TESIS

DISASTER MULTI-ROBOT COMMUNICATION WITH MQTT AND COAP PROTOCOL

Muhammad Ikrar Yamin
NRP. 1020151009

DOSEN PEMBIMBING

Dr. Ir. Son Kuswadi
Sritrusta Sukaridhoto, S.T., Ph.D.

PROGRAM STUDI MAGISTER TERAPAN
TEKNIK ELEKTRO
PROGRAM PASCASARJANA TEKNOLOGI REKAYASA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
2019

TESIS

DISASTER MULTI-ROBOT COMMUNICATION WITH MQTT AND COAP PROTOCOL

Muhammad Ikrar Yamin
NRP. **1020151009**

DOSEN PEMBIMBING

Dr. Eng. Son Kuswadi
Sritrusta Sukaridhoto, S.T., Ph.D.

PROGRAM STUDI MAGISTER TERAPAN
TEKNIK ELEKTRO
PROGRAM PASCASARJANA TEKNOLOGI REKAYASA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
2019

HALAMAN PENGESAHAN

DISASTER MULTI-ROBOT COMMUNICATION WITH MQTT AND COAP PROTOCOL

Oleh:
Muhammad Ikrar Yamin
NRP. 1020151009

Tesis ini digunakan sebagai salah satu syarat untuk memperoleh
Gelar Magister Terapan (M.T.)
di
Politeknik Elektronika Negeri Surabaya
2019

Disetujui oleh:

Pembimbing Utama	:	Dr. Ir. Son Kuswadi NIP. 196201151988031003	
Pembimbing Pendamping	:	Sritrusta Sukaridhoto, S.T., Ph.D. NIP. 197903062002121002	( )
Penguji	:	Amang Sudarsono, S.T., Ph.D. NIP. 197409202002121001	( )
Penguji	:	Dr. Eng. Bima Sena Bayu D, S.ST.,M.T. NIP. 197612151999031003	( )
Penguji	:	M. Udin Harun Al Rasyid, S.Kom., Ph.D. NIP. 198108082005011001	( )

Mengetahui,

Ketua Program Pascasarjana
Politeknik Elektronika Negeri Surabaya

Iwan Syarif, S.Kom., M.Sc.,Ph.D.
NIP. 196904041995121002

HALAMAN PERNYATAAN ORISINALITAS

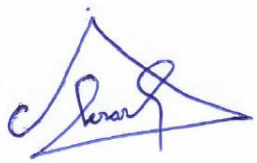
Dengan ini saya menyatakan bahwa bagian atau keseluruhan tesis ini:

1. adalah hasil karya sendiri dan tidak mengandung unsur plagiat dari pihak lain
2. tidak pernah diajukan untuk mendapatkan gelar akademis pada suatu Perguruan Tinggi
3. tidak pernah dipublikasikan atau ditulis oleh pihak lain
4. mencantumkan rujukan dan kutipan dengan jujur dan benar terhadap sumber referensi lain yang menunjang pembahasan pada tesis.

Apabila ditemukan bukti bahwa pernyataan saya diatas tidak benar, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Politeknik Elektronika Negeri Surabaya.

Surabaya, 30 Januari 2019

Yang menyatakan,



(Muhammad Ikrar Yamin)

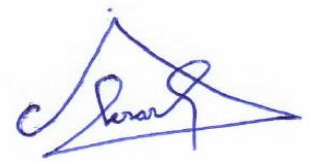
HALAMAN KESEPAKATAN PUBLIKASI

Demi pengembangan ilmu pengetahuan, dengan ini saya menyatakan:

1. memberikan persetujuan kepada Politeknik Elektronika Negeri Surabaya untuk menyimpan, mengolah dalam bentuk pangkalan data, merawat, mengalihmedia/formatkan dan mempublikasikan tesis ini selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.
2. tidak melakukan alihmedia/format dan publikasi dalam bentuk makalah ilmiah dari bagian atau keseluruhan tesis ini ke suatu publikasi ilmiah, pada seminar ataupun jurnal, skala nasional ataupun internasional, kecuali ada persetujuan dari saya dan Dosen Pembimbing Utama, dan mencantumkan nama saya, Dosen Pembimbing Utama dan nama-nama lain (jika ada) yang berkontribusi pada makalah.

Surabaya, 30 Januari 2019

Yang menyatakan,



(Muhammad Ikrar Yamin)

KATA PENGANTAR

Segala puji dan syukur ke hadirat Allah *Subhanahu Wa Ta'ala* atas rahmat, taufik dan hidayah hingga penulis dapat menyelesaikan penyusunan penelitian yang berjudul **"Disaster Multi-robot Communication with MQTT and CoAP Protocol".** Sholawat dan salam kepada Rasulullah *Shollallohu 'alaihi wassalam*, utusan Allah *Subhanahu Wa Ta'ala* yang membawa manusia dari zaman kegelapan menuju zaman terang benderang. Penelitian ini disusun dalam rangka penyusunan Tesis yang menjadi salah satu persyaratan untuk memperoleh gelar Magister Terapan (M.T.) di Politeknik Elektronika Negeri Surabaya.

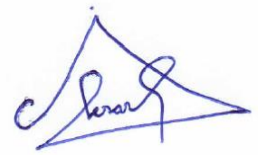
Dalam penyelesaian tesis ini, penulis banyak mendapatkan bantuan dari berbagai pihak. Untuk itu penulis menyampaikan ucapan terima kasih setulusnya kepada:

1. Bapak Dr. Eng. Son Kuswadi, selaku Dosen Pembimbing Utama yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga penyelesaian tesis ini.
2. Bapak Sritrusta Sukaridhoto, S.T., Ph.D., selaku Dosen Pembimbing Pendamping yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga tesis ini.
3. Bapak Dr. Zainal Arief, S.T., M.T., selaku Direktur Politeknik Elektronika Negeri Surabaya atas segala fasilitas yang telah disediakan.
4. Bapak Iwan Syarif, S.Kom., M.Kom., MSc., Ph.D., selaku Ketua Program Pasca Sarjana atas segala fasilitas yang telah disediakan.
5. Bapak Dosen Penguji yang telah memberikan saran dan masukan dalam penyempurnaan dan pengembangan tesis ini, Bapak Amang Sudarsono, S.T., Ph.D., Bapak Dr. Eng. Bima Sena Bayu Dewantara, S.ST., M.T., Bapak M. Udin Harun Al Rasyid, S.Kom., Ph.D., Bapak Dr. Eng. Indra Adji Sulistijono, ST., M.Eng., Bapak Novie Ayub Windarko, ST., MT., Ph.D., Bapak Dr. Sanggar Dewanto, Bapak Dr. Ir. Endra Pitowarno, M.Eng., Bapak Dr. Eng. I Gede Puja Astawa, ST., MT., Bapak Riyanto Sigit, ST., M.Kom., PhD.

6. Bapak dan Ibu dosen pengajar dan staf progam Pascasarjana Terapan Politeknik Elektronika Negeri Surabaya yang telah memberikan ilmu dan bantuan selama masa studi, terimakasih banyak atas ilmu dan bantuan yang diberikan selama ini.
7. Teman-teman Magister Terapan Teknik Elektro angkatan 2015: Indra Ferdiansyah, Safriudin Rifandi, Cindha Riri Pratiwi, Rafina Destiarti Ainul, Ibu Nofriyani, Gezaq Abror, Ocsirendi, Novrian Eka Sandhi, Pak Djoko Santoso dan Pak Justiawan. Terimakasih atas pertemanan yang terjalin selama ini.
8. Teman-teman Magister Terapan Teknik Elektro angkatan 2016 dan 2017: Dimas Pristovani, Bayu, Angga, Pak Ridwan, Toar dan Putu. Terimakasih atas bantuannya dalam banyak hal.
9. Teman-teman seperjuangan Magister Terapan Teknik Informatika dan Komputer angkatan 2015: Pak Rengga Asmara, Muh.Subhan, Munsyi, Ubaidillah Umar, Khotibul Umam, Adhe Widianjaya, Irsal Shabirin, Yesta Medya Mahardhika, Galih Hendra Wibowo, Ardiansyah Alfarouq, Angga Rahagiyanto dan Tresna Maulana Fahrudin. Telah memberikan pengalaman dan menjadi teman seperjuangan.
10. Anggota laboratorium lantai 9 (Iseng4h 2017/2018) dan anggota *Robotics and Automation Based on Biologically-Inspired Technology* (RABBIT) atas bantuan, dukungan dan pertemanan selama penelitian dan penyelesaian tesis ini.
11. Bapak dan Ibu tercinta (Bapak Alm. Ir. Sudjirin MSME, Bapak Suparno dan Ibu Ernawati), yang terus mendoakan dan mendukung penulis hingga saat ini. Semoga rahmat Allah *Subhanahu Wa Ta'ala* selalu tercurahkan kepada beliau.
12. Kakak saya tercinta dan suami (Rina Fatma Aisyah dan Muhammad Yasin, Ike Indri Fatimah dan Rusdi Wijaya), dan adik saya tersayang dan suami (Detty Fitri Anti dan Ahmed), serta adik-adik saya terkasih (Sucik Indah Apri Setiyani dan Ahmad Zainuri Fachri) atas segala dorongan dan motivasi untuk menjadi lebih baik.
13. Semua pihak yang tidak dapat disebutkan satu-persatu atas semua dukungannya.

Semoga Allah *Subhanahu Wa Ta'ala* senantiasa memberikan berkat dan anugrah-Nya berlimpah bagi beliau-beliau yang tersebut di atas. Sangat disadari dalam tesis ini terdapat banyak kekurangan oleh karena itu semua saran dan kritik penulis terima dengan lapang dada demi kesempurnaan penulisan tesis ini. Akhirnya harapan penulis semoga tesis ini bermanfaat bagi kita semua.

Surabaya, 30 Januari 2019



Penulis,

ABSTRAK

Multi-robot dapat mengambil peran penting dalam area bencana untuk mencari dan menyelamatkan korban. Diperlukan komunikasi yang baik di antara robot untuk melakukan tugas mereka dengan cepat dan efisien. Komunikasi dapat digunakan antara operator dan multi-robot atau di antara multi-robot itu sendiri. Proses ini berhubungan dengan *queque message*. Dalam penelitian ini, kami menerapkan protokol komunikasi pada *queque message* dengan topologi *mesh* dan mengintegrasikannya pada *platform* robot. Seperti yang kita ketahui, perkembangan protokol komunikasi tumbuh cepat seperti perkembangan teknologi IoT (*Internet of Things*). MQTT dan CoAP termasuk protokol komunikasi yang digunakan untuk kebutuhan IoT. Kedua protokol tersebut digunakan dan diimplementasikan ke dalam *disaster multi-robot* dan kinerja mereka dibandingkan dari segi jumlah data yang diterima, *error* dan *transfer rate*nya. Kami juga mengintegrasikan protokol komunikasi tersebut ke dalam *platform* robot berbasis python. Hasilnya menunjukkan bahwa protokol MQTT lebih mudah untuk diimplementasikan pada *platform disaster multi-robot* pada topologi *mesh* daripada CoAP, dan bahwa *transfer rate* data protokol MQTT lebih tinggi sebesar 98.28 % daripada CoAP, dengan selisih eror 100 % dan total data yang diterima 98.28 % lebih besar dibanding CoAP.

Kata Kunci

Bencana, Mesh, Robot, Protokol komunikasi, CoAP, MQTT, Internet of Things

ABSTRACT

Abstract—Multi-robot can take an important role in a disaster area to search and rescue victims. It needs good communication among the robots to perform their tasks quickly and efficiently. Communication can be used between the operator and multi-robot or among the multi-robot themselves. This relates with process in the queue message. In this research, we implemented the queue message protocol on mesh topology and integrated it on the robot platform. As we know, the development of communication protocol grows fast like the development of IoT (Internet of Things) Technology. MQTT and CoAP are among the communication protocols used for IoT needs. Both protocols were used and implemented into disaster multi-robot and their performances were compared. We also integrated the protocol into robot platform python based. The result shows that MQTT protocol is easier to be implemented on to disaster multi-robot platform on mesh topology than CoAP, and that the data transfer of the MQTT protocol is higher 98.28 than CoAP, with an error difference of 100% and the total data received by MQTT is 98.28% greater than CoAP.

Keywords: Disaster, Mesh, Robot, Communication Protocol, CoAP, MQTT, Internet of Things

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1. LATAR BELAKANG

Indonesia merupakan salah satu negara yang sering terjadi bencana. Selama kurun waktu 5 tahun antara tahun 2010 – 2014 jumlah kejadian bencana di Indonesia mencapai 1.907 kejadian bencana, terdiri dari 1.124 bencana alam, 626 bencana non alam dan 157 bencana social [1]. Hal yang sulit ketika bencana itu terjadi adalah mencari dan menyelamatkan para korban yang selamat atau menemukan korban-korban yang telah meninggal di area bencana. Diperlukan waktu yang terus menerus dan teknologi yang mendukung hal tersebut.

Salah satu cara pemanfaatan teknologi mutakhir dalam menanggunggulangi bencana adalah menggunakan robot. Salah satu pelopor dalam bidang robot penyelamat untuk bencana di perkotaan, adalah tim yang dipimpin Prof. John Blitch dari University of South Florida, yang memimpin CRASAR (*Center for Robotic Assisted Search and Rescue*) dalam rangka untuk operasi pencarian dan penyelamatan di World Trade Center New York, segera setelah kejadian penyerangan teroris 11 September 2001 [2]. Gambar 1.1 adalah contoh robot yang digunakan dalam operasi tersebut.

Sumber: Casper, J., and Murphy, 2003[3]

Gambar 1.1. Robot-robot yang digunakan dalam operasi penyelamatan korban WTC 11 September 2001; (a) ”Foster-Miller Solem” dan (b) Talon

Robot memiliki keunggulan dalam hal mampu menjangkau daerah yang sulit dan berbahaya bagi manusia dan dapat dipersiapkan secara cepat serta mampu melakukan kegiatan penyelamatan seperti pencarian, pengujian, inspeksi (dalam skala terbatas). Pengembangan robotika untuk operasi penyelamatan bertumpu pada dua kegiatan, yaitu pengembangan *platform hardware* seperti robot beroda, robot ular, robot berkaki atau yang sejenisnya [4] dan perangkat lunak yang mendukung kinerja robot tersebut [5]. Diantara pengembangan perangkat lunak meliputi pembuatan perangkat lunak dalam pengendalian robot, kerjasama multi-robot dan lainnya.

Multi-robot adalah sebuah jenis kumpulan robot yang dapat dimanfaatkan untuk mengidentifikasi medan bencana. Komunikasi diantara anggota robot dapat mewujudkan pendelegasian tugas sehingga target dapat tercapai seperti mencari korban untuk menjangkau daerah yang lebih luas dalam waktu yang lebih singkat, sehingga dapat menyelamatkan lebih banyak nyawa korban bencana alam maupun bencana akibat keteledoran manusia.

Dalam membantu proses pencarian dan pertolongan pertama pada korban bencana alam dibuat yang terdiri dari 3 buah robot, 3 buah robot tersebut masing-masing memiliki peran antara lain 1 robot sebagai *leader* dan 2 robot sebagai *follower*. Peran-peran ini diberikan agar dalam penerapannya kinerja robot menjadi lebih efisien dari pada tim SAR (*Search And Rescue*) maupun *single* robot. Apabila kelompok robot ini dijalankan secara bersama, dibutuhkan suatu sistem yang berfungsi sebagai sarana antar robot untuk berkomunikasi dan berkoordinasi agar multi-robot ini dapat bertindak sesuai dengan *task* yang harus dilakukan di lingkungan yang telah mengalami bencana alam.

Kondisi lingkungan yang harus dilalui oleh multi-robot merupakan daerah yang memiliki medan yang sangat berat dan sulit untuk dijangkau serta memiliki sumber daya listrik yang sangat terbatas sehingga teknologi nirkabel sangat diperlukan dalam sistem jaringan. Dari permasalahan tersebut penulis memberikan solusi dengan menerapkan teknologi *Wireless Mesh Network* (WMN) pada jaringan multi-robot. WMN memiliki kelebihan seperti *self organized* and *self configured* dimana WMN dapat membangun dan menpertahankan konektivitas antar multi-robot

. Pertimbangan dalam memilih protokol komunikasi yang tepat pada *disaster*

multi-robot merupakan landasan penelitian ini yaitu menerapkan protokol komunikasi yang digunakan dalam teknologi *Internet of Things* (IoT) pada komunikasi multi-robot pada sebuah model area bencana. Kemudian penelitian ini membandingkan kinerja di antara kedua protokol komunikasi tersebut.

1.2 PERMASALAHAN

Untuk mewujudkan multi-robot yang dapat bekerja pada daerah bencana dan dikendalikan kapan saja dan di mana saja, maka ada beberapa permasalahan yang perlu dipertimbangkan diantaranya:

1. Bagaimana merancang dan membuat sistem jaringan pada multi-robot agar bisa saling berkomunikasi dalam medan pasca-bencana?
2. Bagaimana kinerja protokol komunikasi dari teknologi *Internet of Things* yang sesuai dengan kebutuhan komunikasi *disaster multi-robot*?

1.3 TUJUAN

Tujuan tesis ini adalah merancang dan membangun jaringan komunikasi *disaster multi-robot* dan membandingkan kinerja 2 buah protokol komunikasi pada teknologi *Internet of Things* yang digunakan pada *disaster multi-robot* di area bencana.

1.4 MANFAAT

Manfaat yang diharapkan dari penelitian tesis ini yaitu sebagai berikut: Pada bidang penelitian yaitu mengimplementasikan protokol komunikasi MQTT dan CoAP pada *disaster multi-robot* sehingga koordinasi dan pengiriman informasi dari multi-robot dapat dilakukan. Multi-robot tersebut dapat digunakan oleh tim SAR (*Search And Rescue*) untuk membantu dalam menemukan korban dan memberikan pertolongan pertama berupa pemberian minum. Informasi dari multi-robot tersebut

juga dapat menjadi pertimbangan bagi badan yang berwenang untuk menghasilkan keputusan-keputusan strategis setelah bencana terjadi.

1.5 SISTEMATIKA PENULISAN

Sistematika pembahasan dalam penyusunan buku tesis ini adalah sebagai berikut:

Bab 1 Pendahuluan

Bab ini berisi tentang pendahuluan yang terdiri dari latar belakang, permasalahan dan batasan masalah, tujuan, manfaat, serta sistematika pembahasan dari tesis. Pada latar belakang diuraikan hal-hal yang mendasari peneliti untuk melakukan penelitian, penyusunan masalah dalam rumusan masalah dan tujuan yang ingin dicapai dalam penelitian serta manfaat penelitian.

Bab 2 Kajian Pustaka

Bab ini membahas mengenai teori–teori yang berkaitan dengan penyelesaian tesis, yang didapatkan dari berbagai macam buku serta sumber–sumber terkait lainnya yang berhubungan dengan pembuatan tesis ini.

Bab 3 Desain Sistem

Bab ini berisi tentang desain sistem yang dibuat termasuk desain komunikasi, desain skenario implementasi dan skenario pengujian, serta spesifikasi perangkat keras dan perangkat lunak yang digunakan.

Bab 4 Eksperimen dan Analisis

Bab ini menyajikan dan menjelaskan seluruh hasil eksperimen dan analisa dalam pembuatan tesis ini dan bagaimana penyelesaian dari setiap permasalahan yang terjadi.

Bab 5 Penutup

Bab ini berisi kesimpulan dari uji coba perangkat keras maupun perangkat lunak, dan saran untuk pengembangan, perbaikan serta penyempurnaan terhadap sistem yang telah dibuat.

Halaman ini sengaja dikosongkan

BAB 2
KAJIAN PUSTAKA

2.1 TEORI PENUNJANG

2.1.1 *Swarm Robot*

Swarm Robot adalah sebuah pendekatan untuk koordinasi sistem multi-robot dimana terdiri dari mobil yang fisiknya sederhana. Pendekatan ini muncul seiring dengan studi biologi tentang serangga, semut dan sejenisnya di alam dimana tingkah laku *swarm* terjadi. Komponen kunci dalam sistem adalah komunikasi diantara agen-agen di dalam grup dimana hubungan secara lokal dengan normal dan menjamin sistem memiliki skalabilitas dan *robust*.

2.1.2 Keunggulan *Swarm Robot*

Beberapa kelebihan *swarm robot* dibanding *single robot* adalah:

1. Paralel

   Jumlah populasinya besar sehingga dapat menghadapai beberapa target dalam satu *task.*

2. Skalabilitas

   Interaksi *swarm* bersifat lokal dalam arti mengizinkan individu-individu untuk bergabung atau keluar dalam *task* kapan saja.

3. Stabil

   Swarm robot tidak tidak terpengaruh ketika bagian dari kelompok tersebut berhenti karena suatu faktor tertentu.

4. Ekonomis

   Swarm robot memiliki biaya yang rendah dalam desain, manufaktur dan perawatan sehari-hari.

5. Energi yang efesien

   Swarm robot membutuhkan energi yang lebih kecil dibanding *single robot* karena lebih kecil dan sederhana sehingga *life time*-nya semakin besar.

2.1.3 Aplikasi dari *Swarm Robot*

Beberapa bidang potensial untuk aplikasi *swarm robot* sebagai berikut:

1. *Monitoring* sebuah area
2. Menyelesaikan tugas yang berbahaya bagi manusia. Sebagai contoh, Murphy dkk [6] merangkum penggunaan robot dalam penyelamatan dan *recovery* di area tambang.
3. Menyelesaikan tugas dalam hal *scaling* populasi

Stormont [7] menjelaskan potensi untuk menggunakan *swarm robot* untuk bereaksi di daerah bencana di 24 jam pertama. Dia merangkum *swarm robot* yang dapat mencari korban dengan probabilitas tertinggi untuk menemukan korban yang selamat, dan membuat beberapa saran untuk penelitian di masa depan di daerah ini.

4. Sistem *Swarm Robot* dalam kehidupan nyata

Correl [8] mengusulkan sebuah sistem *swarm robot* untuk inspeksi pisau-pisau pada jet turbin. Lab Senseable City milik MIT [9] telah mengembangkan sebuah armada *swarm robot* yang dapat menyerap minyak yang disebut *Seaswarm* untuk membaca keadaan laut dan menghilangkan minyak yang tumpah.

