APLIKASI JARINGAN KOMUNIKASI ROBOT *MULTIHOP* TERDISTRIBUSI PADA LINGKUNGAN STATIS TERBATAS: IMPLEMENTASI, SIMULASI DAN ANALISIS PADA KASUS ROBOT LEGO MINDSTORM NXT

W. Jatmiko, A.A. Krisnadhi, R. M. Mardian, dan N.W. Pambudi

Fakultas Ilmu Komputer, Universitas Indonesia, Depok, Indonesia
wisnuj@ui.ac.id

Abstrak

Paper ini akan membahas tahap awal (prototipe) pengembangan jaringan komunikasi-robot *multihop* terdistribusi, yaitu suatu aplikasi yang diharapkan dapat dimanfaatkan pada lingkungan yang minim infrastruktur dan fasilitas komunikasi. Pada bagian awal *paper* disebutkan kondisi yang dapat menerapkan aplikasi jaringan komunikasi *multihop* ini, yaitu pada daerah bencana alam yang mengalami kerusakan sarana komunikasi, pada daerah terpencil karena faktor alam yang sulit untuk diadakan fasilitas komunikasi, ataupun daerah berbahaya untuk didatangi manusia ataupun kawasan konflik dan peperangan. Salah satu karakteristik yang diharapkan dari aplikasi ini adalah proses penyebaran informasi secara cepat dan mudah, bahkan walaupun tidak tersedia infrastruktur memadai sebelumnya. Selain itu, sistem ini bersifat terdistribusi sehingga diharapkan dapat memberikan beberapa keuntungan, baik saat diimplementasikan maupun kinerja di lapangan nantinya. *Paper* ini kemudian membahas pengembangan algoritma penyelesaian masalah, yang dilanjutkan dengan verifikasi dan analisis pada level simulasi perangkat lunak. Selanjutkan, algoritma ini diterapkan pada level simulasi perangkat keras dengan menggunakan modul robot Lego *Mindstorm* NXT. Untuk penyederhanaan masalah, pada tahap ini lingkungan yang digunakan masih bersifat statis dan terbatas sebagai salah satu asumsi.

Kata kunci: *Jaringan robot bergerak, robot terdistribusi, jaringan komunikasi multi-hop, robot Lego Mindstorm nxt.*

1. Pendahuluan

Salah satu keunggulan dari suatu sistem terdistribusi adalah sifat *extensibility* yang memungkinkan sistem dapat dikembangkan secara bertahap karena bersifat terbuka. Penambahan atau pengurangan komponen di kemudian hari dapat dilakukan sehingga memberikan keleluasaan pada tahap pengembangan. Selain itu, dalam sistem terdistribusi diterapkan prinsip *resource-sharing* sehingga sistem dapat mengatasi kegagalan total saat salah satu komponen mengalami kerusakan. Karena sistem terdiri atas komponen-kompnen lebih kecil yang berpartisipasi di dalam jaringan, biaya yang dibutuhkan juga akan sangat tergantung pada skala aplikasi, serta berapa banyak perangkat yang terlibat dalam sistem tersebut. Ditambah lagi dengan adanya kemungkinan pengembangan sistem yang bersifat heterogen untuk mendapatkan berbagai keunggulan lainnya [1].

Ide seperti inilah yang kemudian dapat mendasari perkembangan teknologi robot terdistribusi saat sekarang. Beberapa tren pengembangan aplikasi sistem robot terdistribusi untuk menyelesaikan berbagai permasalahan dalam kehidupan nyata, terbukti dapat menunjukkan hasil yang cukup baik, misalnya pada penanganan kasus pencarian sumber asap untuk mengatasi permasalahan kebocoran yang sering terjadi pada industri kimia dan pertambangan [2,3], ataupun masalah pengaturan sarana transportasi dan lalu lintas dengan menggunakan sistem robot terdistribusi [4].

