menggunakan JSON. . . . .	15
2.8	Contoh data mentah dari modul GPS NEO-6M berupa data GPS NMEA. . . . .	15
2.9	Skema pengujian <i>round-trip delay</i> dan <i>packet loss</i> . . . . .	17
3.1	Diagram implementasi sistem. . . . .	18
3.2	Diagram blok sistem. . . . .	19
3.3	Board Ai-Thinker NodeMCU-32S . . . . .	21
3.4	Board DOIT-ESP32-DEVKIT . . . . .	22
3.5	Board Ai-Thinker NodeMCU ESP-12S . . . . .	24
3.6	Modul GPS NEO-6M . . . . .	25
3.7	Drone MJX Bugs 5W . . . . .	26
3.8	Diagram alur sender node. . . . .	27
3.9	Diagram alur node flying receiver, program penarikan data lokasi dari node sender. . . . .	29
3.10	Diagram alur node flying receiver, program pengujian jaringan. . . . .	30
4.1	Penempatan node jaringan pada pengujian non-terbang di Gedung N FTE Telkom University. . . . .	32
4.2	Penempatan node jaringan pada pengujian terbang satu drone di Lapangan BTP Telkom University. . . . .	33
4.3	Penempatan node jaringan pada pengujian terbang dua drone di Lapangan BTP Telkom University. . . . .	34
4.4	Hasil analisis spektrum Wi-Fi 2.4 GHz di Lapangan BTP Telkom University. . . . .	34
4.5	Hasil analisis spektrum Wi-Fi 2.4 GHz di Gedung N Telkom University. . . . .	35
4.6	Penampakan implementasi <i>sender</i> node di drone MJX Bugs 5W. . . . .	36
4.7	Penampakan implementasi <i>sender</i> node di drone MJX Bugs 5W. . . . .	37

4.8	Implementasi <i>Base Station Receiver</i> menggunakan <i>development board</i> Ai-Thinker NodeMCU ESP8266 dan DOIT-ESP32-DEVKIT . . . . .	38
4.9	Penempatan <i>sender</i> node pada pengujian tanpa terbang, di depan ruangan N304. . . . .	39
4.10	Penempatan <i>base station</i> pada pengujian tanpa terbang, di depan ruangan N308. . . . .	39
4.11	Penempatan <i>flying receiver</i> node pada pengujian tanpa terbang, di depan ruangan N314. . . . .	39
4.12	Lingkaran merah menunjukkan posisi <i>flying receiver</i> node dari <i>sender</i> , dengan <i>line-of-sight</i> yang jelas. . . . .	41
4.13	Lingkaran merah menunjukkan posisi <i>sender</i> node dari <i>flying receiver</i> , dengan <i>line-of-sight</i> yang jelas. . . . .	41
4.14	Lingkaran merah menunjukkan posisi <i>base station</i> dari <i>flying receiver</i> , terlihat bahwa <i>line-of-sight</i> tertutup oleh pohon. . . . .	41
4.15	Pengujian satu drone, jarak 30 meter dan ketinggian dari 4 meter ke 10 meter. . . . .	42
4.16	Jarak drone dengan penguji sekitar 32 meter. . . . .	43
4.17	Penempatan node <i>flying receiver</i> di topi penguji. Foto diambil sebelum node <i>flying receiver</i> dinyalakan. . . . .	43
4.18	Pengujian dua drone, jarak antar-drone 30 meter dan ketinggian dari 4 meter ke 10 meter. . . . .	44
4.19	Proses koneksi awal di node <i>receiver</i> . . . . .	45
4.20	Proses koneksi awal di node <i>sender</i> . . . . .	45
4.21	Permintaan status GPS kepada <i>sender</i> oleh <i>receiver</i> . . . . .	45
4.22	Balasan status GPS kepada <i>receiver</i> . . . . .	45
4.23	Permintaan data lokasi kepada <i>sender</i> oleh <i>receiver</i> . . . . .	46
4.24	Balasan data lokasi kepada <i>receiver</i> . . . . .	46
4.25	Data lokasi yang dikirimkan oleh <i>sender</i> . . . . .	46
4.26	Tampilan awal laman web <i>base station</i> . . . . .	46
4.27	Tampilan laman web <i>base station</i> setelah melakukan koneksi ke <i>sender</i> beserta meminta status GPS. . . . .	47
4.28	Grafik <i>throughput</i> dan RSSI terhadap waktu, mode LR . . . . .	47
4.29	Grafik <i>throughput</i> terhadap RSSI, mode LR . . . . .	48
4.30	Grafik <i>round-trip delay</i> dan RSSI terhadap waktu, mode LR . . . . .	49
4.31	Grafik <i>packet loss</i> dan RSSI terhadap waktu, mode LR . . . . .	49
4.32	Grafik <i>throughput</i> dan RSSI terhadap waktu, mode LR . . . . .	50
4.33	Grafik <i>throughput</i> dan jumlah node terhadap waktu, mode LR . . . . .	51

4.34 Grafik <i>throughput</i> terhadap <i>Receiver RSSI</i> , mode LR . . . . .	51
4.35 Grafik <i>round-trip delay</i> dan <i>Receiver RSSI</i> terhadap waktu, mode LR . . . . .	52
4.36 Grafik <i>round-trip delay</i> dan jumlah node terhadap waktu, mode LR . . . . .	53
4.37 Grafik <i>round-trip delay</i> terhadap <i>Receiver RSSI</i> , mode LR . . . . .	53
4.38 Grafik <i>packet loss</i> terhadap waktu, mode LR . . . . .	54
4.39 Grafik <i>throughput</i> dan <i>RSSI</i> terhadap waktu, mode 802.11n . . . . .	54
4.40 Grafik <i>throughput</i> terhadap <i>RSSI</i> , mode 802.11n . . . . .	55
4.41 Grafik <i>round-trip delay</i> dan <i>Receiver RSSI</i> terhadap waktu, mode 802.11n . . . . .	56
4.42 Grafik <i>round-trip delay</i> terhadap <i>RSSI</i> , mode 802.11n . . . . .	56
4.43 Grafik <i>packet loss</i> terhadap waktu, mode LR . . . . .	57
4.44 Grafik <i>throughput</i> dan <i>RSSI</i> terhadap waktu, mode LR . . . . .	58
4.45 Grafik <i>throughput</i> terhadap <i>Receiver RSSI</i> , mode LR . . . . .	59
4.46 Grafik <i>round-trip delay</i> dan <i>RSSI</i> terhadap waktu, mode LR . . . . .	59
4.47 Grafik <i>round-trip delay</i> terhadap <i>Receiver RSSI</i> , mode LR . . . . .	60
4.48 Grafik <i>packet loss</i> terhadap waktu, mode LR . . . . .	60
4.49 Grafik <i>throughput</i> dan <i>RSSI</i> terhadap waktu, mode 802.11n . . . . .	61
4.50 Grafik <i>round-trip delay</i> dan <i>RSSI</i> terhadap waktu, mode 802.11n . . . . .	62
4.51 Grafik <i>throughput</i> setiap pengujian terhadap <i>RSSI</i> , mode LR. . . . .	62
4.52 Grafik <i>round-trip delay</i> setiap pengujian terhadap <i>RSSI</i> , mode LR. . . . .	63

## **DAFTAR TABEL**

1.1	Jadwal pelaksanaan penelitian. . . . .	4
2.1	Penelitian terkait . . . . .	8
3.1	Spesifikasi Ai-Thinker NodeMCU-32S . . . . .	21
3.3	Spesifikasi DOIT-ESP32-DEVKIT . . . . .	22
3.5	Spesifikasi Ai-Thinker NodeMCU ESP-12S . . . . .	24
3.7	Spesifikasi DOIT-ESP32-DEVKIT . . . . .	25
3.9	Spesifikasi drone MJX Bugs 5W . . . . .	26
4.1	Hasil parameter kinerja jaringan dari seluruh skenario pengujian . .	63

## **DAFTAR SINGKATAN**

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang Masalah

Salah satu faktor penentu kesuksesan dalam operasi *Search and Rescue* (SAR) pada bencana alam adalah kecepatan dalam menemukan lokasi dan posisi korban serta pengiriman logistik bantuan penunjang hidup bagi korban tersebut. Namun, kondisi daratan medan bencana alam yang sukar dilewati oleh tim penyelamat dapat menyebabkan lamanya kedua proses tersebut, menurunkan probabilitas keselamatan bagi korban [1]. Pada saat ini, metode yang biasa digunakan untuk pencarian korban adalah menggunakan helikopter dengan pencarian manual dari udara. Akan tetapi, penggunaan helikopter memiliki kelemahan pada sisi biaya operasi serta perubahan cuaca, dimana penerbangan helikopter yang aman hanya dapat dilakukan pada kondisi cuaca cerah tidak berkabut [2]. Oleh karena itu, penggunaan *Unmanned Aerial Vehicles* (UAV) untuk kepentingan SAR dapat meningkat sebagai pendukung terhadap penggunaan helikopter pada operasi SAR.

Menurut (Lakshmi Narayanan, 2015), UAV adalah sebuah tipe pesawat terbang yang dapat mengudara tanpa adanya awak di atas kapal [3]. Terdapat berbagai kegunaan sebuah UAV, salah satunya adalah operasi SAR. UAV memiliki keunggulan yang cocok bagi operasi SAR, yakni kemampuannya untuk melihat suatu area yang luas dengan akses yang cepat tanpa terhalang oleh medan bencana [4]. UAV juga unggul dalam menghadapi cuaca buruk, dimana cuaca berkabut dapat menyebabkan helikopter tidak dapat terbang karena alasan keamanan, sedangkan UAV tetap dapat terbang karena tidak ada personil yang dibahayakan pada kondisi tersebut. Pada lokasi bencana alam, akses medan bencana yang sulit dapat mempersulit operasi SAR, terutama pada pencarian korban dan pengiriman bantuan.

Oleh karena itu, penelitian ini mengusulkan suatu pengembangan pada sistem komunikasi antar-UAV yang kemudian dapat digunakan pada operasi SAR, dimana data yang dikirimkan berupa koordinat lokasi salah satu unit UAV dalam jaringan. Untuk mewujudkan koordinasi antar masing-masing unit UAV, maka diperlukan sistem komunikasi antar-UAV yang memenuhi kriteria kinerja minimum: *throughput* (laju pengiriman data) minimum 16 KBps, *packet loss* (persentase paket data yang hilang dalam transmisi) dibawah 25%, jarak minimum antar unit UAV minimum 25 meter, dan *round-trip delay* (waktu tempuh pengiriman data bolak-balik) dibawah 4000 milisekon.

Terdapat beberapa arsitektur komunikasi yang layak untuk penggunaan pada komunikasi antar-UAV, seperti *Flying Ad-Hoc Network* (FANET) [5] dan *Centralized* berbasis teknologi seluler (LTE, 5G) [6]. Namun, ketergantungan teknologi seluler terhadap infrastruktur yang telah ada di darat membuat komunikasi berbasis seluler

kurang sesuai jika digunakan untuk kondisi bencana, karena rusaknya infrastruktur fisik (menara *Base Transceiver Station* (BTS)), disrupti pada infrastruktur penunjang (listrik), dan juga *overload* oleh melonjaknya jumlah pengguna jaringan di waktu yang bersamaan [7]. Oleh karena itu, penulis akan menggunakan teknologi FANET berbasis IEEE 802.11 WiFi dengan menggunakan mikrokontroler ESP32 dalam sebuah jaringan WiFi Mesh.

Mikrokontroler ESP32 digunakan karena harganya yang ekonomis dan telah memiliki kapabilitas WiFi IEEE 802.11 secara *built-in*, dapat diprogram menggunakan bahasa pemrograman Arduino yang berbasiskan C dan C++, serta memiliki dokumentasi dan *plug-in* yang lengkap. ESP32 juga mendukung beragam mode operasi WiFi IEEE 802.11 dari 802.11b/g/n dan mode khusus Espressif yakni 802.11 *Long Range*. Setiap mikrokontroler ESP32 beroperasi pada pita frekuensi 2.4 GHz. Pada sistem yang dirancang, masing-masing unit UAV akan dipasangkan satu unit board ESP32 yang kemudian akan berkomunikasi satu sama lain pada mode WiFi ad-hoc.

Dalam tugas akhir ini, dirancang sebuah algoritma komunikasi antar-UAV berbasis WiFi Mesh pada mikrokontroler ESP32, serta akan menganalisis sistem yang dihasilkan dengan parameter kinerja jaringan (*throughput, packet loss, round-trip delay, signal strength* (RSSI)) terhadap jarak antar unit UAV, dampak sistem penerbangan dan kendali UAV terhadap kinerja sistem, serta mode IEEE 802.11 yang digunakan ESP32 terhadap kinerja sistem. Diharapkan hasil sistem yang diperoleh dapat dijadikan salah satu metode komunikasi antar-UAV pada kegunaan operasi SAR dalam menemukan posisi korban.

## 1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana korelasi antara jarak antar-UAV terhadap kinerja jaringan (*throughput, packet loss, range, round-trip delay*) yang telah diimplementasikan?
2. Apa mode WiFi 802.11 yang cocok digunakan untuk kegunaan sistem komunikasi antar-UAV?
3. Apakah algoritma komunikasi yang dihasilkan dapat diimplementasikan di lokasi medan bencana alam?

## 1.3 Tujuan dan Manfaat

Tujuan dari perancangan algoritma komunikasi antar-UAV ini adalah:

1. Mengetahui korelasi jarak antar-node dan perbedaan ketinggian antar-UAV terhadap kinerja jaringan yang diimplementasikan, dengan parameter kinerja *throughput, packet loss, range, round-trip delay*.

2. Mengetahui korelasi mode WiFi 802.11 yang digunakan pada sistem terhadap kinerja jaringan yang diimplementasikan
3. Mengetahui apakah algoritma yang dihasilkan berguna untuk operasi SAR pada bencana alam.

Adapun manfaat dari penelitian ini adalah:

1. Mengembangkan sebuah algoritma komunikasi antar-UAV yang cukup handal dengan *link quality* tinggi sehingga UAV dapat berkomunikasi satu sama lain untuk mengirimkan data.
2. Sebagai tahap pertama dari pengembangan sistem UAV SAR otonom.
3. Sebagai sumber pustaka bagi penelitian di masa depan mengenai permasalahan terkait.

#### **1.4 Batasan Masalah**

Agar pembahasan dalam penelitian dapat difokuskan, maka terdapat pembatasan masalah sebagai berikut:

1. Menggunakan 2 unit drone dan satu *Base Station* (BS) untuk menyederhanakan sistem rancangan.
2. Data komunikasi antar-UAV yang dikirimkan berupa data koordinat (Lintang dan Bujur) dengan 5 angka desimal untuk tingkat kepresisionan 1 meter [8].
3. Kendali masing-masing drone dilakukan secara terpisah dari sistem komunikasi yang diuji dan dilakukan secara manual oleh operator menggunakan *remote control* (R/C).
4. Parameter yang digunakan pada analisis algoritma jaringan yang diimplementasikan adalah *throughput* jaringan, *packet loss*, *round-trip delay*, dan *signal strength* dalam RSSI (dBm).
5. Pengujian dilakukan di area Gedung N Fakultas Teknik Elektro Telkom University dan lapangan Bandung Techno Park Telkom University, dengan jarak antar node jaringan 30 - 50 meter antara satu sama lain.
6. Pengujian dilakukan dengan kondisi drone OFF dan ON.

## 1.5 Metode Penelitian

Metode penelitian yang digunakan pada tugas akhir ini antara lain:

### 1. Studi Literatur

Studi literatur dilakukan dengan mempelajari beberapa materi yang berkaitan dengan penelitian ini, dengan sumber yang digunakan berupa jurnal, artikel, buku, dan situs web yang terpercaya.

### 2. Perancangan Sistem

Pada tahap ini, dilakukan perancangan sistem sesuai dengan target yang telah ditentukan. Melalui perancangan sistem, dihasilkan suatu gambaran jelas mengenai struktur penyusunan sistem dan dapat dilakukan analisis secara matematis.

### 3. Implementasi

Sistem yang telah dirancang kemudian diimplementasikan melalui perangkaian komponen-komponen yang telah ditentukan, serta melakukan pemrograman sistem tersebut.

### 4. Pengukuran Empiris

Pada tahap ini, sistem yang telah diimplementasikan diuji melalui beberapa tes yang menguji sistemnya secara kuantitatif untuk menghasilkan data empiris yang dapat diolah dalam bentuk grafik.

### 5. Analisis Statistik

Hasil pengukuran kemudian dianalisis berdasarkan teori yang telah dikemukakan, dan menghitung faktor-faktor lainnya seperti keakuratan alat pengukur dan faktor-faktor eksternal yang mempengaruhi kinerja sistem.

## 1.6 Jadwal Pelaksanaan

Berikut adalah jadwal pelaksanaan penelitian ini, rincian waktu dan *milestone* dirangkum dalam tabel di bawah ini:

Tabel 1.1: Jadwal pelaksanaan penelitian.

No.	Deskripsi Tahapan	Durasi	Tanggal Selesai	Milestone
1	Rumusan masalah dan studi literatur	2 Minggu	21 Oktober 2021	Mengidentifikasi permasalahan dan studi literatur.
2	Desain sistem	2 Minggu	29 Oktober 2021	Diagram blok sistem, sketsa dasar sistem, diagram alur sistem, dan spesifikasi alat.
3	Pemilihan komponen	1 Minggu	5 November 2021	Pendataan komponen sistem yang akan digunakan.
4	Perancangan dan pembuatan sistem	1 Bulan	3 Desember 2021	Implementasi sistem secara fisik.
5	Pengujian sistem	2 Minggu	8 Juli 2022	<i>Test flight</i> dan pengujian jaringan sistem.
6	Penyusunan Laporan/Buku TA	2 Minggu	22 Juli 2022	Laporan/Buku TA selesai.

## 1.7 Sistematika Penulisan

Sistematika penulisan yang digunakan pada tugas akhir ini adalah:

### BAB I: PENDAHULUAN

Bab ini berisi uraian singkat mengenai latar belakang permasalahan, rumusan masalah, tujuan dan manfaat, pembatasan masalah, serta jadwal pelaksanaan penelitian.

### BAB II: TINJAUAN PUSTAKA DAN KONSEP DASAR SISTEM

Bab ini berisi uraian mengenai landasan teori serta membahas konsep dasar sistem yang dibahas dalam tugas akhir ini.

### BAB III: PERANCANGAN SISTEM

Bab ini berisi uraian mengenai rancangan sistem dari sisi desain perangkat keras maupun perangkat lunak, fungsi dan fitur, serta spesifikasi sistem.

### BAB IV: HASIL DAN ANALISIS

Bab ini berisi uraian mengenai hasil pengujian sistem, serta analisis dari hasil pengujian tersebut secara rinci terhadap parameter yang sudah ditentukan.

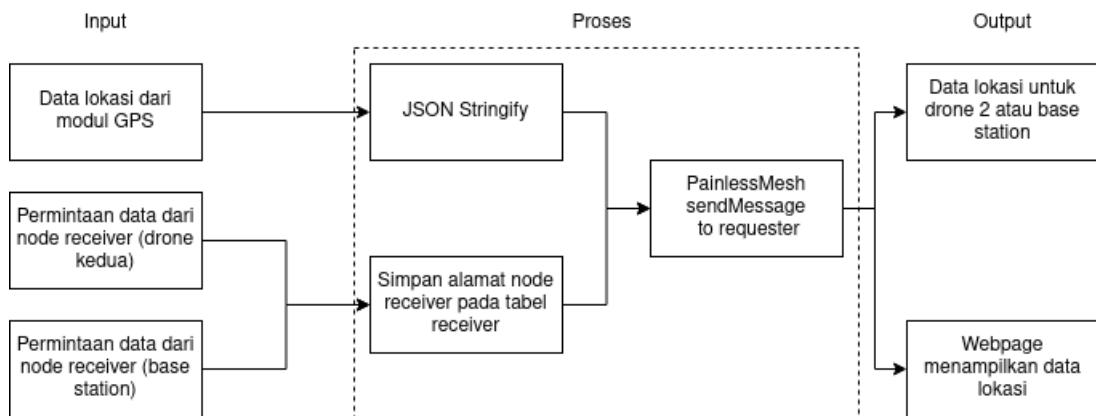
**BAB V: SIMPULAN DAN SARAN**

Bab ini berisi rincian kesimpulan dari penelitian yang telah dikerjakan, serta saran untuk penelitian berikutnya.

## BAB II

### TINJAUAN PUSTAKA DAN KONSEP DASAR SISTEM

#### 2.1 Desain Konsep Sistem



Gambar 2.1: Desain konsep sistem

Gambar 2.1 menunjukkan desain konsep sistem secara dasar, dengan masukan sistem berupa data lokasi dari GPS dan permintaan data dari *receiver*, proses data berupa pembuatan sebuah data JSON dari data lokasi GPS serta pengiriman data ke *receiver*, dan keluaran berupa data lokasi yang diterima *receiver*. Kemudian dari setiap data yang dikirimkan, akan dihitung besar *throughput*, *packet loss*, dan *round-trip delay* jaringan serta mendapatkan besar *signal strength* antar node jaringan.

#### 2.2 Riset Terkait

Terdapat beberapa penelitian yang telah dilaksanakan sebelumnya yang terkait dengan tugas akhir ini, yang akan digunakan sebagai dasar atau referensi dalam pengerjaan. Beberapa penelitian terkait dapat dilihat pada tabel 2.1.