2.1.4 Task yang Fundamental dari *Swarm Robot*

1. *Flocking Strategy* dan *Formation*

Flocking adalah tugas dasar yang penting dalam *swarm robot*. Lawton dkk [10] menyajikan pendekatan berbasis perilaku untuk manuver formasi. Mereka menguraikan pembentukan manuver formasi yang kompleks ke manuver yang berurutan diantara pola-pola formasi. Manuver tersebut menyajikan tiga strategi kontrol formasi untuk berurusan dengan topologi dan tujuan.yang berbeda.

2. *Directed Flocking*

Selain strategi berkelompok, kontrol arah dalam berkelompok adalah penelitian yang paling diberi perhatian dan telah diadopsi secara luas dalam aplikasi-aplikasi navigasi, migrasi dan pencarian. Sampai saat ini, sejumlah besar penelitian telah

dilakukan dalam hal mengarahkan kawanan dengan posisi target tertentu dan menyebarkan informasi dalam *swarm*.

3. *Positioning and Navigation*

a. Navigasi

Dalam berkelompok dan migrasi, *posistioning* tujuan, robot di dekatnya dan berbagai rintangan di medan juga merupakan tugas yang penting. Dalam aplikasi di lingkungan luar, *positioning* global merupakan hal yang mahal dan membutuhkan banyak *hardware* dan tidak terjangkau untuk *swarm robot*. Dengan demikian, *positioning* dalam berkelompok harus khusus dan terfokus. Marjovi dkk [11]. mengusulkan metode navigasi dengan membimbing *swarm* menggunakan koneksi nirkabel ketika sumber bau dicari. Setidaknya tiga robot dalam *swarm* bertindak sebagai *beacon* yang menyiarkan koordinat ke seluruh kawanan untuk mempertahankan sistem koordinat global sementara yang lain mencari bau. Kelemahan dari penelitian ini adalah bahwa penyiaran koordinat *beacon* dalam daerah yang besar sementara robot lain harus mendeteksi jarak *beacon* dari jarak jauh, yang memerlukan perangkat keras yang mahal.

b. **Simulasi *Ant Colonies***

Koloni semut di alam terkenal dalam hal perilaku untuk navigasi dan migrasi dengan bantuan feromon. Para peneliti dari robotika *swarm robot* menggunakan skema tersebut dengan mensimulasikan feromon menggunakan bagian dari robot di *swarm* yang berfungsi sebagai *beacon*. Sebuah studi yang menarik diusulkan oleh Sperati dkk [12]. Dalam percobaan mereka, segerombolan robot berhasil secara kolektif mengeksplorasi lingkungan, membentuk jalur untuk menavigasi antara dua wilayah target, yang terlalu jauh dirasakan oleh satu agen pada waktu yang sama. Robot terus bergerak bolak-balik antara dua lokasi sementara mereka berinteraksi dengan tetangga mereka. Perilaku robot yang dikendalikan oleh jaringan saraf dan kawanan berevolusi untuk mengoptimalkan rute jalan. Mereka mengamati bahwa segerombolan akhirnya menuju ke jalan terpendek. Dalam pekerjaan selanjutnya mereka mengusulkan simulasi *Ant Colonies*.

4. Menghindari penghalang

Min, dkk[13]. Mengusulkan metode baru untuk menghindari hambatan/penghalang dalam lingkungan yang dinamis dengan sebuah model matematika berdasarkan tujuan robot, kecepatan dan arah hambatan dan dioptimalkan menggunakan PSO.

2.1.5 Algoritma Pencarian *Swarm Robot*

1. Inspirasi dari *swarm intelligence algorithms*

Particle Swarm Optimization (PSO) adalah *swarm intelligence algorithms* yang diadopsi sebagian besar pada *swarm* robotik karena kemiripan pada skema dalam berkelompok dan pencarian. Selain PSO, metode lain juga banyak terinspirasi dari pendekatan seperti *Ant Colony Optimization* (ACO) dan *Glowworm Swarm Optimization* (GSO). Ruang lingkup pendekatan ini meliputi pencarian jalan, navigasi, lokalisasi bau,dan lain-lain. Ada 3 tipe metode dalam menggunakan *swarm intelligence algorithms* yaitu:

1. Pengoptimalan parameter-parameter
2. *Modelling* perilaku individu. Pada tipe ini setiap robot dianggap sebagai sebuah partikel atau agen pada *swarm intelligence algorithm*. Lingkungan pencarian biasanya diinterpretasikan sebagai nilai *fitness*. *Swarm* menggunakan *fitness* tersebut untuk mencari target.
3. Gabungan dari kedua metode di atas

2. Inspirasi dari metode lainnya

Diantara inspirasi dari metode lain adalah penciuman berbasis algoritma, *foraging* atau pencarian makanan yang berfokus pada efisiensi energi [14].

2.1.6 *Wireless Mesh Network* (WMN) pada *Disaster Multi-robot*

Ada banyak faktor untuk melakukan komunikasi yang efektif di antara *disaster multi-robot*. Penggunaan wifi adalah salah satu solusi untuk menyelesaikan masalah dengan kawat karena panjang dan tidak fleksibel. Ada banyak teknologi nirkabel

yang digunakan untuk mengendalikan *mobile robot* jarak jauh seperti bluetooth, 3G, dan Wi-Fi. Pemilihan teknologi nirkabel dalam *physical layer* dalam komunikasi tergantung pada jenis aplikasi yang akan dikembangkan dengan mempertimbangkan hal berikut: range, frekuensi dan *data rate* [15]. Dalam *transport layer*, protokol komunikasi yang sesuai perlu mempertimbangkan hal-hal yang berkaitan dengan penggunaan baterai, area cakupan, *delay*, dan *bandwidth*, dan lain-lain. Multi-agent seperti *disaster multi-robot* perlu memilih topologi jaringan dan protokol komunikasi terbaik berdasarkan kondisi dan lingkungan mereka.

Sumber: http://www.pheenet.com.tw/solutions2_content?id=4[16]

Gambar 2.1. Penggunaan *wireless mesh network* pada *smart city*

Komunikasi *wifi* yang cocok dan dapat digunakan untuk multi-agent adalah *multicast* karena menggunakan komunikasi *unicast* dapat menyebabkan kesulitan dalam mengontrol data tinggi, penggunaan *bandwidth* yang besar, dan kepadatan lalu lintas data. *Mesh* adalah topologi yang cocok untuk *multi-agent* untuk mengatasi sinyal yang hilang (*loss-signal*) di area bencana. *Wireless mesh network* merupakan jaringan mobile *ad-hoc* yang terkoneksi *peer-to-peer* ke jaringan. *Wireless mesh network* memiliki jangkauan yang luas karena setiap node tidak hanya bertindak sebagai *host* tetapi juga dapat berfungsi sebagai sebuah router untuk meneruskan paket-paket informasi yang akan dikirim menuju node lain yang mungkin tidak dapat

menjangkau tempat yang ingin ditujunya karena keterbatasan jarak. *Wireless mesh network* memiliki kelebihan, dimana jika koneksi yang dilakukannya gagal, maka akan melakukan routing kembali dan dapat mengorganisasikan dirinya sendiri, sehingga koneksi tidak rentan putus. Gambar 2.1 adalah sebuah contoh penggunaan *wireless mesh network* pada konsep *smart city*.

2.1.7 Protokol *Better Approach To Mobile Ad Hoc Network* (B.A.T.M.A.N)

Better Approach To Mobile Ad-Hoc Network atau B.A.T.M.A.N adalah merupakan sebuah *routing* protokol yang bersifat *proactive* yang dikembangkan oleh *Freifunk Community* yang dikembangkan dari protokol *routing* OLSR. B.A.T.M.A.N dikembangkan dengan konsep membentuk sebuah protokol *routing* yang menggunakan informasi *routing* seminimum mungkin dengan hanya mengkalkulasikan *nexthop*.

Konsep *routing* pada B.A.T.M.A.N adalah setiap keputusan *routing* didistribusikan secara merata kepada seluruh node. Sehingga setiap node memiliki pengetahuan mengenai seluruh node yang tersedia beserta *total metric* untuk menuju ke tujuan dan juga *nexthop* terbaik untuk mencapai tujuan. Pada B.A.T.M.A.N, informasi mengenai perubahan topologi jaringan tidak diperlukan. B.A.T.M.A.N melakukan *flooding Originator Message* (OGM) untuk menghindari informasi *routing* yang berbeda sehingga tidak terjadi *routing loop*. B.A.T.M.A.N merupakan salah satu protokol *routing* yang banyak dikembangkan dan diuji dalam banyak skenario [17].

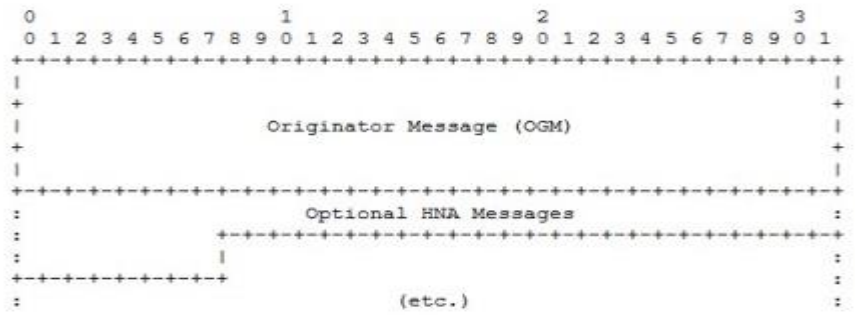
2.1.7.1 Karakteristik B.A.T.M.A.N

Pada dasarnya, B.A.T.M.A.N bekerja pada layer 3, sama seperti OLSR. Sehingga pada mekanisme *routing*, B.A.T.M.A.N menggunakan *IP Address* agar dapat berkomunikasi. Meskipun begitu, B.A.T.M.A.N hanya peduli pada penentuan *best nexthop*. Hal ini membuat mekanisme *routing* B.A.T.M.A.N lebih efisien dan juga lebih cepat. Seperti yang telah dijelaskan sebelumnya, B.A.T.M.A.N menggunakan OGM untuk memberitahu mengenai eksistensi sebuah node kepada

seluruh node di jaringan. Dimana hal inilah yang akan digunakan menjadi salah satu penentuan *best nexthop* terbaik. B.A.T.M.A.N dibuat bukan untuk jaringan yang stabil seperti jaringan dengan menggunakan kabel, melainkan untuk jaringan yang *unreliable* seperti *wireless* yang tidak stabil dan juga selalu mengalami *packet loss* [17].

2.1.7.2 Format Paket B.A.T.M.A.N

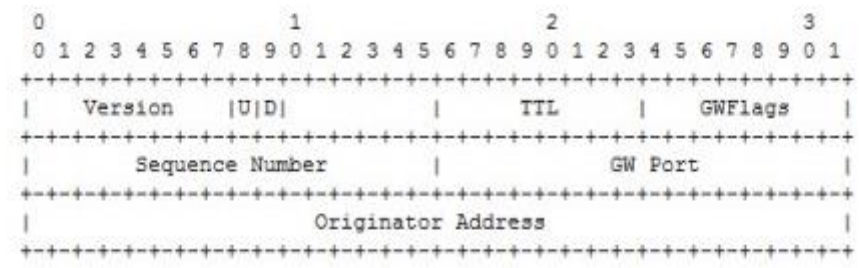
Secara garis besar, format paket B.A.T.M.A.N dapat diilustrasikan seperti Gambar 2.2 dibawah ini:



Sumber: http://www.landasanteori.com/2015/10/pengertian-batman-karakteristik.html[18]

Gambar 2.2. Format Paket B.A.T.M.A.N

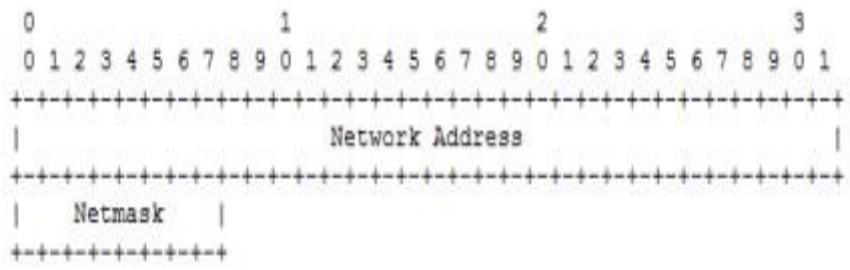
Paket pada B.A.T.M.A.N merupakan paket UDP yang terdiri dari OGM dan *Optional Host Network Announcement* (HNA) Message. OGM memiliki besar paket yang tetap, yaitu 12 oktet. Dimana isi dari OGM digambarkan dalam Gambar 2.3 berikut ini:



Sumber: http://www.landasanteori.com/2015/10/pengertian-batman-karakteristik.html[18]

Gambar 2.3. Format OGM

OGM merupakan paket yang dikirimkan untuk memberitahukan eksistensi node di dalam *mesh.* Untuk paket HNA, *message* dapat digambarkan seperti pada Gambar 2.4 dibawah ini [18].



Sumber: http://www.landasanteori.com/2015/10/pengertian-batman-karakteristik.html[18]

Gambar 2.4. Format Pesan HNA

2.1.7.3 Mekanisme Routing B.A.T.M.A.N

B.A.T.M.A.N menjalankan *routing* daemon untuk terus menjaga *routing table*-nya terus *update*. Routing daemon ini terus menjaga *track* dari OGM-OGM baru dan menjaga *list* dari seluruh originator yang telah mengirimkan OGM. B.A.T.M.A.N juga menjaga satu *entry dedicated routing* untuk setiap OGM dan HNA yang telah dikenal. Setiap *routing entry* menunjukkan *interface outgoing* dari B.A.T.M.A.N dan IP Address dari *nexthop direct link* tetangga menuju originator yang terkait. B.A.T.M.A.N harus memasukkan sebuah rute untuk menuju semua node, bahkan jika node tersebut adalah tetangga dengan status *link-local bidirectional single hop* [19].

2.1.7.4 Pemilihan dan Pembentukan Rute B.A.T.M.A.N

Ketika sebuah node mendapati OGM dari originator yang tidak dikenal ataupun mendapati OGM untuk node yang tidak dikenal oleh jaringan, maka node yang tidak dikenal tersebut akan dimasukkan ke dalam *routing table* dan mekanisme pemilihan tetangga dengan *link-local bidirectional* terbaik akan dilakukan, dimana tetangga dengan *link-local bidirectional* jalur terbaik akan dijadikan sebagai *gateway*

menuju tujuan. Jika terjadi perubahan, misalnya perubahan peringkat tetangga dengan jalur terbaik berubah, maka *routing table* akan di-*update* [19].

2.1.7.5 Penghapusan Rute B.A.T.M.A.N

Penghapusan rute dari *routing table* akan dilakukan secara otomatis jika sebuah node tidak menerima OGM maupun HNA dari sebuah originator dalam rentang waktu yang melebihi WINDOW_SIZE dan interval PURGE_TIMEOUT [19].

2.1.7.6 B.A.T.M.A.N-*advanced*

B.A.T.M.A.N-*advanced* atau B.A.T.M.A.N-adv merupakan implementasi dari protokol *routing* B.A.T.M.A.N dalam bentuk modul kernel yang bekerja pada lapisan ke 2 atau lapisan *data link layer* dari *network layer*. Pada dasarnya protokol routing yang bekerja pada layer 3 saling bertukar informasi *routing* dengan mengirimkan paket UDP dan juga menetapkan keputusan *routing* mereka dengan memanipulasi kernel *routing table*. B.A.T.M.A.N-adv beroperasi sepenuhnya pada layer dua. Itu berarti semua mekanisme *routing* dan juga penetapan jalur *routing* dilakukan di lapisan tersebut menggunakan MAC (*Media Access Control*) Address.

B.A.T.M.A.N-adv menggunakan *tool* batctl untuk melakukan konfigurasi dan *debugging* terhadap kernel modul B.A.T.M.A.N-adv. Batctl juga dapat digunakan untuk melakukan aktifitas *ping*, *traceroute*, dan juga tcpdump pada lapisan dua [20].

2.1.7.7 Interface Virtual bat0

Masalah yang timbul selanjutnya adalah dimana kebanyakan aplikasi memanfaatkan pengalamatan logika di lapisan 3 (*network layer)* yaitu berupa IPv4 untuk melakukan komunikasi dengan aplikasi lain, sementara B.A.T.M.A.N-adv adalah protokol routing yang bekerja pada lapisan 2 (*data-link layer*). Untuk mengatasi hal ini, B.A.T.M.A.N-adv akan membuat sebuah *interface virtual* yang diberi nama bat0 pada setiap node, dimana fungsi dari *interface virtual* ini adalah untuk membantu B.A.T.M.A.N-adv memenuhi kebutuhan pengalamatan logika pada

lapisan 3, sehingga memungkinkan untuk aplikasi yang bekerja dengan memanfaatkan pengalamatan lapisan 3, misalnya VoIP dapat bekerja dengan baik [18].

2.1.8 Sistem Benam (*Embedded System*)

Sistem benam adalah sistem komputer tujuan-khusus, yang seluruhnya dimasukkan ke dalam alat yang dia kontrol. Kata benam (*embedded*) menunjukkan bahwa sistem ini merupakan bagian yang tidak dapat berdiri sendiri. Sebuah sistem benam memiliki kebutuhan tertentu dan melakukan tugas yang telah diset sebelumnya, tidak seperti komputer pribadi serbaguna. Contoh sistem atau aplikasinya antara lain adalah instrumentasi medik, *process control*, *automated vehicles control*, dan perangkat komunikasi.

Salah satu sistem benam yang populer adalah Raspberry Pi. Raspberry Pi sendiri merupakan sebuah *single-board* komputer yang dikembangkan oleh Raspberry Pi Foundation di Inggris dengan tujuan untuk digunakan sebagai media pembelajaran dan pengenalan dasar-dasar dari ilmu komputer. Raspberry Pi berdasarkan pada Sistem Broadcom BCM2835 pada chip (SoC), yang termasuk di dalamnya terdapat ARM1176JZF-S 700 MHz, VideoCore IV GPU dan dilengkapi dengan RAM 256 MB yang kemudian di*upgrade* menjadi 512 MB pada model B dan B+. Gambar 2.5 merupakan salah satu model dari Raspberry Pi.

Pada awal February 2015, Raspberry Pi 2 dirilis di pasaran. *Board* komputer baru tersedia hanya satu dalam sekali konfigurasi (model B) dan menyediakan fitur baru diantaranya Broadcom BMC2836 SoC dengan CPU ARM Cortex-A7 quad-core dan dual-core GPU Video IV. RAM sebesar 1 GB dengan spesifikasi yang mirip dengan generasi B+ sebelumnya [21].

Sumber:https://upload.wikimedia.org/wikipedia/commons/c/c7/Raspberry_Pi_2_Model_B_v1.1_top_new.jpg[22]

Gambar 2.5. Raspberry Pi

2.1.9 Sensor *Thermal*

Sensor *Thermal Array* TPA 81 adalah sensor yang membaca radiasi panas. Sensor ini digunakan untuk mendeteksi infra merah pada panjang gelombang 2µM – 22µM, yang merupakan panjang gelombang dari radiasi panas. Sensor ini memiliki 8 buah sensor panas yang tersusun dalam satu baris. TPA 81 dapat mengukur suhu pada 8 titik yang berdekatan secara bersamaan dan dapat mendeteksi api lilin pada jarak 2 meter dengan tidak terpengaruh oleh cahaya luar. Secara keseluruhan, TPA 81 memiliki range horizontal sebesar 41° dan range vertikal sebesar 6°. Sensor ini dapat mendeteksi api lilin dari jarak sekitar 2 meter.

Data yang dihasilkan dari sensor *thermal array* berupa data biner 8 bit dari masing-masing piksel sensor yang merupakan data suhu yang terukur. Misalkan pada salah satu sensor mendeteksi suhu sebesar 48°C, maka data yang dihasilkan pada sensor tersebut adalah 48 (30H). Sensor *thermal array* memiliki 10 register yang dapat diakses dengan menggunakan protokol I2C. Data suhu dari tiap-tiap piksel sensor terdapat pada register-register yang ditunjukkan pada Tabel 2.1.

Modul sensor *thermal array* dengan komunikasi protokol I2C ini sama dengan modul kompas elektronik. Alamat fix dari sensor ini adalah 0xD0. Selanjutnya membaca data register dengan mengirimkan nilai alamat register yang diinginkan.

Data sensor ada pada alamat register 0×02-0×09 untuk data sensor pixel 1-pixel 8. Untuk sistem komunikasi I2C secara keseluruhan sama dengan modul

kompas elektronik, yang berbeda hanyalah alamat dari modul dan register-register yang dibaca.

Tabel 2.1. Alamat register pada Sensor TPA 81

Register	*Read*	*Write*
0	*Software Version*	*Command Register*
1	*Ambient Temperatur °C*	*Servo Range*
2	*Pixel 1 Temperatur °C*	*N/A*
3	*Pixel 2 Temperatur °C*	*N/A*
4	*Pixel 3 Temperatur °C*	*N/A*
5	*Pixel 4 Temperatur °C*	*N/A*
6	*Pixel 5 Temperatur °C*	*N/A*
7	*Pixel 6 Temperatur °C*	*N/A*
8	*Pixel 7 Temperatur °C*	*N/A*
9	*Pixel 8 Temperatur °C*	*N/A*

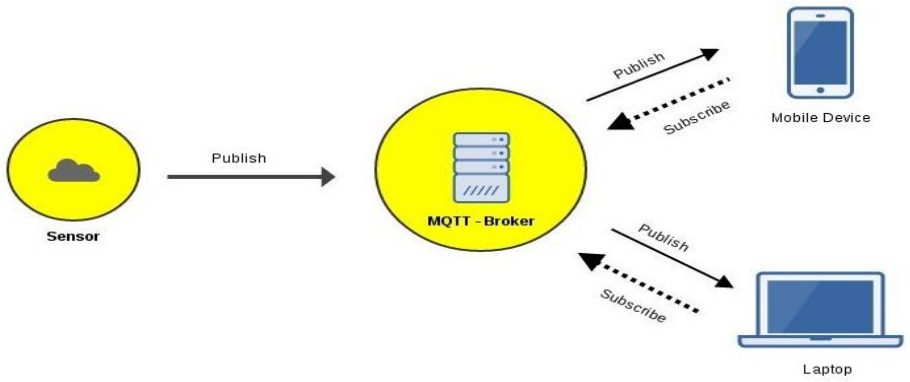
Sumber: https://fahmizaleeits.wordpress.com/tag/cara-kerja-sensor-thermal-tpa-81/#jp-carousel-1069 [23]

Sedangkan cara-cara komunikasinya sama, yaitu dengan menggunakan sistem komunikasi standard I2C. Data yang terbaca pada register-register yang menyimpan data sensor tiap *piksel* adalah data 8 bit yang mempresentasikan nilai suhu yang terukur. Secara umum, cara untuk mendapatkan nilai-nilai suhu dari sensor *thermal array* sama seperti pada kompas elektronik, yang berbeda hanyalah pada alamat register yang akan dibaca dan alamat *device*-nya.