Pada penelitian ini, akan diuraikan penerapan aplikasi robot bergerak terdistribusi untuk masalah komunikasi. Saat sekarang, adanya media komunikasi yang dapat menyebarkan informasi secara cepat dan mudah sudah menjadi kebutuhan yang vital. Akan tetapi, kebutuhan seperti ini tentu menuntut tersedianya infrastruktur yang memadai agar dapat dilakukan proses komunikasi dan perhubungan. Dalam kehidupan nyata sering kali dijumpai masalah yang menyebabkan fasilitas komunikasi tersebut sulit untuk diadakan, sedangkan kebutuhan untuk distribusi informasi pada daerah tersebut saat itu sangat tinggi misalnya pada kawasan lokasi yang sedang mengalami bencana alam ataupun daerah yang sedang mengalami

kerusakan. Infrastruktur jaringan telekomunikasi seringkali ikut rusak saat terjadi bencana alam pada suatu daerah. Akibatnya daerah lain yang diharapkan dapat memberikan suplai bantuan ataupun evakuasi sering terlambat untuk mendatangkan bantuan [5]. Masalah serupa juga sering dialami pada lokasi terpencil yang sulit dijangkau oleh fasilitas komunikasi. Pengadaan infrastruktur jaringan komunikasi bisa jadi terhambat oleh keadaan alam yang mungkin dapat menghambat, atau bahkan membahayakan keberadaan fasilitas tersebut. Contoh lain dari permasalahan jaringan komunikasi yang terhambat karena masalah infrastruktur adalah proses penyebaran informasi pada kawasan yang sedang terlibat konflik atau peperangan. Penggunaan manusia untuk membawa suatu informasi tentu dapat membahayakan nyawa manusia tersebut.

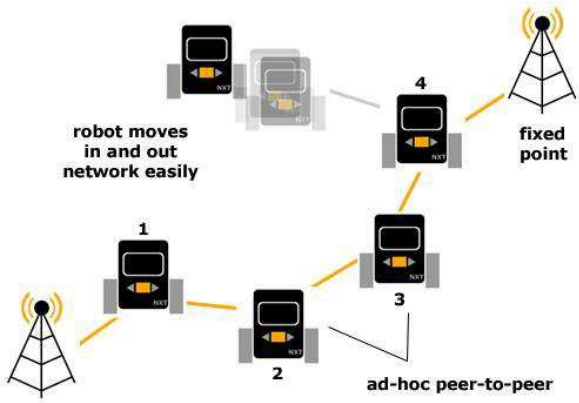
Untuk itulah pengembangan aplikasi jaringan komunikasi robot bergerak akan dapat memberikan banyak keuntungan jika diterapkan pada beberapa contoh yang disebutkan di atas. Robot sebagai agen cerdas dapat difungsikan untuk mendistribusikan informasi dari suatu tempat ke tempat yang lain. Penggunaan banyak robot dalam kelompok dimaksudkan untuk membangun suatu sistem yang terdistribusi dalam rangka mendapatkan keuntungan biaya dan waktu, skala aplikasi, dan sebagainya.

2. Metodologi

Pada penelitian ini dikembangkan implementasi jaringan robot bergerak sebagai fungsi komunikasi pada lingkungan statis dan terbatas. Pengembangan tahap ini diharapkan menjadi prototipe bagi pengembangan tahap selanjutnya hingga akhirnya akan dicapai aplikasi jaringan robot yang dapat menyelesaikan masalah komunikasi pada kehidupan di dunia nyata.

Jaringan robot bergerak ini kemudian akan kita sebut sebagai jaringan *multihop* karena memang dalam penerapannya, informasi dipindahkan dalam jaringan komunikasi dalam bentuk banyak loncatan antar robot sebelum akhirnya mencapai perangkat yang dihubungi. Untuk itu, robot di sini akan berperan sebagai *host* dalam proses komunikasi yang didukung dengan jenis teknologi *peer-to-peer*. Sementara itu, karena topologi jaringan terbentuk sebagai akibat interaksi antar robot yang saling bergerak untuk menemukan solusi konfigurasinya, robot juga berperan sebagai *router* untuk menentukan *path* yang menghubungkan lokasi atau perangkat komunikasi [6,7].