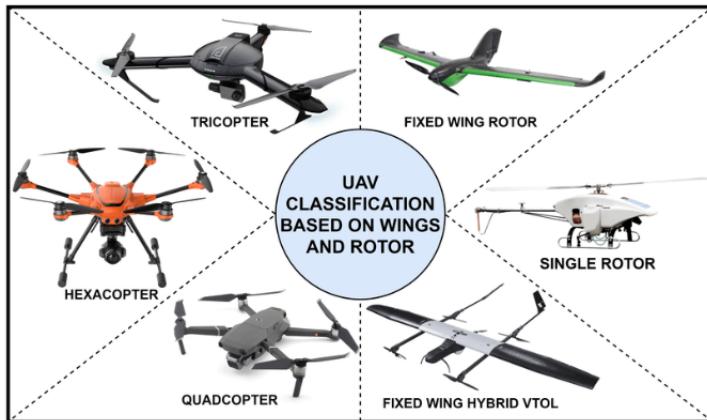
Tabel 2.1: Penelitian terkait

No.	Judul	Metode	Kesimpulan	Kelebihan	Kekurangan
[9]	Use of High Mobility Nodes to Improve Connectivity in Wireless Sensor Networks	PainlessMesh-based Opportunistic Mobile Ad Hoc Networks	Menggunakan high mobility nodes berupa drone sebagai messenger data antara cluster sensor dengan server	Penelitian mengimplementasikan metode security berbasis secret-key cryptography. Packet loss antara sensor dan server rendah, hanya 4,48% pada kondisi terburuk walaupun packet melewati 2 hop.	Tidak menguji round-trip delay packet
[10]	Performance Evaluation of ESP8266 Mesh Networks	Pengujian one-way delay dan data rate pada jaringan PainlessMesh ESP8266	Jumlah node berkorelasi dengan meningkatnya single hop delay dan menurunnya stabilitas jaringan, serta besar payload menentukan data rate dan korupsi data.	Penelitian menguji dari 2 hingga 16 node sehingga terlihat gambaran kasar stabilitas jaringan PainlessMesh. Kinerja jaringan 2 node memiliki delay 2.49 ms sehingga cukup untuk aplikasi yang tidak terlalu kompleks.	ESP8266 tidak memiliki performa yang cukup untuk mengirimkan dan menerima payload data yang besar. Pengujian data rate dibatasi pada 2 node.

[11]	Implementing Wireless Mesh Network Topology Between Multiple Wi-Fi Powered Nodes for IoT Systems	Penggunaan 3 node PainlessMesh berbasis ESP32 untuk komunikasi multidirectional output sensor dan input push button	Mesh bersifat self-healing, pada saat terjadi disrupsi maka mesh secara otomatis mengatur diri.	Implementasi 3 node dan arah data bolak balik dapat dilakukan	Tidak ada pengujian jarak jauh
[12]	A dust sensor monitoring system using Wi-Fi mesh network	Implementasi 9 node berbasis ESP32 dan ESP-Mesh untuk monitoring tingkat debu pada suatu ruangan.	Sistem efektif dengan measurement error dibawah 5%	Mesh dapat berfungsi tanpa intervensi manusia, memiliki sifat self-healing dan auto-configuration.	Jaringan memiliki topologi tree, bukan mesh murni, sehingga jika terjadi gangguan pada node akar maka proses self-heal berjalan lebih lama.

Berdasarkan penelitian yang telah dilakukan sebelumnya, maka pada penelitian tugas akhir ini menggunakan 3 node jaringan, masing-masing berupa board mikro-kontroler ESP32. 2 node ditempatkan pada UAV/drone quadcopter, dimana 1 node memiliki modul GPS sebagai pengirim data lokasi dan 1 node memiliki board MicroSD untuk *data logging* serta sebagai penerima data lokasi dari pengirim. Node terakhir berada di darat dan berfungsi sebagai penerima data lokasi dari node pengirim serta menampilkan data tersebut kepada pengguna pada sebuah *Web server*.

### 2.3 Unmanned Aerial Vehicles (UAV)



Gambar 2.2: Beberapa tipe-tipe UAV [13]

UAV adalah sebuah pesawat terbang yang dapat mengudara tanpa awak [3], dikendalikan secara *remote* atau secara otonom. Salah satu tipe UAV adalah *quadcopter drone*. Pada sistem ini, UAV digunakan sebagai pengirim data lokasi GPS sebagai node *sender*, dan penerima data lokasi GPS sebagai node *receiver*.

### 2.4 Quadcopter



Gambar 2.3: Salah satu contoh *quadcopter*.

Drone *quadcopter* adalah suatu jenis UAV yang memiliki 4 rotor pada masing-masing sudut. Sama seperti helikopter, *quadcopter* memiliki kemampuan untuk *hover*. Terdapat sepasang rotor yang berputar searah jarum jam dan sepasang rotor yang berputar berlawanan arah jarum jam, sehingga pada kondisi *steady state* total torsi pada *drone* adalah nol. Hal tersebut juga menyebabkan konfigurasi *quadcopter* tidak membutuhkan *tail rotor*. Keempat rotor tersebut juga menghasilkan daya angkat yang besar, sehingga cocok digunakan untuk membawa *payload*. Pada

penelitian ini, setiap drone membawa payload berupa board mikrokontroler yang membuat sebuah jaringan nirkabel bersifat ad-hoc.

## **2.5 Jaringan Nirkabel**

*Wireless Network* atau jaringan nirkabel adalah sebuah jaringan komputer yang menggunakan media nirkabel untuk koneksi data antar node [14]. Melalui media nirkabel, sebuah jaringan bersifat lebih fleksibel dalam sebuah ruangan karena tidak terbatasi oleh perkabelan untuk berkomunikasi. Jaringan nirkabel dapat dikelompokkan berdasarkan besar lingkupnya:

### **2.5.1 Wireless Wide Area Network (WWAN)**

Sebuah *Wide Area Network* (WAN) adalah jaringan dengan lingkup besar, yakni melingkupi sebuah daerah regional, negara, dan seluruh dunia [15]. Sebuah jaringan nirkabel yang melingkupi sebuah WAN memerlukan teknologi yang dapat melayani node yang bergerak-gerak. Contoh dari teknologi WWAN adalah jaringan seluler dan *wireless ad-hoc networks* (WANET).

### **2.5.2 Wireless Local Area Network (WLAN)**

*Local Area Network* (LAN) adalah sebuah jaringan dengan lingkup lokal, seperti sebuah gedung/bangunan, atau sebuah daerah kecil seperti sebuah kampus. Sebuah jaringan WLAN dapat melayani pengguna yang bergerak dalam jaringan itu sendiri. Contoh dari teknologi WLAN adalah Wi-Fi (IEEE 802.11).

### **2.5.3 Wireless Personal Area Network (WPAN)**

*Personal Area Network* adalah sebuah jaringan dengan lingkup pribadi dan biasa digunakan untuk komunikasi antar-piranti jarak pendek. Contoh dari teknologi WPAN adalah Bluetooth.

## **2.6 Flying Ad-hoc Network**

Menurut Khan (2017) [5], *Flying Ad-hoc Network* adalah sebuah kumpulan UAV-UAV kecil dalam konfigurasi ad-hoc, mengadopsi model jaringan WANET. Terdapat beberapa pertimbangan dalam pemilihan jenis teknologi komunikasi nirkabel untuk sebuah jaringan FANET, yakni kemampuan untuk *autoconfiguration*, tidak membutuhkan infrastruktur yang sudah ada, serta lingkup jaringan yang luas.

### **2.6.1 Bluetooth untuk penggunaan FANET**

Terdapat dua jenis teknologi Bluetooth yang dipasarkan oleh Bluetooth SIG, yakni Classic Bluetooth dan Bluetooth Low Energy (LE). Masing-masing memiliki

*range* teoritis sebesar 100 meter, tetapi dengan batasan daya radio dan interferensi maka *range* secara praktis hanya sekitar 10 sampai 20 meter [16]. Dengan target jarak antar node hingga 100 meter, maka teknologi Bluetooth kurang cocok untuk digunakan untuk FANET.

### 2.6.2 Jaringan Seluler untuk penggunaan FANET

Sebuah jaringan seluler membutuhkan infrastruktur penunjang untuk membangun jaringan tersebut, seperti *Base Transceiver Station*, *Base Station Controller*, dan *Mobile Switching Center* [17]; sehingga teknologi jaringan seluler kurang cocok untuk digunakan pada aplikasi FANET penelitian ini karena tidak dapat berfungsi tanpa infrastruktur yang sudah ada.

### 2.6.3 Wi-Fi untuk penggunaan FANET

Implementasi protokol IEEE 802.11 tidak menentukan *range* maksimum dari jaringan Wi-Fi, sehingga *range* maksimum dapat bervariasi sangat drastis. Dengan mengubah implementasi IEEE 802.11 pada sebuah piranti, maka jarak antar piranti dapat ditingkatkan dengan mengorbankan kemampuan interoperabilitas dengan piranti Wi-Fi lain. Contohnya adalah implementasi Espressif 802.11 Long Range, yang mengklaim dapat memiliki jarak maksimum hingga 1 km dengan *line-of-sight* [18].

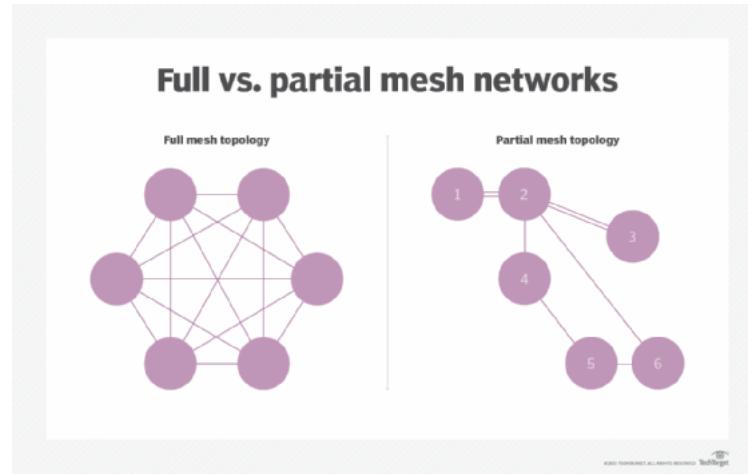
Wi-Fi juga dapat berfungsi tanpa adanya infrastruktur yang sudah ada menggunakan mode ad-hoc, sehingga piranti berbasis Wi-Fi dapat berkomunikasi satu sama lain tanpa memerlukan sebuah titik sentral seperti Wi-Fi router. Sehingga Wi-Fi cocok digunakan untuk jaringan FANET pada tugas akhir ini.

## 2.7 Jaringan *mesh*

Jaringan *mesh* adalah sebuah topologi jaringan dimana masing-masing node jaringan terhubung satu sama lain secara non-hierarkis. Pada sebuah jaringan *mesh* penuh, setiap node memiliki sebuah *routing* terhadap satu sama lain, sedangkan pada jaringan *mesh* parsial, hanya beberapa titik node yang terhubung satu sama lain, sehingga pada beberapa kasus data perlu melewati node lain untuk menuju node tujuan.

## 2.8 ESP32

ESP32 adalah sebuah mikrokontroler *system-on-a-chip* dengan integrasi Wi-Fi dan Bluetooth dan biaya yang murah. Dengan modul dibuat oleh Espressif Systems, ESP32 dapat digunakan dalam bentuk chip sendiri, atau menggunakan *development board* buatan produsen lain.



Gambar 2.4: Bentuk topologi jaringan *mesh* penuh dan parsial

Terdapat beberapa variasi modul ESP32, dengan versi CPU *single core* berbasis RISC-V dan *dual core* berbasis Xtensa LX7. Setiap modul ESP32 mendukung protokol Wi-Fi 802.11b/g/n pada band 2.4 GHz. ESP32 mendukung mode operasi klien Wi-Fi (*station mode*), mode operasi *Access Point* Wi-Fi, dan mode keduanya sekaligus (APSTA). Fitur ini dapat digunakan untuk membuat sebuah jaringan *mesh* ad-hoc.

ESP32 juga mendukung protokol *proprietary* buatan Espressif bernama 802.11 LR (*Long Range*), dengan jarak hingga 1 km selama ada *line-of-sight* antara *node*. [19].



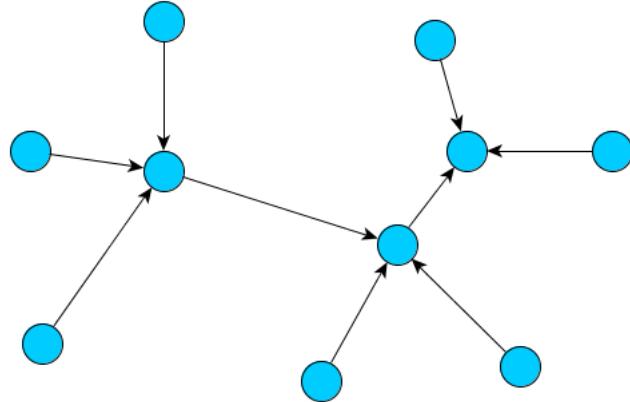
Gambar 2.5: ESP32

## 2.9 PainlessMesh

PainlessMesh merupakan sebuah library yang memudahkan pengguna *board* ESP32 dalam membuat sebuah jaringan *mesh* ad-hoc berbasis WiFi. Karena batasan dari *board* ESP32, maka jaringan yang dihasilkan merupakan *partial mesh* sehingga pada kondisi standar, semua node memiliki kedudukan yang sama dan diatur dalam topologi campuran antara *tree* dengan *mesh*.

Jaringan PainlessMesh menggunakan mode APSTA pada board ESP32, dimana setiap board ESP32, disebut dengan "node", berfungsi sebagai WiFi *Access Point*

(AP) sekaligus sebagai klien WiFi *Station Mode*. Setiap node berbasis ESP32 dapat memiliki sebanyak 10 klien yang terhubung pada 1 AP, dan dapat terhubung pada AP lainnya sebagai sebuah klien.



Gambar 2.6: Topologi jaringan PainlessMesh [20], panah menunjukkan arah koneksi dari klien ke AP

Keistimewaan dari library PainlessMesh adalah kemampuannya untuk *autoconfigure*, dimana setiap node dapat memutus sambungan dan menyambung ulang setiap saat, dan jaringan mesh dapat berjalan melalui proses konfigurasi secara otomatis.

### 2.9.1 Protokol PainlessMesh

Setiap pesan yang dikirimkan dalam sebuah jaringan PainlessMesh adalah berbasis JavaScript Object Notation (JSON), dan dapat dibagi menjadi dua tipe pesan: *control messages* dan *user messages*.

*Control messages* adalah pesan yang dikirimkan secara otomatis oleh *library* PainlessMesh untuk menjaga kelangsungan jaringan mesh secara asinkron, seperti sinkronisasi waktu antar node dan informasi *routing* antar node. *Control messages* hanya dikirimkan antara node yang berhubungan langsung satu sama lain.

*User messages* adalah pesan yang dikirimkan oleh pengguna dalam jaringan PainlessMesh, dan dapat berupa *string*, *binary data*, dan data JSON yang disematkan dalam *user message* tersebut. Setiap *user message* dapat ditentukan tujuannya antara tujuan spesifik (*single addressed*) maupun *broadcast*.

### 2.9.2 JavaScript Object Notation (JSON)

PainlessMesh menggunakan JavaScript Object Notation untuk setiap pengiriman pesan antar-node dalam jaringan. Penggunaan JSON memudahkan penafsiran

(*parsing*) setiap pesan yang diterima untuk kemudian diolah menjadi data yang ditampilkan [21].

```
{
    "dest": 887034362,
    "from": 37418,
    "type": 9,
    "msg": "The message I want to send"
}
```

Gambar 2.7: Setiap pesan pada jaringan PainlessMesh menggunakan JSON.

Setiap paket data PainlessMesh menggunakan skema JSON untuk menentukan tujuan (dari *key* "dest"), dari mana paket data tersebut berasal (*key* "from"), jenis paket berdasarkan nilai integer pada *key* "type", serta pesan yang akan dikirim pada *key* "msg".

## 2.10 Global Positioning System (GPS)

GPS adalah sebuah sistem navigasi satelit yang dikembangkan oleh United States Department of Defense (US DoD) dan dimiliki oleh pemerintah Amerika Serikat. Dengan menggunakan data waktu atomik dan beberapa satelit, maka lokasi (lintang, bujur, dan ketinggian) sebuah *receiver* GPS dapat didapatkan dengan akurasi sisi satelit yang dijamin oleh pemerintah Amerika Serikat kurang dari 2 meter [22].

Data lokasi dapat diterima dari satelit GPS menggunakan GPS *receiver*. Pada beberapa *receiver* GPS, data yang diterima adalah berupa data National Marine Electronics Association (NMEA) 0183.

## 2.11 GPS NMEA Data

Modul GPS NEO-6M menghasilkan output serial mentah berupa data NMEA 0183, sebuah standar komunikasi piranti GPS berupa karakter-karakter ASCII yang membentuk kalimat-kalimat NMEA [23].

```
Error: No GPS data received. Check wiring and reset.
Board will now dump GPS stream:
$GPTXT,01,01,02,u-blox ag - www.u-blox.com*50
$GPTXT,01,01,02,Hw UBX-G60xx 00040007 FFF9FFFp*5D
$GPTXT,01,01,02,ROM CORE 7.03 (45969) Mar 17 2011 16:18:34*59
$GPTXT,01,01,02,ANTSUPERV=AC SD PDoS SR*20
$GPTXT,01,01,02,ANTSTATUS=DONTKNOW*33
$GPRMC,,V,,N*53
$GPVTG,,N*30
$GPGGA,,,0,00,99.99,,,*48
$GPGSA,A,1,,99.99,99.99,99.99*30
$GPGSV,1,1,00*79
$GPGLL,,V,N*64
```

Gambar 2.8: Contoh data mentah dari modul GPS NEO-6M berupa data GPS NMEA.

## 2.12 Pengukuran Kinerja Jaringan

### 2.12.1 Round-trip delay

*Round-trip delay* adalah waktu yang diperlukan untuk sebuah paket data dikirimkan dari node pengirim ke node penerima, ditambah waktu yang diperlukan untuk paket *acknowledge* diterima oleh node pengirim dari node penerima. Pengukuran *round-trip delay* dapat menggunakan fungsi yang sudah diimplementasikan dalam *library* PainlessMesh yakni `painlessMesh::startDelayMeas()`.

Setiap node pada jaringan PainlessMesh memiliki waktu internal yang tersinkronisasi satu sama lain melalui proses "*Time sync*" [21] yang berlangsung secara periodis setiap 10 menit. PainlessMesh menjaga waktu jaringan pada tingkat kepresisan mikrosekon menggunakan fungsi `micros()` pada mikrokontroler ESP32. Kemudian setelah fungsi `startDelayMeas()` dipanggil, maka *round-trip delay* dapat dihitung menggunakan rumus berikut:

$$t_{\text{roundtrip}} = (t_3 - t_0) - (t_2 - t_1) \quad (2.1)$$

$t_0$  = waktu internal pada pengirim *delay measurement*

$t_1$  = *timestamp* pada saat paket *delay measurement* diterima

$t_2$  = *timestamp* pada saat respon terhadap *delay measurement* dikirimkan

$t_3$  = *timestamp* pada saat respon diterima oleh pengirim *delay measurement*

### 2.12.2 Throughput

*Throughput* adalah banyaknya data yang dapat ditransmisikan dalam suatu satuan waktu. Pada penelitian ini, *throughput* dihitung setiap pengiriman paket data, dibandingkan dengan perbedaan *timestamp* antara pengiriman dan penerimaan paket data tersebut.

$$\text{Throughput} = \frac{\text{sizeof(packet)}}{t_1 - t_0} \quad (2.2)$$

$t_0$  = waktu internal pada saat pengirim mengirim paket data

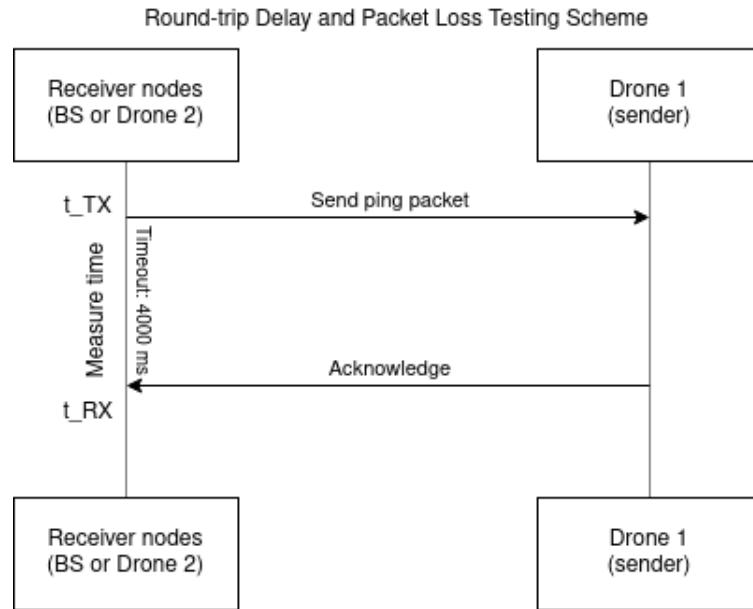
$t_1$  = *timestamp* pada saat penerima menerima paket data

### 2.12.3 Packet loss

Packet loss adalah jumlah packet data yang hilang dibandingkan packet data yang berhasil dikirimkan.

$$\text{PacketLoss} = (1 - \frac{\text{ack}_{RX}}{p_{TX}}) \times 100\% \quad (2.3)$$

$p_{TX}$  = jumlah packet data yang dikirimkan  
 $ack_{RX}$  = jumlah packet respon yang diterima

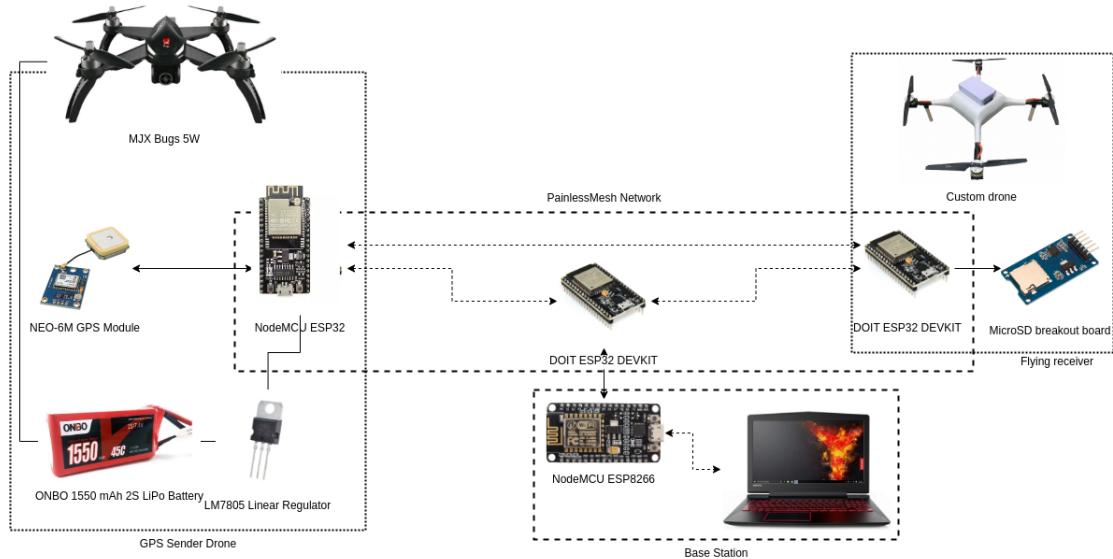


Gambar 2.9: Skema pengujian *round-trip delay* dan *packet loss*

## BAB III

### PERANCANGAN SISTEM

#### 3.1 Desain Sistem



Gambar 3.1: Diagram implementasi sistem.