2.1.10 Protokol MQTT

Protokol MQTT (*Message Queuing Telemetry Transport*) adalah protokol yang berjalan pada diatas stack TCP/IP dan mempunyai ukuran paket data dengan *low overhead* yang kecil (minimum 2 bytes) sehingga berefek pada konsumsi catu daya yang juga cukup kecil.

Protokol ini adalah jenis protokol *data-agnostic* yang artinya kita bisa mengirimkan data apapun seperti data binary, text bahkan XML ataupun JSON dan protokol ini memakai model *publish/subscribe* daripada model *client-server*.



Sumber: https://jsiot.pw/mengenal-MQTT-998b6271f585[24]

Gambar 2.6. Sistem IoT menggunakan MQTT

Stack TCP/IP sekarang sudah banyak di dukung oleh mikrokontroler seperti seri STM32Fx7 maupun *device board* yang umum dipasaran seperti Arduino Yun, Arduino + Ethernet Shield, ESP8266 WiFi SoC, Raspberry Pi dll.

Sistem umum MQTT seperti pada gambar diatas membutuhkan dua komponen perangkat lunak utama yaitu:

1. MQTT *Client* yang nantinya akan diinstal di *device*. Untuk Arduino bisa digunakan *pubsubclient*, pustaka seperti MQTT.js bisa dipakai pada platform Node.js di Raspberry Pi ataupun laptop.
2. MQTT Broker yang berfungsi untuk menangani *publish* dan *subscribe* data. Untuk platform Node.js bisa digunakan broker *mosca* sedangkan untuk platform yg lain banyak broker tersedia seperti mosquitto, HiveMQ dll.

Keuntungan dari sistem *publish/subscribe* adalah antara sumber pengirim data (*publisher*) dan penerima data (klien) tidak saling mengetahui karena ada broker diantara mereka atau istilah kerennya yaitu *space decoupling* dan yang lebih penting lagi yaitu adanya *time decoupling* dimana *publisher* dan klien tidak perlu terkoneksi secara bersamaan, misalnya klien bisa saja *disconnect* setelah melakukan *subscribe* ke broker dan beberapa saat kemudian klien *connect* kembali ke broker dan klien tetap akan menerima data yang ter-*pending* sebelumnya proses ini dikenal dengan mode *offline*. Gambar 2.6 menunjukkan penggunaan MQTT pada sistem *Internet of Things*.

2.1.10.1 Sinyal Kontrol

MQTT mempunyai 14 tipe sinyal kontrol seperti berikut:

- CONNECT—*Client request to connect to Server*
- CONNACK—*Connection Acknowledgement*
- PUBLISH—*A message which represents a new/separate publish*
- PUBACK—*QoS 1 Response to a PUBLISH message*
- PUBREC—*First part of QoS 2 message flow*
- PUBREL—*Second part of QoS 2 message flow*
- PUBCOMP—*Last part of the QoS 2 message flow*
- SUBSCRIBE—*A message used by clients to subscribe to specific topics*
- SUBACK—*Acknowledgement of a SUBSCRIBE message*
- UNSUBSCRIBE—*A message used by clients to unsubscribe from specific topics*
- UNSUBACK—*Acknowledgement of an UNSUBSCRIBE message*
- PINGREQ—*Heartbeat message*
- PINGRESP—*Heartbeat message acknowledgement*
- DISCONNECT—*Graceful disconnect message sent by clients before disconnecting*

Dari sinyal sinyal tersebut hanya 4 sinyal utama yang dipakai langsung oleh klien seperti PUBLISH, SUBSCRIBE, UNSUBSCRIBE, CONNECT dan sinyal lainnya merupakan bagian dari mekanisme kerja *publish/subscribe*.

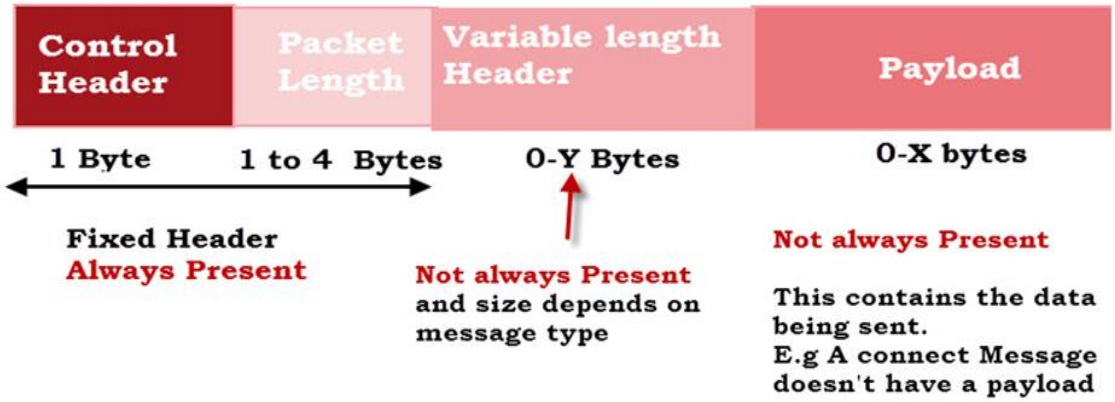
2.1.10.2 *Topic*

Dalam MQTT juga dikenal istilah *topic* yaitu berupa UTF-8 string yang perannya hampir sama seperti topik pada chat hanya saja lebih sederhana dan berfungsi sebagai filter untuk broker dalam mengirimkan pesan ke tiap klien yang terhubung atau dengan kata lain *topic* adalah kanal bagi klien untuk *subscribe* [24]

2.1.10.3 Format Pesan MQTT

MQTT adalah protokol berbasis biner dimana elemen kontrolnya adalah byte biner dan bukan string teks. MQTT menggunakan format sebuah perintah dan perintah *acknowledgement*. Hal Itu berarti setiap perintah memiliki *acknowledgement* yang terkait. Nama *topic, Client ID, User name* dan *Password* dikodifikasi sebagai UTF-8 string.

Payload yang diluar informasi protokol MQTT seperti ID klien, dll adalah data biner dan konten serta formatnya spesifik aplikasi. Paket MQTT atau format pesan terdiri dari header yang tetap 2 byte (selalu ada) + *variabel-header* (tidak selalu ada) + *payload* (tidak selalu ada). Gambar 2.7 menunjukkan struktur paket standar MQTT.



Sumber: http://www.steves-internet-guide.com/mqtt-protocol-messages-overview/[25]

Gambar 2.7. Struktur paket standar MQTT

a. *Fixed Header*

Bagian *header* yang tetap/*fixed* adalah bagian *control* dan *packet length* yang bervariasi. Ukuran minimum bagian *packet length* adalah 1 byte untuk pesan-pesan yang panjangnya kurang dari 127 byte (tidak termasuk bagian kontrol dan panjang). Ukuran paket maksimum adalah 256 MB. Paket-paket kecil kurang dari 127 byte memiliki bagian panjang 1 byte. Sedangkan paket yang lebih besar dari 127 dan kurang dari 16383 akan menggunakan 2 byte dan seterusnya. Sebagai catatan, bagian ini menggunakan 7 bit dengan bit ke-8 sebagai *continuation bit*.

Jadi ukuran paket minimum hanya 2 byte dengan perincian 1 byte bagian

control dan 1 byte bagian *packet length*. Contoh: *disconnect message* hanya 2 byte.

b. Bagian kontrol

8 bit bagian kontrol adalah byte pertama dari 2 byte *header* yang tetap. Ini dibagi menjadi dua bagian 4 bit, dan berisi semua perintah protokol dan *response*. 4 bit pertama (*most significant bits*) adalah bagian perintah atau jenis atau tipe pesan dan 4 bit lainnya digunakan sebagai *control flag*. Bagian *control flag* menunjukkan detail dari tipe pesan tersebut seperti DUP, QoS, dan Retain.

c. *Variable Length Header*

Bagian ini tidak selalu ada dalam pesan MQTT. Bagian ini menunjukan detail tambahan informasi *control*. Contoh *Variable Length Header* dalam *connection packet* sebagai berikut:

- MQTT packet = control + length + protocol name + Protocol Level + Connect Flags + keep alive + Payload
- Payload = client ID = client ID length + client ID

d. *Payload*

Tipe pesan yang memiliki *payload* diantaranya: CONNECT, SUBSCRIBE, SUBACK [25].

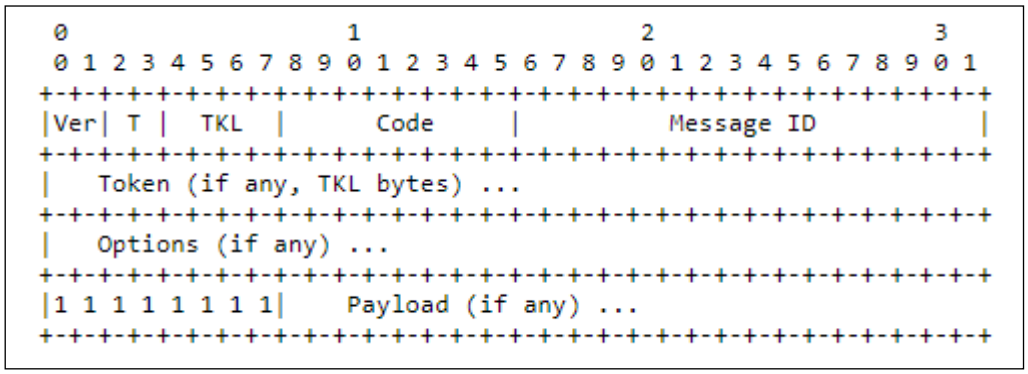
2.1.11 Protokol CoAP

Constrained Application Protocol adalah suatu web transfer protokol khusus untuk pengunaan dengan node terbatas dan jaringan yang dibatasi. CoAP dikembangkan oleh *International Engineering Task Force* (IETF) merupakan suatu protokol layer aplikasi dan termasuk ke dalam standar RFC 7252. CoAP yang juga merupakan suatu *lightweight protocol* dimaksudkan untuk digunakan sebagai pengganti HTTP untuk menjadi protokol pada layer aplikasi di dalam IoT (*Internet of Things*). CoAP menyediakan model interaksi *request/response* antara aplikasi dan *endpoint*, dikarenakan CoAP memiliki beberapa fitur yang menyerupai HTTP. Tidak seperti HTTP yang beroperasi pada TCP, CoAP beroperasi pada UDP untuk

menghindari *congestion control* yang kompleks. CoAP memiliki *Representational State Transfer* (REST) arsitektur, sehingga menyediakan URI dan metode seperti GET, POST, PUT, dan DELETE. Untuk mengimbangi dari kekurangan UDP, CoAP memiliki sebuah mekanisme retransmisi. Untuk mengatasi kelemahan dalam keterbatasan sumber daya, CoAP perlu mengoptimalkan panjang datagram untuk menyediakan komunikasi yang handal. Terdapat empat tipe pesan yang digunakan pada CoAP untuk melakukan pertukaran data antara *client* dan server, berikut tipe – tipe pesan tersebut :

1. *Confirmable* (CON), merupakan pesan yang berisi request dan memerlukan *Acknowledgment*.
2. *Non-Confirmable* (NON), merupakan pesan yang digunakan berulang secara teratur tanpa memerlukan *Acknowledgment.*
3. *Acknowledgment* (ACK), merupakan pesan yang berisi *response*.
4. *Reset* (RST), merupakan pesan yang digunakan ketika pesan CON tidak diterima dengan benar atau terdapat konteks yang hilang [26].

2.1.11.1 Format Pesan CoAP



Sumber: https://tools.ietf.org/html/rfc7252[27]

Gambar 2.8. Struktur paket standar CoAP

Gambar 2.8 menunjukkan struktur paket standar CoAP dimana pesan CoAP dikodekan dalam format biner sederhana. Format pesan dimulai dengan header 4-

byte berukuran tetap. Lalu diikuti oleh nilai Token *variable-length,* yang panjangnya antara 0 dan 8 byte.

Setelah nilai Token merupakan nol atau lebih CoAP *option* dalam format *Type-Length-Value* (TLV), lalu diikuti oleh *payload* yang mengambil sisa datagram yang bersifat opsional.

Bagian di header didefinisikan sebagai berikut:

a. Versi (Ver): *Unsigned integer* 2-bit. Bagian ini menunjukkan versi CoAP. Nilainya diset ke 1 (01 biner). Nilai lain disediakan untuk versi mendatang. Pesan dengan nomor versi tidak dikenal maka diabaikan.

b. *Type* (T): *Unsigned integer* 2-bit. Bagian ini menunjukkan tipe dari pesan berupa *Confirmable*(0), *Non-confirmable*(1), *Acknowledgement*(2), atau *Reset* (3).

c. *Token Length* (TKL): *Unsigned integer* 4-bit. Bagian ini menunjukkan panjang dari bagian *variable-length* Token (0-8 bytes). Panjang 9-15 dicadangkan, tidak harus dikirim, dan harus diproses sebagai kesalahan format pesan.

d. *Code*: *Unsigned integer* 8-bit. Dibagi menjadi kelas 3-bit (*most significant bits*) dan 5-bit (*least significant bits*), didokumentasikan sebagai "c.dd" di mana "c" adalah digit dari 0 hingga 7 untuk sub bagian 3-bit dan "dd" adalah dua digit dari 00 hingga 31 untuk sub bagian 5-bit. Kelas tersebut dapat menunjukkan sebuah *request* (0), *success response* (2), *client error response* (4), *server error response* (5) (semua nilai kelas lain telah disediakan). Pada kasus tertentu, code 0.0 menunjukkan pesan yang kosong. Pada tipe pesan *request*, bagian *code* menunjukkan *Request Method*; pada tipe pesan *response* pun berarti menunjukkan sebuah kode *response*.

e. *Message ID*: *Unsigned integer* 16-bit di dalam perintah pada jaringan. Digunakan untuk mendeteksi duplikasi pesan dan untuk mencocokkan pesan tipe *Acknowledgement/Reset* ke pesan bertipe *Confirmable/Non-confirmable*.

f. Token bernilai 0 hingga 8 byte, seperti yang diberikan oleh bagian *Token Length*. Nilai Token digunakan untuk menghubungkan antara *request* dan *response*.

g. *Option*. *Option* dapat diikuti oleh akhir pesan, oleh *Option* yang lain, atau oleh *Payload Marker* dan *payload*.

h. *Optional payload*. Jika ada nilainya dan panjang tidak nol, maka diawali oleh sebuah satu-byte *Payload Marker* (0xFF) yang tetap dan menunjukkan akhir *optional* dan awal *payload.* Data *payload* memanjang dari setelah penanda ke ujung datagram UDP, yaitu, *Payload Length* dihitung dari ukuran datagram. Tidak adanya *Payload Marker* menunjukkan sebuah *payload zero-length*. Kehadiran sebuah penanda diikuti oleh *payload zero-length* harus diproses sebagai kesalahan format pesan [27].

2.1.12 *Ubiquitous Network Robot Platform* (UNR-PF)

Layanan robot yang terkoneksi menempatkan hal-hal yang penting seperti kerjasama diantara banyak robot yang berbeda, lingkungan-lingkungan sensor, dan *database-database* dalam jaringan-jaringan. Untuk mengembangkan layanan jaringan robot, developer layanan harus mempersiapkan sebuah kemampuan jaringan yang mampu menyesuaikan diri dan fleksibel sehingga tidak tergantung pada desain sebelum konstruksi. Seiring pengembangan, jaringan-jaringan sistem harus dapat dimodifikasi untuk lingkungan masing-masing dan ditingkatkan sesuai dengan permintaan dari pengguna jasa melalui percobaan lapangan di lingkungan nyata.

Untuk menyediakan layananan di tempat yang berbeda, sistem seharusnya disesuaikan berdasarkan lingkungan lokal. Juga, modul-modul fungsional robot diasumsikan dan dimodifikasi untuk meningkatkan kinerjanya. Jika peningkatan kinerja memaksa modifikasi dalam seluruh sistem, maka hal itu akan menghalangi seluruh proses pengembangan. Karena itu lapisan aplikasi layanan yang mendefenisikan proses dari konten layanan harus independen dari fungsi robot. Dengan demikian tingkat abstraksi dari fungsi-fungsi robot harus didesain hati-hati agar dapat dikendalikan dari aplikasi layanan.

Layanan robot terkoneksi tidak hanya disiapkan berdasarkan luasnya namun juga non-kontinu area-area seperti rumah user, toko, rumah sakit, restoran, dsb untuk membantu kehidupan user dimanapun berada (*ubiquitous*). Dalam hal mengkoordinasikan layanan multi-lokasi, sebuah mekanisme harus dibangun untuk *sharing* informasi diantara area-area yang berbeda.

Sesuai dengan kemajuan perkembangan robot maka semakin banyak tipe-tipe robot yang diciptakan untuk mengerjakan macam-macam *task*. Kemampuan-kemampuan robot akan juga meningkat dengan cepat mencakup semua bidang dan begitupula dengan jenis spesialisasi robot. Untuk hal seperti kelompok robot yang bekerjasama, tipe-tipe kemampuan robot harus diklasifikasi dan diatur dengan tepat.

Berdasarkan hal yang disebutkan di atas, desain sebuah **sistem robot jaringan *ubiquitous*** seharusnya menempatkan masalah-masalah sebagai berikut: abstraksi yang sesuai dan klasifikasi fungsi-fungsi robot, *sharing* informasi dan manajemen diantara lokasi-lokasi yang tersebar, dan *platform-platform* manajemen umum untuk komponen-komponen robot dengan kapabilitas dan ketersediaanya.

a. Multi-robot Management

Platform ini dibutuhkan untuk mengklasifikasi kemampuan-kemampuan robot dan mengalokasi robot-robot yang sesuai berdasarkan konten layanan dan lingkungan eksekusi. Dengan mengimplementasikan sebuah fungsi manajemen dalam *platform*, setiap fungsional robot dapat melakukan *discover* sehingga koordinasi robot dapat ditingkatkan dalam situasi dimana beraneka ragam robot tersedia. Kompatibilitas juga harus ditingkatkan diantara modul-modul yang memiliki fungsi yang mirip pada robot yang berbeda.

b. Multi-Area Management

Untuk menyediakan layanan-layanan pada sebuah area yang luas, hal yang penting untuk meingkatkan koordinasi diantara area-area yang banyak. Untuk menghubungkan beberapa poin fisik, *platform* perlu mekanisme untuk berbagi informasi spasial dari masing-masing daerah seperti informasi peta. Hal ini penting untuk tidak hanya berbagi informasi spasial statis seperti informasi peta tetapi juga informasi lokasi dinamis (yaitu, lokasi robot, pengguna, halangan dan sasaran objek)

yang merubah lokasi relatif dan mutlak sesuai dengan keadaan. Untuk koordinasi multi-daerah yang efisien, informasi spasial dan lokasi ini harus dikelola pada struktur manajemen berlapis-lapis yang berisi lapisan area lokal, yang mengelola informasi lokasi masing-masing daerah, dan lapisan global, yang mengelola hubungan antara daerah setempat.

c. User Management

Untuk memberikan layanan yang sesuai bagi pengguna layanan, informasi terkait dengan pengguna harus dikelola. Terutama di layanan robot yang melakukan tugas fisik bagi pengguna, sistem robot harus memilih peralatan yang tepat sesuai dengan atribut pengguna. *Platform* ini akan menyediakan sebuah mekanisme untuk mengelola atribut pengguna secara umum sehingga semua layanan dapat merujuk pada informasi pengguna secara efisien.

d. Service Management

Untuk melaksanakan sejumlah layanan di beberapa daerah, hal yang penting untuk memantau keadaan daerah masing-masing dan menentukan awal eksekusi layanan sehingga layanan dapat dieksekusi dalam situasi yang tepat. *Platform* ini memerlukan mekanisme untuk mengelola status lingkungan eksekusi layanan. Selanjutnya, *platform* tidak hanya membutuhkan mekanisme untuk mengeksekusi setiap layanan secara mandiri tetapi juga mekanisme untuk berbagi informasi antara masing-masing layanan.

Untuk mengatasi masalah ini, UNR-PF yang dibuat terdiri dari tiga lapisan: lapisan modul fungsional, lapisan aplikasi layanan, dan lapisan *platform* umum. Lapisan pertama terdiri dari modul fungsional yang dapat digunakan terus milik robot individu, dan lapisan kedua mengelola logika untuk isi layanan. Lapisan *platform* umum harus independen dari keduanya dan melayani fungsi umum untuk dua lapisan lainnya.

Platform umum dikembangkan dan disediakan secara stabil, maka fungsi dan layanan robot akan terpisah sehingga seluruh sistem robot akan dapat dikembangkan dengan biaya yang lebih rendah dan menjadi lebih diandalkan dalam hal kegunaan, skalabilitas, dan ketersediaan. Aplikasi-aplikasi layanan, yang menentukan aliran isi

layanan, dan robot di lingkungan masing-masing menghubungkan dan mendaftar ke UNR-PF pada awalnya. Kemudian, interaksi antara aplikasi dan layanan robot didirikan melalui UNR-PF. Jadi UNR-PF bertindak sebagai *middleware* diantara layanan-layanan dan peralatan-peralatan robot [28].

Jadi *Ubiquitous Network Robot Platform* (UNR-PF) adalah sebuah *middleware* yang dirancang khusus untuk aplikasi *ubirobots*.