Bentuk teknologi seperti dikenal juga sebagai jenis teknologi *wireless ad-hoc* yang mendukung mobilitas pergerakan robot karena tidak harus terhubung dengan infrastruktur yang diam (kecuali satu atau beberapa robot yang terhubung langsung dengan perangkat atau lokasi yang akan komunikasi melalui hubungan yang dapat bersifat *fixed*). Hal inilah yang kemudian menyebabkan aplikasi jaringan *multihop* ini dapat dibangun pada lokasi yang tidak memadai secara infrastruktur, serta mungkin digunakan pada daerah berbahaya karena robot bergerak dapat dioperasikan secara tele-operation. Gambar 1 berikut merupakan representasi visual dari jaringan multihop ini.



Gambar 1. Jaringan Komunikasi Multihop Berbasis Robot-Bergerak

Dari ilustrasi tersebut dapat dilihat bahwa dalam proses pencarian topologi jaringan, robot dapat keluar masuk jaringan dengan mudah. Hal ini menyebabkan topologi jaringan dapat dibentuk tanpa dipersiapkan infrastrukturnya terlebih dahulu. Akan tetapi, hal ini juga mengakibatkan bahwa jaringan
rentan terhadap perubahan topologi, sehingga metode *routing* yang diterapkan harus bersifat adaptif [8].

Penelitian ini kemudian akan mengambil fokus pada masalah navigasi dan algoritma pergerakan robot dalam mencari konfigurasi atau topologi jaringan. Dalam pengembangan aplikasi yang berbasis robot sebenarnya perlu diperhatikan juga beberapa masalah lain, mulai dari bagaimana robot dapat mengambil informasi dari lingkungannya (*sensing*), bagaimana robot dapat mengekstrak informasi tersebut sehingga bermanfaat untuk pengetahuannya (*perception*), hingga bagaimana robot dapat mengetahui posisinya relatif terhadap lingkungan sehingga dapat dibuat rencana untuk mencapai tujuannya (*localization*) [9]. Karena kita tidak akan membahas ketiga masalah ini pada *paper*

ini, maka pada penelitian ini diasumsikan robot menggunakan mekanisme yang baik hingga dapat diperoleh berbagai masukan yang dibutuhkan untuk keperluan perencanaan navigasi. Dalam perkembangan teknologi robotika sendiri sebenarnya sudah sering dilakukan juga berbagai penelitian lain untuk menyelesaikan masalah *sensing*, *perception* ataupun *localization* ini [10,11]. Dengan demikian, pada penelitian ini melalui suatu metode *localization* tertentu, misalnya dengan menggunakan perangkat GPS, robot dapat menentukan posisinya relatif terhadap suatu titik koordinat (0, 0). Selain itu, karena diharapkan untuk menghubungkan suatu posisi yang sudah ditentukan, maka robot juga diberikan mengenai informasi lokasi yang saling berkomunikasi tersebut.

2.1 Keadaan Simulasi

Penelitian ini berfokus pada perencanaan pergerakan robot dalam kelompok atau dikenal dengan masalah *navigation planning*. Untuk dapat menyelesaikan permasalahan secara terstruktur, penelitian dibagi atas beberapa tahap dengan membagi kondisi pengembangan sebagai berikut.

a. Lingkungan yang bersifat ideal: statis dan terbatas. Pada simulasi hanya ada objek-objek seperti: perangkat atau lokasi yang dihubungkan oleh jaringan komunikasi (untuk selanjutnya disebut sebagai target), serta robot-robot otonom bergerak yang akan menemukan topologi jaringan dan mendistribusikan informasi.
b. Lingkungan yang bersifat semi-ideal: dinamis dan terbatas. Pada simulasi selain target dan robot, juga ada beberapa komponen lain seperti: rintangan, daerah terlarang yang tidak dicapai robot dan sebagainya. Pada kasus seperti ini akan dilihat pengaruh halangan tersebut terhadap performa algoritma yang dikembangkan.
c. Lingkungan yang bersifat tidak ideal: dinamis dan tidak terbatas. Pada simulasi tahap ini, keadaan lingkungan semakin dikondisikan mendekati kehidupan sebenarnya. Berbagai rintangan mungkin bisa bergerak, lantai ruangan yang tidak rata ataupun kondisi lingkungan yang dapat berubah sewaktu-waktu. Hasil yang diperoleh pada tahap ini sudah dapat diujikan pada implementasi di dunia nyata.