Desain dan implementasi tugas akhir ini bertujuan untuk merancang sebuah sistem algoritma komunikasi antar-UAV menggunakan jaringan mesh berbasis ESP32. Perancangan sistem terdiri dari proses perancangan blok diagram sistem, penjelasan sistem secara umum, perancangan perangkat keras sistem, serta perancangan dan penjelasan perangkat lunak sistem yang meliputi algoritma komunikasi tersebut.

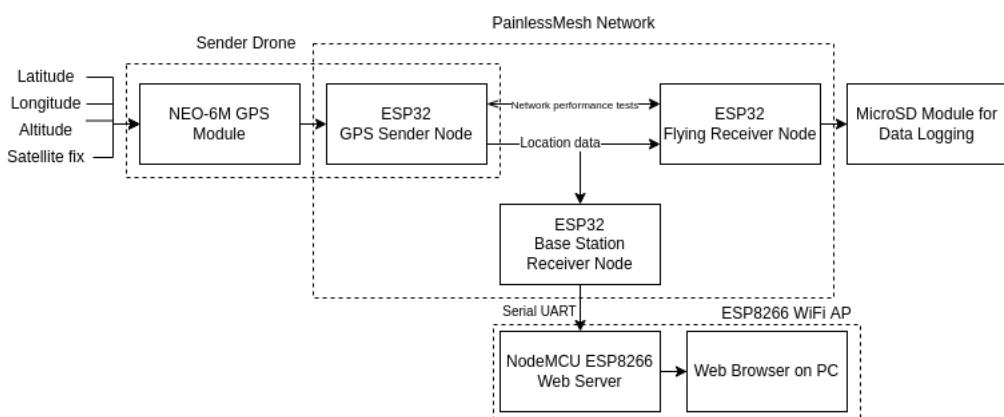
##### 3.1.1 Prinsip Kerja Sistem

Berdasarkan rumusan masalah yang telah dipaparkan pada bab 1, berikut adalah konsep prinsip kerja sistem yang dikembangkan:

1. Terdapat 2 drone dan 1 *Base Station* (BS), masing-masing menggunakan mikrokontroler ESP32 yang berkomunikasi satu sama lain menggunakan JSON *messages* pada jaringan PainlessMesh. Drone 1 disebut sebagai "*sender node*", dan drone 2 disebut sebagai "*receiver node*".
2. ESP32 Drone 1 mengaktifkan modul GPS dan mendata koordinat lokasi, ketinggian terbang drone, dan jumlah satelit yang mendapat *lock*, kemudian menghidupkan LED hijau pada *board*.
3. Drone 2 mengirimkan permintaan data lokasi kepada drone 1.

4. Drone 1 mengirimkan data melalui jaringan mesh kepada drone 2. Drone 2 menghidupkan LED hijau setelah menerima data lokasi yang valid.
5. BS mengirimkan permintaan data pada drone 1 berupa *string* dalam *message*.
6. Drone 1 mengirimkan data melalui jaringan mesh kepada BS.
7. BS menampilkan data kepada pengguna melalui *WiFi Access Point* yang dapat diakses dari sebuah *website*. WiFi AP tersebut dibangun menggunakan sebuah board ESP8266 yang menerima data dari board ESP32 yang terhubung dengan jaringan mesh melalui sambungan serial.

### 3.1.2 Diagram Blok



Gambar 3.2: Diagram blok sistem.

Gambar 3.1 merupakan diagram blok dari sistem yang dirancang. Secara keseluruhan, sistem dapat dibagi menjadi 3 bagian, yakni **Sender**, **Flying Receiver**, dan **Base Station Receiver**.

#### 3.1.2.1 Sender Node

Sender node merupakan board ESP32 yang diterbangkan menggunakan drone *Victim Finder* dan bertugas menerima data lokasi dari modul GPS dan kemudian mengirimkan data lokasi tersebut berdasarkan permintaan dari *receiver*. Board ESP32 mendapatkan data lokasi dari modul GPS dengan menggunakan komunikasi serial UART pada baud rate 9600, yang kemudian modul GPS mengirimkan data lokasi berupa kalimat-kalimat NMEA. Data tersebut kemudian diolah menggunakan library TinyGPS++ untuk memudahkan penafsiran kalimat-kalimat NMEA tersebut.

Karena sistem *messaging* di PainlessMesh menggunakan JSON, maka data lokasi yang telah didapatkan (lintang, bujur, ketinggian, dan jumlah satelit yang sudah *lock*) dikirimkan berupa JSON *values* dengan *key* berupa *latitude*, *longitude*, *altitude*, dan *satellite*. Data JSON tersebut dihitung besarnya dalam byte untuk kepentingan pengukuran *throughput*, kemudian ditambahkan *timestamp* pengiriman.

### **3.1.2.2 Flying Receiver Node**

Receiver node ini merupakan board ESP32 yang dilengkapi board MicroSD untuk kepentingan *data logging*. Board ini akan diterbangkan oleh sebuah drone dan bertugas melakukan permintaan data lokasi kepada node Sender, sekaligus melakukan pengujian kinerja jaringan, yakni pengujian *throughput*, pengujian *packet loss*, dan pengujian *round-trip delay*. Hasil dari pengujian tersebut kemudian disimpan dalam sebuah kartu microSD untuk diolah dan dianalisis.

### **3.1.2.3 Base Station Receiver Node**

Receiver node ini merupakan board ESP32 yang disambungkan secara serial dengan sebuah board ESP8266 yang bertugas menjadi web server untuk menampilkan data lokasi pada sebuah situs web. Konfigurasi ini diperlukan karena keterbatasan pengujian, dimana mode 802.11LR pada ESP32 merupakan mode *proprietary* yang tidak didukung oleh piranti WiFi lainnya, serta jaringan PainlessMesh yang membutuhkan mode AP dari ESP32, sedangkan board ESP32 hanya mendukung maksimal 1 AP per board. Oleh karena itu, dibutuhkan satu board lain untuk menjadi web server.

## **3.1.3 Fungsi dan Fitur**

Sistem yang dirancang untuk tugas akhir ini memiliki fungsi komunikasi data lokasi dari satu drone kepada drone lain dan *base station*. Karena posisi antar drone dan *base station* yang bervariasi serta target penggunaan di daerah tanpa infrastruktur, maka dibutuhkan metode komunikasi yang fleksibel terhadap disrupti dan tidak bergantung pada infrastruktur yang sudah ada, oleh karena itu digunakan sebuah jaringan mesh menggunakan PainlessMesh sehingga komunikasi dapat terjalin pada jaringan yang mampu melakukan *self-healing* dan *autoconfiguration* ketika terjadi disrupti.

## **3.2 Desain Perangkat Keras**

Perangkat yang digunakan pada sistem ini adalah tiga buah board DOIT ESP32 Development Kit, sebuah board NodeMCU ESP8266, sebuah modul GPS NEO-

6M, sebuah board modul MicroSD, baterai Lithium-Polymer (LiPo) 2S 1500 mAh, sebuah linear regulator LM7805, dan dua buah drone. Sistem dapat menerima daya dari baterai LiPo melalui konektor *balance* JST XH, yang kemudian tegangannya diturunkan menggunakan linear regulator LM7805 ke tegangan 5V.

Perancangan sistem membutuhkan komponen-komponen pendukung untuk dapat merealisasikan sistem, diantaranya adalah board DOIT-ESP32-DEVKIT, modul GPS NEO-6M, drone MJX Bugs 5W, dan *power supply*, dengan spesifikasi sebagai berikut:

### 1. Ai-Thinker NodeMCU-32S

Board Ai-Thinker NodeMCU-32S adalah sebuah *development kit* berbasis mikrokontroler ESP32S dari Ai-Thinker. Board ini memiliki kapabilitas Wi-Fi dan Bluetooth, serta mendukung penggunaan antena eksternal melalui konektor Hirose U.FL. Menggunakan kabel *pigtail* konverter U.FL ke konektor SubMiniature A (SMA), maka board ini dapat dengan mudah dihubungkan dengan sebuah antena eksternal 2.4 GHz. Pada perancangan sistem ini, board ini digunakan sebagai modul komunikasi pada drone *sender*.



NodeMCU-32

Gambar 3.3: Board Ai-Thinker NodeMCU-32S

Tabel 3.1: Spesifikasi Ai-Thinker NodeMCU-32S

SOC	2x Xtensa 32-bit LX6 microprocessor up to 240MHz, 384KB ROM, 512KB SRAM
Konektivitas	WiFi 802.11 (B/G/N/LR), Bluetooth LE
IO	18 channel ADC, 10 capacitive sensing GPIO, 3 UART interfaces, 3 SPI interfaces, 2 I <sup>2</sup> C interfaces, 16 PWM output channels, 2 channel DAC, 2 I <sup>2</sup> S interfaces
Masukan tegangan	5V regulated, 6V - 20V unregulated
Tegangan operasional	3.3V
Temperatur operasi	-40°C - 85°C

## 2. DOIT-ESP32-DEVKIT

Board DOIT-ESP32-DEVKIT adalah sebuah *development kit* yang menggunakan mikrokontroler ESP32, sehingga memiliki kapabilitas Wi-Fi dan Bluetooth. Pada perancangan sistem ini, ESP32 digunakan sebagai modul komunikasi pada drone *flying receiver* dan *base station*.



Gambar 3.4: Board DOIT-ESP32-DEVKIT

Tabel 3.3: Spesifikasi DOIT-ESP32-DEVKIT

SOC	2x Xtensa 32-bit LX6 microprocessor up to 240MHz, 448KB ROM, 520KB SRAM
Konektivitas	WiFi 802.11 (B/G/N/LR), Bluetooth
IO	18 channel ADC, 10 capacitive sensing GPIO, 3 UART interfaces, 3 SPI interfaces, 2 I <sup>2</sup> C interfaces, 16 PWM output channels, 2 channel DAC, 2 I <sup>2</sup> S interfaces
Masukan tegangan	5V regulated, 6V - 20V unregulated
Tegangan operasional	3.3V
Temperatur operasi	-40°C - 85°C

### 3. Ai-Thinker NodeMCU ESP-12S (ESP8266)

Board Ai-Thinker NodeMCU ESP-12S adalah sebuah *development kit* yang menggunakan mikrokontroler ESP8266. Pada perancangan sistem ini, ESP8266 digunakan sebagai *web server* yang menampilkan data lokasi drone sender yang diterima oleh *base station* ESP32 melalui UART.



NodeMCU-12S(V1.2 CH340C)

Gambar 3.5: Board Ai-Thinker NodeMCU ESP-12S

Tabel 3.5: Spesifikasi Ai-Thinker NodeMCU ESP-12S

SOC	Tensilica L106 32-bit 160 MHz
Konektivitas	WiFi 802.11 (B/G/N)
IO	9 IO Port, 1.5 UART Interfaces
Masukan tegangan	5V regulated, 6V - 20V unregulated
Tegangan operasional	3.3V
Temperatur operasi	-20°C - 85°C

#### 4. NEO-6M

NEO-6M adalah modul GPS *receiver standalone* yang menggunakan U-Blox 6 Positioning Engine. Pada perancangan sistem ini, NEO-6M digunakan sebagai modul GPS untuk mendapatkan data lokasi untuk dikirimkan.



Gambar 3.6: Modul GPS NEO-6M

Tabel 3.7: Spesifikasi DOIT-ESP32-DEVKIT

<i>Time to first fix</i>	27 detik <i>cold start</i> , 27 detik <i>warm start</i> , 1 detik <i>hot start</i>
Sensitivitas	-161dBm ( <i>tracking and navigation</i> ), -160dBm ( <i>reacquisition</i> ), -147dBm ( <i>cold start</i> ), -156dBm ( <i>hot start</i> ).
Akurasi posisi horizontal	2.5m
Masukan tegangan	2.7V - 3.6V
Maksimum arus	67mA
Temperatur operasi	-40°C - 85°C

5. *Power supply* yang digunakan pada *Sender node* dan *Flying receiver* node menggunakan sebuah linear regulator LM7805 yang dihubungkan pada sebuah baterai LiPo 2S 7.4V, sehingga menghasilkan tegangan keluaran 5V.

## 6. MJX Bugs 5W

Pada perancangan sistem, drone ini digunakan untuk membawa board ESP32 dan modul GPS sebagai node *sender*.



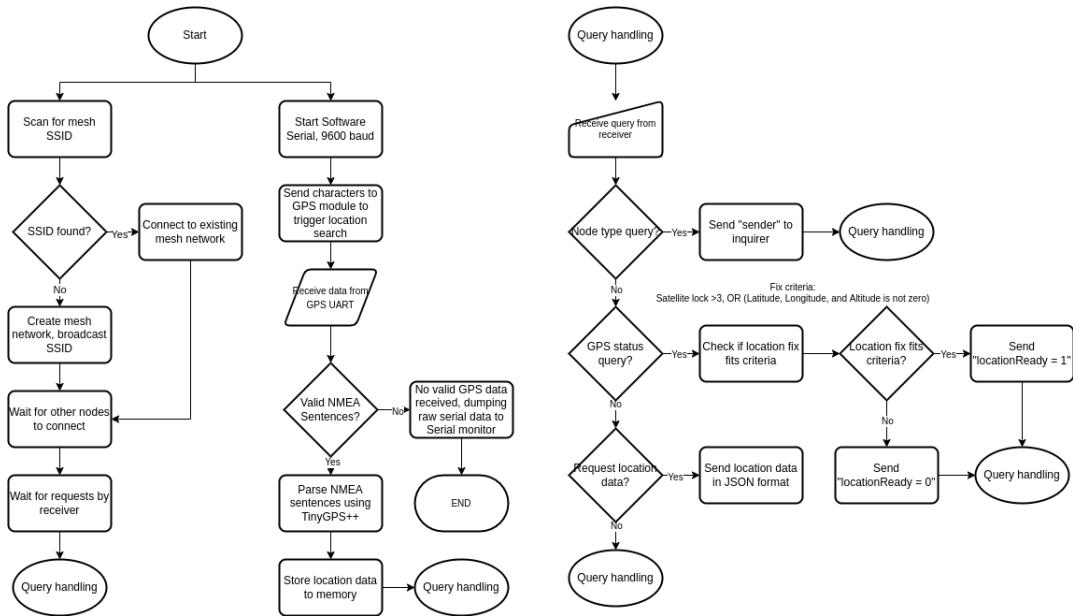
Gambar 3.7: Drone MJX Bugs 5W

Tabel 3.9: Spesifikasi drone MJX Bugs 5W

Frekuensi Remote Control	2.4GHz
Baterai drone	7.4V 2S 1800mAh
Jarak Remote Control	300-500m
Motor	1800KV BLDC
ESC	6A Brushless

### 3.3 Desain Perangkat Lunak

#### 3.3.1 Sender Node



Gambar 3.8: Diagram alur sender node.

Gambar 3.6 menunjukkan diagram alur sistem pada node sender. Pada bagian kiri adalah proses inisialisasi sistem, yakni inisialisasi jaringan mesh dan inisialisasi sistem GPS. Pada proses inisialisasi jaringan mesh, board ESP32 melakukan *scanning* terhadap SSID jaringan mesh yang sudah ditetapkan, dan jika ditemukan akan mencoba bergabung dalam jaringan mesh tersebut. Jika SSID tidak ditemukan, maka board ESP32 akan membuat jaringan mesh tersebut dengan sendirinya. Setelah inisialisasi jaringan sudah selesai, maka board ESP32 akan menunggu koneksi dari node lain serta menunggu permintaan dari sebuah node receiver.

Untuk proses inisialisasi GPS, board ESP32 memulai sebuah koneksi serial UART dengan modul GPS NEO-6M menggunakan *library* Software Serial dengan baud rate 9600 bps. Modul GPS NEO-6M membutuhkan sebuah input konstan dari koneksi UART, oleh karena itu board ESP32 secara terus-menerus mengirimkan karakter kepada modul GPS tersebut. Terdapat waktu tunggu lima detik untuk menunggu respon dari modul GPS, jika tidak ada input dari modul GPS setelah lima detik sejak board pertama dinyalakan, maka board akan menghentikan eksekusi program dan menampilkan data mentah dari pin RX di Serial Monitor. Sedangkan jika modul GPS berhasil mengirimkan data melalui UART, maka eksekusi program diteruskan, dan data dari modul GPS tersebut yang berupa kalimat-kalimat NMEA akan ditafsirkan menggunakan *library* TinyGPS++ untuk mendapatkan data lokasi

yang kemudian disimpan dalam memori.

Setelah proses inisialisasi sistem selesai, maka board ESP32 akan menunggu permintaan dari node receiver. Terdapat tiga jenis permintaan/*query* yang diterima oleh node sender, yaitu:

1. *Node Type Query*

*Query* ini merupakan sebuah permintaan dari node receiver yang menanyakan tipe node apa yang telah terhubung dalam jaringan mesh, menggunakan pesan dengan "type" bernilai 1. Respon dari *query* ini untuk node sender adalah mengirimkan sebuah pesan JSON dengan *key* "nodeType" dan *value* "sender".

2. *GPS Status Query*

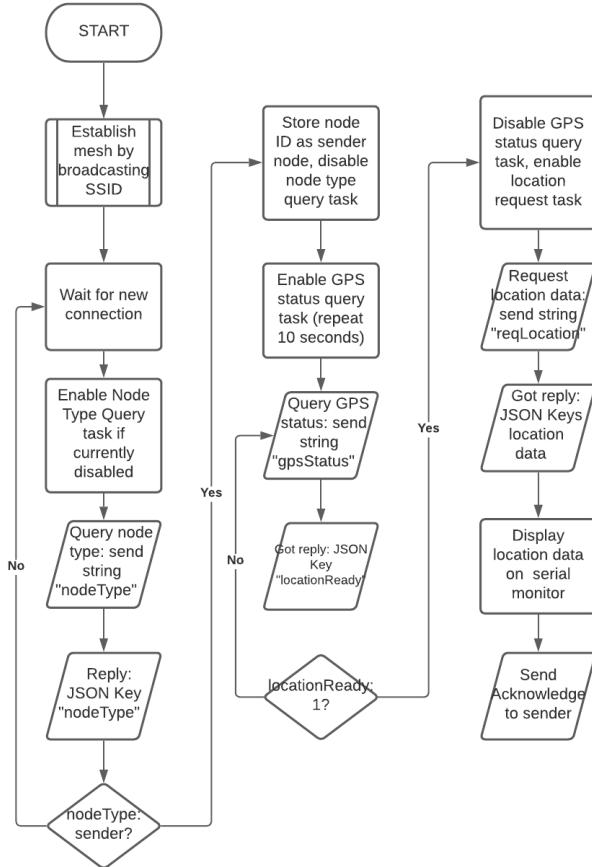
*Query* ini adalah permintaan dari node receiver untuk mengecek status GPS node sender dan kesiapan node sender untuk mengirimkan data lokasi, dengan menggunakan pesan yang bertipe senilai 2. Kriteria kesiapan tersebut adalah jumlah satelit yang sudah  $fix \geq 3$ ; atau nilai lintang, bujur, dan ketinggian tidak nol. Node sender akan mengirimkan sebuah pesan JSON dengan *key* "locationReady" dan *value* 0 jika kriteria kesiapan GPS belum terpenuhi, dan 1 jika kriteria sudah terpenuhi.

3. *Request Location Query*

*Query* ini adalah permintaan data lokasi dari node receiver menggunakan pesan bertipe senilai 3. Node sender akan mengirimkan sebuah pesan JSON dengan *key* *latitude*, *longitude*, *altitude*, dan *satellite*, berdasarkan data yang sudah didapatkan dari modul GPS.

Node sender juga menerima permintaan pengecekan *round-trip delay* dari node receiver yang menggunakan fungsi `startDelayMeas()`. Fungsi tersebut juga digunakan untuk pengujian *packet loss*.