2.2 PENELITIAN TERKAIT

2.2.1 *Disaster Multi-robot* (Multi-robot pada daerah bencana)

Rancangan robot untuk daerah bencana mempertimbangkan banyak aspek termasuk bidang, mekanisme dan mobilitas, rintangan, misi, dan banyak lagi. Beberapa penelitian telah dilakukan untuk mengatasi hal-hal ini dengan membuat robot dengan roda yang cocok dengan medan yang tidak rata, robot yang dapat belajar untuk menghindari rintangan dengan algoritma tertentu [29], robot yang dapat melakukan manuver dengan adaptasi dengan mengubah diri mereka sendiri (lihat Gambar 2.9) [30], robot yang dikendalikan dari jarak jauh yang dilengkapi dengan kemampuan navigasi [31], dan kerjasama multi-robot untuk menyelamatkan korban bencana [32].

sumber: Son Kuswadi, dkk ,2017[30]

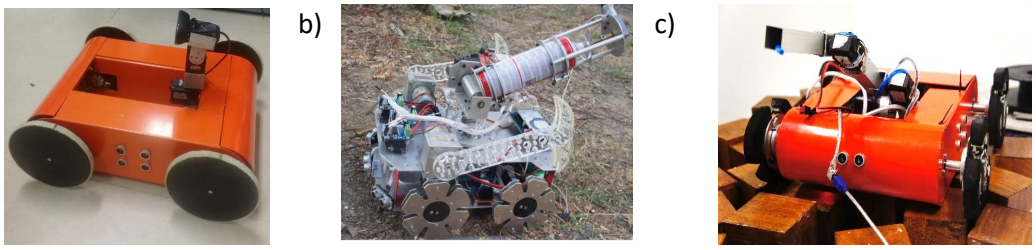
Gambar 2.9. Robot PENS-FlyCrawl yang dapat melakukan transformasi

Penelitian ini menggunakan yang terdiri dari 3 robot dengan spesialisasi

masing-masing, yaitu 1 robot sebagai pemimpin dan 2 robot sebagai pengikut/*follower* seperti pada Gambar 2.10. Multi-robot akan melakukan pencarian dan penyelamatan sederhana ketika menemukan korban di bidang eksperimental.

Semua robot dapat mendeteksi api dan manusia berdasarkan termal. Ketika satu atau lebih robot menemukan api di suatu lokasi, mereka mengirim sinyal ke sistem atau operator, kemudian memerintahkan robot yang dilengkapi dengan pemadam api untuk mendekati lokasi untuk memadamkan api. Selain itu, jika mereka menemukan atau mendeteksi manusia, robot yang dilengkapi dengan tabung air akan mendekati korban dan melakukan penyelamatan sederhana dengan memberikan air yang dikendalikan oleh operator.

a) b) c)



Gambar 2.10. (a) *Leader*; (b) *Follower* 1(*Fire Extinguisher*); (c) *Follower* 2(*Water-Provider*)

2.2.2 Protokol Komunikasi IoT

Baru-baru ini, penelitian tentang teknologi IoT berkembang pesat. Teknologi ini bertujuan untuk membuat semua perangkat di sekitar kita terhubung ke Internet sehingga tujuan pemantauan dan otomatisasi dapat dilakukan. Terkait dengan teknologi ini, kami berpikir tentang bagaimana menerapkannya pada multi-robot. Komunikasi antar sensor atau perangkat dalam IoT akan diterapkan ke komunikasi multi-robot. Di antara protokol komunikasi IoT yang dapat digunakan untuk multi-robot adalah MQTT dan CoAP.

Penelitian tentang protokol komunikasi ringan seperti MQTT untuk mengendalikan robot dan pelacakan GPS juga telah dilakukan dengan mengintegrasikannya pada platform *cloud*. Tetapi penelitian ini berlaku untuk satu robot saja [33]. Ada beberapa makalah yang telah membahas perbedaan antara CoAP

dan MQTT dan membandingkan kinerja antara mereka. Seperti yang dibahas sebelumnya, CoAP dan MQTT menggunakan lapisan *transport* yang berbeda (UDP vs TCP). Sebuah penelitian menunjukkan bahwa MQTT lebih baik daripada CoAP dalam penundaan komunikasi jika tingkat kehilangan jaringan adalah 20% dan CoAP lebih baik daripada MQTT jika tingkat kehilangannya 25% [34].

Studi lain menunjukkan bahwa MQTT lebih cepat dan lebih hemat daya daripada HTTP [35], dan penelitian lain menunjukkan bahwa CoAP lebih hemat daya dibandingkan dengan MQTT [36]. MQTT dan CoAP juga mengkonsumsi bandwith rendah [37]. Penelitian lain membandingkan protokol komunikasi MQTT-SN dan CoAP dalam jaringan internet menggunakan kabel ethernet LAN untuk robot. MQTT-SN adalah protokol yang menggunakan lapisan *transport* serupa dengan CoAP [38].

2.2.3 *Wireless Mesh* untuk *Disaster Multi-robot*

Koneksi komunikasi antara robot di daerah bencana perlu dipertahankan dan hilangnya koneksi juga harus diprediksi untuk kebutuhan kontrol multi-robot dan untuk membuat transmisi data dapat terus berlanjut. Oleh karena itu, manajemen topologi jaringan *mesh* dalam jaringan multi-robot diperlukan. Ada studi tentang implementasi jaringan *mesh* nirkabel antara node (laptop) menggunakan aplikasi [39].

The Better Approach To Mobile Adhoc Networking (BATMAN) adalah protokol *routing* untuk jaringan *adhoc* seluler multi-hop yang sedang dikembangkan oleh komunitas "Freifunk" Jerman dan dimaksudkan untuk menggantikan protokol *routing* jalur-OLSR. Hal-hal penting dari BATMAN adalah desentralisasi pengetahuan tentang rute terbaik melalui jaringan - tidak ada simpul tunggal yang memiliki semua data. Kharisma Babu et al. [40] telah membandingkan antara OLSR dan B.A.T.M.A.N. dan hasilnya adalah B.A.T.M.A.N. lebih efisien daripada OLSR. Davinder dkk. [41] telah membandingkan kinerja protokol B.A.T.M.A.N., DSR, dan OLSR untuk 100 node berdasarkan rasio pengiriman paket, *delay* ujung ke ujung, beban perutean, dan pengukuran *throughput*; hasilnya adalah B.A.T.M.A.N. lebih baik dari yang lain. Daniel Satier dkk. [42] telah membuktikan kinerja yang baik dari

B.A.T.M.A.N. dalam percobaan nyata.

2.3 KONTRIBUSI

Kontribusi penelitian ini adalah mengimplementasikan protokol komunikasi *Internet of Things* (IoT) pada area bencana dengan topologi *mesh*. Selain itu penelitian ini juga membandingkan kinerja antara protokol MQTT dan CoAP pada *disaster multi-robot*.

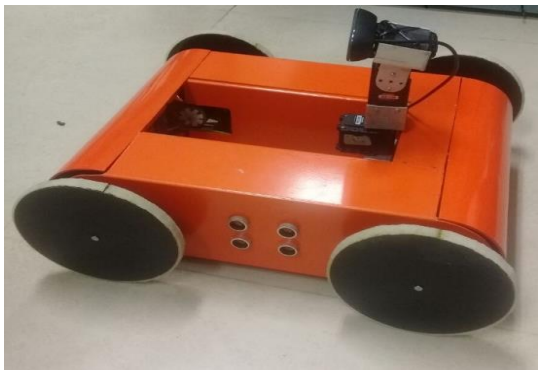
Batasan penelitian yang dilakukan dengan menggunakan 3 buah robot yang dapat mendeteksi (*search)* keberadaan korban bencana pasca gempa dan melakukan penyelamatan (*rescue*) dengan memberikan air yang dibawa. Selain itu juga dapat mendeteksi titik api (*hotspot/fire point*) sekitar area bencana.

Halaman ini sengaja dikosongkan

BAB 3
DESAIN SISTEM

Desain sistem dari *multimobile robot* ini terdiri atas desain mekanik robot dan desain elektronik. *Wireless Mesh Network* (WMN) akan diimplementasikan untuk membangun jaringan antar robot yang digunakan dalam proses pertukaran informasi baik antar robot dengan robot maupun antar robot dengan *Ground Station.* Secara keseluruhan, pada rencana besarnya, robot-robot ini akan diintegrasikan menjadi satu tim agar dapat bekerja sama dengan efektif pada saat beraksi di area pasca bencana.

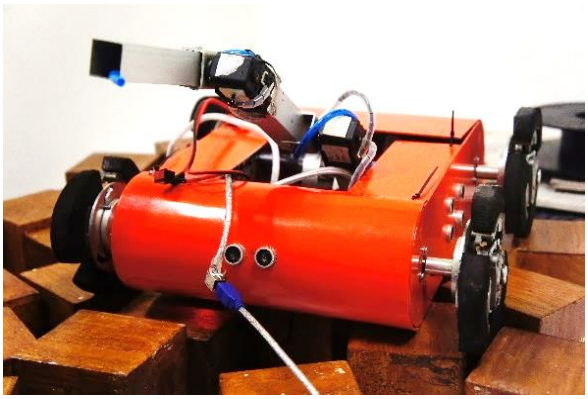
3.1.1 Desain Mekanik



Gambar 3.1. Robot *Leader*

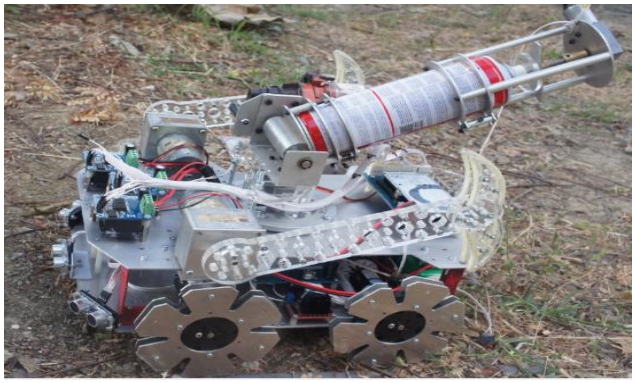
Desain *rescue mobile robot* untuk 3 robot menggunakan roda tipe *leg wheel transformable* yang dapat berubah bentuk sesuai dengan medan. Sebuah robot sebagai *leader* dan 2 robot sebagai *follower*. Robot *leader* seperti yang ditunjukkan pada Gambar 3.1 dilengkapi dengan sebuah kamera dan sensor *thermal* untuk deteksi korban.

Sebuah robot *follower* pertama dilengkapi lengan. Lengan pada *mobile* robot ini bekerja pada saat melakukan evakuasi korban saja, yaitu dengan cara memberikan pertolongan pertama pada korban berupa minuman. Robot ini juga dilengkapi tabung air seperti yang ditunjukan pada Gambar 3.2.



Gambar 3.2. Robot *Follower* Pemberi Minuman

Sedangkan desain *fire extinguisher mobile robot* sebagai *follower* kedua menggunakan roda untuk segala medan dan dibantu mekanisme lengan berbentuk seperti pacul untuk menghadapi jalan yang bergelombang dan tidak rata. Robot ini juga dilengkapi sebuah lengan yang dapat menggenggam botol pemadam api seperti pada Gambar 3.3.



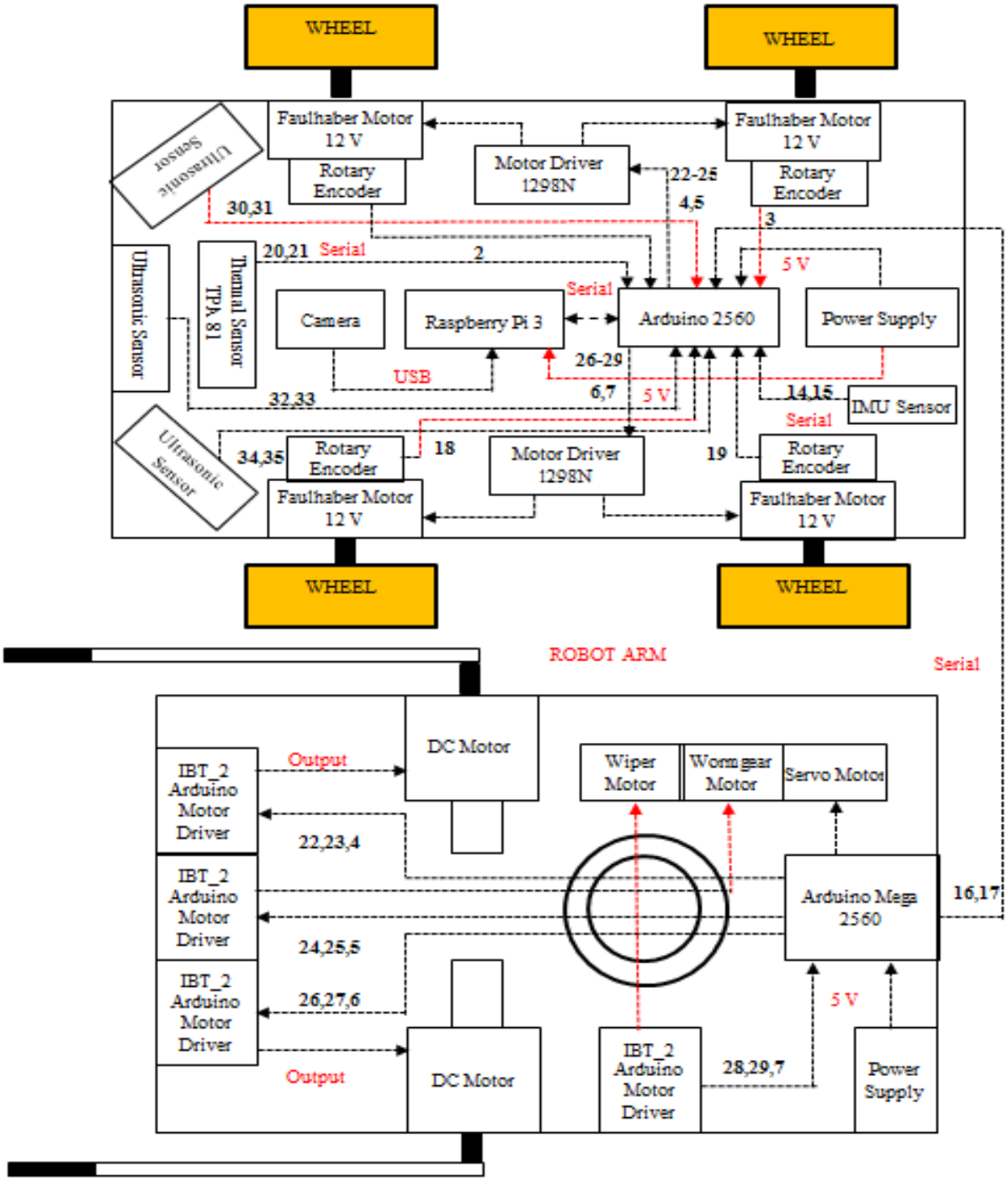
Gambar 3.3. Robot *Fire Extinguisher*

3.1.2 Desain Elektronik

Sistem elektonik pada proyek ini mengintegrasikan beberapa sensor untuk keperluan sensing,navigasi dan komunikasi seperti:

- Sensor IMU
- Sensor *Thermal* TPA 81

- Kamera *Logitech* C270
- *Rotary Encoder* (Sensor Kecepatan)
- Sensor Ultrasonik



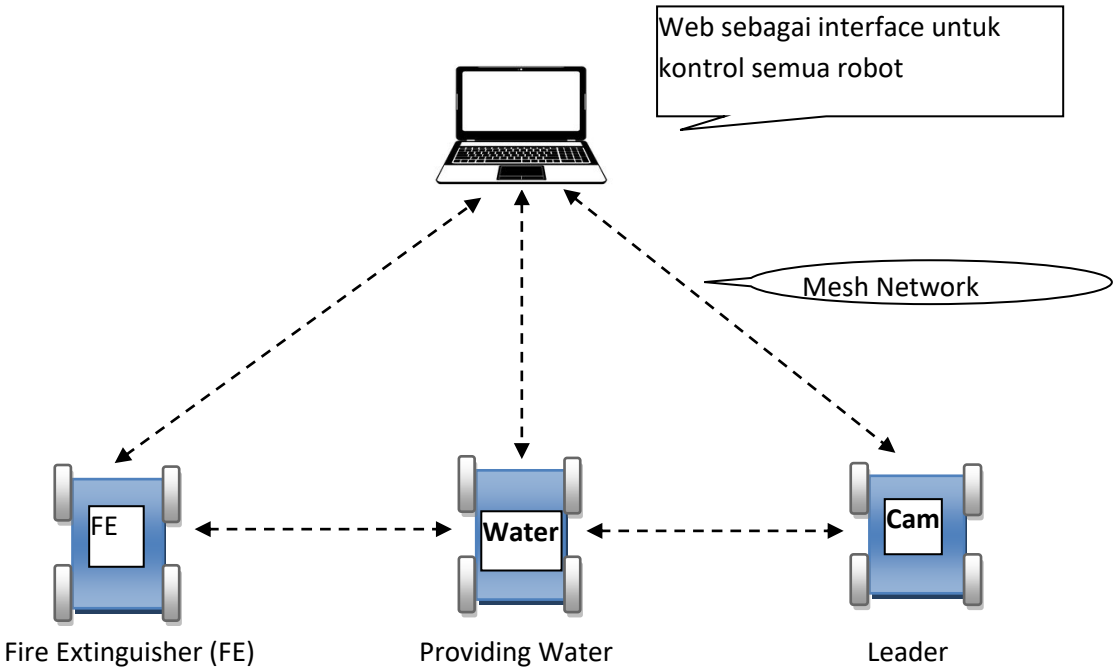
Gambar 3.4. Diagram Sistem Elektronik

Gambar 3.4 adalah gambar susunan komponen-komponen yang terpasang pada robot. Semua sensor terhubung ke Arduino Mega 2560 sebagai mikrokontroler agar mudah diproses langsung untuk proses pencarian korban, deteksi korban, deteksi titik api dan *obstacle avoidance*. Hasilnya akan dikirim secara serial menggunakan jalur UART ke Raspberry Pi lalu Raspberry Pi yang telah terinstal MQTT akan mengirim data tersebut. Raspberry Pi di robot juga berperan dalam komunikasi antara robot dan juga operator.

Multi-robot yang digunakan merupakan *semiautonomous* multi-robot. Multi-robot dapat bergerak otomatis dalam melakukan proses *behaviour base control* namun ketika didapatkan situasi di luar perkiraan maka operator dapat berperan langsung atau mengontrol robot secara manual dengan menggunakan *user interface* yang ada.

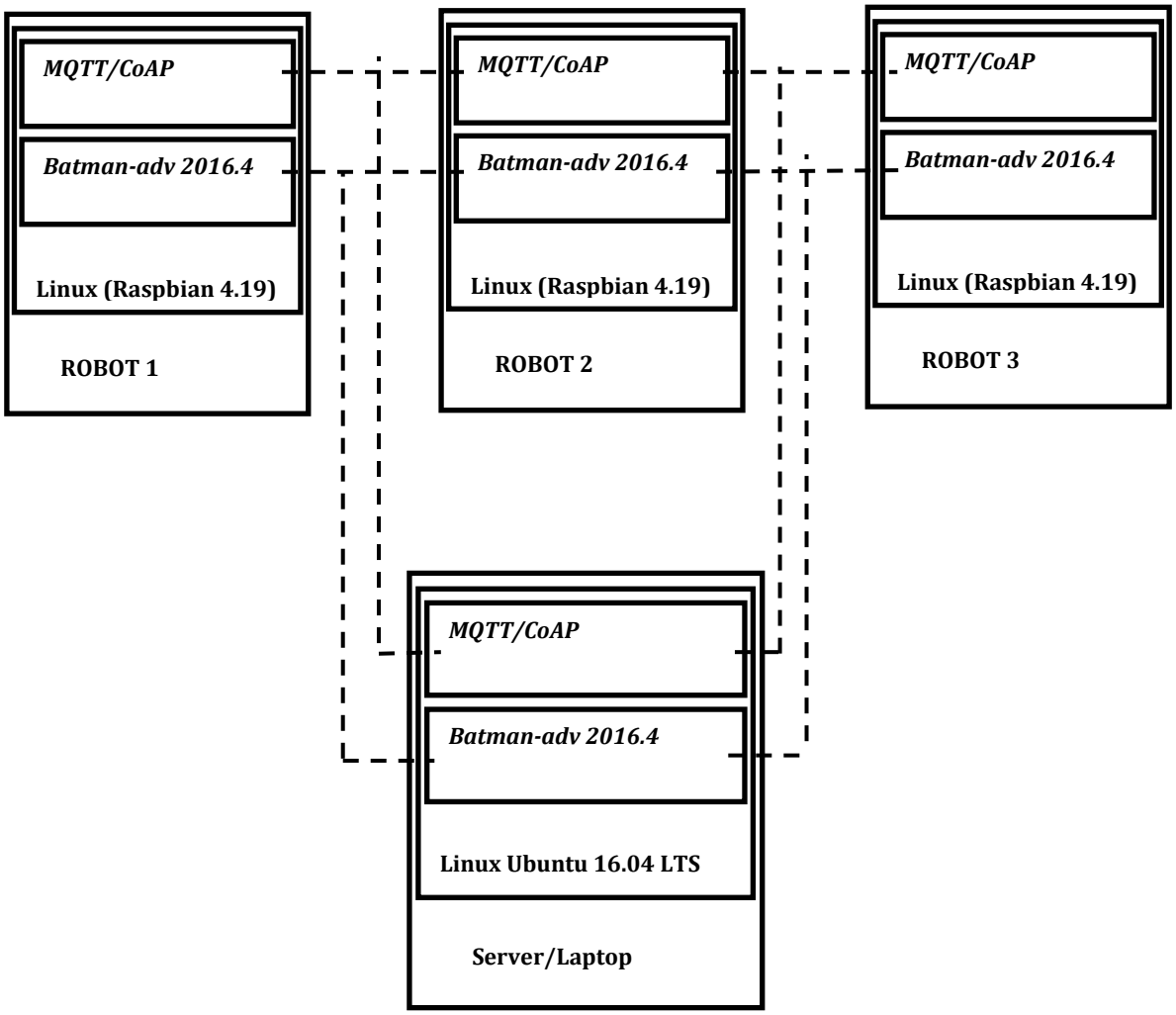
Sebagai catatan khusus, operator memegang kendali penuh atas robot *fire extinguisher* dan robot pemberi minuman ketika sudah mendekati target. Hal ini diatur agar kesalahan bisa diminimalisasi dan tindakan penyelamatan dapat dilakukan secara efektif.