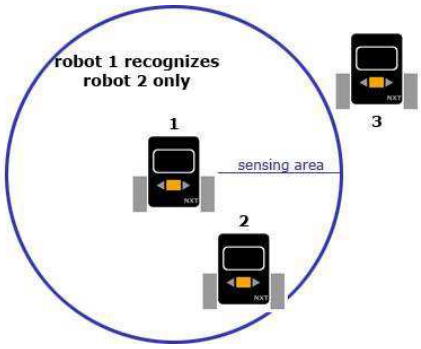
Kemudian, selain asumsi mengenai lingkungan yang dijelaskan tersebut perlu juga diperhatikan masalah sumber daya untuk membangun jaringan komunikasi-robot multihop ini. Jumlah robot yang digunakan tentu tidak terbatas, begitu juga waktu untuk menemukan topologi jaringan, sehingga solusi yang ditemukan hendaknya bersifat optimum. Paper ini akan mencoba menjabarkan hasil penelitian yang diperoleh pada penerapan yang dibatasi sesuai dengan poin nomor a, yaitu pada lingkungan yang ideal: statis dan terbatas.

2.2 Permasalahan dengan Dua Sumber / Pair-source

Permasalahan dengan dua sumber atau *Pair-source*, merupakan kondisi ketika terdapat dua target di dalam lingkungan simulasi yang dihubungkan oleh jaringan komunikasi-robot multihop. Sebelum menjelaskan algoritma yang digunakan untuk *navigation planning*, akan didefinisikan spesifikasi robot yang digunakan pada simulasi ini.

Robot yang digunakan bersifat otonom, yang berarti masing-masing robot di dalam simulasi memproses informasi yang diterimanya sendiri. Robot kemudian dapat memanfaatkan informasi tersebut dalam rangka mencapai tujuannya masing-masing. Untuk dapat menyelesaikan suatu tujuan, robot otonom ini memiliki beberapa modul sebagai berikut:

- Modul *sensing*, yaitu fungsi pada robot untuk memperoleh informasi dari lingkungan di sekitarnya. Dengan modul ini robot dapat mengenali objek lain di sekelilingnya selama dalam rentang s*ensing area* robot tersebut. Gambar 2 di bawah menunjukkan fungsi dari modul *sensing* ini.



Gambar 2. Robot hanya dapat mengenali objek lain selama berada dalam rentang area sensing-nya

- Modul *computation*, yaitu fungsi pada robot untuk mengolah dan memanipulasi informasi yang diperoleh dari lingkungan dalam rangka mencapai tujuan yang ditentukan, misalnya

kemampuan robot untuk menghitung jarak dari objek lain untuk mencegah terjadinya benturan.

- Modul *communication*, yaitu fungsi pada robot untuk saling bertukar informasi dan memindahkan data. Fungsi inilah yang mendasari adanya jaringan komunikasi-robot multihop. Serupa dengan modul *sensing*, fungsi komunikasi ini hanya dapat dilakukan dalam rentang communication area tertentu saja. Untuk itu jarak ini harus menjadi jarak maksimum antar robot hingga dapat terbentuk suatu *link* komunikasi.
- Modul *locomotion*, yaitu fungsi pada robot untuk bergerak sesuai dengan kemampuan motor yang dimilikinya sehingga dapat mencapai tujuan yang ditentukan sebelumnya.

  Untuk dapat menentukan arah pergerakan, robot memanfaatkan informasi posisi awal dan posisi tujuannya untuk setiap iterasi. Dengan menyimpan kedua informasi tersebut, robot bergerak dengan menggunakan vektor pergerakan sebagai berikut.

  $$\mathbf{P_1' = P_1 + \Delta P} \quad \mathbf{(1)}$$

  $$\mathbf{P_1' = (x_1 + \Delta x, y_1 + \Delta y, z_1 + \Delta z)}$$

  Di mana P_1' adalah posisi tujuan setelah iterasi ke-1, P_1 adalah posisi awal pada iterasi ke-1, dan ΔP adalah perpindahan robot pada setiap iterasinya. Kemudian dengan mengasumsikan simulasi akan dilakukan pada ruangan yang berlantai rata, maka nilai pada koordinat z dapat dianggap selalu 0. Dengan demikian, pergerakan robot menggunakan ruang berdimensi dua dengan persamaan sebagai berikut.