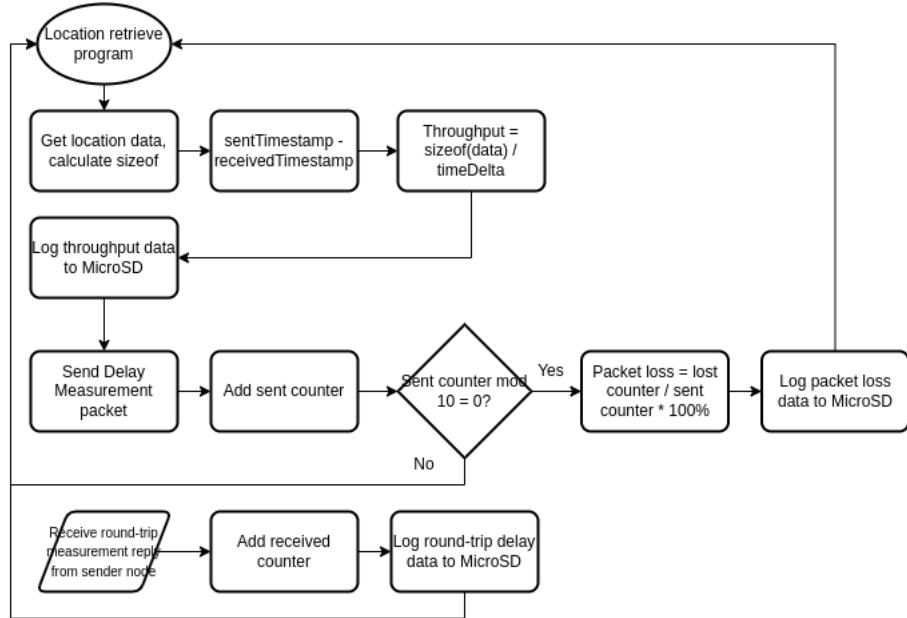
### 3.3.2 Flying Receiver Node



Gambar 3.9: Diagram alur node flying receiver, program penarikan data lokasi dari node sender.

Flying receiver node adalah node ESP32 yang diterbangkan menggunakan sebuah drone yang bertugas menerima data lokasi dari sender node, menguji kinerja jaringan terbang, sekaligus mencatat ke sebuah kartu memori MicroSD. Tugas-tugas tersebut dibagi menjadi dua program pada satu node, yaitu program pertama yang menarik data lokasi dari sender node, dan program kedua yang menguji kinerja jaringan setelah selesainya pelaksanaan program pertama.

Pelaksanaan masing-masing task pada node ini terjadwal menggunakan fitur *Task Scheduler*, dan dilaksanakan secara berurutan, yakni melakukan koneksi pada jaringan mesh, melakukan *query* tipe node-node lainnya pada jaringan, melakukan *query* kondisi GPS pada node sender, dan melakukan *query* data lokasi dari sender node.



Round-trip measurement packet diterima sebagai interrupt dalam library PainlessMesh, sehingga receiver node mengirim paket sender terus-menerus beriringan dengan location request dari receiver tanpa menunggu balasan dari receiver, dijalankan setiap 2 detik.

Gambar 3.10: Diagram alur node flying receiver, program pengujian jaringan.

Berdasarkan yang dipaparkan pada Bab 2, pengujian kinerja jaringan yang dilakukan adalah pengujian *throughput*, *round-trip delay*, dan *packet loss*.

### 1. *Throughput*

Pengujian *throughput* dilakukan bersamaan dengan program penarikan data lokasi, yakni dengan menghitung besar paket data lokasi yang dikirim, membandingkan waktu pengiriman dan waktu penerimaan paket data, lalu membagi besar paket data tersebut dengan selisih waktu pengiriman dan penerimaan paket data.

Misalkan dengan paket data sebesar 12 byte, waktu pengiriman paket data dari *sender* senilai 1000 ms, dan waktu penerimaan paket data di *receiver* senilai 1250 ms, maka nilai *throughput* sebesar:

$$\begin{aligned}
 \text{Throughput} &= \frac{\text{sizeof(packet)}}{t_1 - t_0} \\
 &= \frac{12}{1250 - 1000} \\
 &= \frac{12}{250} \\
 &= 0.048B/ms \\
 &= 48B/s
 \end{aligned} \tag{3.1}$$

## 2. Round-trip delay

Pengujian *round-trip delay* menggunakan fungsi `startdelayMeas()` pada library PainlessMesh yang menghasilkan *return value* sebesar nilai *delay* dalam mikrosekon. *Receiver node* melakukan tes ini tepat setelah menerima data lokasi dari *sender node* dengan memanggil fungsi `startDelayMeas()`. Setelah fungsi tersebut dipanggil, maka *receiver* menambahkan jumlah *counter* untuk paket data yang dikirim.

Pada sistem ini,  $t_0$  adalah *timestamp internal clock* board ESP32 saat permintaan pengukuran *delay* dibuat oleh *receiver node*,  $t_1$  adalah *timestamp jaringan* saat permintaan pengukuran *delay* diterima oleh *sender node*,  $t_2$  adalah *timestamp jaringan* saat balasan dari *sender node* dibuat, dan  $t_3$  adalah *timestamp internal clock* board ESP32 saat balasan pengukuran *delay* diterima oleh *receiver node*. Misalkan nilai  $t_0 = 20000 \mu\text{s}$ ,  $t_1 = 25000 \mu\text{s}$ ,  $t_2 = 35000 \mu\text{s}$ , dan  $t_3 = 60000 \mu\text{s}$ :

$$\begin{aligned}
 t_{\text{roundtrip}} &= (t_3 - t_0) - (t_2 - t_1) \\
 &= (60000 - 20000) - (35000 - 25000) \\
 &= 40000 - 10000 \\
 &= 30000 \mu\text{s} \\
 &= 30 \text{ms}
 \end{aligned} \tag{3.2}$$

## 3. Packet loss

Pengujian *packet loss* dilakukan setelah 10 kali *receiver* mengirim paket data lalu membandingkan berapa kali fungsi `startdelayMeas()` dipanggil terhadap jumlah paket balasan dari *sender node*.

Misalkan telah dikirimkan 80 paket pengukuran *delay* dan telah diterima 75 paket respon pengukuran *delay*, maka:

$$\begin{aligned}
 \text{PacketLoss} &= \left(1 - \frac{\text{ack}_{\text{RX}}}{\text{PTX}}\right) \times 100\% \\
 &= \left(1 - \frac{75}{80}\right) \times 100\% \\
 &= 0.0625 \times 100\% \\
 &= 6.25\%
 \end{aligned} \tag{3.3}$$

## 4. Signal strength

Pendataan *signal strength* dilakukan setiap kali ada permintaan data dari *receiver*, menggunakan fungsi `WiFi.RSSI()` dari ESP32.

## BAB IV

### HASIL DAN ANALISIS

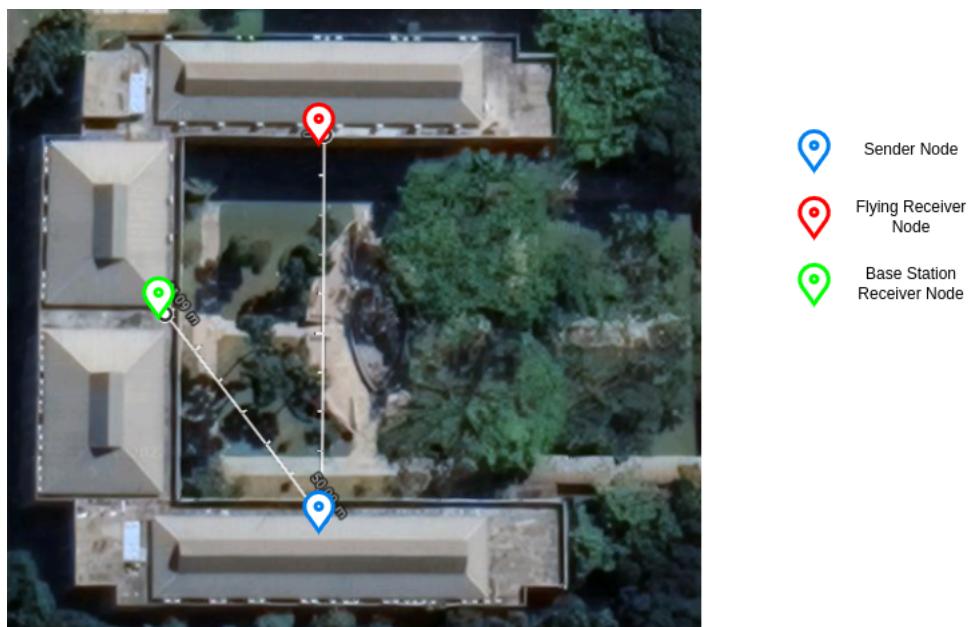
#### 4.1 Pendahuluan

Untuk implementasi algoritma komunikasi antar-UAV ini, digunakan satu buah unit NodeMCU ESP32 dengan membandingkan konfigurasi antena *built-in* dan antena eksternal sebagai *sender*, dua buah unit DOIT ESP32 DEVKIT sebagai *receiver* dilengkapi dengan board MicroSD *reader/writer* untuk *logging* data, satu buah unit NodeMCU ESP8266 sebagai unit *web server*, dan satu buah *laptop* untuk menampilkan data lokasi dari *sender*.

#### 4.1.1 Pelaksanaan Pengujian Sistem

Pengujian sistem dilaksanakan pada 2 tempat, yakni Gedung N Fakultas Teknik Elektro (FTE) Telkom University dan Lapangan Bandung Techno Park (BTP). Terdapat x skenario pengujian yang dilakukan:

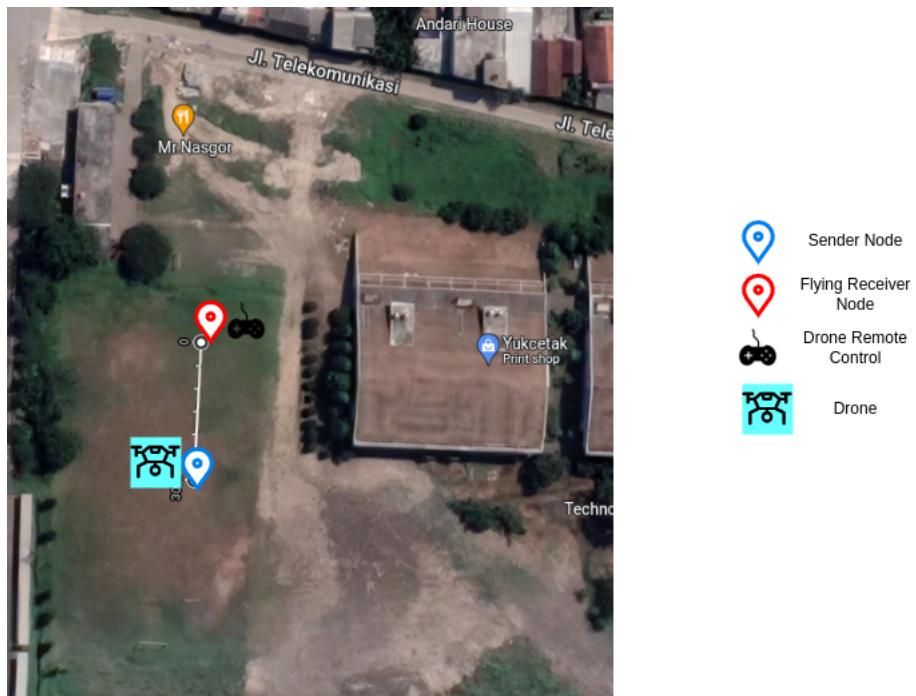
1. **Pengujian tanpa terbang:** Pengujian sistem dilakukan tanpa menerbangkan *drone*, dilakukan di Gedung N Fakultas Teknik Elektro Telkom University. Node sender ditempatkan di depan ruang N304, dan node "flying receiver" di depan ruang N314. Pada pengujian 3 node, maka node *base station* ditempatkan di koridor depan ruang N308. Masing-masing node ditempatkan di lantai tiga.



Gambar 4.1: Penempatan node jaringan pada pengujian non-terbang di Gedung N FTE Telkom University.

2. **Pengujian terbang - 1 drone - Lapangan BTP:** Pengujian sistem dilakukan

dengan menerbangkan *drone sender* dengan jarak 30-50 meter ke arah selatan dan dengan ketinggian hingga 10 meter di atas permukaan tanah. Node "flying receiver" diletakkan di topi penguji untuk menjaga *line-of-sight* dengan node *sender*.



Gambar 4.2: Penempatan node jaringan pada pengujian terbang satu drone di Lapangan BTP Telkom University.

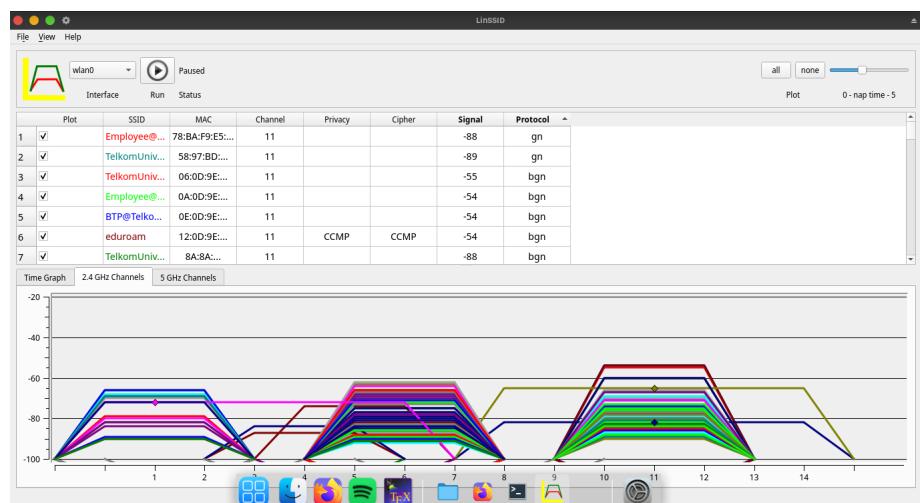
**3. Pengujian terbang - 2 drone - Lapangan BTP:** Pengujian sistem 2 drone dilakukan dengan menerbangkan *drone sender* dengan jarak 30 hingga 50 meter terhadap *drone receiver*. Kedua drone ditargetkan mendapatkan ketinggian 10 meter di atas permukaan.



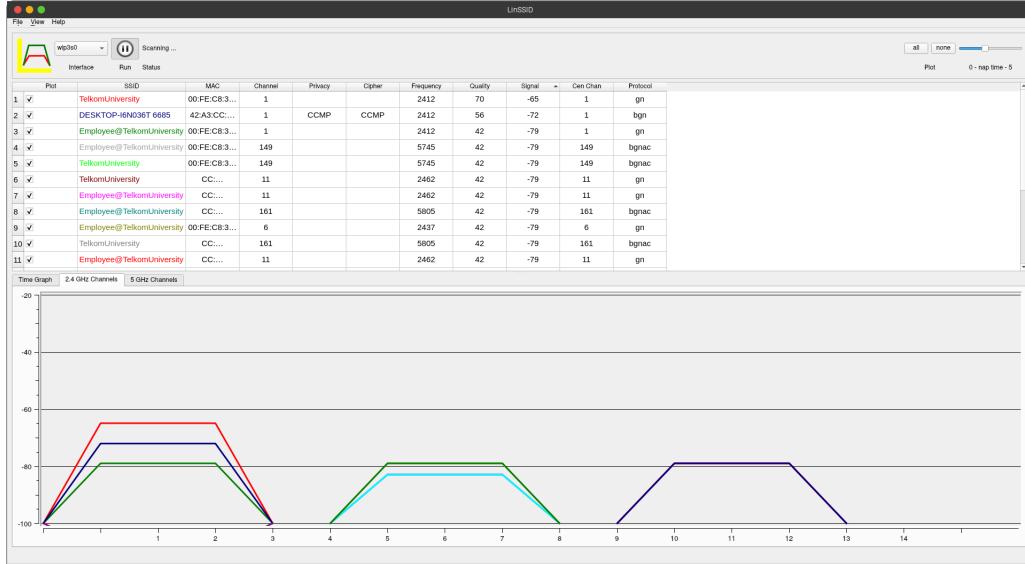
Gambar 4.3: Penempatan node jaringan pada pengujian terbang dua drone di Lapangan BTP Telkom University.

#### 4.1.2 Kondisi Lingkungan Pengujian

Faktor lingkungan dapat mempengaruhi hasil dari pengujian, dengan faktor paling menentukan adalah banyaknya interferensi di pita frekuensi 2.4 GHz terutama dari piranti Wi-Fi di sekitar. Banyaknya interferensi Wi-Fi dapat dilihat menggunakan sebuah program Wi-Fi *Spectrum Analyzer* seperti LinSSID.



Gambar 4.4: Hasil analisis spektrum Wi-Fi 2.4 GHz di Lapangan BTP Telkom University.



Gambar 4.5: Hasil analisis spektrum Wi-Fi 2.4 GHz di Gedung N Telkom University.

Gambar 4.4 dan 4.5 menunjukkan banyaknya *access point* (AP) Wi-Fi 2.4 GHz di daerah Lapangan BTP dan Gedung N FTE Telkom University. Untuk menjaga konsistensi pengujian dan sekaligus menghindari interferensi yang berlebih, maka jaringan mesh diatur untuk menggunakan kanal 6 2.4 GHz dengan pita frekuensi 2426-2448 MHz.

#### 4.1.3 Desain Alat

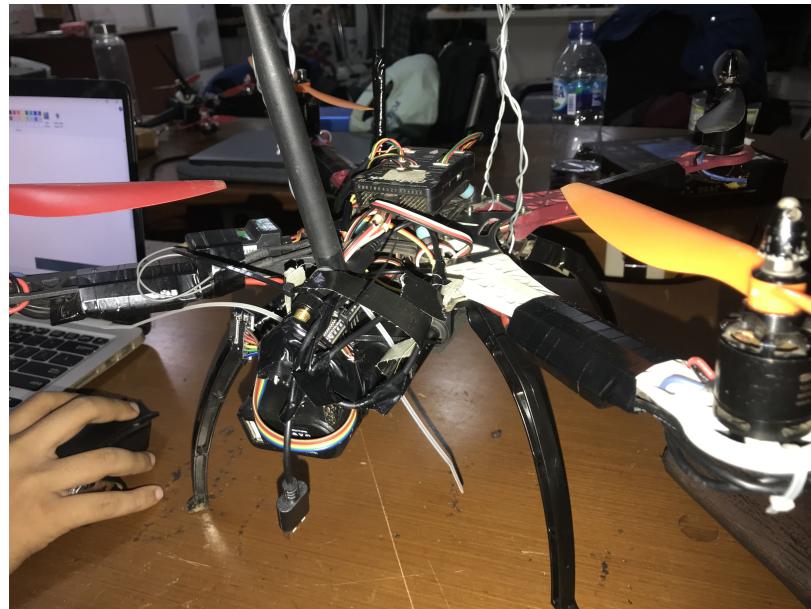
##### 4.1.3.1 Drone *Sender*



Gambar 4.6: Penampakan implementasi *sender* node di drone MJX Bugs 5W.

Drone *sender* merupakan drone yang membawa *board sender* node dan dilengkapi dengan kapabilitas GPS. Karena penggunaan drone yang tidak didesain untuk membawa *payload*, maka untuk mengurangi bobot, implementasi *board* ESP32 tidak menggunakan PCB melainkan sepenuhnya menggunakan kabel-kabel *jumper*, serta tidak ada *shroud* pelindung *board*. Antena *omnidirectional* ditempatkan di sisi atas drone untuk menjaga *line-of-sight* terhadap *flying receiver*.

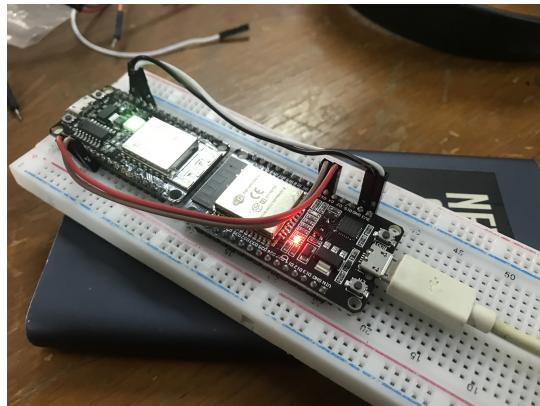
#### 4.1.3.2 Drone *Receiver*



Gambar 4.7: Penampakan implementasi *sender node* di drone MJX Bugs 5W.

Drone *receiver* merupakan drone yang membawa *board receiver* node dan dilengkapi dengan sebuah *breakout board* MicroSD untuk *logging* data. *Board receiver* menggunakan sebuah *power bank* sebagai sumber daya dan dihubungkan menggunakan kabel USB Micro-B. Pada sistem ini, *board* DOIT-ESP32-DEVKIT dimodifikasi untuk dapat dipasang antena eksternal berbasis SubMiniature-A (SMA) 2.4 GHz, diletakkan di atas *board* dan diposisikan di sisi atas untuk menjaga *line-of-sight* terhadap *sender*.

#### 4.1.3.3 Base Station Receiver



Gambar 4.8: Implementasi *Base Station Receiver* menggunakan *development board* Ai-Thinker NodeMCU ESP8266 dan DOIT-ESP32-DEVKIT

Pada node *base station*, sebuah board DOIT-ESP32-DEVKIT dan Ai-Thinker NodeMCU ESP8266 dipasang di sebuah *breadboard* dan dihubungkan secara UART menggunakan kabel jumper. Pin VIN dan GND dari masing-masing *board* dihubungkan satu sama lain menggunakan kabel jumper, sehingga kedua *board* dapat dinyalakan secara bersamaan dengan mencolok kabel USB ke salah satu port USB dari kedua *board* tersebut. Pada sistem ini, catu daya yang digunakan adalah sebuah *power bank*.