3.1.3 Desain Komunikasi Antar Robot



Gambar 3.5. Struktur Komunikasi Antar Robot

Pada Gambar 3.5 ditunjukkan skema komunikasi dan jaringan yang diatur antar robot. *Web server* diinstal di laptop yang berperan sebagai server. *Web Server* ini akan menjadi *user interface* yang menerima data dan mengirim data pada jaringan robot. Dalam hal pertukaran data menggunakan protokol MQTT atau CoAP, sedangkan pada jaringannya menggunakan topologi "*mesh*" dengan maksud untuk mengatasi terputusnya sinyal di salah satu node atau robot saat berada di daerah bencana. Topologi *mesh* dapat mencari jalur lain jika jalur utama terputus. Jangkauan *wifi* yang lebih dari 100 meter dengan didukung topologi ini juga akan memperlebar jangkauan komunikasi.

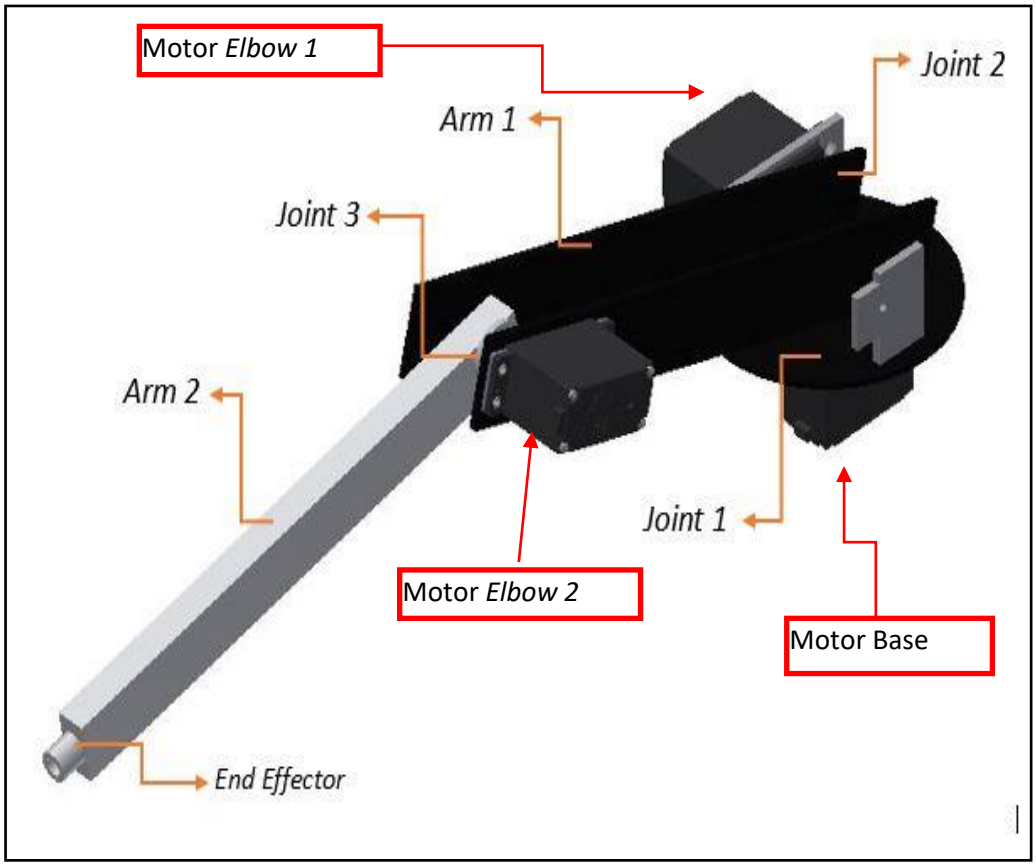


Gambar 3.6. Desain Komunikasi Antar Robot

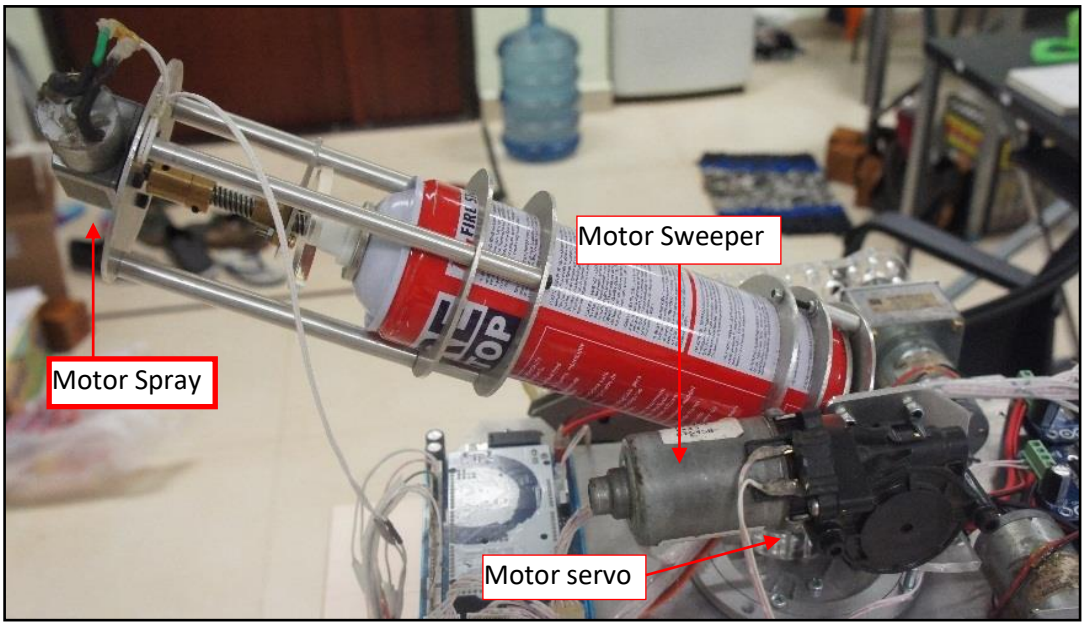
Pada Gambar 3.6 menunjukkan software yang dibutuhkan untuk komunikasi antara *disaster robot*. Sistem operasi yang diinstal adalah produk *open source* yang memang dibuat untuk Raspberry Pi. *Library* atau software untuk keperluan protokol komunikasi diinstal di masing-masing robot dan server/PC.

3.1.4 Skema *Command*/Perintah Untuk Komunikasi

Gambar 3.7 menunjukkan bagian-bagian penyusun *arm* robot untuk memberikan minuman bagi korban. Motor-motor yang menyusun *arm* tersebut akan dikendalikan melalui *user interface* yang telah digunakan. Sedangkan Gambar 3.8 menunjukkan *arm* khusus untuk robot *fire extinguisher* yang digunakan untuk membantu memadamkan titik api. Sistem akan menggunakan driver motor yang terhubung ke Arduino Mega .



Gambar 3.7. *Arm* Robot Pemberi Minuman



Gambar 3.8. *Arm* Robot *Fire Extinguisher*

-PC/Server

A. Pengiriman Perintah

Tabel 3.1 menunjukkan komunikasi antara server dan multi-robot berupa pengiriman perintah dari server/PC. Salah satu contoh sintaks untuk perintah dalam protokol MQTT adalah sebagai berikut:

- Client.publish("topic/moveRobot1","F") = mengirim string "F" yang memberi perintah robot 1 untuk bergerak maju

Publish dalam MQTT menunjukkan perintah pengiriman. Sedangkan *subscriber* menunjukkan kesiapan menerima data. *Topic* antara *publisher* dan *subscriber* harus sama agar lalu lintas data terwujud. Sedangkan kode-kode yang ada tergantung spesifikasi perintah yang diinginkan. Pengendalian multi-robot berpusat pada server sehingga detail perintah yang diperlukan harus dijabarkan.

Tabel 3.1. Pengiriman Perintah dari PC

Sumber	Tujuan	Topic	Kode	Perintah
PC/Server	Robot1	moveRobot1	F	Maju
			B	Mundur
			R	Belok Kanan
			L	Belok Kiri
			S	Berhenti
	Robot2	moveRobot2	G	Maju
			H	Mundur
			I	Belok Kanan
			J	Belok Kiri
			K	Berhenti
			O	*Base Arm* berputar 90^o searah jarum jam
			U	*Base Arm* berputar 90^o berlawanan jarum jam
			T	Motor *Sweeper* terangkat 90^o
			V	Motor *Sweeper* turun 90^o
			W	Memgaktifkan pemadam api
			X	Menonaktifkan pemadam api
	Robot3	moveRobot3	Y	Maju
			Z	Mundur
			A	Belok Kanan
			M	Belok Kiri
			C	Berhenti
			D	*Base Arm* berputar 90^o searah jarum jam
			E	*Base Arm* berputar 90^o berlawanan jarum jam
			P	Motor *servo elbow 1 arm* berputar 90^o searah jarum jam
			Q	Motor *servo elbow 1 arm* berputar 90^o berlawanan arah jarum jam
			N	Motor *servo elbow 2 arm* berputar 90^o searah jarum jam
			2	Motor *servo elbow 2 arm* berputar 90^o berlawanan arah jarum jam
			3	Pompa aktif
			4	Pompa berhenti
	Robot1, Robot 2, Robot 3	formasirobot	1	Membentuk formasi

B. Penerimaan Data

Tabel 3.2. Penerimaan Data ke PC

Sumber	Tujuan	Topic	Aksi
Server	Robot 1	DataRobot1	Siap menerima data dari robot 1
	Robot 2	DataRobot2	Siap menerima data dari robot 2
	Robot 3	DataRobot3	Siap menerima data dari robot 3

Tabel 3.2 menunjukkan kesiapan server untuk menerima data dari masing-masing robot. Salah satu contoh sintaks untuk menerima data dalam protokol MQTT adalah sebagai berikut:

- Client.subscriber("topic/DataRobot1") = mengirim sinyal ke robot 1 yang menandakan server siap menerima data dari robot 1

-Multi-robot

A. Pengiriman Perintah

Tabel 3.3. Pengiriman Perintah dari Robot

Sumber	Tujuan	Topic	Aksi
Robot 1	Server/PC	DataRobot1	Mengirim data ke Server
	Robot 2	pemadam	Memberi perintah robot 2 menuju titik api
	Robot 3	air	Memberi perintah robot 3 menuju korban
Robot 2	Server/PC	DataRobot2	Mengirim data ke Server
	Robot 3	air	Memberi perintah robot 3 menuju korban
Robot 3	Server/PC	DataRobot3	Mengirim data ke Server
	Robot 2	air	Memberi perintah robot 3 menuju korban

Tabel 3.3 menujukkan pengiriman data dari robot terhadap server dan robot lainnya. Robot akan mengirimkan data terus menerus ke server berdasarkan pembacaan sensor padanya dan juga mengirimkan data ke robot-robot yang

lain pada waktu-waktu tertentu, diantaranya saat menemukan korban dan saat ingin membentuk formasi karena perannya sebagai *leader*. Format data yang dikirimkan sebagai berikut:

%Y-%m-%d_%H:%M:%S.%f, A,B,C,Y,P,R,To,T1,T2,T3,T4,T5,T6,T7,T8

-%Y- %m -%d %H:%M:%S.%f = Format *time stamp* yang akan dikirim

- A,B,C= Posisi robot berupa X,Y dan *velocity*/kecepatan
- Y,P,R = Yaw, Pitch, Roll dari robot hasil pembacaan sensor IMU
- To,T1,T2,T3,T4,T5,T6,T7,T8 = Hasil pembacan sensor *thermal*

Data tersebut akan disimpan dalam variabel *"data"* yang akan dikirim ke tujuannya masing-masing. Sintaks pengiriman data ke server dan robot lain sebagai berikut:

- Client.publish("topic/DataRobot1",data) = mengirim isi variabel data berupa string ke server

B. Penerimaan Perintah

Tabel 3.4. Penerimaan Perintah untuk Robot

Sumber	Tujuan	Topic	Aksi
Robot 1	Server/PC	moveRobot1	Mengirim sinyal ke server bahwa robot 1 siap menerima perintah bergerak
	Server/PC	formasirobot	Mengirim sinyal ke server bahwa robot 1 siap menerima perintah membentuk formasi
Robot 2	Server/PC	moveRobot2	Mengirim sinyal ke server bahwa robot 1 siap menerima perintah bergerak
	Server/PC	formasirobot	Mengirim sinyal ke server bahwa robot 2 siap menerima perintah membentuk formasi
Robot 3	Server/PC	moveRobot3	Mengirim sinyal ke server bahwa robot 3 siap menerima perintah bergerak
	Server/PC	formasirobot	Mengirim sinyal ke server bahwa robot 1 siap menerima perintah membentuk formasi

Tabel 3.4 adalah daftar perintah yang diterima oleh robot dari server untuk pergerakan dan formasi. Contoh sintaksnya adalah sebagai berikut:

- Client.subscriber(“topic/moveRobot1”) = mengirim sinyal ke server yang menandakan robot 1 siap menerima perintah/*request* pergerakan dari server

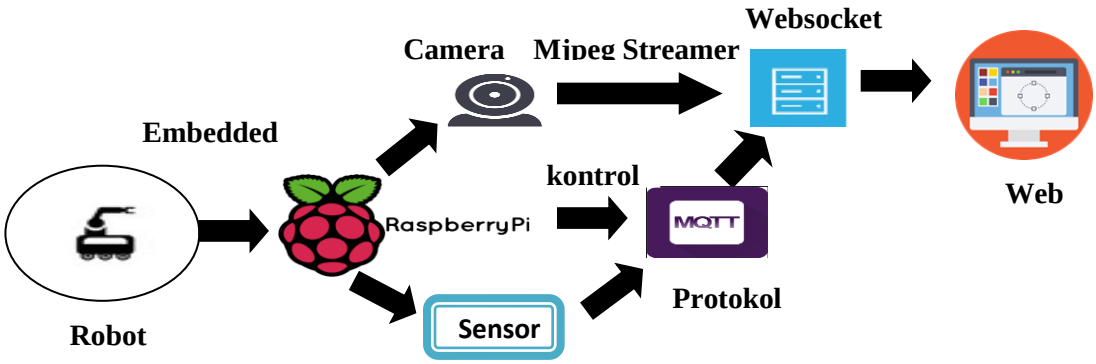
3.1.5 Desain *Multi-robot Platform*

3.1.5.1 Proses Visualisasi Data

Gambar 3.9 menunjukan proses pengiriman seluruh data dari robot sampai divisualisasikan dalam web sebagai *user interface*. Pusat komunikasi robot dilakukan di Raspberry Pi. Dari Raspberry Pi tersebut data sensor yang dikirim dari mikrokontroler diteruskan dengan menggunakan protokol komunikasi MQTT ke PC yang telah terinstal *web server* dan Mosquitto karena berperan sebagai broker atau pengumpul data juga. Web menggunakan teknologi Websocket pada gerbang akhirnya agar data yang ditampilkan di web *real time* dan *continous*.

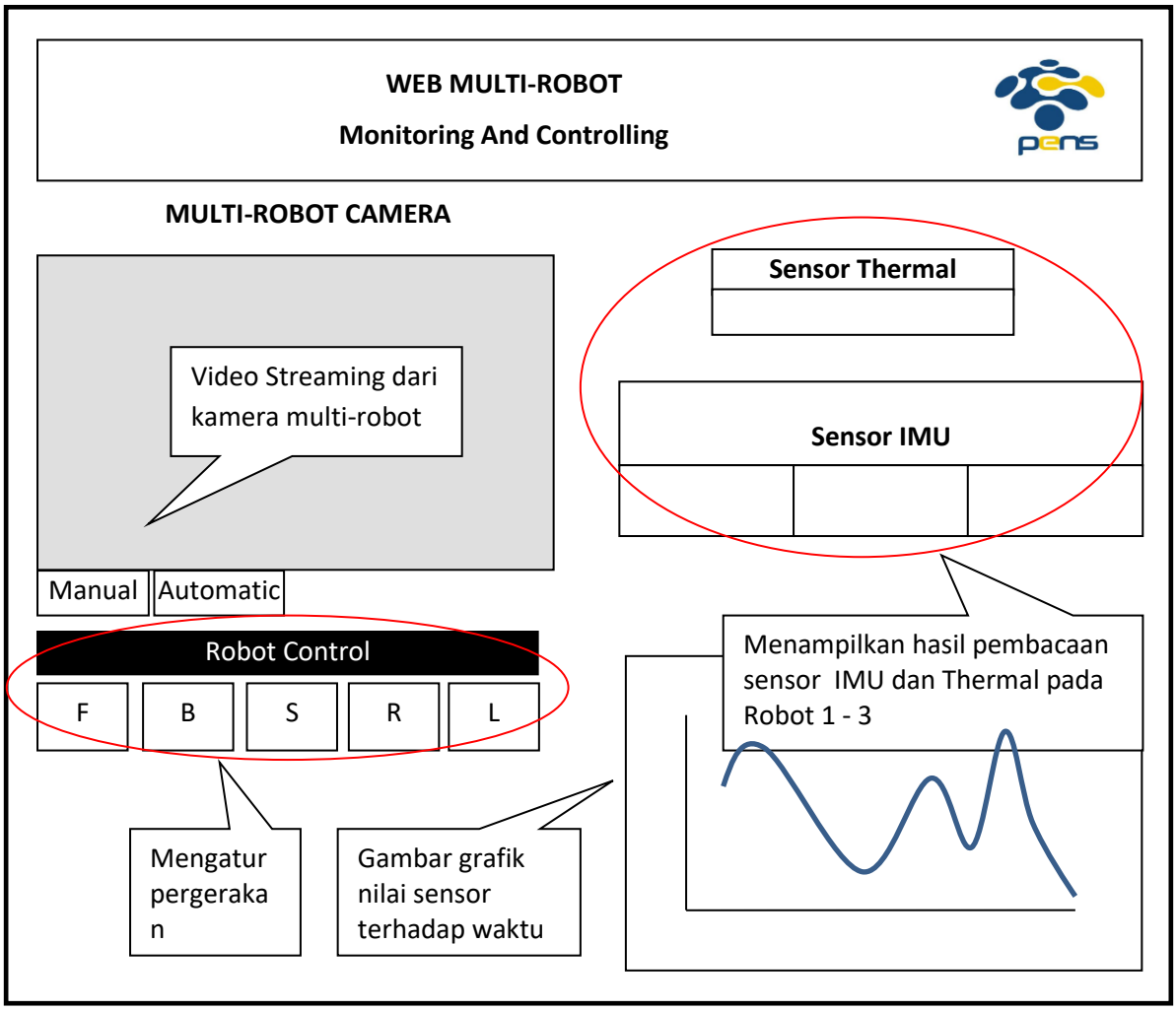
Pada proses kontrol pergerakan robot juga digunakan protokol MQTT dengan mengatur *topic* yang ada, hal ini dijelaskan secara panjang lebar dalam skema komunikasi robot. Jalur yang berbeda untuk data gambar atau video yang berasal dari *webcam*. Data tersebut tidak melalui protokol komunikasi MQTT namun memanfaatkan jaringan *wifi* yang ada dengan membuat Mjpeg-streamer sendiri menggunakan openCV Python sehingga dapat dikembangkan dalam pengolahan gambar untuk keperluan deteksi dan sebagainya.

Seluruh program untuk komunikasi jaringan dan data di simpan dalam *script-script* yang akan berjalan otomatis ketika Raspberry Pi *booting*.



Gambar 3.9. Proses Visualisasi Data di *Platform*

3.1.5.2 Desain *Web Interface*



Gambar 3.10. Desain Web untuk Multi-robot

Gambar 3.10 adalah rancangan *web interface* untuk mengontrol *disaster multi-robot*. Web ini adalah untuk *monitoring* pembacaan sensor-sensor yang ada pada multi-robot serta untuk pengendalian. Prosesnya adalah sebagai berikut:

1. Sensor TPA yang terhubung di pin 20,21 dan sensor IMU di pin 14, 15 pada Arduino di dalam multi-robot, akan mengirim data ke Raspberry, lalu Raspberry akan mengirimkan data tersebut ke server dengan protokol komunikasi. Server pada akhirnya yang akan menampilkannya secara *real time* di web.

2. Tombol F (*Forward*), B (*Back*), S (*Stop*), R (*Right*), L (*Left*) adalah tombol untuk pergerakan multi-robot yang akan mengirimkan data berupa string berdasarkan perintahnya masing-masing dari server laptop ke raspberry di dalam multi-robot. Dari raspberry tersebut data diolah lalu akan mengirim data string juga ke Arduino dengan mengatur *driver* motor yang terhubung di pin 22 , 25 serta 26, 29. Pengaturan tersebut juga berlaku untuk pergerakan *arm* robot dengan pin-pin pada Arduino yang terhubung dengan *driver* motornya.

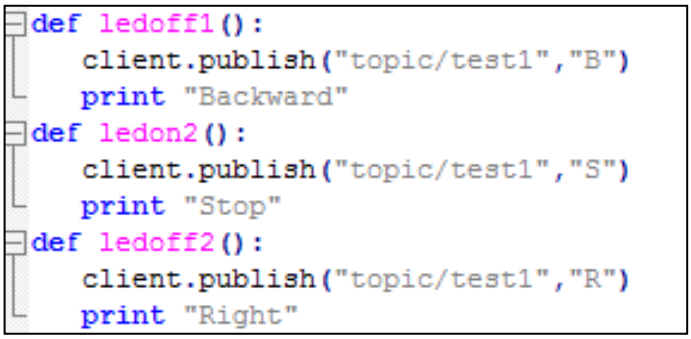
3.1.5.3 Integrasi Web dan Perintah MQTT

Web yang dibuat menggunakan *framework python* sehingga untuk integrasi dengan protokol komunikasi MQTT membutuhkan *library paho mqtt client*.

```
def on_connect(client, userdata, flags, rc):
    print("Connected with result code" + str(rc))
    client.subscribe("robotmonitoring1")
    client.subscribe("robotmonitoring2")
    client.subscribe("robotmonitoring3")

def on_message(client, userdata, msg):
  global data_3,data_4,data_5,data_6,data_7,data_8
  global datarobot1,datarobot2,datarobot3
  data=msg.payload
  print data
```

Gambar 3.11. Metode untuk menerima data dari multi-robot

Gambar 3.11 menunjukkan fungsi-fungsi untuk menerima data dari multi-robot. Fungsi ini akan berjalan selama ada koneksi diantara server dan multi-robot. Setelah data-data tersebut diterima maka akan di*parsing* agar dapat ditampilkan pada web.