  $$\mathbf{P_1' = (x_1 + \Delta d * \cos \theta, y_1 + \Delta d * \sin \theta)} \quad \mathbf{(2)}$$

  Di mana θ adalah sudut yang dibentuk oleh posisi robot terhadap titik koordinat (0, 0), dan Δd adalah jarak dapat ditempuh robot dalam satu iterasi atau satuan waktu. Selanjutnya robot akan berhenti saat jarak antara posisi awal dan posisi tujuannya memenuhi persamaan:

  $$\mathbf{| P_1' - P_1 | <= Threshold} \quad \mathbf{(3)}$$

Selanjutnya, untuk dapat menemukan topologi jaringan komunikasi-robot multihop, robot harus menemukan posisi-posisi tujuan (*feasible positions*) sehingga nantinya semua robot dapat saling meneruskan informasi dalam bentuk 'banyak lompatan'. Untuk memudahkan penamaan, untuk selanjutnya kita akan menyebut salah satu target sebagai *source*, yaitu lokasi atau perangkat yang akan mengirimkan informasi kepada target lainnya (cukup disebut sebagai target). Ide dasar untuk menentukan semua *feasible positions* ini kemudian dilakukan dengan cara menentukan lintasan yang menghubungkan *source* dengan target terlebih dahulu, kemudian ditentukan titik-titik yang mungkin ditempati robot sepanjang lintasan tersebut. Pada saat semua robot menempati setiap titik tersebut, itu berarti solusi konfigurasi jaringan komunikasi-robot multihop pun berhasil ditemukan.

Pada kasus *pairform-source* dalam lingkungan ideal yang statis dan terbatas, solusi ini kemudian dapat ditentukan dengan memanfaatkan informasi posisi *source* dan *target*, serta informasi *communication range* masing-masing robot. Solusi paling optimum untuk permasalahan ini adalah lintasan terdekat yang dapat ditempuh oleh robot yang menghubungkan posisi *source* dan *target*. Karena di dalam ruangan simulasi diasumsikan memiliki lantai rata, dan hanya terdapat *source*, target dan robot, maka solusi paling optimum dapat ditemukan dengan menemukan jarak langsung antara *source* dan target yang dapat dihitung dengan persamaan sebagai berikut.

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \mathbf{(4)}$$

dengan D adalah jarak langsung antara *source* pada posisi (x_1, y_1) dan target pada posisi (x_2, y_2).

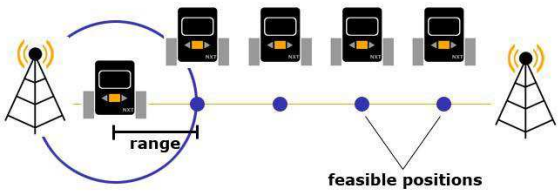
Selanjutnya semua titik *feasible positions* dapat ditentukan dengan cara menemukan semua titik yang berjarak sama dengan jarak *communication range* masing-masing robot di dalam simulasi, dengan *pseudocode* sebagai berikut.

For i=0 until i=(D/r)

$$\mathbf{P_i = (i * r * \cos \theta + x_1, i * r * \sin \theta + y_1)} \quad \mathbf{(5)}$$

End loop

Selanjutnya, robot diminta untuk menentukan salah satu *feasible positions* sebagai posisi tujuannya. Gambar 3 berikut memberikan ilustrasi mengenai cara menentukan *feasible positions*.



Gambar 3. *Feasible positions* dapat ditentukan dengan cara membagi total jarak dengan rentang area *communication* masing-masing robot

Setelah menemukan semua posisi tersebut, kemudian robot akan menentukan jarak dari semua *feasible positions* dan bergerak menuju salah satu *feasible position* terdekat. Robot baru berhenti ketika berhasil menempati posisi tujuan tersebut, atau jika tidak berhasil meraihnya karena sudah ditempati robot lain, maka robot akan memilih *feasible position* terdekat berikutnya sampai tidak ada lagi posisi yang dapat didekati.