## 4.2 Skenario Pengujian

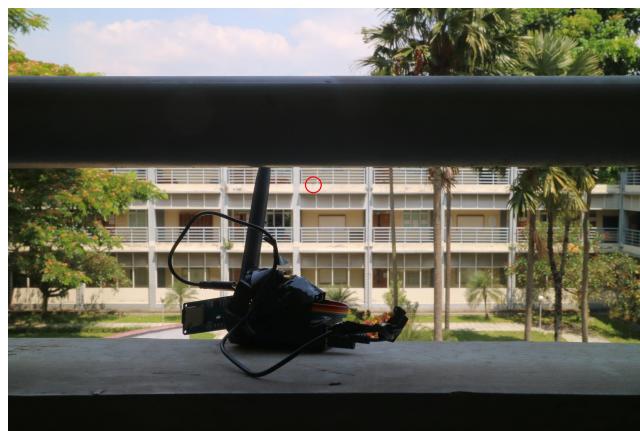
### 4.2.1 Pengujian Tanpa Terbang



Gambar 4.9: Penempatan *sender* node pada pengujian tanpa terbang, di depan ruangan N304.



Gambar 4.10: Penempatan *base station* pada pengujian tanpa terbang, di depan ruangan N308.



Gambar 4.11: Penempatan *flying receiver* node pada pengujian tanpa terbang, di depan ruangan N314.

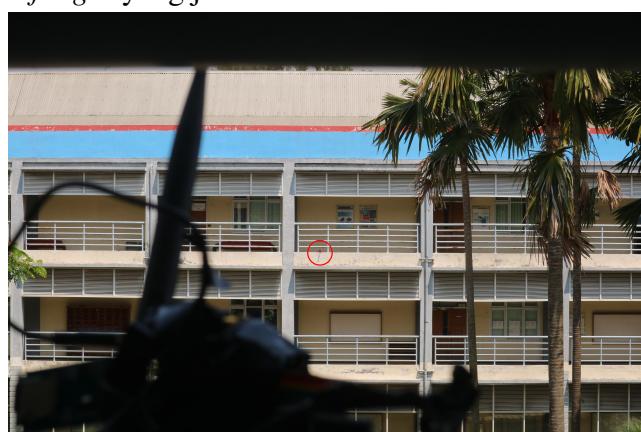
Pengujian tanpa terbang dilakukan sebanyak dua kali, menguji perbedaan mode Wi-Fi *Long Range* dan Wi-Fi 802.11n. Data yang didapatkan berupa *throughput*, *round-trip delay* dan *packet loss* komunikasi antara dua node. Karena fungsi WiFi.RSSI() pada mikrokontroler ESP32 mengukur kekuatan sinyal antara *station* dengan *access point* dan pada jaringan PainlessMesh terdapat kemungkinan koneksi antara *sender* dan *receiver* tidak terhubung secara langsung melainkan melalui *base station*, maka ada kemungkinan nilai RSSI yang didapatkan pada setiap pengukuran memiliki dua nilai. Oleh karena itu, agar keadaan jaringan dapat ditebak, maka masing-masing node dinyalakan pada waktu yang berbeda, dengan *base station* diyalakan pertama, kemudian *sender* node, dan paling terakhir adalah *flying receiver* node. Dari urutan tersebut, maka dapat ditebak *sender* node akan terhubung ke AP *base station*, kemudian *flying receiver* akan terhubung ke AP *sender*.

Berdasarkan hasil pengukuran dari fungsi sizeof(msg), besar paket data lokasi yang dikirimkan dari *sender* ke *receiver* adalah sebesar 12 byte. Satuan *throughput* yang digunakan adalah Byte per detik (B/s). Satuan *round-trip delay* yang digunakan library PainlessMesh adalah mikrosekon yang kemudian dikonversi menjadi milisekon dengan membagi nilainya dengan 1000. *Packet loss* dihitung setelah 10 kali pengiriman paket pengujian *round-trip delay*.

Pada pengujian ini, node *sender* memiliki *line-of-sight* yang jelas terhadap node *flying receiver* dan *base station*. Akan tetapi, node *flying receiver* hanya memiliki *line-of-sight* jelas terhadap node *sender*, sedangkan posisi node *base station* tertutup pohon. Hal ini mempengaruhi hasil pengujian, terutama pada nilai RSSI.



Gambar 4.12: Lingkaran merah menunjukkan posisi *flying receiver* node dari *sender*, dengan *line-of-sight* yang jelas.



Gambar 4.13: Lingkaran merah menunjukkan posisi *sender* node dari *flying receiver*, dengan *line-of-sight* yang jelas.



Gambar 4.14: Lingkaran merah menunjukkan posisi *base station* dari *flying receiver*, terlihat bahwa *line-of-sight* tertutup oleh pohon.

#### 4.2.2 Pengujian Terbang Satu Drone



Gambar 4.15: Pengujian satu drone, jarak 30 meter dan ketinggian dari 4 meter ke 10 meter.

Pengujian terbang satu drone dilakukan di Lapangan BTP Telkom University, dengan drone *sender* diterbangkan dengan jarak 30 meter hingga 50 meter dari posisi *receiver*. Ketinggian drone bervariasi dari 4 meter di atas tanah hingga 10 meter di atas tanah. Pengujian dilakukan sebanyak dua kali, menguji perbedaan mode 802.11n dan mode LR.

Pada Gambar 4.13, lingkaran merah menunjukkan posisi drone *sender*, lingkaran biru menunjukkan posisi patok 30 meter dari lokasi penguji, dan lingkaran kuning menunjukkan posisi patok 50 meter dari lokasi penguji. Pengujian dilakukan selama 250 detik hingga mendapat peringatan *low battery* dari remote kontrol drone. Pada 50 detik pertama, drone melakukan *hover* pada ketinggian 5 meter, lalu naik ke ketinggian 10 meter, kemudian pada detik ke-90 drone dijauhkan ke jarak 50 meter. Setelah peringatan *low battery* muncul, drone didekatkan ke penguji hingga akhirnya mendarat di patok 30 meter.



Gambar 4.16: Jarak drone dengan penguji sekitar 32 meter.



Gambar 4.17: Penempatan node *flying receiver* di topi penguji. Foto diambil sebelum node *flying receiver* dinyalakan.

Node *flying receiver* ditempatkan di topi penguji untuk menjaga *line-of-sight* dengan antena *sender* node. Dengan penempatan tersebut, posisi antena *flying receiver* terletak sekitar 1.8 meter di atas tanah. Pengujian terbang satu drone tidak menggunakan node *base station*, sehingga jaringan hanya memiliki 2 node.

#### 4.2.3 Pengujian Terbang Dua Drone



Gambar 4.18: Pengujian dua drone, jarak antar-drone 30 meter dan ketinggian dari 4 meter ke 10 meter.

Pengujian terbang dua drone dilakukan dengan menempatkan dua drone dengan jarak perkiraan antar-drone 30 meter dan perbedaan ketinggian antar-drone bervariasi antara 4 hingga 10 meter. Masing-masing drone dikendalikan secara manual dari darat oleh dua pilot.

Pengujian ini dilakukan menggunakan dua node dan hanya menggunakan mode LR. Drone *flying receiver* tidak memiliki fitur *altitude hold*, sehingga ketinggian drone *flying receiver* berubah-ubah dengan rentang antara 4 hingga 10 meter. Drone *sender* ditahan pada ketinggian 10 meter dari permukaan tanah.

### 4.3 Realisasi Algoritma Sistem

```

CONNECTION: scanComplete():--> Cleared old APs.
CONNECTION: scanComplete(): num = 1
CONNECTION:     found : UAV, -47dBm
CONNECTION:     Found 1 nodes
CONNECTION: connectToAP(): Best AP is 84313829<---
CONNECTION: connectToAP(): Trying to connect, scan rate set to 4*normal
CONNECTION: eventSTA Got IP Handler: ARDUINO_EVENT_WIFI_STA_GOT_IP
CONNECTION: New STA connection incoming
CONNECTION: painlessmesh::Connection: New connection established.
CONNECTION: newConnectionTask():
CONNECTION: newConnectionTask(): adding 84313829 now= 3526829
New Connection, nodeId = 84313829
Changed connections
Querying node type...
Sent node type query. Timestamp is
3530620Querying node type...
Sent node type query. Timestamp is
3536440
Message from 84313829 msg = {"type":1,"timestamp":8530852,"nodeType":1,"rssi":0}

```

Gambar 4.19: Proses koneksi awal di node *receiver*.

```

CONNECTION: New AP connection incoming
CONNECTION: painlessmesh::Connection: New connection established.
CONNECTION: newConnectionTask():
CONNECTION: newConnectionTask(): adding 1830119401 now= 186992428
New Connection, nodeId = 1830119401
Changed connections
Message from 1830119401 msg={"type":1,"timestamp":4214663}
Sending Node Type

```

Gambar 4.20: Proses koneksi awal di node *sender*.

Gambar 4.19 dan 4.20 menunjukkan proses awal koneksi antara node *sender* dengan node *receiver*. Node *sender* menerima pesan dengan *key* type senilai 1 yang merupakan sebuah pesan *node type query*, sebagaimana yang telah dijelaskan di Bab 3.3.1. *Sender* kemudian membalas dengan tipe yang sama dan *key* *nodeType* senilai 1.

```

Message from 84313829 msg = {"type":1,"timestamp":8530852,"nodeType":1,"rssi":0}
    appended. Sender Node ID recorded, 84313829Disabling node type query task...
Querying GPS status...
Sent GPS status query. Timestamp is
3624987
Message from 84313829 msg = {"type":2,"locationReady":1,"timestamp":3948661,"rssi":0}

```

Gambar 4.21: Permintaan status GPS kepada *sender* oleh *receiver*.

```

Message from 1830119401 msg={"type":2,"timestamp":4301355}
Sending GPS Status Query
Sent message: {"type":2,"locationReady":1,"timestamp":4607533,"rssi":0}

```

Gambar 4.22: Balasan status GPS kepada *receiver*.

Khusus pada pengujian, kriteria kesiapan GPS yang dipaparkan di Bab 3 dinaktifkan, sehingga node *sender* dapat terus mengirim data lokasi dari GPS wala-

upun status GPS masih belum memenuhi kriteria. Permintaan status GPS dilakukan dengan mengirimkan pesan bertipe senilai 2 kepada node *sender*, dan dibalas dengan sebuah *key* locationReady dengan nilai boolean 0 jika belum memenuhi kriteria, dan 1 jika sudah.

```
.....,.....,..... appended.Sender Node reports GPS system is ready. Commencing location data request..  
Disabling GPS status query...  
Requesting location data from sender node...  
Sent location request. Timestamp is  
3994723 Sending ping test...
```

Gambar 4.23: Permintaan data lokasi kepada *sender* oleh *receiver*.

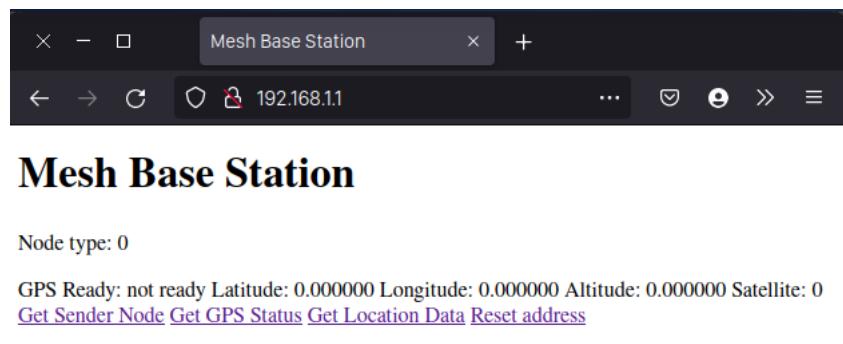
```
Adjusted time 1809389.01551 offset + 42151  
Message from 1830119401 msg={"type":3,"timestamp":4642365}  
Sending Location Data  
Sent message: {"type":3,"latitude":0,"longitude":0,"altitude":0,"satellites":0,"timestamp":4699858,"rssi":0}
```

Gambar 4.24: Balasan data lokasi kepada *receiver*.

Permintaan data lokasi dilakukan dengan mengirimkan pesan dengan tipe senilai 3, yang kemudian dibalas oleh node *sender* dengan pesan yang memiliki data lokasi dalam *JSON keys*.

```
Message from 84313829 msg = {"type":3,"latitude":0,"longitude":0,"altitude":0,"satellites":0,"timestamp":6041513,"rssi":0}  
Sender-side RSSI:  
017768  
0.017768  
12  
Throughput:  
675.371460 bpsAppending file: /throughput.csv  
Message 6065,675.371460,0,-50  
 appended.  
lat 0.000000, lon 0.000000, alt 0.000000, sat 0Appending file: /location.csv  
Message 6080,0.000000,0.000000,0.000000,0,0,-50  
 appended.Round-trip delay:  
31445  
3 received ping counter, 3 sent ping counter
```

Gambar 4.25: Data lokasi yang dikirimkan oleh *sender*.



Gambar 4.26: Tampilan awal laman web *base station*.

## Mesh Base Station

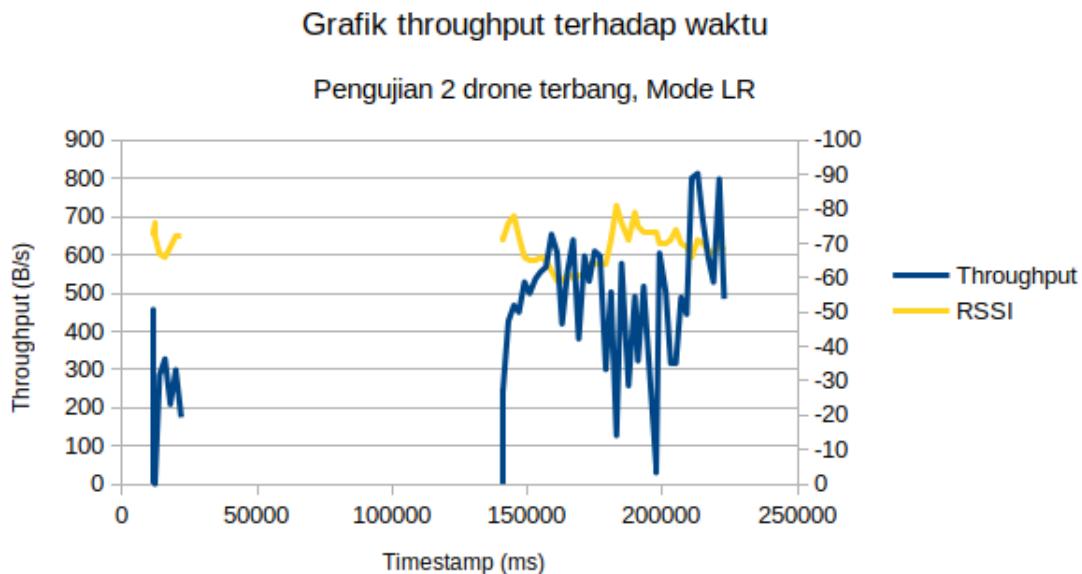
Node type: 1

GPS Ready: ready Latitude: 0.000000 Longitude: 0.000000 Altitude: 0.000000 Satellite: 0 [Get Sender Node](#) [Get GPS Status](#) [Get Location Data](#) [Reset address](#)

Gambar 4.27: Tampilan laman web *base station* setelah melakukan koneksi ke *sender* beserta meminta status GPS.

Implementasi laman web jaringan mesh pada dasarnya adalah implementasi algoritma antara *receiver* dan *sender* yang dilakukan secara manual, dimana pengguna mengklik tautan *Get Sender Node* untuk menghubungkan *base station* dengan *sender*, *Get GPS Status* untuk meminta status GPS kepada *sender*, dan *Get Location* untuk meminta data lokasi kepada *sender*. Laman web diatur untuk melakukan muat ulang setiap 2 detik.

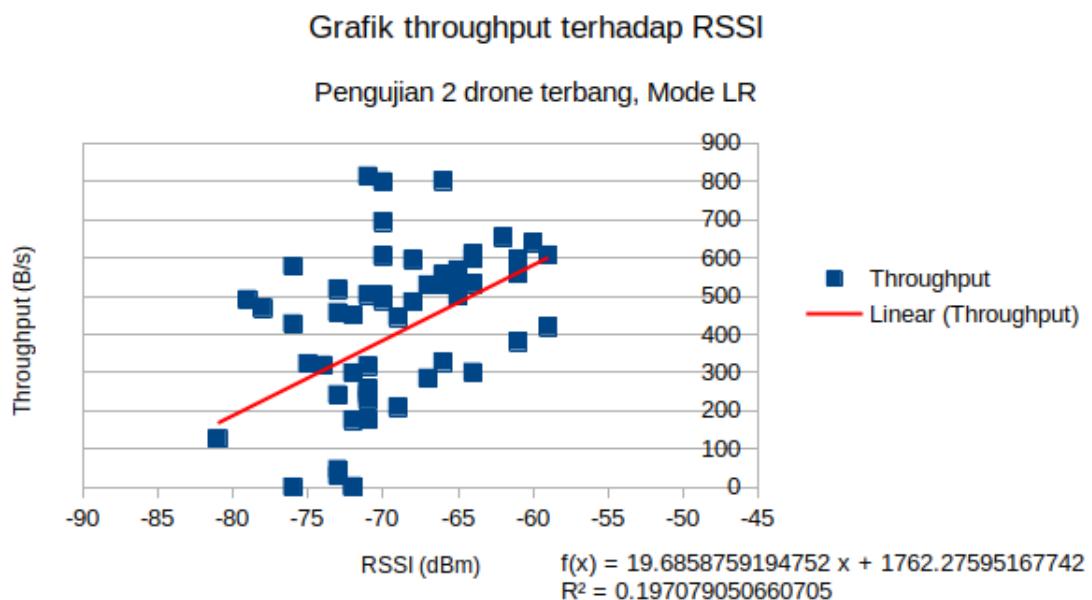
### 4.4 Hasil dan Analisis Pengujian Skenario Komunikasi Dua Drone



Gambar 4.28: Grafik throughput dan RSSI terhadap waktu, mode LR

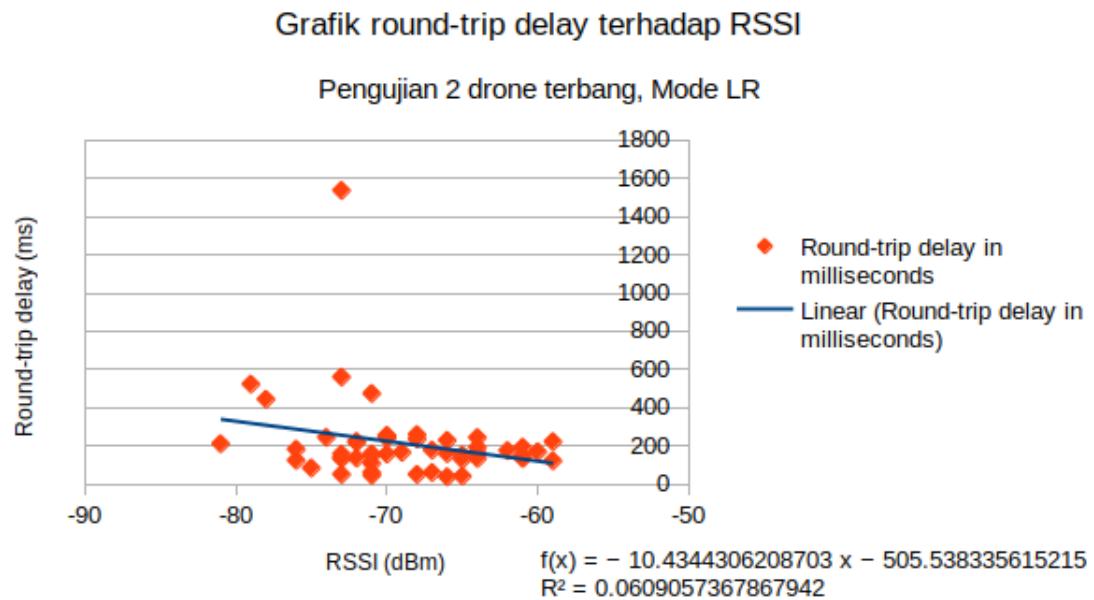
Pada pengujian komunikasi dua drone, drone *sender* diterbangkan terlebih dulu dengan jarak 30 meter dari drone *flying receiver*. Karena posisi antena di drone *flying receiver* yang berada di belakang drone, maka posisi *heading/yaw* drone sangat menentukan koneksi *line-of-sight* antar node jaringan. Ketinggian drone *sender* ditahan di tingkat 10 meter, sedangkan ketinggian drone *flying receiver* bervariasi karena tidak ada fitur *altitude hold*.

Kedua drone diposisikan dengan *heading* menghadap barat. Dengan posisi antena *flying receiver* di belakang drone, maka jaringan terganggu jika posisi drone *sender* lebih ke barat dan di atas drone *flying receiver*. Hal tersebut terjadi pada detik ke-21 pengujian, dimana kedua node gagal berkomunikasi hingga posisi drone *flying receiver* kembali lebih ke barat dibandingkan drone *sender* pada detik ke-140. Jaringan didapatkan nilai RSSI rata-rata -69,31 dBm dengan standar deviasi 5,01 dBm, serta nilai *throughput* rata-rata 397,84 B/s dengan standar deviasi 222,3 B/s.