Gambar 3.12. Contoh metode atau fungsi untuk mengirim perintah ke multi-robot

Gambar 3.12 menunjukkan fungsi-fungsi untuk mengirim data ke multi-robot sesuai dengan *topic*nya masing-masing. Adapun tentang perincian kode string telah dibahas pada bagian skema perintah untuk komunikasi multi-robot.

BAB 4
EKSPERIMEN DAN ANALISIS

Pada bab ini dilakukan pengujian dan analisa terhadap keseluruhan sistem yang ada pada tesis ini. Baik berupa perangkat keras dan juga perangkat lunak. Pengujian ini dilakukan agar dapat mengetahui kinerja per-bagian dari sistem. Selain itu kita juga bisa mengetahui kinerja sistem secara keseluruhan dan mengetahui apakah sistem sudah sesuai dengan perencanaan atau tidak.

4.1 PARAMETER EKSPERIMEN

Dalam penelitian ini terdapat beberapa parameter eksperimen, diantaranya: pengujian performa MQTT dan CoAP dalam hal besar total data yang diterima, *error* yang muncul serta *transfer rate* pengiriman data.

4.2 KARAKTERISTIK DATA

Data pada penelitian ini merupakan hasil pengambilan data dengan menggunakan *real hardware*. Bentuk data berupa data uji pengukuran dan performansi sistem yang telah dibuat dengan dilengkapi grafik perbandingan hasil pengukuran. Data pengukuran mencakup jumlah robot, protokol komunikasi yang digunakan.

4.3 TEMPAT UJICOBA

Untuk pengujian performansi sistem dilakukan di Laboratorium Pengujian milik grup penelitian *Robotics and Automation Based on Biologically-Inspired Technology* (RABBIT) Politeknik Elektronika Negeri Surabaya.

4.4 WAKTU UJICOBA

Penelitian ini akan berlangsung selama 12 bulan. Jadwal pelaksanaan kegiatan penelitian dijelaskan pada Tabel 4.1.

Tabel 4.1. Jadwal Pelaksanaan

No	Rincian Kegiatan	2017				2018	
		K1	K2	K3	K4	K1	K2
1	Pengumpulan Data						
3	Perancangan dan Desain Sistem						
4	Integrasi Sistem						
5	Pengujian Sistem						
6	Pelaporan dan Publikasi						

4.5 SPESIFIKASI PERALATAN UJI COBA

Pada penelitian ini digunakan beberapa peralatan yaitu: Lenovo G40-30 Laptop, yang berbasis pada unit prosesor Intel Celeron N2840 sebagai server. Setiap robot sebagi klien yang dilengkapi di dalamnya dengan sebuah Raspberry Pi 3 model B V1.2. Semua perangkat menggunakan modul *wifi onboard* untuk mendukung jaringan *Wifi-Mesh*. Server atau laptop akan mengirim data ke semua robot (3 robot) dan kinerja protokol komunikasi akan diamati dengan menggunakan Wireshark. Peralatan dan spesifikasinya untuk percobaan dirangkum dalam Tabel 4.2.

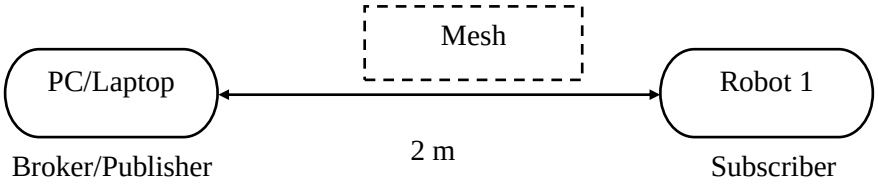
Table 4.2. Spesifikasi Peralatan

Peralatan	**Spesifikasi**
Laptop/Server	
Hardware	
Prosesor	Intel® Celeron® N2830
Wireless	RTL8723 PCIe
Software	
Routing Protocol	Batman-Adv 2016.4
Operating System	Ubuntu 16.04.4 LTS
MQTT	Paho-MQTT 1.3.1
CoAP	CoAPthon 4.0.2
Robot 1,2 & 3	
Hardware	
Prosesor	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
Wireless	BCM43438
Software	
Routing Protocol	Batman-Adv 2016.4
Operating System	Raspian Jessie 8.0
MQTT	Paho-MQTT 1.3.1
CoAP	CoAPthon 4.0.2
Test Software	
Server/PC	Wireshark 2.9.0
Raspberry	Tcpdump 4.9.2

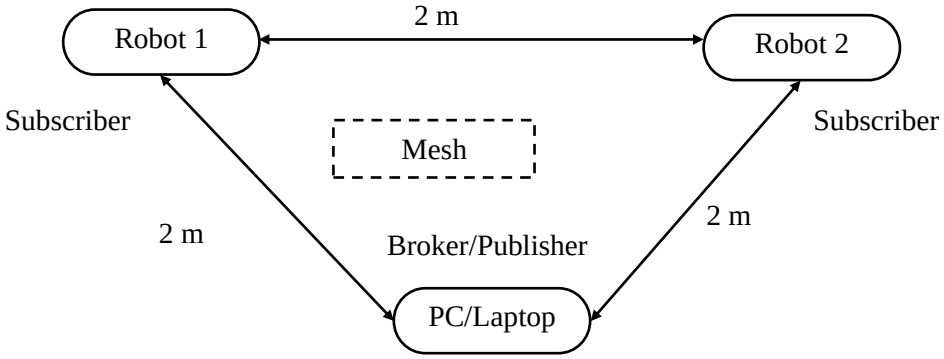
Sistem perlu dikonfigurasi untuk melihat kinerja protokol MQTT dan *Wifi-Mesh*. Gambar 4.1-4.3 menunjukan desain sistem untuk menguji protokol MQTT. Pada penelitian ini laptop digunakan sebagai broker karena *web server* diinstal di dalam laptop tersebut. *Web server* akan menampilkan data yang diterima dari multi-robot dan berfungsi sebagai GUI (*Graphical User Interface*) untuk mengatur pergerakannya. Sistem ini menggunakan bahasa pemrograman Python untuk komunikasi dan membangun webnya. Setiap percobaan diatur jarak pemisah antara robot.

Laptop atau PC akan mengirim 988 byte data string ke semua robot selama 1 jam untuk melihat kinerjanya. Percobaan ini dilakukan secara bertahap dengan mengirim ke satu, dua lalu ketiga robot. Pengiriman data dengan menggunakan

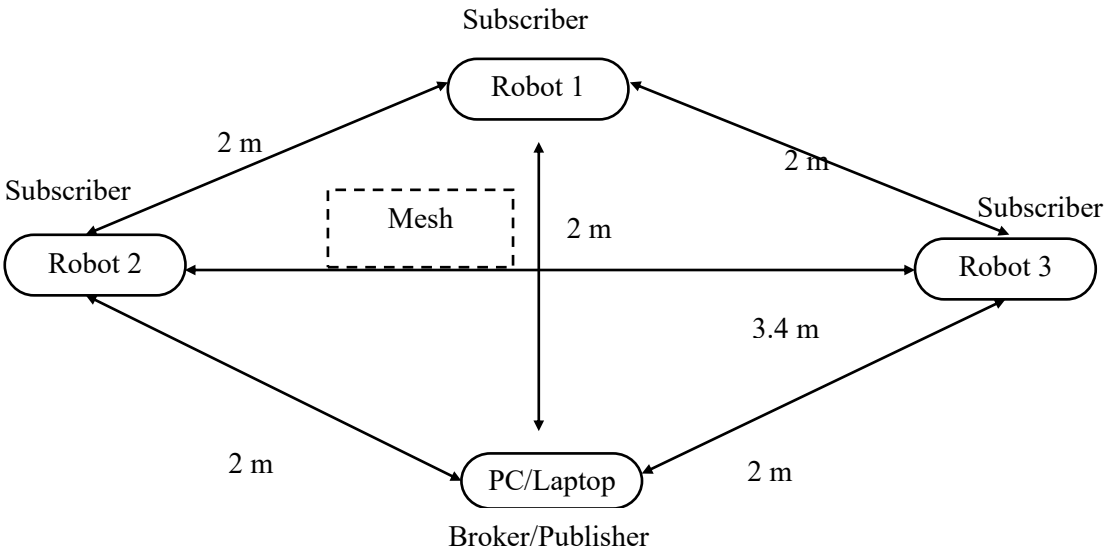
protokol MQTT ini membutuhkan sebuah *topic*. *Topic* ini menentukan arah atau lalu lintas data. Pada penelitian ini digunakan satu *topic* yang sama untuk semua robot. Nama *topic* ini adalah *"robotmonitoring".* Kesuksesan atau kegagalan penerimaan data dari sisi dapat dilihat dengan menggunakan *tool software* tcpdump.



Gambar 4.1. Desain MQTT untuk 1 robot

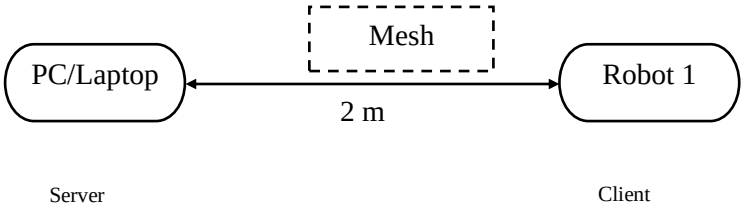


Gambar 4.2. Desain MQTT untuk 2 robot

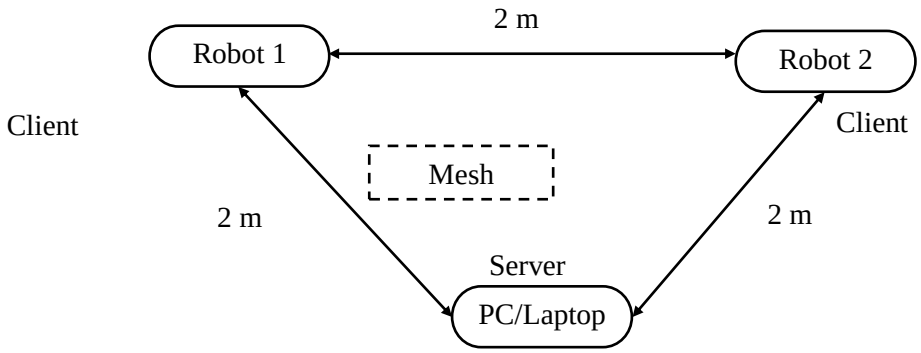


Gambar 4.3. Desain MQTT untuk 3 robot

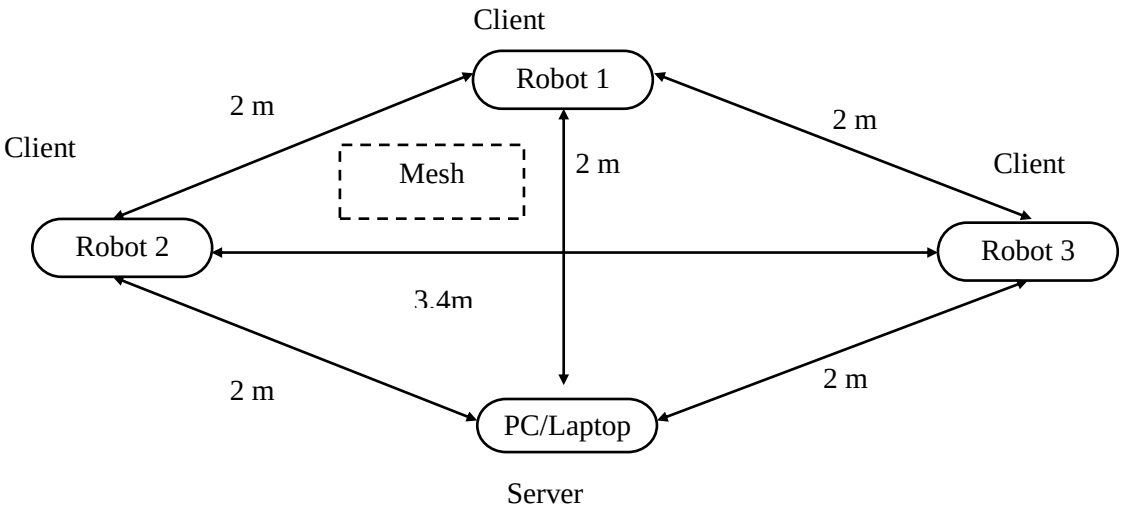
Desain sistem untuk eksperimen protokol CoAP tidak jauh berbeda dengan MQTT. Gambar 4.4-4.6 menunjukan konfigurasi protokol CoAP untuk komunikasi *disaster multi-robot*.



Gambar 4.4. Desain CoAP untuk 1 robot



Gambar 4.5. Desain CoAP untuk 2 robot



Gambar 4.6. Desain CoAP untuk 3 robot

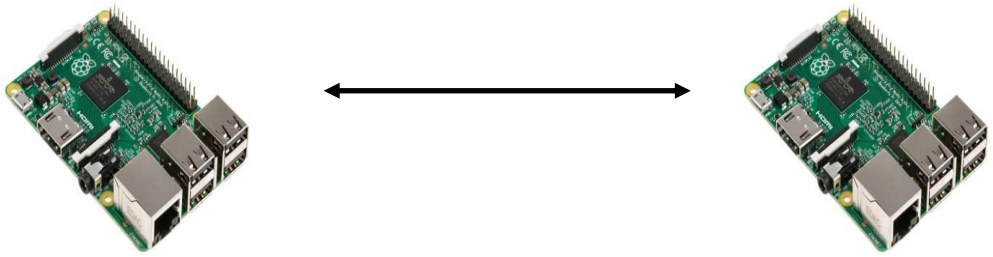
PC/Laptop dan semua robot telah diinstal dengan CoAPthon dan robot diatur sebagai klien. Sedangkan laptop sebagai server yang akan mengirim data 988 byte selama 1 jam seperti percobaan sebelumnya.Tahapan eksperimen yang akan dilakukan adalah sebagai berikut;

- Mencoba jaringan *mesh* diantara Raspberry.
- Melakukan eksperimen dengan mengirimkan data sebuah sensor yang terhubung ke Ardunio Mega lalu mengirimkannya secara serial ke Raspberry kemudian dilanjutkan dengan mengirimnya menggunakan protokol MQTT ke Raspberry lain pada jaringan *mesh* yang terbentuk.
- Melakukan eksperimen dengan mengirimkan data dari PC/Server ke multi-robot dengan protokol MQTT dan CoAP lalu mengamati data yang diterima oleh multi-robot dengan desain komunikasi yang telah dibuat masing-masing untuk 1, 2 dan 3 robot.
- Menghitung jumlah data yang diterima, *error* dan *transfer rate data* pada eksperimen sebelumnya.
- Melakukan eksperimen dengan membandingkan *throughput* komunikasi multi-robot dengan kedua protokol komunikasi tersebut saat berjalan pada berbagai jarak dari Server/PC.

4.6 HASIL EKSPERIMEN

Pada sub bab ini akan dijabarkan hasil eksperimen yang dilakukan berdasarkan beberapa parameter yang telah dikemukakan. Beberapa eksperimen yang dilakukan adalah uji coba komunikasi antara Raspberry Pi dengan jaringan *Wifi-Mesh* lalu akan diuji dengan tes koneksi dan pengiriman data.

4.6.1 Komunikasi Jaringan *Mesh* diantara 2 Raspberry



Gambar 4.7. Jaringan *mesh* diantara Raspberry Pi

Dalam membentuk jaringan *mesh* diantara 2 Raspberry Pi seperti Gambar 4.7, digunakan aplikasi batman-adv. B.A.T.M.A.N adalah protokol *routing* dalam bentuk sebuah modul kernel Linux yang beroperasi pada layer 2 dalam standar komunikasi dalam sebuah jaringan.

Hal-hal yang perlu diinstal untuk membentuk jaringan *mesh* dalam penelitian ini adalah:

- Raspbian 4.1.19
- Batman-adv 2015.0.x (tar.gz)
- Batctl 2015.0 (tar.gz)
- Linux headers 4.1.19
- Libnl-3-dev
- Gcc and g++

Setelah memastikan semuanya terpasang dengan baik lalu dilakukan pengaturan hal-hal sebagai berikut:

- Load Kernel Module
- Add the bat0 device, (e.g. form wlan0)
- Create Ad-hoc Network on wlan0
- Configure an IP Address Static on bat0

Pada Gambar 4.8 terlihat jaringan *mesh* terbentuk yang memiliki ESSID *mesh* dengan mode Ad-hoc. Gambar hubungan diantara node-node yang ada dalam jaringan *mesh* tersebut ditunjukkan pada Gambar 4.9

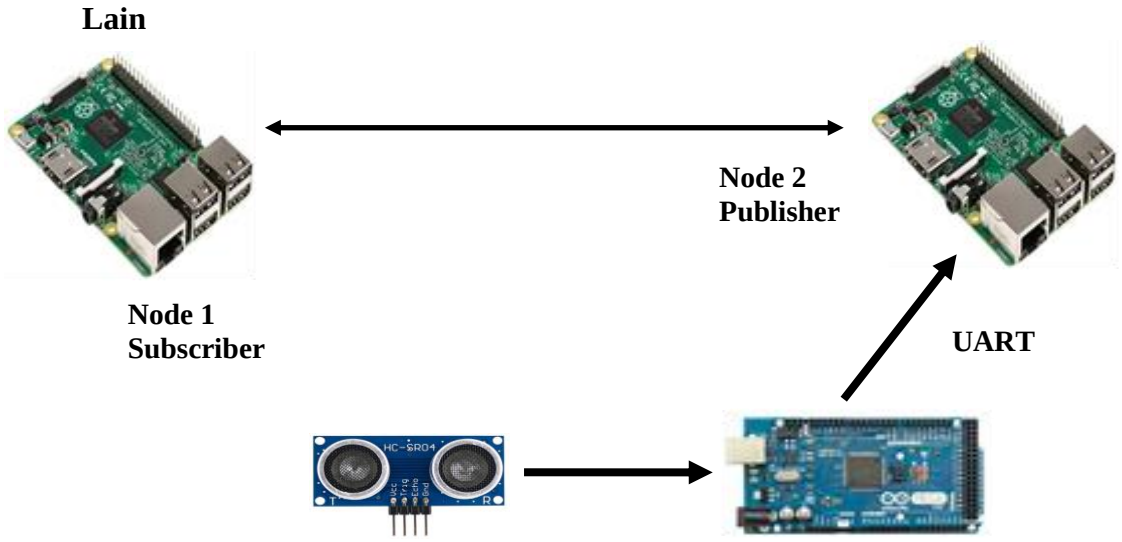
```
pi@raspberrypi:~/batctl-2015.0 $ sudo iwconfig
wlan0     IEEE 802.11bgn  ESSID:"mesh"
          Mode:Ad-Hoc  Frequency:2.422 GHz  Cell: Not-Associated
          Tx-Power=1496 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:on

bat0      no wireless extensions.

lo        no wireless extensions.

eth0      no wireless extensions.
```

Gambar 4.8. Konfigurasi yang terbentuk di jaringan *mesh*

```
pi@raspberrypi:~/batctl-2015.0 $ sudo batctl o
[B.A.T.M.A.N. adv 2015.0, MainIF/MAC: wlan0/b8:27:eb:08:c4:b3 (bat0 BATMAN_IV)]
  Originator      last-seen (#/255)          Nexthop [outgoingIF]:   Potential nexthops ...
b8:27:eb:c9:6c:53    0.070s   (247) b8:27:eb:c9:6c:53 [     wlan0]: b8:27:eb:c9:6c:53 (247)
pi@raspberrypi:~/batctl-2015.0 $
```

Gambar 4.9. Penampilan node-node yang terhubung

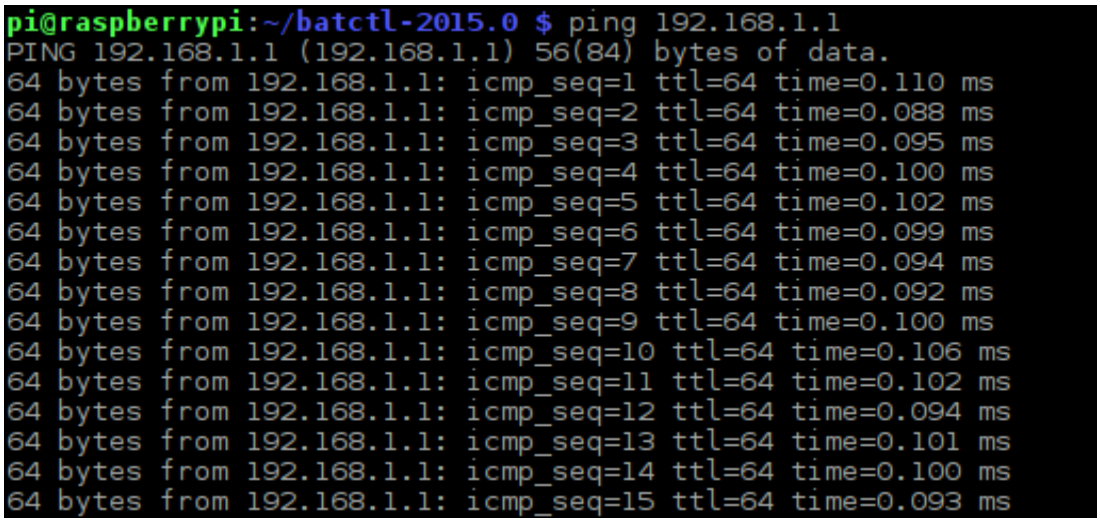
4.6.2 Pengiriman Data Sensor Ultrasonik *Real Time* dari Satu Node ke Node Lain



Gambar 4.10. Skema percobaan pengiriman data dari node 1 ke node 2

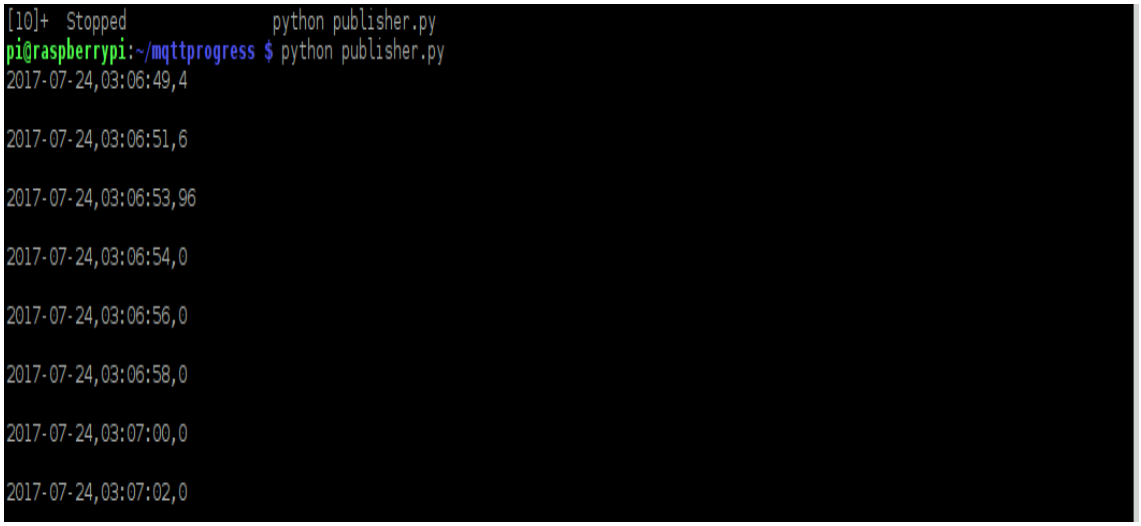
Pada percobaan ini yang sesuai dengan skema yang ditunjukkan pada Gambar 4.10, dilakukan pengiriman data sensor ultrasonik dari mikrokontroler Arduino Mega 2560 yang terhubung serial dengan node 2 yang dalam sebuah jaringan *mesh* dengan menggunakan protokol MQTT. Protokol ini digunakan menimbang karena beberapa hal diantaranya konsumsi energi yang rendah, *bandwith* jaringan yang ringan, *latency* rendah, dll

Konsep dalam MQTT menggunakan konsep *publisher-subscriber* dimana dalam percobaan ini node 2 sebagai *publisher* atau pengirim data sedangkan node 1 sebagai *subscriber* atau penerima data yang dikirimkan. Pertama kali hal yang dilakukan sebelum proses pengiriman data adalah memastikan koneksi diantara kedua node seperti yang ditunjukkan pada Gambar 4.11. Kemudian mengeksekusi program berbasis Python untuk mengirimkan data di node 2 dan mengeksekusi program untuk menerima data di node 1. Data hasil percobaan tersebut dapat dilihat pada Gambar 4.12 dan 4.13. Data berupa *timestamp* dan hasil pembacaan sensor ultrasonik.