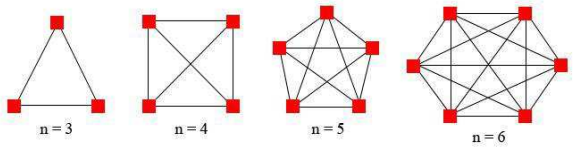
Sebagai langkah terakhir untuk menentukan apakah solusi konfigurasi jaringan sudah ditemukan atau tidak, masing-masing robot kemudian melakukan pengecekan koneksi. Artinya robot akan mengecek apakah objek yang terhubung dengannya merupakan posisi *source* atau target. Jika robot tersebut terhubung dengan *source*, robot akan menyimpan nilai *flag* bahwa dirinya telah terhubung terhadap *source* dan melanjutkan pengecekan koneksi terhadap target. Jika ternyata robot tersebut terhubung dengan robot lain, maka melalui *modul communication*, robot akan meminta nilai *flag* koneksi robot tetangganya tersebut. Jika *flag* robot tetangga bernilai benar, maka robot juga akan mengubah nilai *flag*-nya sendiri. Begitu seterusnya, model pengecekan seperti ini dapat dilakukan dengan menggunakan algoritma *searching Depth First Search* untuk mengecek koneksi antar robot [12]. Dengan demikian, pada saat semua robot telah mengeset nilai *flag* koneksinya, telah terdapat suatu konfigurasi jaringan komunikasi-robot multihop yang menghubungkan *source* dan target.

2.3 Permasalahan dengan Banyak Sumber/Multi-source

Setelah berhasil menyelesaikan permasalahan *Pair-source* di lingkungan statis dan terbatas, ruang lingkup permasalahan pada penelitian ini kemudian diperluas untuk menangani kasus *multi-source* atau banyak perangkat atau lokasi yang akan dihubungkan oleh jaringan komunikasi-robot multihop ini. Jika pada permasalahan *pair-source*, solusi dapat langsung ditemukan dengan cara menentukan lintasan terdekat antara *source* dan target. Pada permasalahan *multi-source* kita harus terlebih dahulu menentukan berbagai kombinasi lintasan yang mungkin menghubungkan semua target dalam jaringan yang sama, kemudian baru ditentukan *feasible positions* dari kombinasi lintasan yang paling optimum.

Dalam menentukan kombinasi lintasan ini, cara yang dapat ditempuh adalah dengan merepresentasikan semua target dan lintasan dalam bentuk *graph* terhubung, di mana target digambarkan sebagai *vertex* dan kombinasi lintasannya sebagai *edge*. Karena semua target bisa saja terhubung satu sama lain, maka konfigurasi *graph* terhubung yang kita miliki akan menjadi suatu bentuk *complete graph* pada saat ada koneksi langsung antar semua target di dalam jaringan komunikasi multihop ini [13].

Pada Gambar 4 berikut akan dijelaskan visualisasi konfigurasi *graph* yang terbentuk untuk jumlah target=3 hingga target=6 dimana terdapat koneksi langsung antar masing-masing target.



Gambar 4. Representasi *graph* dari permasalahan dengan banyak sumber pada jumlah target yang berbeda-beda

Selanjutnya, perlu diperhatikan bahwa kondisi di mana terdapat koneksi langsung antar masing-masing target tidak selamanya dapat dicapai, disebabkan adanya keterbatasan jumlah robot yang digunakan. Jika hal ini terjadi, perlu dilakukan pengecekan kombinasi lintasan untuk mengeliminasi lintasan-lintasan yang tidak perlu, yaitu lintasan yang secara berulang menghubungkan target-target dalam jarak yang tidak optimal, sehingga dengan jumlah robot tertentu masih terdapat kemungkinan untuk menghubungkan semua target, walaupun melalui bentuk koneksi tidak langsung (dua target terhubung melalui koneksi dengan target lain sebagai perantara).

Dari penjelasan di atas, dapat ditarik kesimpulan bahwa kombinasi lintasan solusi di sini merupakan *spanning tree* dari konfigurasi *graph* tersebut. Saat jumlah robot minimum, berarti solusinya akan menjadi konfigurasi dengan jumlah *edge* minimum yang dapat membentuk *connected graph*, dalam hal ini berarti konfigurasi *Minimum Spanning Tree* (MST) dari *graph* tersebut.