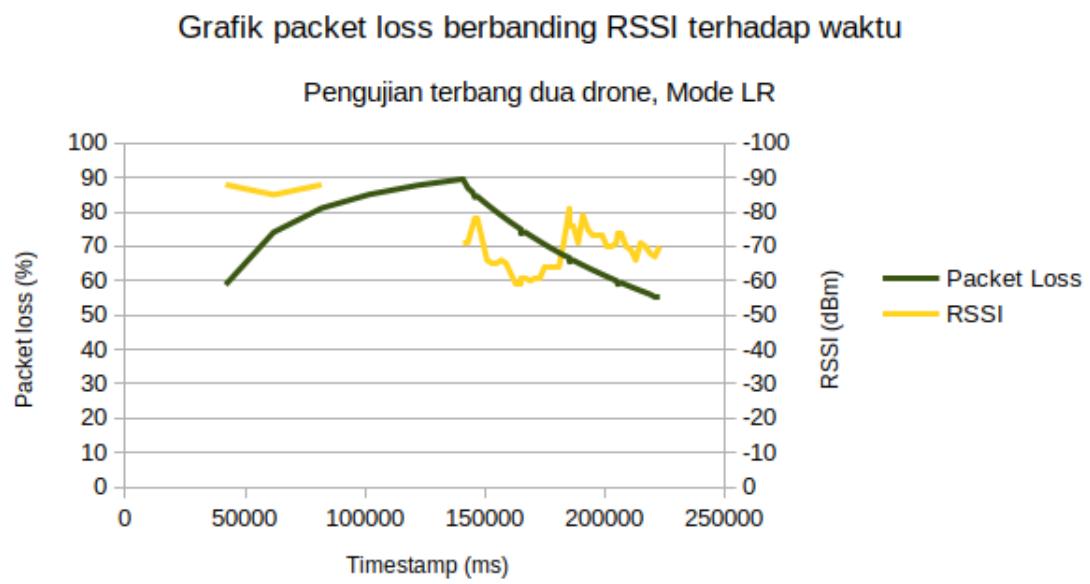
Gambar 4.29: Grafik *throughput* terhadap RSSI, mode LR

Terlihat pada Gambar 4.29 korelasi yang kuat antara *throughput* dengan RSSI, terutama dengan nilai sebaran RSSI yang cukup luas, dengan nilai maksimum RSSI sebesar -59 dBm dan nilai minimum RSSI dimana kedua node masih terhubung sebesar -81 dBm. Menggunakan perhitungan regresi linear, nilai gradien garis tren sebesar 19,69, sehingga dapat diartikan setiap peningkatan 1 dBm RSSI meningkatkan nilai *throughput* sebesar 19,69 B/s.



Gambar 4.30: Grafik *round-trip delay* dan RSSI terhadap waktu, mode LR

Gambar 4.30 menunjukkan korelasi penurunan nilai *round-trip delay* terhadap meningkatnya nilai RSSI. Menggunakan perhitungan regresi linear didapatkan nilai gradien (m) sebesar -10,434, sehingga dapat diartikan setiap peningkatan 1 dBm RSSI menghasilkan nilai *round-trip delay* yang menurun sebesar 10 ms.



Gambar 4.31: Grafik *packet loss* dan RSSI terhadap waktu, mode LR

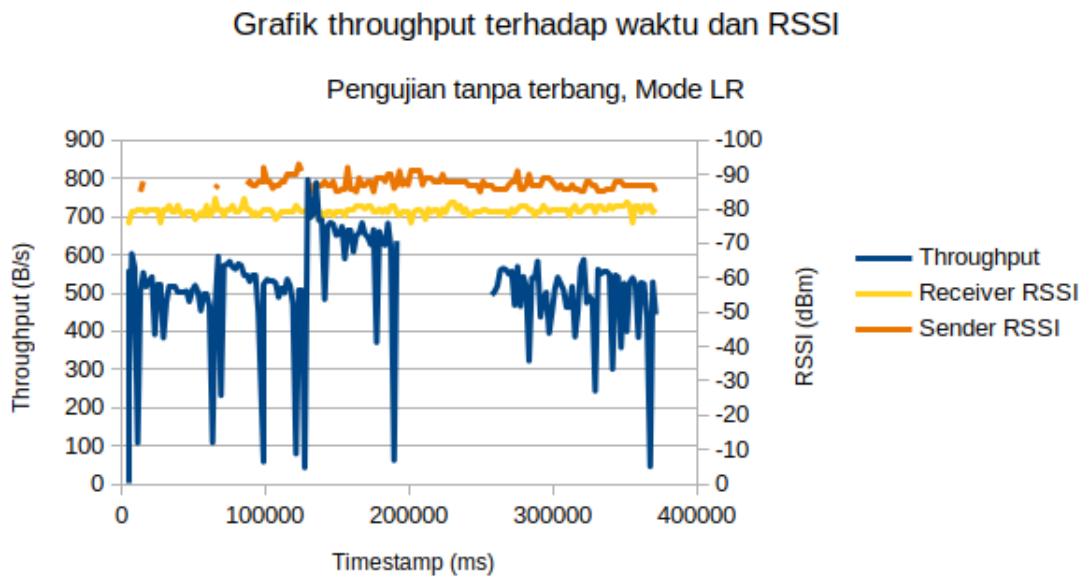
Pada pengujian 2 drone terbang, didapatkan nilai *packet loss* yang tinggi dengan nilai maksimum sebesar 89,55 persen, disebabkan oleh terjadinya *disconnect*

antara kedua node dari detik ke-81 hingga detik ke-140. Pengukuran *packet loss* berlangsung secara independen terhadap kondisi jaringan, sehingga masih ada nilai yang didapat pada saat jaringan masih terhubung namun gagal berkomunikasi, seperti yang terjadi pada pengujian *throughput* dan *round-trip delay* dari detik ke-21 hingga detik ke-81. Setelah detik ke-140 dimana kedua node dapat berkomunikasi kembali, nilai *packet loss* berangsur menurun hingga nilai terendahnya sebesar 55,04 persen.

#### 4.5 Hasil Pengujian

##### 4.5.1 Pengujian Tanpa Terbang

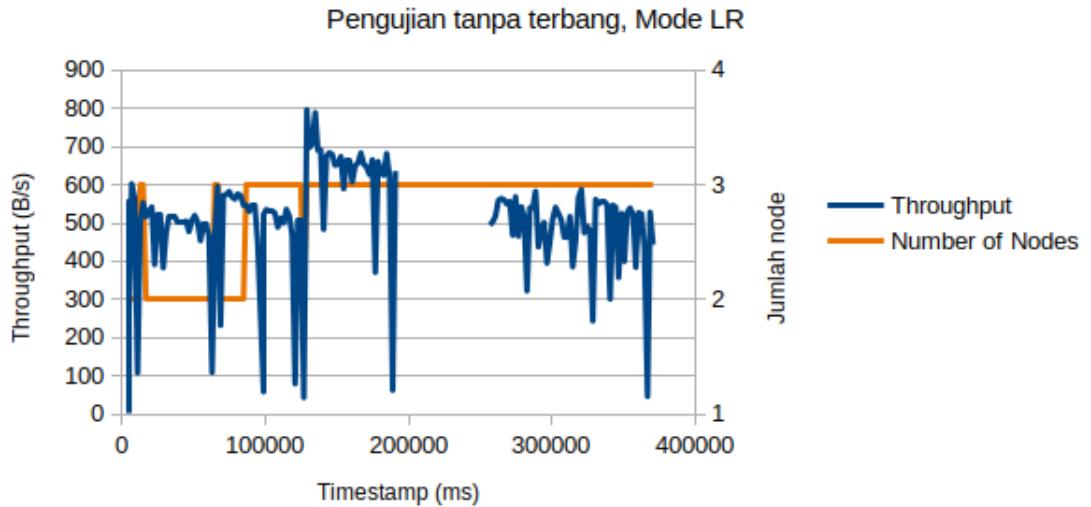
###### 4.5.1.1 Wi-Fi Mode LR



Gambar 4.32: Grafik *throughput* dan RSSI terhadap waktu, mode LR

Gambar 4.32 adalah data yang didapatkan dari pengujian tanpa terbang di Gedung N FTE selama 370 detik menggunakan mode LR dan 3 node jaringan. Pada saat pengujian, nilai *throughput* dari detik ke-192 hingga detik ke-254 tidak valid karena terjadi *overflow* pada algoritma perhitungan *throughput* sehingga nilai *throughput* pada rentang waktu tersebut tidak dimasukkan dalam perhitungan. *Sender RSSI* adalah data kekuatan sinyal dari *sender node* ke *base station*, dan *Receiver RSSI* adalah kekuatan sinyal dari *flying receiver node* ke *sender node*.

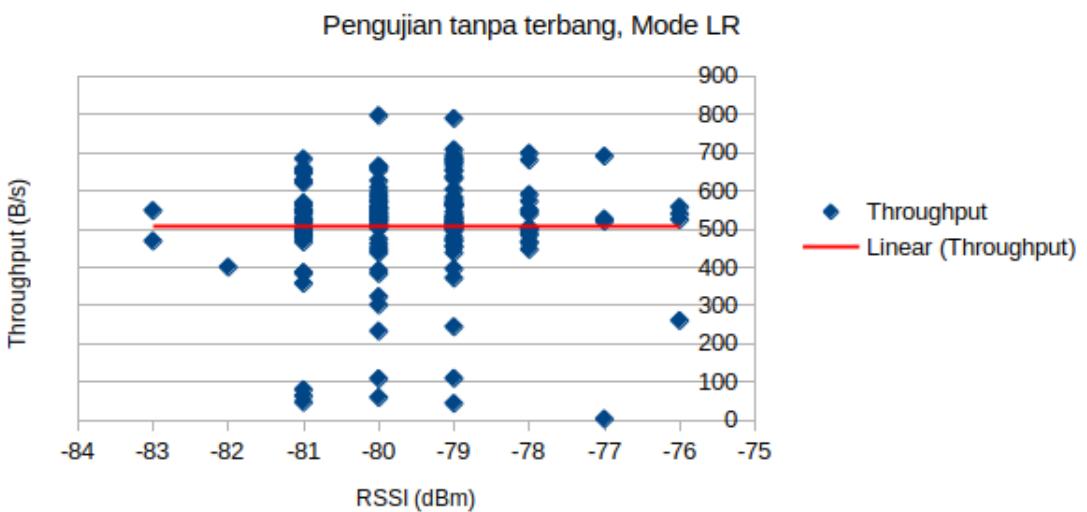
Grafik throughput terhadap waktu dan jumlah node



Gambar 4.33: Grafik *throughput* dan jumlah node terhadap waktu, mode LR

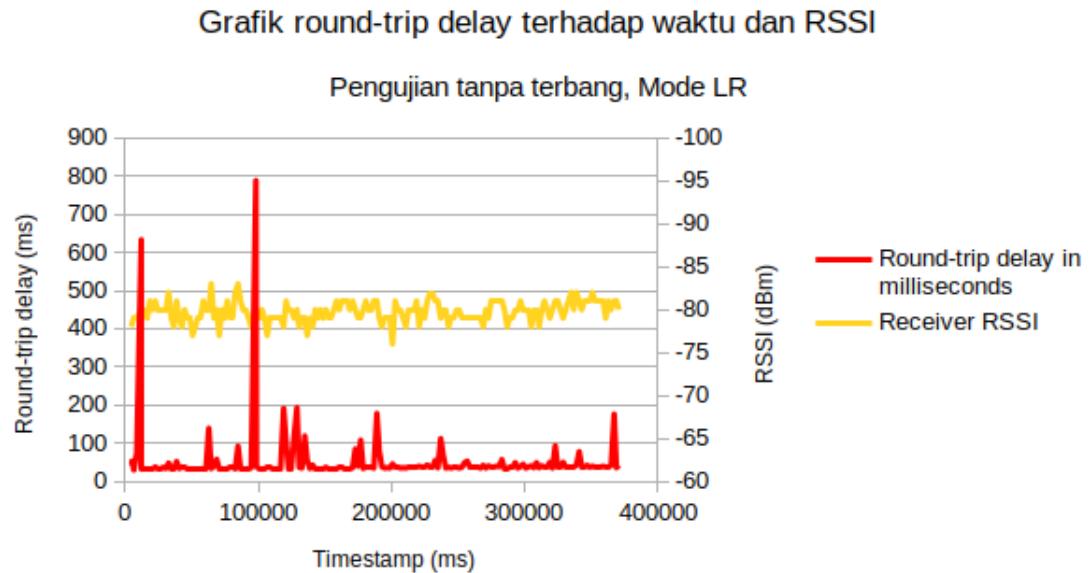
Dengan membandingkan Gambar 4.32 dan 4.33, dapat terlihat pada saat nilai *Sender RSSI* tidak ada, jumlah node yang terhubung di jaringan menurun ke 2 node. Terlihat juga pada Gambar 4.18 bahwa setiap perubahan jumlah node diiringi dengan menurunnya nilai *throughput*, sehingga dapat disimpulkan bahwa proses rekonfigurasi jaringan saat terjadi perubahan node adalah salah satu faktor menuurnya nilai *throughput*.

Grafik throughput terhadap RSSI



Gambar 4.34: Grafik *throughput* terhadap *Receiver RSSI*, mode LR

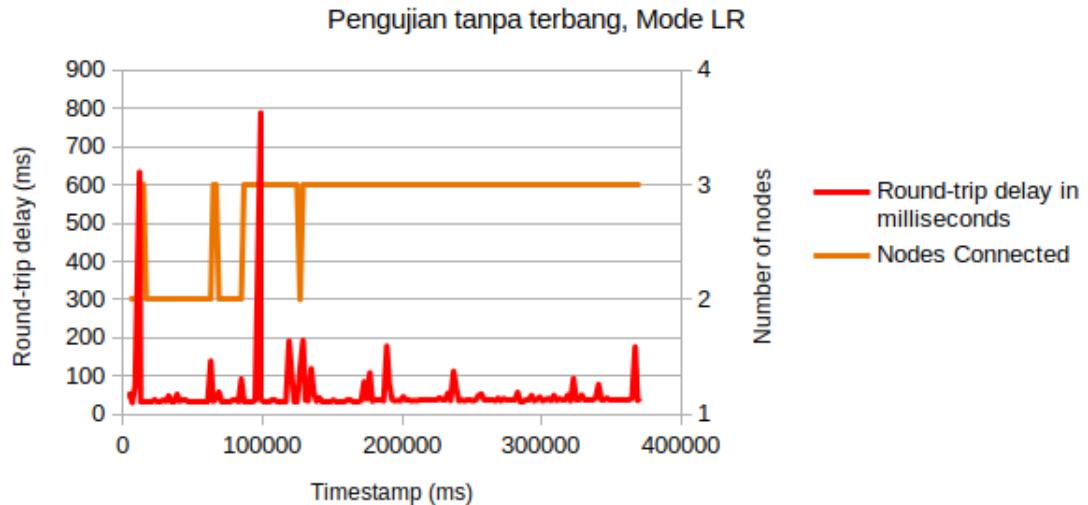
Karena posisi node yang diam di tempat, maka sebaran nilai RSSI tidak drastis dan tidak terlalu berpengaruh terhadap nilai *throughput*. Nilai rata-rata RSSI yang didapat adalah -79,5 dBm dengan standar deviasi 1,25 dBm.



Gambar 4.35: Grafik *round-trip delay* dan *Receiver RSSI* terhadap waktu, mode LR

Gambar 4.35 menunjukkan perubahan nilai *round-trip delay* terhadap waktu. Terlihat bahwa selain beberapa kali terjadi lonjakan *delay*, secara keseluruhan nilai *round-trip delay* cukup stabil dengan nilai rata-rata 52,3 ms dan standar deviasi 74,47 ms.

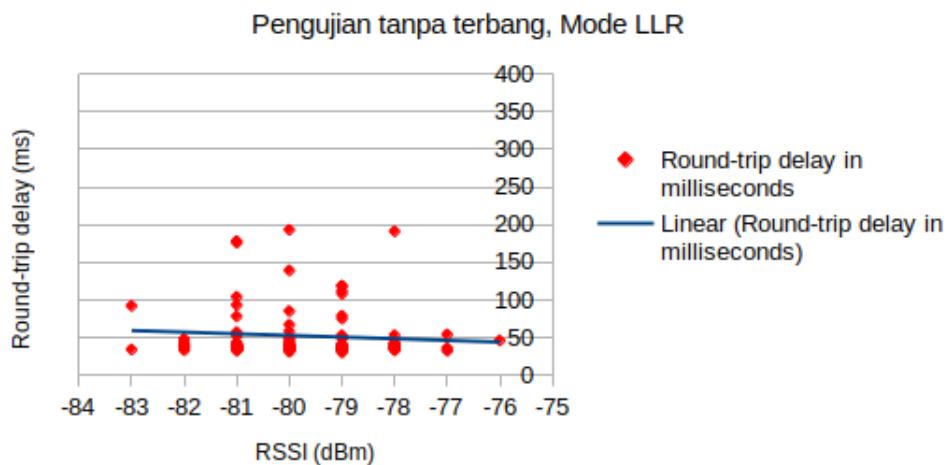
Grafik round-trip delay terhadap waktu dan jumlah node



Gambar 4.36: Grafik *round-trip delay* dan jumlah node terhadap waktu, mode LR

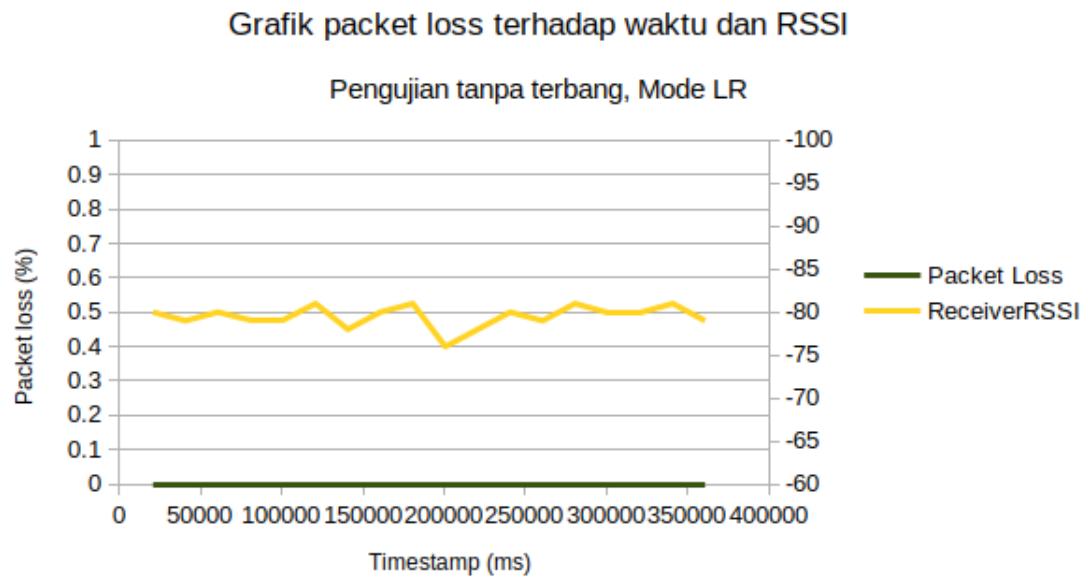
Gambar 4.36 menunjukkan lonjakan *round-trip delay* terbesar terjadi karena proses rekonfigurasi jaringan seiring dengan *connect* dan *disconnect* node-node dalam jaringan.

Grafik round-trip delay terhadap RSSI



Gambar 4.37: Grafik *round-trip delay* terhadap *Receiver RSSI*, mode LR

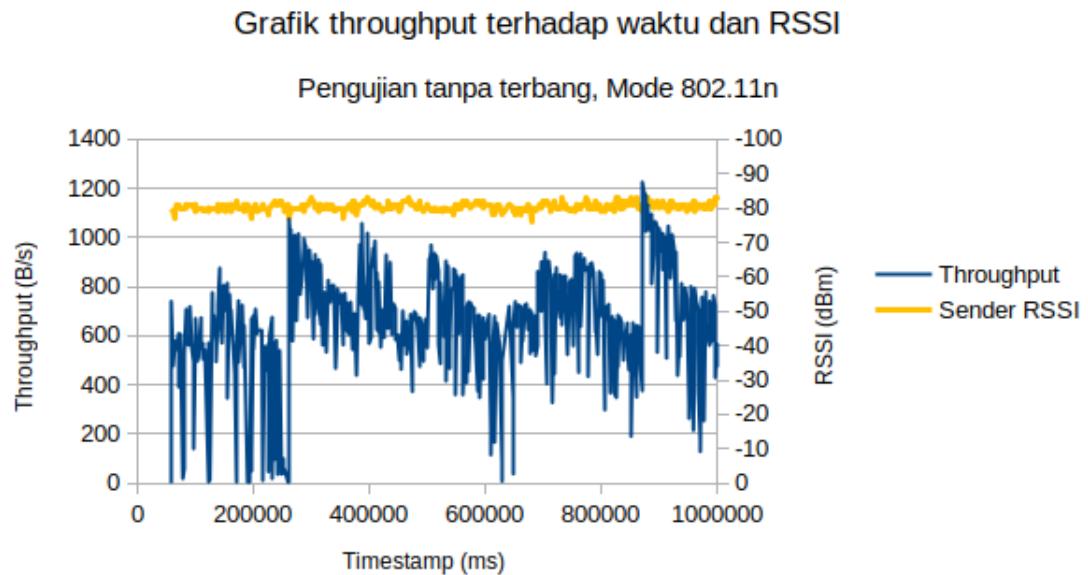
Pada Gambar 4.37, dapat terlihat sebaran nilai RSSI yang sedikit menyebabkan nilai *round-trip delay* tidak banyak berubah, dengan garis tren yang menurun sedikit seiring meningkatnya kekuatan sinyal.