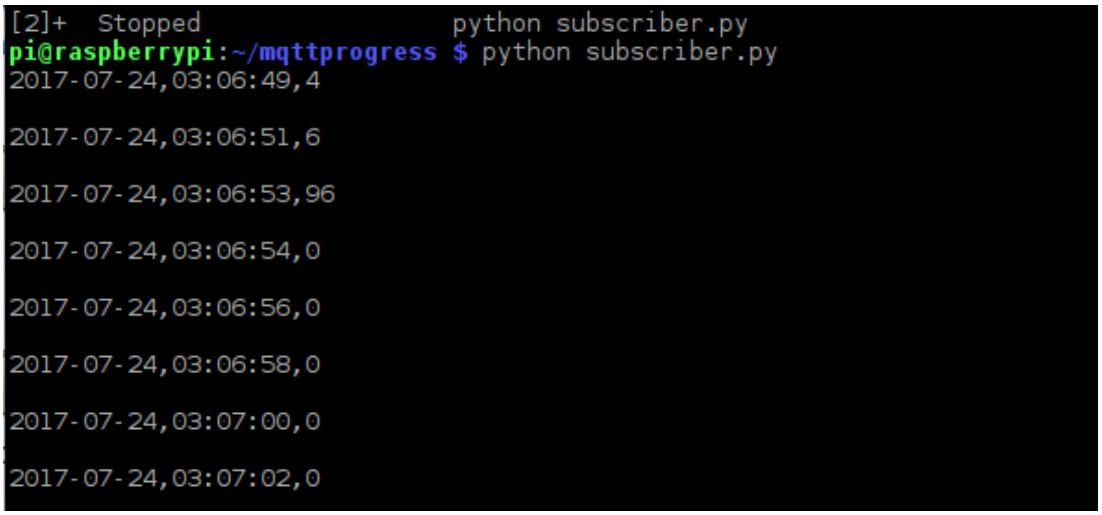
Gambar 4.11. Koneksi antara node 1 dan node 2

Gambar 4.11 uji konektivitas dengan 2 buah node yang sudah diatur ipnya 192.168.1.1 dan 192.168.1.2. Pada sisi pengirim diatur format data berupa *timestamp* lalu nilai dari sensor ultrasonik.



Gambar 4.12. *Publisher* mengirim data

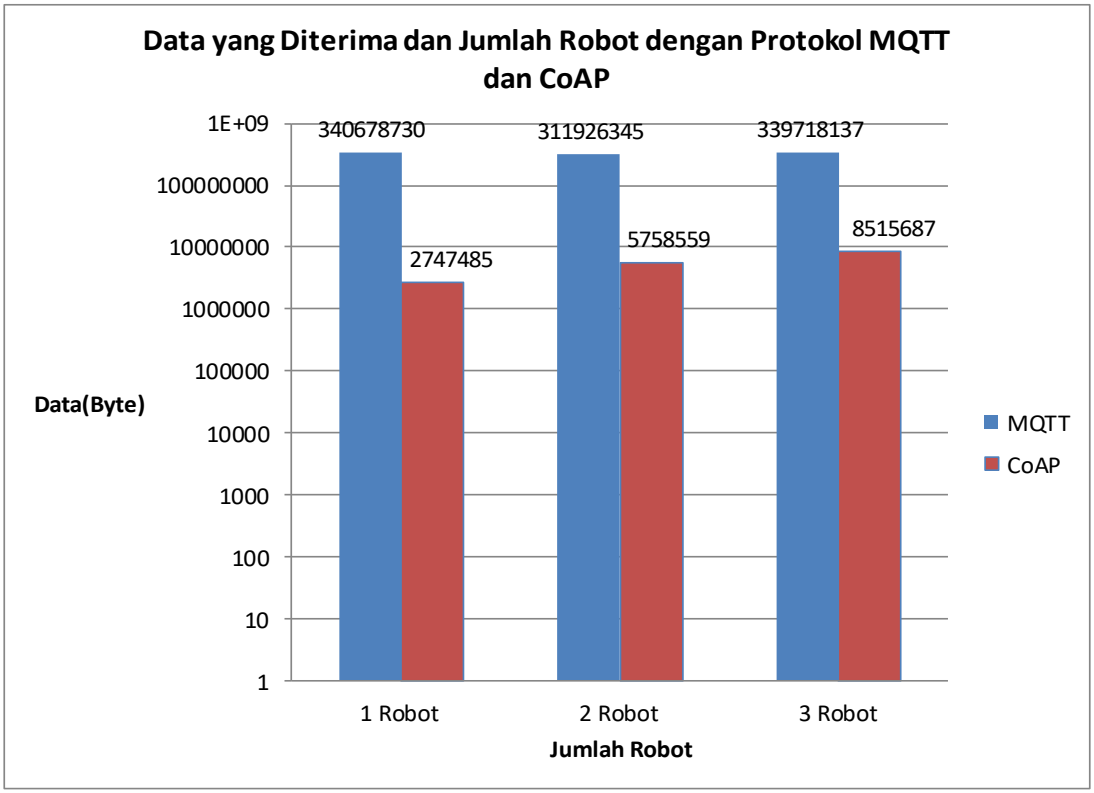
Pada Gambar 4.13 terlihat bahwa data dapat diterima dengan baik dengan format yang sama dari sisi pengirim. Pengirim dan penerima harus memiliki *topic* yang sama dan pengaturan *address*/alamat broker pada sisi penerima juga dilakukan.



Gambar 4.13. *Subscriber* menerima data

4.6.3 Pengujian kinerja MQTT dan CoAP untuk Multi-robot

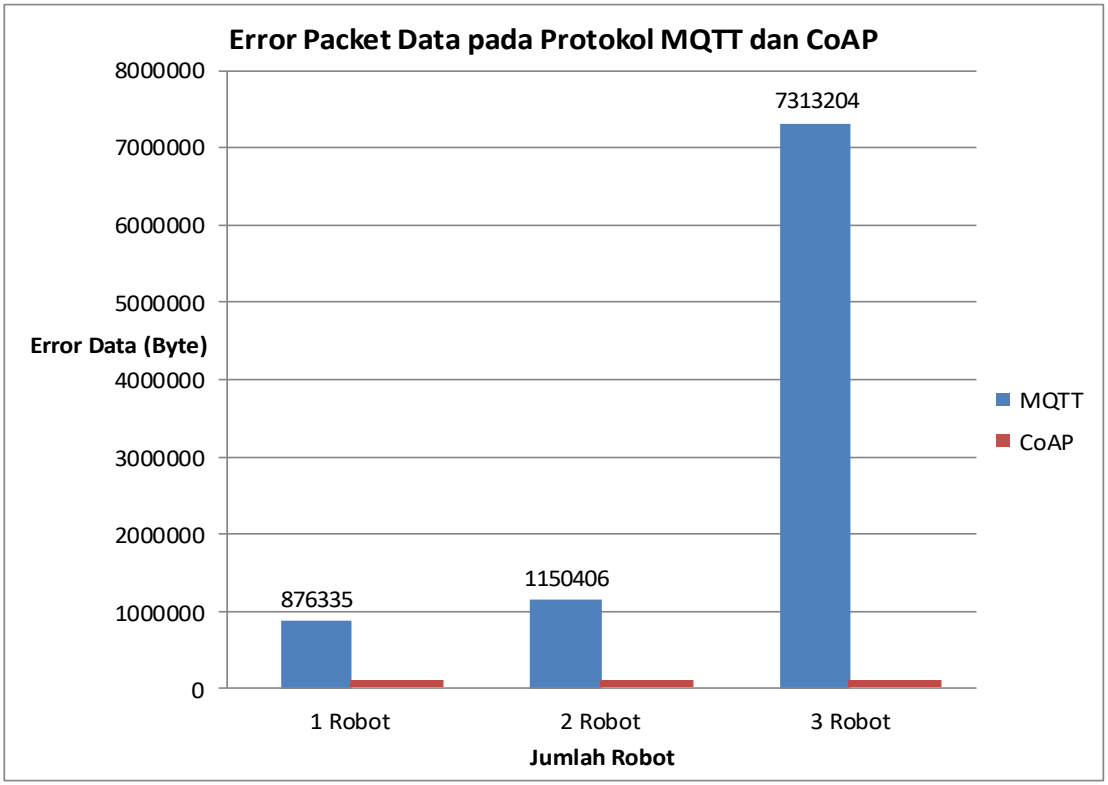
Hasil dari eksperimen untuk pengiriman data ke 1,2 dan 3 robot diplot dalam Gambar 4.14 untuk protokol MQTT dan CoAP.



Gambar 4.14. Data yang diterima multi-robot dengan protokol MQTT dan CoAP

Pada Gambar 4.14 dapat dilihat bahwa untuk protokol MQTT dengan berapapun jumlah robotnya akan menerima data yang hampir sama. Hal ini dikarenakan MQTT bekerja pada layer TCP/IP dimana tidak dapat melakukan *broadcast* data secara paralel ke semua klien. Pada CoAP, total data yang diterima semakin besar seriring dengan semakin banyaknya robot yang menerima data. Namun jumlah data yang diterima melalui protokol CoAP dengan jumlah robot yang sama jauh lebih sedikit dibanding dengan protokol MQTT. Selain itu dari segi penggunaan dan integrasi ke *user interface*, protokol MQTT lebih mudah digunakan dibanding CoAP.

Perbandingan paket data yang *error* antara MQTT dan CoAP dapat dilihat pada Gambar 4.15. CoAP tidak memiliki data *error*. Hal ini dikarenakan *transfer rate* yang rendah dan masih dalam jarak yang ideal bagi protokol tersebut untuk mengirim data, namun hal ini tidak menandakan bahwa protokol tersebut lebih baik dari MQTT dalam hal *error*.

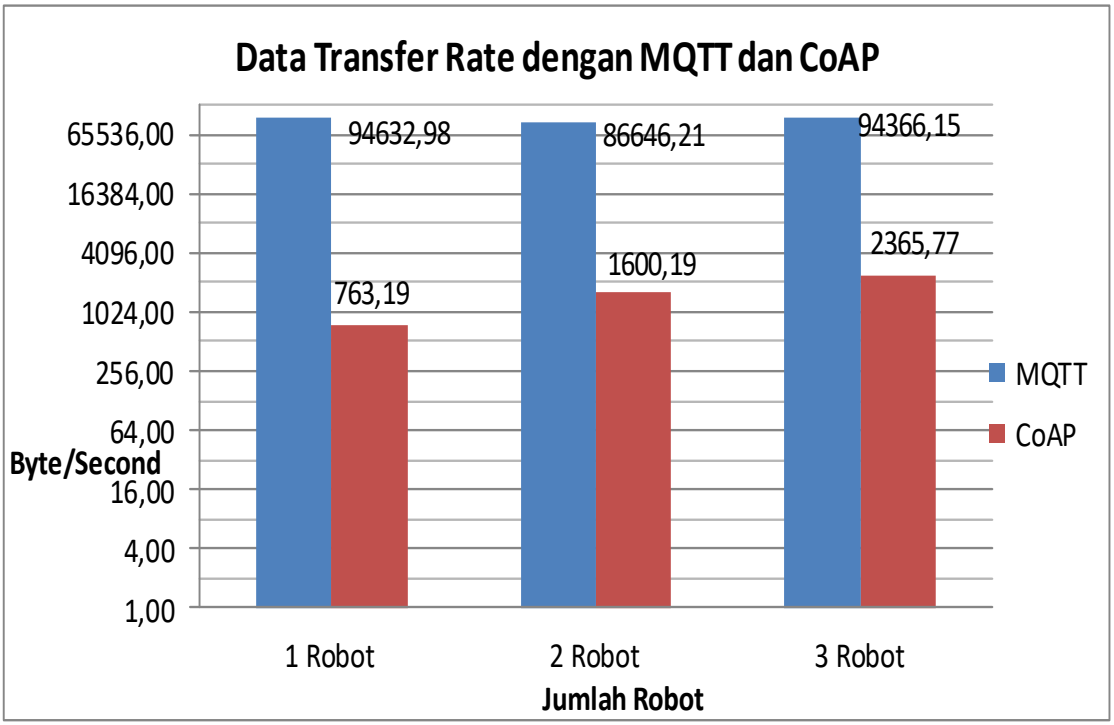


Gambar 4.15. *Error Packet Data* pada protokol MQTT dan CoAP

Pada eksperimen yang dilakukan, protokol komunikasi MQTT lebih mudah digunakan dalam pengiriman data ke robot yang menjadi target dan dapat secara penuh dikontrol dari PC/laptop yang bertindak sebagai *publisher* atau *sender* bukan dari *receiver* atau *subscriber.*

MQTT menggunakan *topic* untuk membangun alur data antara pengirim dan penerima data. Sehingga jalur komunikasi data dapat diatur dengan mudah. Sedangkan CoAP membutuhkan sebuah alamat dalam pengiriman data dan juga membutuhkan metode GET yang dieksekusi oleh tiap klien supaya memperoleh data yang diinginkan dari server sehingga yang berperan besar dalam memperoleh data adalah klien. Metode GET seperti sebuah permintaan dari klien kepada server. Kemudian server akan melayani permintaan tersebut dengan mengirim data ke klien. Metode ini perlu diatur sebelum eksekusi dalam potongan program berbasis Python mengingat digunakan CoAPthon yang merupakan *library* dari Python.

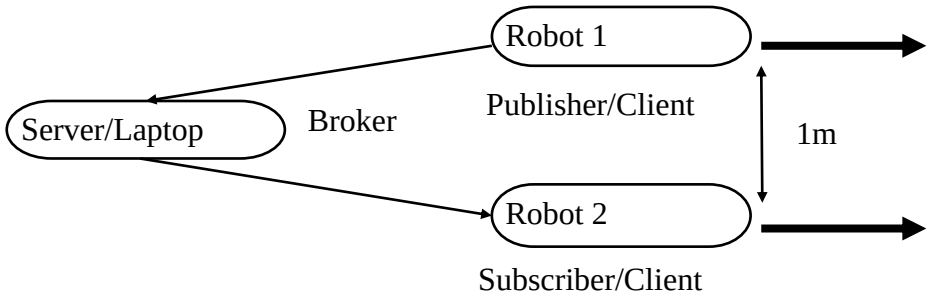
Gambar 4.16 menunjukan *transfer rate data* antara MQTT dan CoAP.



Gambar 4.16. *Transfer Rate Data* antara MQTT dan CoAP

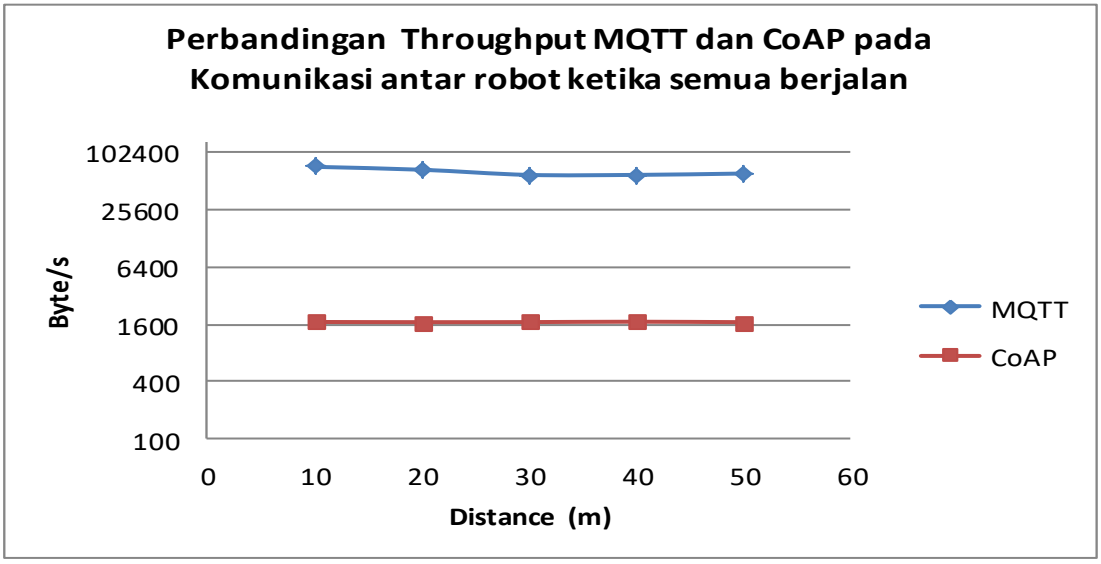
Transfer rate data untuk protokol MQTT lebih tinggi di banding CoAP sehingga menggunakan protokol MQTT menerima lebih banyak data selama satu jam (berdasarkan eksperimen). Hasil ini membawa ke kesimpulan bahwa protokol MQTT dapat digunakan untuk komunikasi *real time* dibanding CoAP.

Untuk tes komunikasi diantara multi-robot maka dibuat konfigurasi sebagai berikut.



Gambar 4.17. Skema komunikasi antar 2 robot dengan protokol MQTT dan CoAP

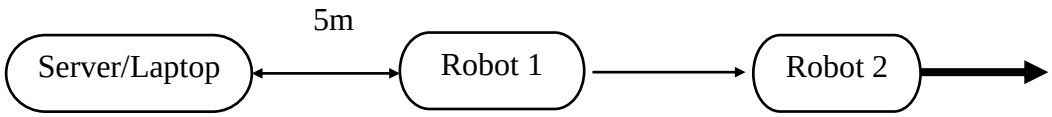
Robot 1 mengirim 988-byte data string ke robot 2 melalui laptop yang berfungsi sebagai broker dalam konfigurasi protokol MQTT dan sebagai server pada konfigurasi protokol CoAP. Jarak antara robot adalah 1 m dan posisi laptop telah ditentukan. 2 robot bergerak maju bersama dan menjauhi laptop lalu diamati data *throughput* pada jarak-jarak yang ditentukan. Gambar 4.18 menunjukkan hasil percobaan.



Gambar 4.18. *Throughput* MQTT dan CoAP pada komunikasi antar robot ketika semua berjalan

Gambar 4.18 menunjukkan bahwa kinerja protokol MQTT lebih baik dibanding CoAP karena memiliki *throughput* yang lebih besar . Hal ini menambah keunggulan protokol tersebut.

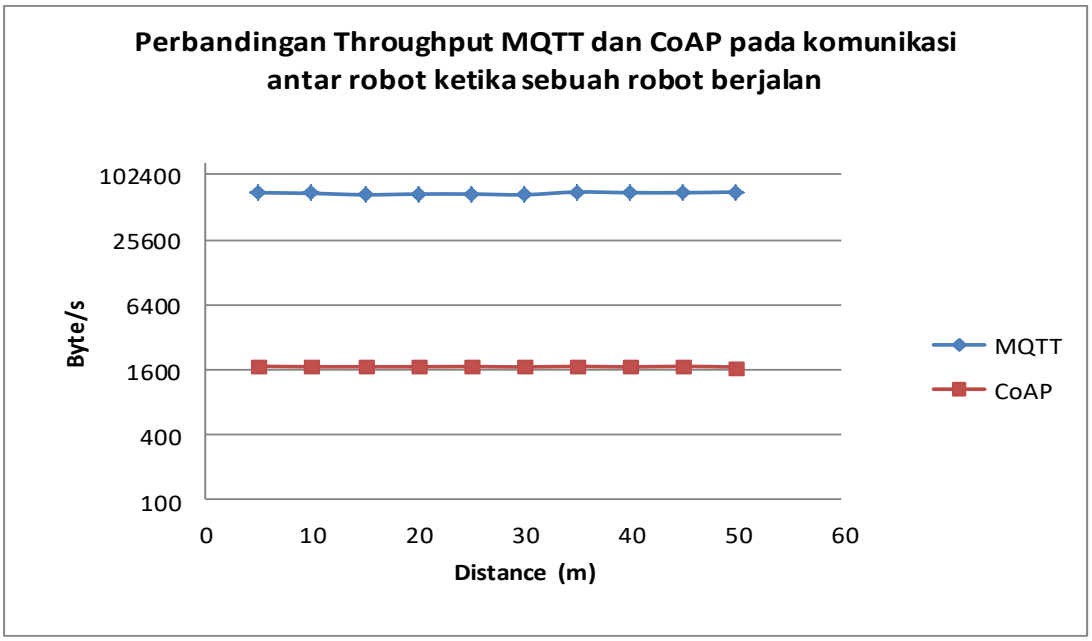
Setelah itu, dilakukan juga eksperimen dengan konfigurasi seperti pada Gambar 4.19.