Pada saat jumlah robot lebih banyak dari yang dibutuhkan untuk membentuk konfigurasi MST, diperlukan suatu metode untuk menentukan kombinasi lintasan yang paling optimum. Untuk keperluan ini, kemudian didefinisikan dua jenis variabel yang akan digunakan untuk menemukan kemungkinan kombinasi lintasan yang paling optimum ini, yaitu nilai *Cost* dan *Distance*.

Cost adalah nilai satuan yang diperlukan untuk mendapatkan semua konfigurasi lintasan atau *edge* di dalam *graph* terhubung. Untuk memahami

pengertian ini dapat dilihat pada contoh berikut. Jika dalam suatu *graph* terdapat tiga *vertex* A, B dan C di mana hanya terdapat koneksi langsung antara A-B dan B-C masing-masingnya 3 satuan dan 4 satuan; maka besar Cost yang diperlukan konfigurasi *graph* tersebut adalah 3+4 = 7 satuan. *Distance* adalah masing-masing jarak satuan yang terjadi antara setiap *vertex* pada suatu kombinasi lintasan atau *edge* tertentu di dalam *graph* terhubung. Pada contoh yang sebelumnya, nilai *Distance* akan menjadi sebagai berikut. A-B bernilai 3 satuan, B-C bernilai 4 satuan, serta A-C ditentukan dengan menjumlahkan nilai jarak A-B dengan B-C, yaitu 3+4=7 satuan; maka besar *Distance* pada konfigurasi *graph* tersebut adalah 3+4+7=14. Hal yang berbeda akan diperoleh pada saat terdapat koneksi langsung antara A-C dengan nilai 5 satuan. Maka, nilai *Cost* akan bertambah besar menjadi 3+4+5 = 12 satuan, sedangkan nilai *Distance* akan bertambah kecil menjadi 3+4+5 = 12 satuan.

Dari penjelasan tersebut, dapat diperhatikan bahwa nilai *Cost* yang dimaksud di sini yaitu jumlah robot yang diperlukan untuk membentuk suatu konfigurasi jaringan komunikasi multihop pada jumlah target tertentu; sedangkan nilai *Distance* yang dimaksud adalah jarak tempuh antara masing-masing target. Karena besarnya nilai *Cost* dan *Distance* ini saling berbanding terbalik satu sama lain, proses pencarian solusi konfigurasi jaringan komunikasi multihop dengan kasus banyak sumber ini sebenarnya dapat dilihat sebagai penyelesaian masalah untuk meminimalkan nilai *Distance* pada saat *Cost* yang dimiliki juga bernilai minimal (karena adanya keterbatasan jumlah robot yang mungkin disediakan).

Adapun nilai *Cost* paling minimum yang diperlukan agar semua target tetap terhubung dalam konfigurasi jaringan yaitu jumlah satuan robot yang diperlukan untuk mendapatkan solusi MST dari konfigurasi *graph* tersebut. Sementara itu, nilai *distance* paling minimum justru tercapai pada saat terdapat lintasan langsung antara masing-masing target, yaitu jumlah satuan robot yang diperlukan untuk mendapatkan solusi *complete* dari konfigurasi *graph* tersebut. Dengan demikian, kemungkinan kombinasi lintasan solusi yang dapat diambil akan mengikuti persamaan berikut.

$$\mathbf{MST \leq NR \leq CG} \qquad \mathbf{(6)}$$

dengan MST adalah jumlah robot yang diperlukan untuk membentuk konfigurasi *Minimum Spanning Tree* dari graph, NR adalah jumlah robot yang tersedia di dalam simulasi, dan CG adalah jumlah robot yang diperlukan untuk membentuk konfigurasi *Complete Graph* pada jumlah target tertentu. Solusi konfigurasi jaringan dianggap tidak ada pada saat jumlah robot yang tersedia lebih sedikit dari jumlah satuan yang diperlukan untuk membentuk konfigurasi MST, sehingga simulasi tidak perlu dilanjutkan lebih jauh. Jika jumlah robot yang tersedia melebihi jumlah satuan yang diperlukan untuk membentuk konfigurasi *complete graph*, maka solusi yang diambil adalah konfigurasi *complete graph* tersebut.