Gambar 4.38: Grafik *packet loss* terhadap waktu, mode LR

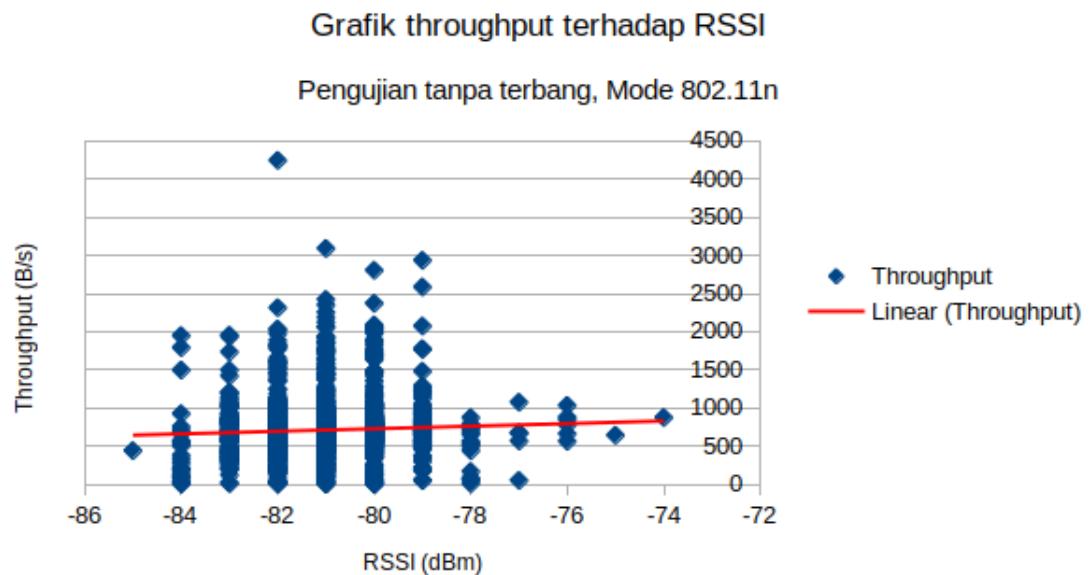
Selama pengujian, *sender node* dan *flying receiver node* tidak pernah terputus, sehingga nilai *packet loss* konstan 0 selama durasi pengujian.

#### 4.5.1.2 Wi-Fi Mode 802.11n



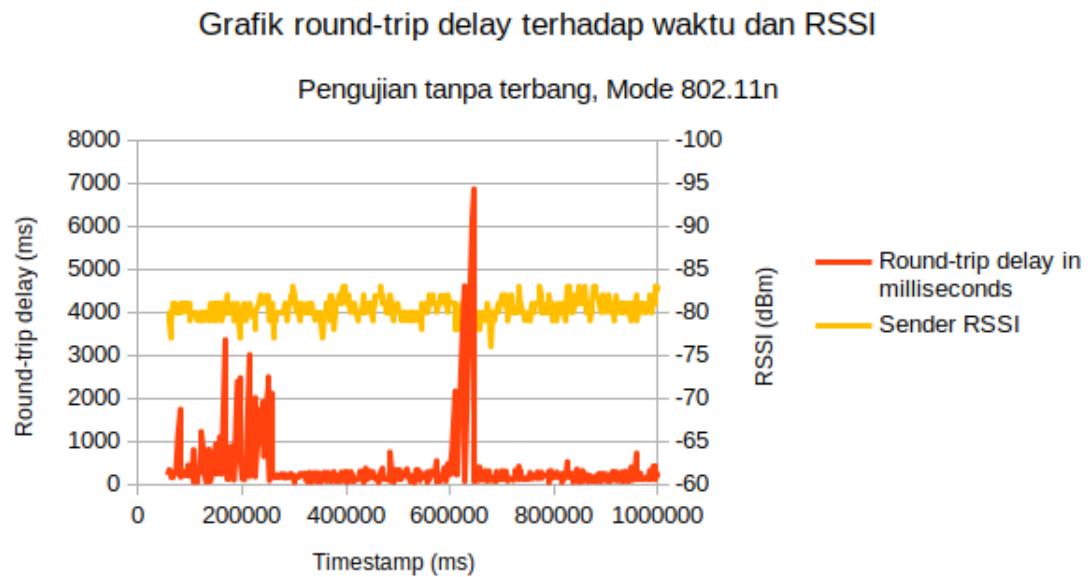
Gambar 4.39: Grafik *throughput* dan RSSI terhadap waktu, mode 802.11n

Pengujian tanpa terbang mode 802.11n dilakukan selama 1000 detik, dengan posisi masing-masing node sama dengan pada pengujian mode LR. Selama pengujian mode 802.11n, node *base station* tidak pernah berhasil bergabung dengan jaringan mesh sehingga pada pengujian ini jaringan mesh hanya memiliki 2 node.



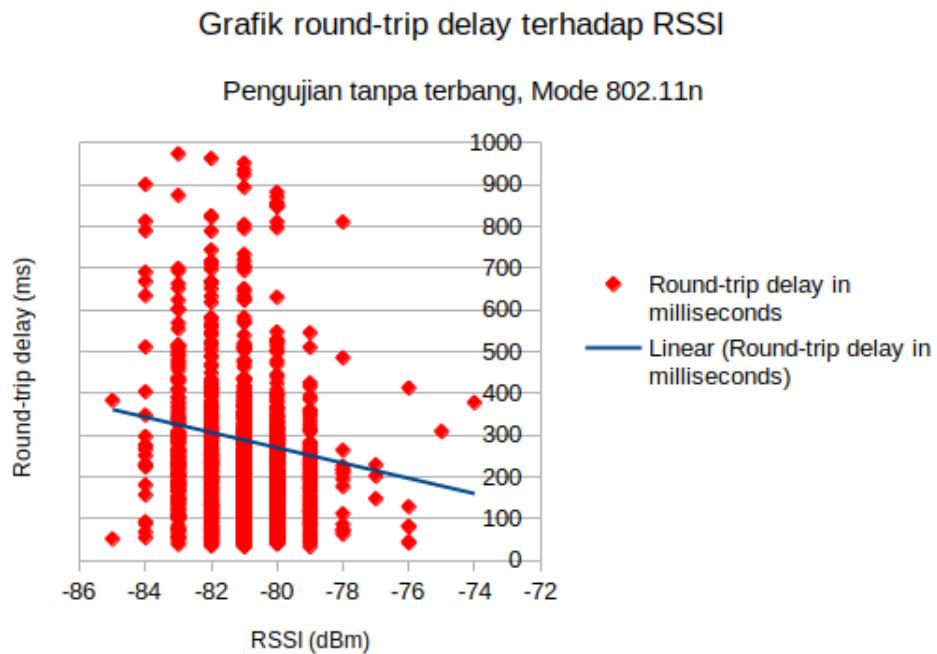
Gambar 4.40: Grafik *throughput* terhadap RSSI, mode 802.11n

Terlihat pada Gambar 4.40 terdapat suatu tren kenaikan *throughput* seiring meningkatnya RSSI, tetapi dengan tidak ada pergerakan node maka sebaran nilai RSSI tidak besar dengan nilai rata-rata RSSI yang didapat adalah -80,93 dBm dan standar deviasi 1,197 dBm.



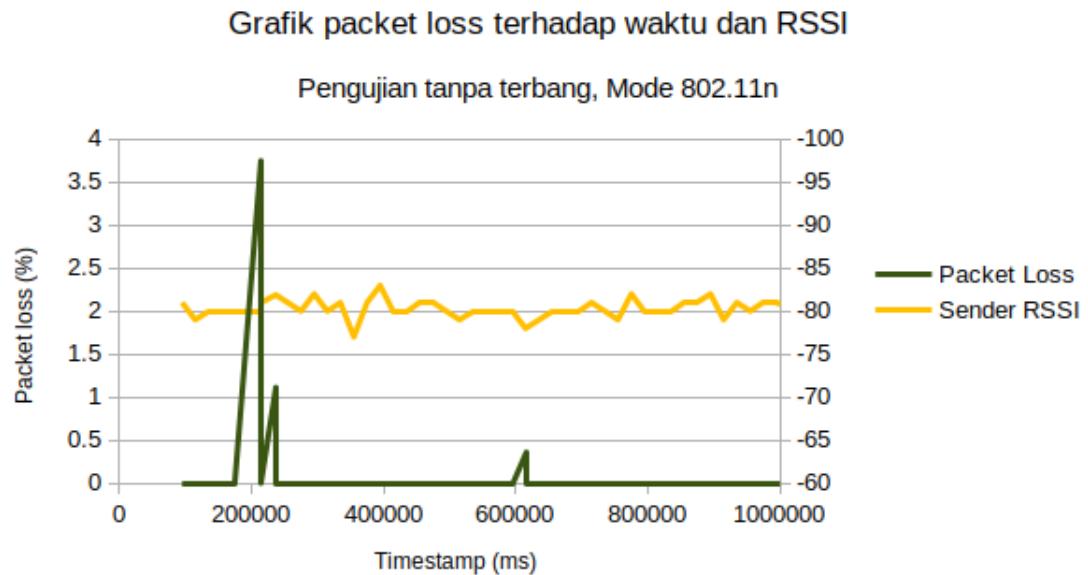
Gambar 4.41: Grafik *round-trip delay* dan *Receiver RSSI* terhadap waktu, mode 802.11n

Pada mode Wi-Fi 802.11n, didapatkan nilai rata-rata *round-trip delay* yang lebih tinggi dibanding mode LR, yakni 287,9 ms dengan standar deviasi 450,72 ms, menunjukkan *delay* yang lebih lama dan tidak sestabil mode LR.



Gambar 4.42: Grafik *round-trip delay* terhadap RSSI, mode 802.11n

Tidak seperti pada mode LR, pada pengujian mode 802.11n, terlihat tren penurunan *round-trip delay* terhadap meningkatnya kekuatan sinyal yang lebih jelas.

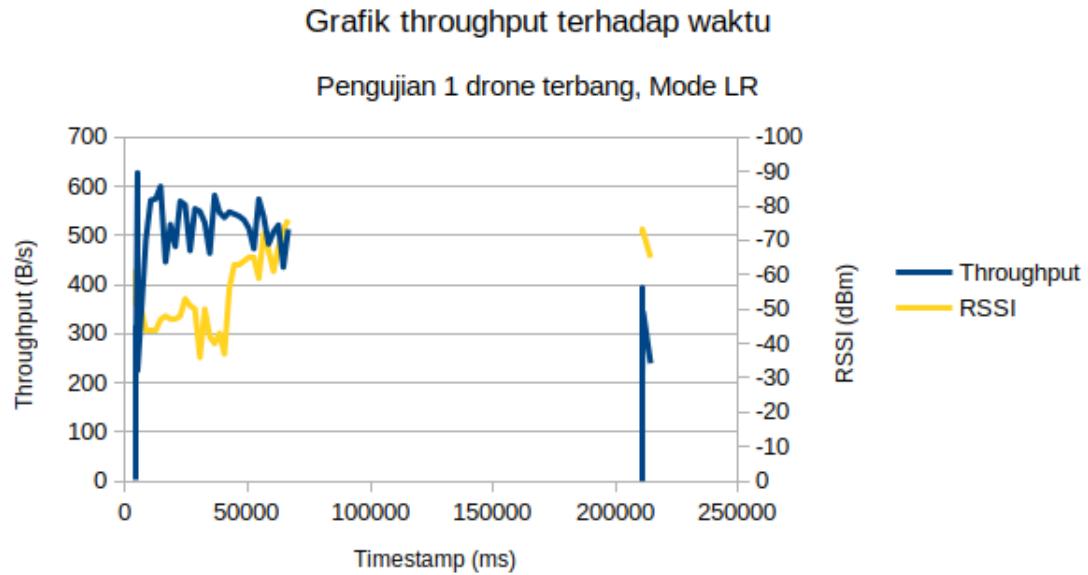


Gambar 4.43: Grafik *packet loss* terhadap waktu, mode LR

Gambar 4.43 menunjukkan pergerakan nilai *packet loss* seiring waktu pada saat pengujian. Sempat terjadi peningkatan *packet loss* pada detik ke-200 beriringan dengan pergerakan nilai kekuatan sinyal. Pengukuran *packet loss* merupakan sebuah indikator yang terlambat (*lagging indicator*) karena dilakukan setiap 10 kali pengiriman permintaan pengukuran *round-trip delay*, sehingga dibutuhkan waktu untuk sebuah peningkatan kualitas jaringan untuk bisa terlihat di data penurunan nilai *packet loss*.

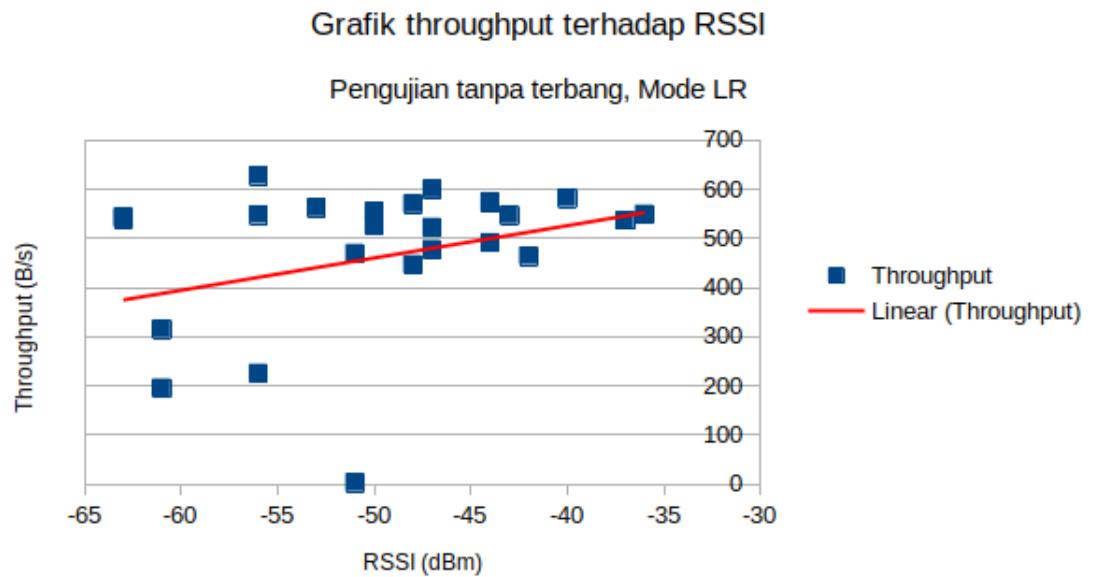
## 4.5.2 Pengujian Terbang Satu Drone

### 4.5.2.1 Wi-Fi Mode LR



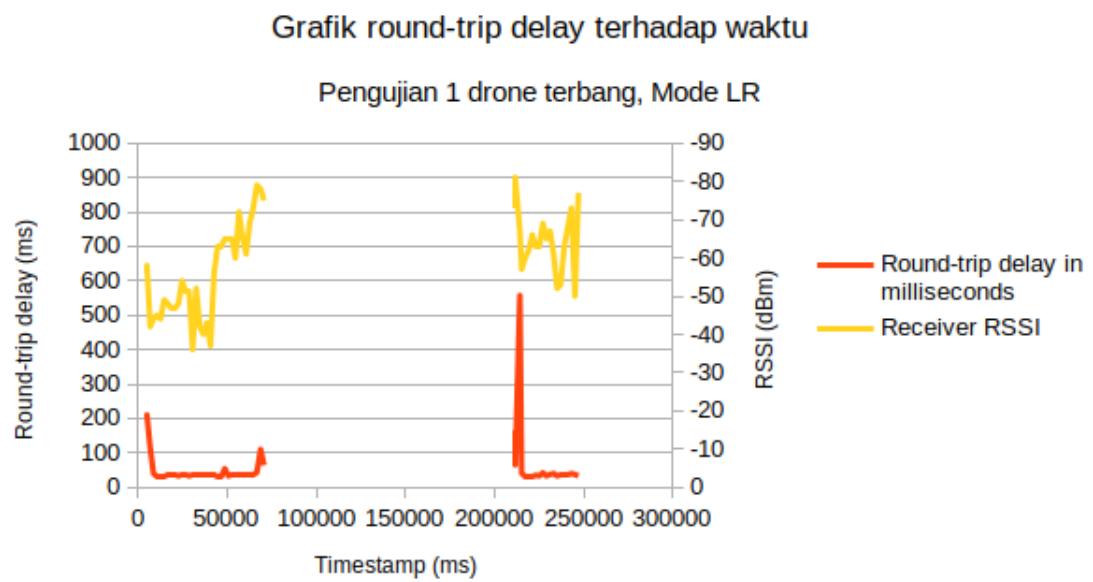
Gambar 4.44: Grafik *throughput* dan RSSI terhadap waktu, mode LR

Sebagaimana yang telah dipaparkan pada Bab 4.2.2, drone *sender* diterbangkan dengan jarak 30-50 meter dari penguji, dengan ketinggian awal 5 meter di atas permukaan tanah selama 50 detik dan naik ke ketinggian 10 meter di atas permukaan tanah pada jarak 30 meter, lalu dijauhkan ke jarak 50 meter pada detik ke-90, lalu kemudian dikembalikan ke jarak 30 meter saat baterai drone mulai habis. Dapat terlihat penurunan kekuatan sinyal pada sekitar detik ke-42 dengan menurunnya nilai RSSI dari -37 dBm ke -63 dBm seiring dengan proses naiknya ketinggian drone, lalu pada akhirnya *disconnect* pada detik ke-66 seiring dengan menjauhnya posisi drone. Dapat terlihat juga proses *reconnect* antara kedua node setelah posisi drone mendekat pada saat baterai drone mulai habis.



Gambar 4.45: Grafik *throughput* terhadap *Receiver RSSI*, mode LR

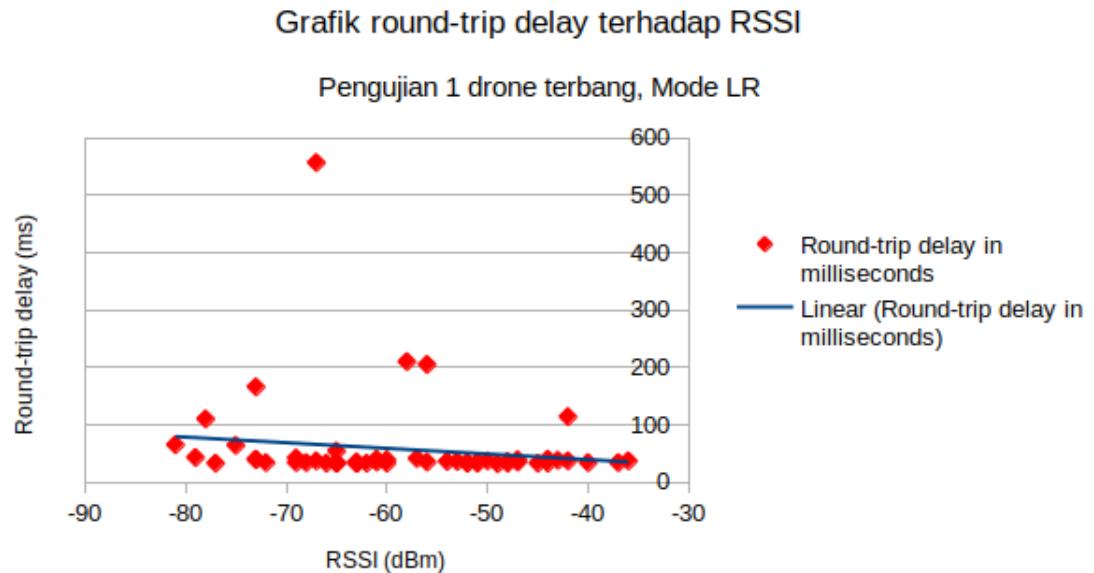
Pada pengujian ini, tren peningkatan *throughput* terhadap peningkatan RSSI lebih terlihat. Didapatkan nilai RSSI rata-rata sebesar -57,23 dBm dengan standar deviasi 11,62 dBm, menunjukkan jaringan yang tidak sestabil pada pengujian tanpa terbang karena jarak antar node yang berubah-ubah.



Gambar 4.46: Grafik *round-trip delay* dan RSSI terhadap waktu, mode LR

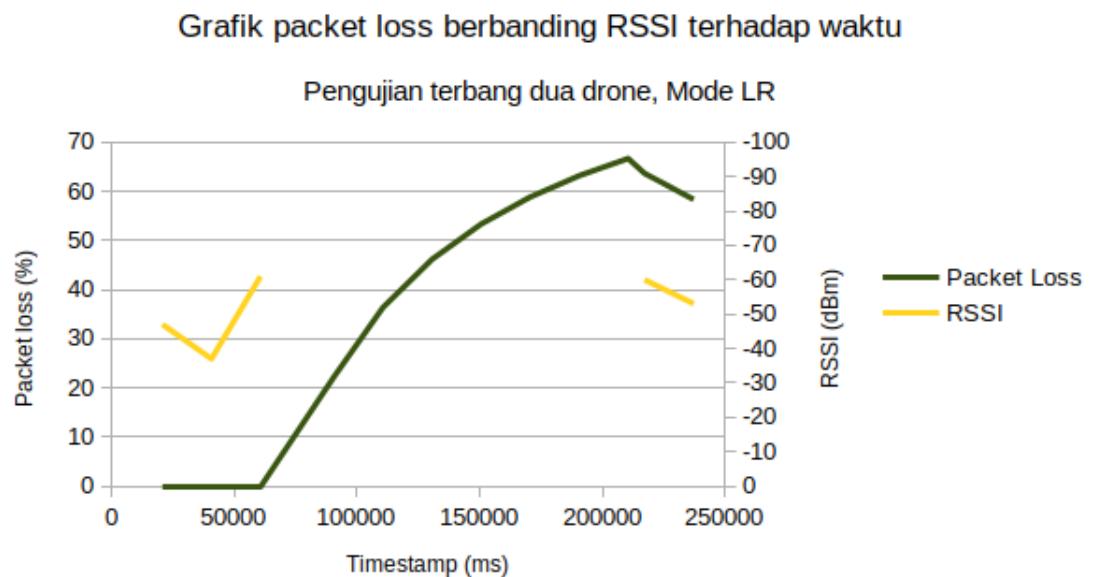
Dapat terlihat pada Gambar 4.46 bahwa nilai *round-trip delay* cukup stabil pada saat fase pengujian jarak 30 meter dan terjadi lonjakan pada saat proses awal koneksi.

si antar node dan rekonfigurasi jaringan mesh, terlihat dengan lonjakan pada awal pengujian dan pada saat kedua node terhubung kembali.