Gambar 4.19. Skema komunikasi antar 2 robot dengan protokol MQTT dan CoAP ketika salah satu robot bergerak

Pada eksperimen ini, robot 1 mengirim data ke robot 2 namun posisi robot 1 dan posisi laptop/server tetap dan berjarak 5 m satu sama lain. Robot 2 bergerak

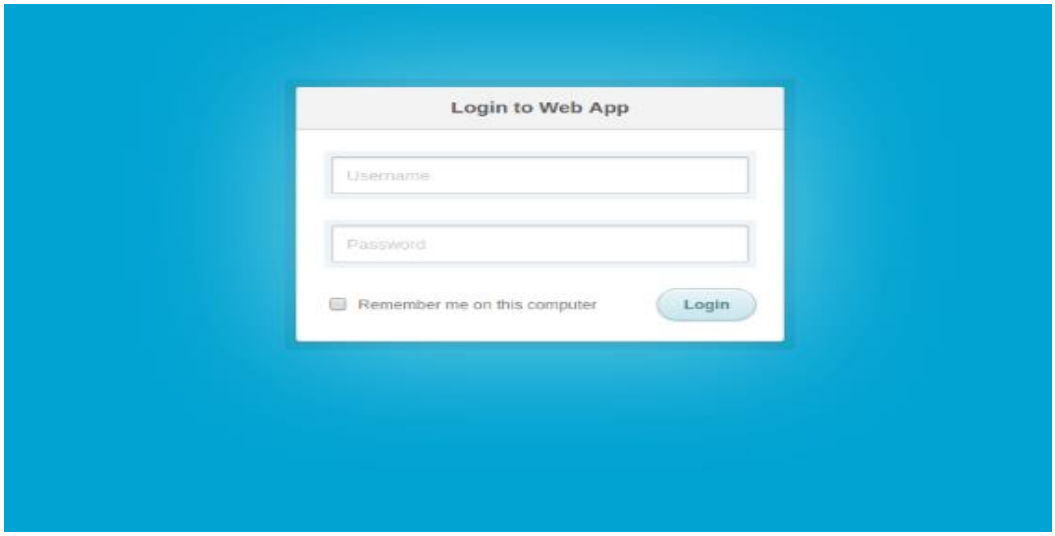
menjauhi keduanya kemudian diukur kinerja dari protokol MQTT dan CoAP saat jarak-jarak tertentu. Hasil percobaan ini dapat dilihat pada Gambar 4.20.



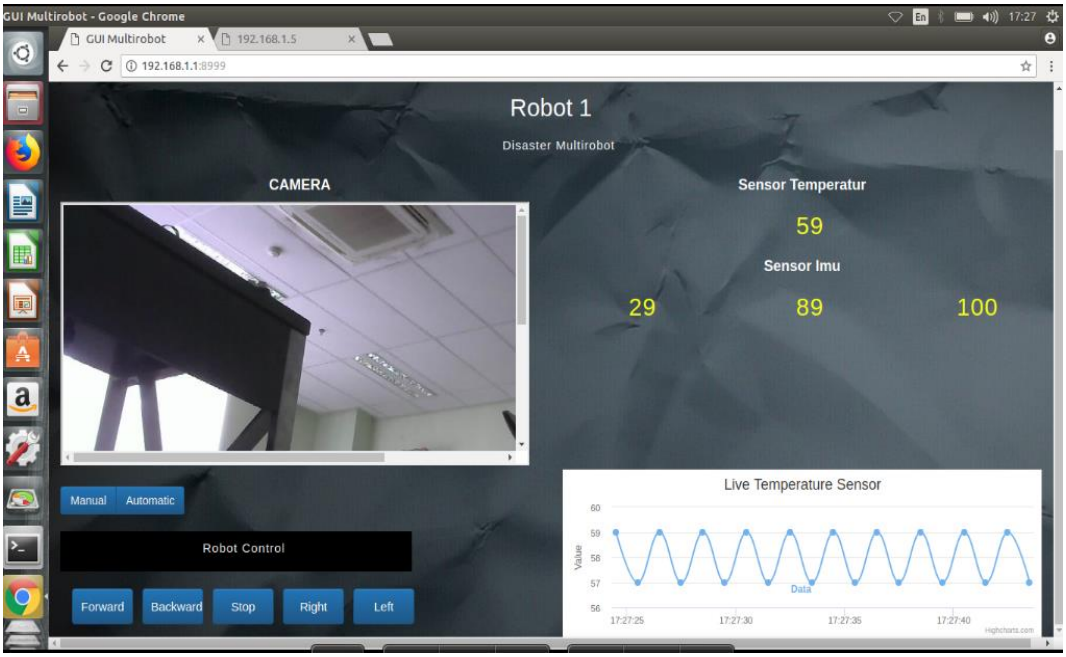
Gambar 4.20. *Throughput* MQTT dan CoAP pada komunikasi antar robot ketika salah satu robot berjalan

Gambar 4.20 juga menunjukan kinerja MQTT yang memiliki *throughput* besar dan stabil pada range tertentu melebihi CoAP.

4.6.4 *Web server* untuk *Disaster Multi-robot*



Gambar 4.21. Halaman utama *login*



Gambar 4.22. Tampilan Web di PC

Gambar 4.21 menunjukkan halaman login untuk keamanan *web interface* sedangkan Gambar 4.22 menunjukkan *web interface* untuk mengendalikan multi-robot dengan menggunakan *framework* Tornado yang berbasis Python. *Framework* tersebut dipilih karena ringan dan *powerful* untuk membangun sebuah aplikasi web. Web tersebut berisi tombol untuk mengatur pergerakan, sedangkan layar pada bagian sebelah kiri untuk menampilkan gambar hasil kamera pada robot. Pada bagian sebelah kanan berfungsi menghasilkan data sensor secara *real time*. Pada bagian tombol dan pembacaan sensor digunakan protokol Websocket untuk komunikasi *duplex* (2 arah) melalui koneksi TCP tunggal.

4.7 ANALISIS HASIL EKSPERIMEN

Untuk mengetahui performansi sistem yang telah dibuat dilakukan analisis mengenai kinerja sistem yaitu:

1. Jaringan *wifi-mesh* dapat berjalan dengan baik dengan mengatur ip masing-masing node dan diuji dengan tes konektivitas dan pengiriman data dari satu node ke node lain dari hasil pembacaan sensor ultrasonik

2. Perbandingan kinerja antara protokol MQTT dan CoAP pada *platform* diantaranya:
   a. Total data yang diterima keseluruhan robot dengan menggunakan MQTT jauh lebih banyak sekitar ± 300 juta byte untuk semua jumlah robot atau 98.28% lebih besar dari yang diterima oleh CoAP
   b. *Error* yang dihasilkan dalam pengiriman data menggunakan MQTT memiliki nilai yang besar dibanding CoAP mengingat besarnya juga data yang diterima oleh robot dengan maksimal *error* ± 73 juta byte sedangkan CoAP mendekati nol atau tidak mengalami *error* sehingga besar data yang dikirim 100 % diterima oleh robot.
   c. *Transfer rate data* dengan menggunakan MQTT memiliki nilai maksimal 94366.15 byte/second jauh meninggalkan protokol CoAP sebesar 2365.77 byte/second untuk 3 robot atau rata-rata 98.28% lebih cepat dibanding CoAP.
3. Secara singkat perbedaan antara MQTT dan CoAP ditunjukkan pada Tabel 4.3

Tabel 4.3. Perbedaan Protokol MQTT dan CoAP

	MQTT	**CoAP**
Transport Layer	TCP	UDP
Arsitektur yang didukung	Publish- Subscribe	Request-Response,Resource-Observe/Publish-Subscribe
Header	2 byte	4 byte
Power Consumption	Lebih tinggi dibanding CoAP	Lebih rendah dibanding MQTT
Transfer Rate	Lebih tinggi dibanding CoAP	Lebih rendah dibanding MQTT
Throughput	Lebih tinggi dibanding CoAP	Lebih rendah dibanding MQTT
Data	Cocok untuk pengiriman pesan secara *real time*	Tidak cocok untuk pengiriman pesan secara *real time*

4. *Web interface* dapat menampilkan data-data hasil *sensing* sensor pada masing-masing robot kemudian dikirim menggunakan protokol MQTT ke PC/server lalu akan di*parsing* dan ditampilkan secara *real time*
5. Pengujian perintah-perintah untuk multi-robot dengan *user interface* menunjukkan hasil yang baik dalam hal pergerakan dan me*monitoring* data hasil *sensing*.

4.8 DISKUSI

Penelitian ini berangkat dari sebuah ide untuk menemukan sebuah metode untuk menyelesaikan sebuah persoalan yaitu komunikasi efektif pada multi-robot di daerah bencana.

Hal ini lalu dilanjutkan berdasarkan pada perkembangan teknologi *Internet of Things* yang bertujuan menjadikan segala perangkat di sekeliling kita terkontrol dan terawasi. Kemudian kami mencoba menerapkan protokol tersebut pada *disaster multi-robot*. Disamping itu kami mencoba membandingkan 2 protokol yang sering digunakan dalam teknologi tersebut dari sisi kinerjanya.

Ke depannya, penelitian ini dapat dilanjutkan dengan menggali lebih dalam penerapan yang lebih real di daerah bencana bersama tim SAR milik pemerintah sehingga segala hambatan yang ada di lapangan dapat dilengkapi dan target dapat tercapai dengan baik. Penelitian ini juga dapat dikembangkan dengan mengumpulkan data-data di area bencana yang akan di simpan dalam Big Data sehingga memungkinkan memanfaatkan *Artificial Intellegence* dalam menghasilkan keputusan-keputusan strategis berdasarkan hasil pengolahan data-data tersebut.

BAB 5
PENUTUP

5.1 KESIMPULAN

Penelitian ini telah bertujuan untuk membangun jaringan komunikasi *mesh* multi-robot dengan memanfaatkan protokol komunikasi teknologi *Internet of Things* agar dapat menampilkan data yang dibutuhkan secara *real time* dan berkoordinasi dengan baik.

Salah satu bagian dari eksperimen ini adalah membandingkan protokol komunikasi antara MQTT dan CoAP untuk mengendalikan *disaster multi-robot* secara jarak jauh. Hasil eksperimen menunjukkan bahwa protokol MQTT sangat cocok untuk komunikasi *real time* karena memiliki *transfer rate* yang tinggi dalam mengirim data di banding CoAP dengan selisih perbandingan sebesar 98.28 %, dan selisih jumlah data sebesar 98,28%. Selain itu protokol MQTT memiliki kemudahan dalam penggunaannya di dalam topologi *mesh* dan mengintegrasinya dengan *user interface* (*Web*) dibanding CoAP. Protokol MQTT bekerja pada TCP/IP dan protokol CoAP bekerja pada UDP sehingga koreksi *error* tidak begitu dibutuhkan.

5.2 SARAN

Berdasarkan hasil penelitian tesis maka dapat disarankan:

a. Pengembangan komunikasi dapat dilakukan dengan menambah variasi koordinasi multi-robot dari jaringan yang telah terbentuk beserta target-target yang ingin dicapai. Penggunaan protokol *routing* yang lain juga mungkin bisa digunakan sebagai pembanding.
b. Pengembangan sistem untuk memperluas jaringan *wifi* yang ada sehingga dapat beroperasi di area yang lebih lebar
c. Pengembangan dengan menambahkan penggunaan robot terbang seperti *drone* dan sejenisnya atau robot berkaki untuk membangun kerjasama yang efektif

dalam *task searching* dan *rescue.* Selain itu dapat juga berupa penambahan *task-task* seperti menampilkan area bencana secara 3D menggunakan Lidar, mengangkut korban, membawa tabung oksigen, mendeteksi gas beracun, dll

d. Pengembangan *web interface* bisa dengan menampilkan konektivitas robot yang ada sehingga dapat melakukan *maintenance* dengan cepat jika diperlukan.

DAFTAR PUSTAKA

[1] E. Mediastianto, “Statistik Kejadian Bencana Tahun 2014,” 2015. [Online]. Available: http://www.penanggulangankrisis.depkes.go.id/statistik-kejadian-bencana-tahun-2014. [Accessed: 11-Feb-2016].

[2] K. Nagatani *et al.*, “Emergency Response to the Nuclear Accident at the Fukushima Daiichi Nuclear Power Plants using Mobile Rescue Robots,” *J. F. Robot.*, vol. 24, no. 5, pp. 421–434, 2007.

[3] J. Casper and R. R. Murphy, “Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center.,” *IEEE Trans. Syst. Man. Cybern. B. Cybern.*, 2003.

[4] R. M. Voyles, “TerminatorBot: a robot with dual-use arms for manipulation and\nlocomotion,” *Proc. 2000 ICRA. Millenn. Conf. IEEE Int. Conf. Robot. Autom. Symp. Proc. (Cat. No.00CH37065)*, vol. 1, pp. 61–66, 2000.

[5] R. Masuda, T. Oinuma, and A. Muramatsu, “Multi-sensor control system for rescue robot,” *Proc. 1996 IEEEISICEIRSJ Int. Conf. Multisens. Fusion Integr. Intell. Syst. Multi-sensor*, no. 4, pp. 381–387, 1996.

[6] R. R. Murphy, J. Kravitz, S. L. Stover, and R. Shoureshi, “Mobile robots in mine rescue and recovery,” *IEEE Robot. Autom. Mag.*, vol. 16, no. 2, pp. 91–103, 2009.

[7] D. P. Stormont, “Autonomous rescue robot swarms for first responders,” in *Proceedings of the 2005 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety, CIHSPS 2005*, 2005, vol. 2005, pp. 151–157.

[8] Nikolaus Correll and Alcherio Martinoli, “Towards Optimal Control Of Self-Organized Robotic Inspection Systems,” *Robotics*, 2003.

[9] “Seaswarm is a fleet of low-cost oil absorbing robots,” 2010. [Online]. Available: http://senseable.mit.edu/seaswarm/index.html. [Accessed: 24-Jul-2017].

[10] J. R. T. Lawton, R. W. Beard, and B. J. Young, “A decentralized approach to formation maneuvers,” *IEEE Trans. Robot. Autom.*, vol. 19, no. 6, pp. 933–

941, 2003.

[11] A. Marjovi, J. Nunes, P. Sousa, R. Faria, and L. Marques, “An olfactory-based robot swarm navigation method,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 4958–4963.

[12] V. Sperati, V. Trianni, and S. Nolfi, “Self-organised path formation in a swarm of robots,” *Swarm Intell.*, vol. 5, no. 2, pp. 97–119, 2011.

[13] H.-Q. M. H.-Q. Min, J.-H. Z. J.-H. Zhu, and X.-J. Z. X.-J. Zheng, “Obstacle avoidance with multi-objective optimization by PSO in dynamic environment,” *2005 Int. Conf. Mach. Learn. Cybern.*, vol. 5, no. August, pp. 18–21, 2005.

[14] J.-H. Lee and C. W. Ahn, “Improving Energy Efficiency in Cooperative Foraging Swarm Robots Using Behavioral Model,” *2011 Sixth Int. Conf. Bio-Inspired Comput. Theor. Appl.*, pp. 39–44, 2011.

[15] S. Kahar, R. Sulaiman, A. S. Prabuwono, and N. A. Ahmad, “A Review of Wireless Technology Usage for Mobile Robot Controller,” vol. 34, no. Icsem, pp. 7–12, 2012.

[16] P. T. Corp., “Smart-City built by Well-Managed and High Quality Wireless Mesh Infrastructure.” [Online]. Available: http://www.pheenet.com.tw/solutions2_content?id=4. [Accessed: 18-Dec-2018].

[17] Lina Afriana, “Implementasi dan analisis kinerja Routing Protocol B.A.T.M.A.N-ADV (Better Approach To Mobile Ad-Hoc Networking Advanced) pada jaringan berbasis Wireless Mesh,” 2013.

[18] “Pengertian B.A.T.M.A.N Karakteristik, Mekanisme Routing, (Protokol Better Approach To Mobile Ad Hoc Network (B.A.T.M.A.N),”2015.[Online].Available:http://www.landasanteori.com/2015/10/pengertian-batman-karakteristik.html.[Accessed: 10-Aug-2018].

[19] S. Lindner, M., Neumann, A., Aichele, C. & Wunderlich, *Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.) : draft- wunderlich-openmesh-manet-routing-00*. Internet-Draft Network Working Group (IETF), 2008.

[20] “B.A.T.M.A.N. advanced.” [Online]. Available: https://www.open-mesh.org/projects/batman-adv/wiki/Wiki. [Accessed: 10-Aug-2018].

[21] “Raspberry Pi 2 Model B,” 2015. [Online]. Available: http://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Accessed: 11-Feb-2016].

[22] Multicherry, “Top half of Raspberry Pi 2 Model B v1.1 viewed directly from above,” 2015. [Online]. Available: https://commons.wikimedia.org/wiki/File:Raspberry_Pi_2_Model_B_v1.1_top_new.jpg. [Accessed: 13-Aug-2018].

[23] Fahmizal, “cara kerja Sensor Thermal TPA 81,” 2010. [Online]. Available: https://fahmizaleeits.wordpress.com/tag/cara-kerja-sensor-thermal-tpa-81/#jp-carousel-1069. [Accessed: 26-Jul-2018].

[24] E. Pr., “Mengenal MQTT, Protokol IoT,” 2015. [Online]. Available: http://jsiot.pw/mengenalmqtt998b6271f585. [Accessed: 24-Jul-2017].

[25] “Understanding the MQTT Protocol Packet Structure,” 2018. [Online]. Available: http://www.steves-internet-guide.com/mqtt-protocol-messages-overview/. [Accessed: 18-Dec-2018].

[26] Y. F. Wiryawan, D. P. Kartikasari, and M. Data, “Implementasi Constrained Application Protocol (CoAP) pada Sistem Pengamatan Kelembaban Tanah,” vol. 2, no. 8, pp. 2480–2487, 2017.

[27] IETF, “RFC-7252-The Constrained Application Protocol (CoAP),” 2014. [Online]. Available: https://tools.ietf.org/html/rfc7252. [Accessed: 18-Dec-2018].

[28] M. Sato, K. Kamei, S. Nishio, and N. Hagita, “The ubiquitous network robot platform: Common platform for continuous daily robotic services,” *2011 IEEE/SICE Int. Symp. Syst. Integr. SII 2011*, pp. 318–323, 2011.

[29] T. Sun, X. Xiang, W. Su, H. Wu, and Y. Song, “A transformable wheel-legged mobile robot: Design, analysis and experiment,” *Rob. Auton. Syst.*, vol. 98, pp. 30–41, 2017.

[30] S. Kuswadi, M. N. Tamara, D. A. Sahanas, G. I. Islami, and S. Nugroho, “Adaptive morphology-based design of multi-locomotion flying and crawling robot ‘PENS-FlyCrawl,’” *2016 Int. Conf. Knowl. Creat. Intell. Comput. KCIC 2016*, pp. 80–87, 2017.

[31] B. Doroodgar, Y. Liu, and G. Nejat, “A learning-based semi-autonomous

controller for robotic exploration of unknown disaster scenes while searching for victims," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2719–2732, 2014.

[32] J. S. Jennings, G. Whelan, and W. F. Evans, "Cooperative search and rescue with a team of mobile robots," *1997 8th Int. Conf. Adv. Robot. Proceedings. ICAR'97*, pp. 193–200, 1997.

[33] N. Aroon, "Study of using MQTT cloud platform for remotely control robot and GPS tracking," *2016 13th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol. ECTI-CON 2016*, 2016.

[34] D. Thangavel, X. Ma, A. Valera, H. X. Tan, and C. K. Y. Tan, "Performance evaluation of MQTT and CoAP via a common middleware," *IEEE ISSNIP 2014 - 2014 IEEE 9th Int. Conf. Intell. Sensors, Sens. Networks Inf. Process. Conf. Proc.*, no. April, pp. 21–24, 2014.

[35] N. Stephen, "Power Profiling: HTTPS Long Polling vs. MQTT with SSL, on Android," 2012. .

[36] N. De Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing," in *IEEE SCVT 2013 - Proceedings of 20th IEEE Symposium on Communications and Vehicular Technology in the BeNeLux*, 2013.

[37] S. Bandyopadhyay and A. Bhattacharyya, "Lightweight Internet protocols for web enablement of sensors using constrained gateway devices," in *2013 International Conference on Computing, Networking and Communications, ICNC 2013*, 2013, pp. 334–340.

[38] M. H. Amaran, N. A. M. Noh, M. S. Rohmad, and H. Hashim, "A Comparison of Lightweight Communication Protocols in Robotic Applications," *Procedia Comput. Sci.*, vol. 76, no. Iris, pp. 400–405, 2015.

[39] H. Yuliandoko, S. Sukaridhoto, U. H. Al Rasyid, and N. Funabiki, "Performance of Implementation IBR-DTN and Batman-Adv Routing Protocol in Wireless mesh Networks," *Emit. Int. J. Eng. Technol.*, vol. 3, no. 1, 2015.

[40] L. M. Cortés-peña, "Wireless Mesh Network Implementation," *Comput. Eng.*, 2007.

[41] Davinder Singh Sandhu and Sukesha Sharma, "Performance Evaluation of

BATMAN, DSR, OLSR Routing Protocols - A Review," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 2, no. 1, pp. 184–188, 2012.

[42] D. Seither, A. König, and M. Hollick, "Routing performance of wireless mesh networks: A practical evaluation of BATMAN advanced," in *Proceedings - Conference on Local Computer Networks, LCN*, 2011, pp. 897–904.