Gambar 4.47: Grafik *round-trip delay* terhadap *Receiver RSSI*, mode LR

Gambar 4.47 menunjukkan sebuah tren penurunan kecil *round-trip delay* terhadap meningkatnya RSSI.

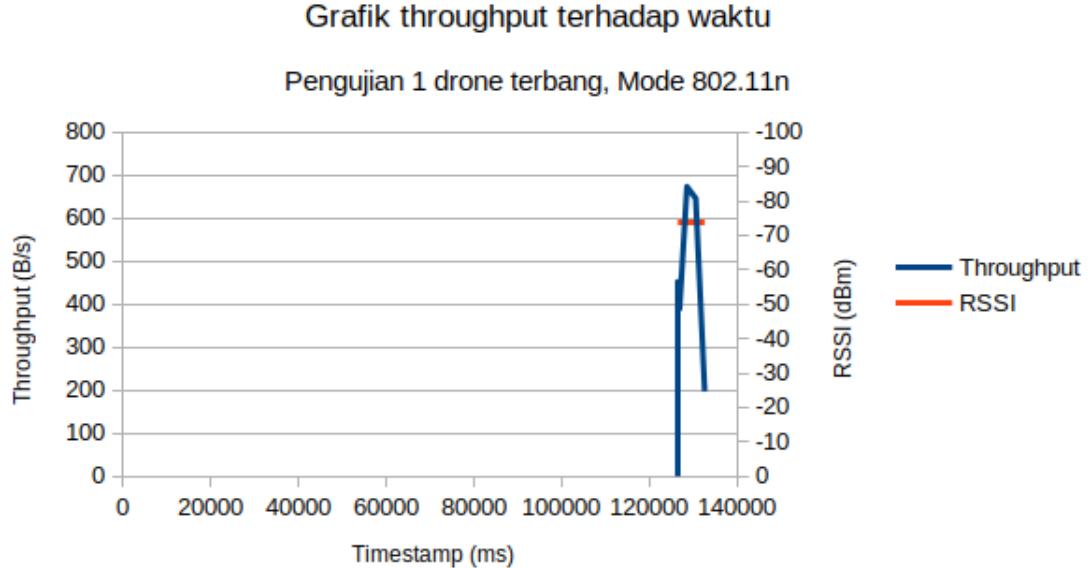


Gambar 4.48: Grafik *packet loss* terhadap waktu, mode LR

Dengan terjadinya *disconnect* antara kedua node, maka nilai *packet loss* me-

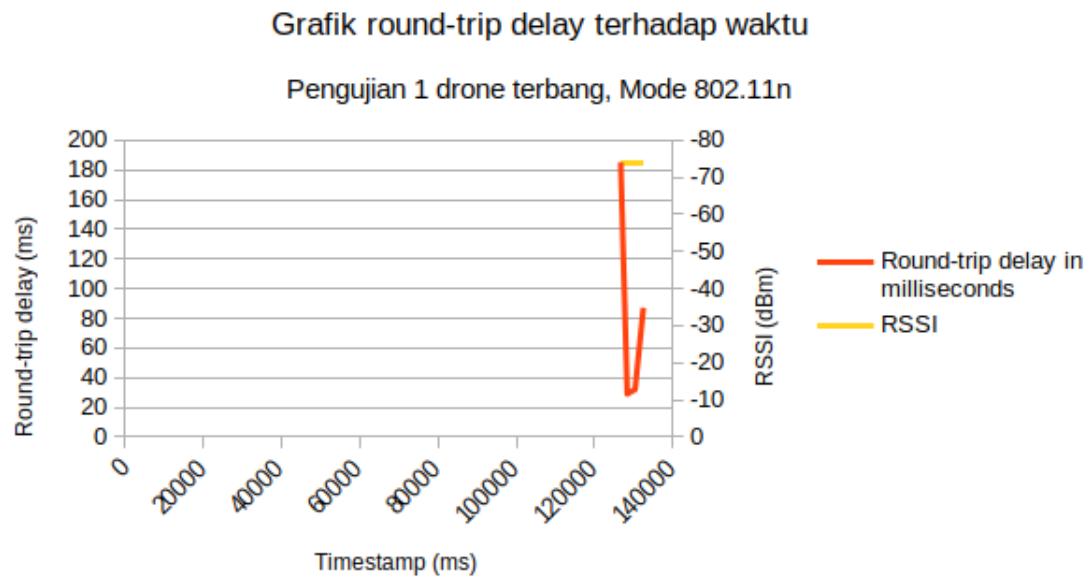
lonjak ke tingkat tertinggi sebesar 66,66 persen. Pengukuran *packet loss* sebagai *lagging indicator* juga terlihat pada Gambar 4.33 dengan nilai *packet loss* baru mulai menurun beberapa saat setelah nilai RSSI meningkat.

#### 4.5.2.2 Wi-Fi Mode 802.11n



Gambar 4.49: Grafik *throughput* dan RSSI terhadap waktu, mode 802.11n

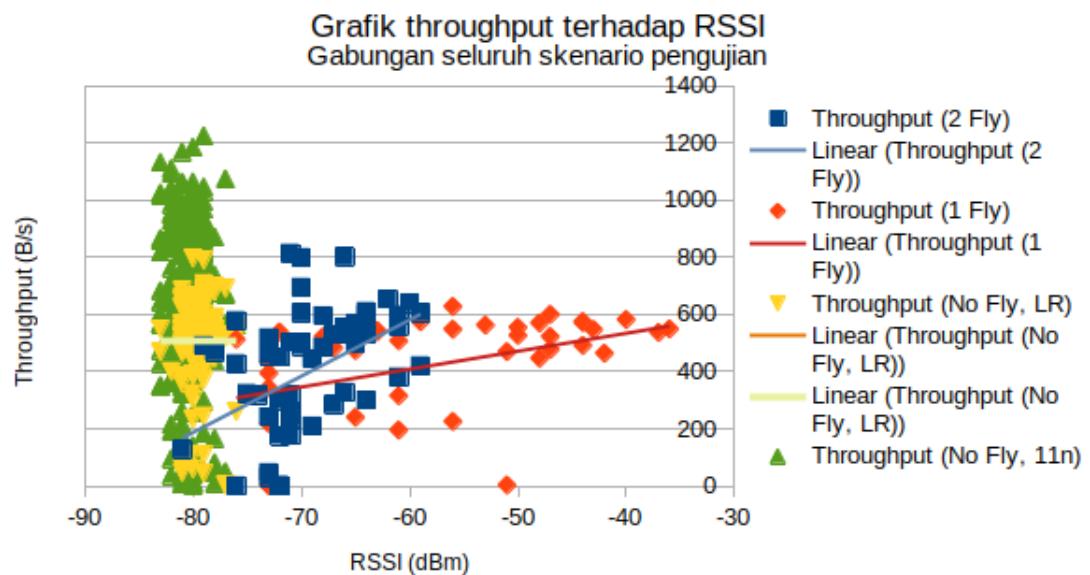
Pengujian skenario 1 drone terbang menggunakan Wi-Fi mode 802.11n menghasilkan sebuah jaringan yang sangat buruk, dengan kedua node tidak dapat terhubung satu sama lain hingga pada akhirnya drone mendekat ke posisi penguji.



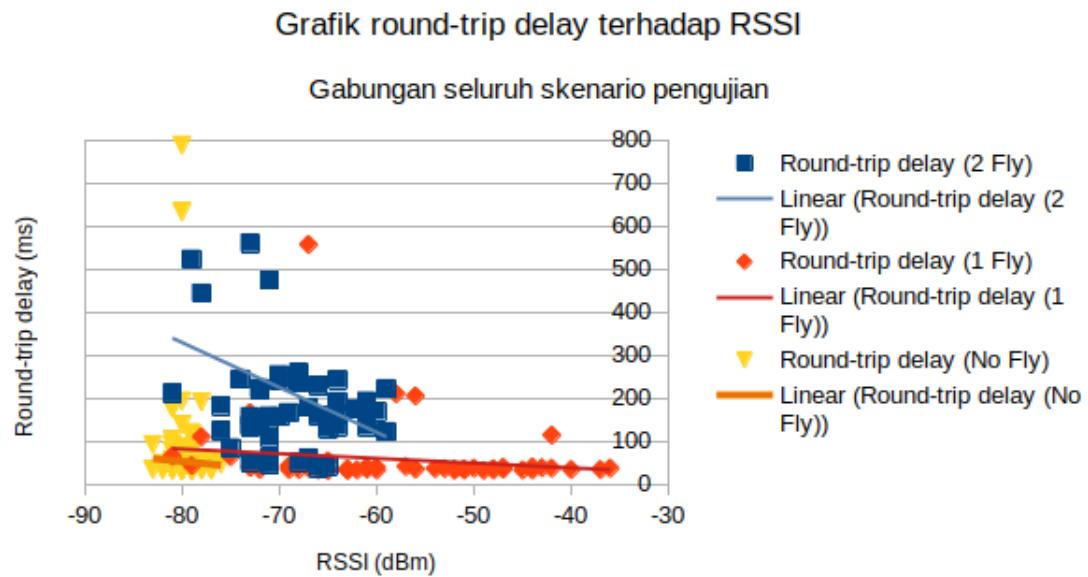
Gambar 4.50: Grafik *round-trip delay* dan RSSI terhadap waktu, mode 802.11n

Dapat terlihat pada Gambar 4.49 dan 4.50 tidak adanya data sebelum detik ke-126 karena gagalnya kedua node untuk berkomunikasi dan membentuk jaringan. Hal ini menunjukkan mode 802.11n tidak cocok untuk digunakan sebagai mode Wi-Fi untuk komunikasi antar-UAV.

#### 4.6 Analisis Hasil Pengujian Keseluruhan



Gambar 4.51: Grafik *throughput* setiap pengujian terhadap RSSI, mode LR.



Gambar 4.52: Grafik *round-trip delay* setiap pengujian terhadap RSSI, mode LR.

Tabel 4.1: Hasil parameter kinerja jaringan dari seluruh skenario pengujian

		Tanpa terbang LR	Tanpa terbang 802.11n	Terbang 1 drone LR	Terbang 2 drone LR
RSSI (dBm)	<b>Median</b>	-80	-80	-56	-70,5
	<b>Mean</b>	-79,508	-80,418	-57,232	-69,310
	<b>Standar deviasi</b>	1,250	1,114	11,625	5,013
Throughput (B/s)	<b>Median</b>	524,934	664,562	512,711	453,255
	<b>Standar deviasi</b>	144,293	257,180	187,624	222,296
Round-trip delay (ms)	<b>Median</b>	37	190,5	35	164
	<b>Standar deviasi</b>	74,472	408,078	78,765	221,426
Packet loss (%)	<b>Maksimum</b>	0	0	66,666	89,552
	<b>Mean</b>	0	0	39,057	70,219

Pada setiap skenario pengujian, dapat terlihat tren kenaikan nilai *throughput* dan menurunnya nilai *round-trip delay* seiring dengan meningkatnya nilai kekuatan si-

nyal (RSSI) jaringan. Didapatkan juga skenario pengujian yang menghasilkan jaringan yang paling stabil secara sebaran nilai RSSI adalah pengujian tanpa terbang dengan nilai RSSI median sebesar -80 dBm dan standar deviasi 1,25 dBm pada mode LR dan 1,114 dBm pada mode 802.11n.

Terlihat juga pada Tabel 4.1 bahwa mode 802.11n memiliki keunggulan yaitu nilai *throughput* yang lebih tinggi dibandingkan pada mode LR. Hal tersebut sesuai dengan dokumentasi ESP dimana *throughput* PHY maksimum mode LR adalah 1/2 Mbps [19]. Namun dengan nilai standar deviasi yang lebih tinggi serta nilai median *round-trip delay* yang lebih tinggi juga, mode 802.11n memiliki kestabilan jaringan yang lebih rendah dibanding mode LR untuk skenario pengujian yang sama.

Dengan jarak antar node yang lebih dekat dibanding pengujian tanpa terbang, dapat terlihat pada pengujian terbang 1 drone dan 2 drone didapatkan nilai median RSSI yang lebih tinggi. Namun karena posisi node yang bergerak-gerak menyebabkan nilai RSSI yang lebih tersebar, terlihat dengan tingginya nilai standar deviasi RSSI pada pengujian terbang. Pergerakan posisi node juga menyebabkan tingginya nilai *packet loss*, hingga 66 persen pada pengujian 1 drone dan 89 persen pada pengujian 2 drone.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1 Kesimpulan

Dari penelitian yang telah dilakukan, dapat disimpulkan bahwa:

1. Board-board ESP32 dapat membuat sebuah jaringan mesh yang mampu mengirimkan pesan berupa JSON *string* menggunakan *library* PainlessMesh.
2. Wi-Fi mode *Long Range* (LR) ESP32 memiliki keunggulan dimana masing-masing node dapat berkomunikasi pada kekuatan sinyal yang lebih buruk dibandingkan Wi-Fi mode 802.11n.
3. Wi-Fi mode 802.11n dapat menghasilkan nilai *throughput* yang lebih tinggi dibandingkan pada Wi-Fi mode LR.
4. Jaringan yang dihasilkan tidak memenuhi kriteria *packet loss*, dengan hasil pengujian terbang 2 drone menghasilkan nilai *packet loss* maksimum sebesar 89,552 %.
5. Jaringan yang dihasilkan memenuhi kriteria *round-trip delay* dan jarak, dengan jaringan masih bisa berkomunikasi dengan jarak antar-node hingga 48 meter pada pengujian tanpa terbang, dan 30 meter pada pengujian terbang. Pada masing-masing pengujian, nilai median *round-trip delay* maksimum adalah 190,5 ms, dibawah nilai kriteria minimum.
6. Wi-Fi mode 802.11n tidak cocok digunakan untuk komunikasi antar-UAV, berdasarkan hasil pengujian terbang 1 drone dengan mode 802.11n dimana kedua node tidak pernah berhasil untuk terkoneksi satu sama lain dan membangun jaringan.
7. Dengan *throughput* median pada mode LR terbesar senilai 524,934 bps, jaringan yang dihasilkan cukup cepat untuk mengirimkan data lokasi berbasis JSON sebesar 12 byte.

#### 5.2 Saran

Untuk perbaikan hasil penelitian agar lebih baik untuk di masa depan, penulis memiliki beberapa saran untuk meningkatkan kinerja sistem yang dihasilkan, yakni:

1. Dilakukan penelitian pemilihan dan penempatan antena untuk dapat menjaga *line-of-sight* antar node jaringan.
2. Gunakan drone *custom* agar implementasi sistem lebih fleksibel dari sisi penempatan antena, catu daya, dan waktu terbang. Penggunaan drone *custom* juga dapat membantu memajukan penelitian ini ke tahap kontrol otonom.

3. Menggunakan mikrokomputer seperti Raspberry Pi untuk menguji skema-skema *routing* FANET yang lebih kompleks.
4. Perbaikan algoritma perhitungan *throughput* untuk mencegah masalah *overflow* pada *timestamp* data.

## DAFTAR PUSTAKA

- [1] N. SYAFITRI, R. SUSANA, I. AMMARPRAWIRA, M. FAUZI, and A. JAB-Baar, ?The Autonomous Disaster Victim Search Robot using the Waypoint Method?, *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 8, p. 347, May 2020. DOI: 10.26760/elkomika.v8i2.347.
- [2] C. Shimanski, *Risks in Mountain Rescue*. Mountain Rescue Association, 2008.
- [3] R. G. Lakshmi Narayanan and O. C. Ibe, ?6 - Joint Network for Disaster Relief and Search and Rescue Network Operations?, in *Wireless Public Safety Networks 1*, D. Câmara and N. Nikaein, Eds., Elsevier, Jan. 2015, pp. 163–193, ISBN: 978-1-78548-022-5. DOI: 10.1016/B978-1-78548-022-5.50006-6.
- [4] *Drones for Search and Rescue — Learn How and Why They are Used*, <https://flytnow.com/drones-for-search-rescue/>.
- [5] M. A. Khan, A. Safi, I. M. Qureshi, and I. U. Khan, ?Flying ad-hoc networks (FANETs): A review of communication architectures, and routing protocols?, in *2017 First International Conference on Latest Trends in Electrical Engineering and Computing Technologies (INTELLECT)*, Karachi: IEEE, Nov. 2017, pp. 1–9, ISBN: 978-1-5386-2969-7. DOI: 10.1109/INTELLECT.2017.8277614.
- [6] X. Lin, V. Yajnanarayana, S. D. Muruganathan, *et al.*, ?The Sky Is Not the Limit: LTE for Unmanned Aerial Vehicles?, *IEEE Communications Magazine*, vol. 56, no. 4, pp. 204–210, Apr. 2018, ISSN: 0163-6804, 1558-1896. DOI: 10.1109/MCOM.2018.1700643.
- [7] A. M. Townsend and M. L. Moss, *Telecommunications Infrastructure in Disasters: Preparing Cities for Crisis Communications*. New York University, Apr. 2005.
- [8] *Precision of coordinates - OpenStreetMap Wiki*, <https://wiki.openstreetmap.org/wiki/Precision>
- [9] L. Santos, P. Nascimento, L. Bento, R. Machado, P. Ferrari, and C. Amorim, ?Use of High Mobility Nodes to Improve Connectivity in Wireless Sensor Networks?, in Jan. 2021, pp. 528–545, ISBN: 978-3-030-63091-1. DOI: 10.1007/978-3-030-63092-8\_36.
- [10] Y. Chia, R. Arjadi, E. Setyaningsih, P. Wibowo, and M. Sudrajat, ?Performance Evaluation of ESP8266 Mesh Networks?, *Journal of Physics: Conference Series*, vol. 1230, p. 012023, Jul. 2019. DOI: 10.1088/1742-6596/1230/1/012023.

- [11] M. Manvi and S. Maakar, ?Implementing Wireless Mesh Network Topology between Multiple Wi-Fi Powered Nodes for IoT Systems?, vol. 7, pp. 2395–0056, Oct. 2020.
- [12] Z. Guo, X. Ma, P. Zhang, and Z. Liu, ?A dust sensor monitoring system using Wi-Fi mesh network?, *Journal of Physics: Conference Series*, vol. 1754, no. 1, p. 012 129, Feb. 2021, ISSN: 1742-6588, 1742-6596. DOI: 10.1088/1742-6596/1754/1/012129.
- [13] V. Chamola, P. Kotesh, A. Agarwal, N. Gupta, M. Guizani, and N. Naren, ?A Comprehensive Review of Unmanned Aerial Vehicle Attacks and Neutralization Techniques?, *Ad Hoc Networks*, vol. 111, Oct. 2020. DOI: 10.1016/j.adhoc.2020.102324.
- [14] T. J. Wheat J. Hiser R., *Designing a Wireless Network*, 1st. Syngress, 2001, ISBN: 1928994148,192899427X,1928994458,192899458X. [Online]. Available: <https://libgen.fun/book/index.php?md5=101ea841867848bb45e62ad5e4ab4071>
- [15] B. A. Forouzan, *Data Communications and Networking* (McGraw-Hill Forouzan Networking), 4th ed. McGraw-Hill Higher Education, 2007, ISBN: 9780072967753,0072967753 [Online]. Available: <https://libgen.fun/book/index.php?md5=f90841d02431af5010fb9cea31665e4e>.
- [16] *All the Internet of Things - Episode One: Transports*, en-US. [Online]. Available: <https://learn.adafruit.com/alltheiot-transports/bluetooth-btle> (visited on 08/16/2022).
- [17] *Cellular Network Architecture — Cellular Operators Association of India*. [Online]. Available: <https://www.coai.com/indian-telecom-infocentre/telecom-infrastructurenetworks> (visited on 08/16/2022).
- [18] *Wi-Fi Driver - ESP32 - — ESP-IDF Programming Guide latest documentation*. [Online]. Available: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/wifi.html> (visited on 01/12/2022).
- [19] *Wi-Fi Driver - ESP32 - — ESP-IDF Programming Guide latest documentation*, <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/wifi.html>.
- [20] *Home · Wiki · painlessMesh / painlessMesh*, <https://gitlab.com/painlessMesh/painlessMesh/-/wikis/home>.
- [21] *Mesh protocol · Wiki · painlessMesh / painlessMesh*, <https://gitlab.com/painlessMesh/painlessMesh/-/wikis/mesh-protocol>.
- [22] *GPS.gov: GPS Accuracy*. [Online]. Available: <https://www.gps.gov/systems/gps/performance/accuracy/> (visited on 08/16/2022).

[23] *GPS - NMEA sentence information*, [http://aprs.gids.nl/nmea/.](http://aprs.gids.nl/nmea/)