Permasalahan optimisasi nilai *Cost* dan *Distance* ini hanya perlu dilakukan pada saat jumlah robot yang tersedia berada di antara kedua nilai satuan yang diperlukan untuk mendapatkan solusi MST serta *complete graph* tersebut.

Selanjutnya untuk dapat menentukan konfigurasi jaringan yang diharapkan dilakukan dengan cara sebagai berikut.

1. Ambil setiap kemungkinan kombinasi konfigurasi jaringan yang mungkin terbentuk untuk nilai *Cost* yang ditentukan / jumlah robot yang tersedia.
2. Periksa nilai *Distance* dari semua kemungkinan kombinasi konfigurasi jaringan yang diambil pada langkah sebelumnya. Pilih kombinasi yang memiliki nilai *Distance* paling minimum.
3. Kombinasi yang kemudian terpilih merupakan solusi dari konfigurasi jaringan multihop yang sedang dicari.

3. Implementasi

Pada penelitian ini akan dilakukan pengembangan hingga tahap simulasi dan implementasi sederhana pada perangkat keras. Untuk keperluan simulasi akan dibangun simulator untuk verifikasi algoritma sebelum akhirnya diimplementasikan pada perangkat kerasnya. Simulator dikembangkan dengan menggunakan Java (J2SE) dan OpenGL untuk visualisasi animasi. Robot yang digunakan adalah robot Mindstorm NXT dari Lego. Alasan penggunaan perangkat ini adalah karena robot lego memiliki fitur yang dibutuhkan untuk modul-modul yang sudah disebutkan, mudah untuk diterapkan, serta biaya pengembangannya relatif murah.

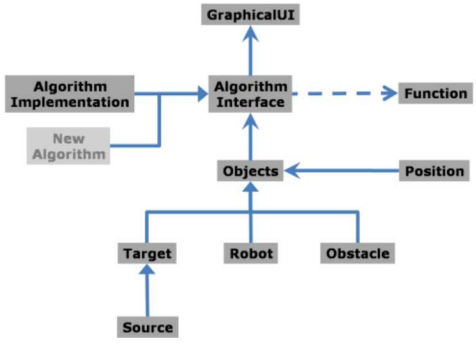
Pengujian algoritma dan metodologi penyelesaian masalah yang sudah dikembangkan akan diverifikasi melalui dua cara, yaitu penerapan pada simulasi yang berbasis perangkat lunak serta pengujian pada simulasi yang berbasis perangkat keras. Adapun penerapan dengan berbasis perangkat lunak dipilih sebagai langkah yang pertama karena lebih mudah untuk dilakukan, serta tidak terkait

dengan kendala fisik yang sering ditemui pada keadaan nyata, misalnya kondisi lingkungan tempat dilakukannya simulasi. Selain itu, keterbatasan perangkat keras juga dapat diatur dengan lebih fleksibel pada level perangkat lunak sehingga kemampuan algoritma untuk menyelesaikan permasalahan dapat diuji pada berbagai *setting* simulasi, pada jumlah robot dan spesifikasi yang berbeda-beda pula.

Adapun kemudian setelah diperoleh hasil yang menunjukkan kinerja dari algoritma yang sedang dikembangkan, pengujian seterusnya dapat dilanjutkan pada penerapan yang berbasis perangkat keras. Hal ini bertujuan untuk menguji apakah evaluasi algoritma benar-benar dapat dijalankan pada kehidupan nyata. Pada tahapan ini, dalam rangka menghemat waktu pengembangan dan biaya pengerjaan, simulasi dibatasi dengan lingkungan yang tidak terlalu luas dan jumlah robot dan target yang juga tidak terlalu banyak.

3.1 Penerapan Berbasis Perangkat Lunak

Perangkat lunak yang dirancang dikembangkan sedemikian rupa agar nantinya dapat digunakan kembali jika akan dilakukan pengembangan algoritma yang baru. Oleh karena itu, dibuat pemodelan *class diagram* yang memudahkan kebutuhan ini, yaitu seperti pada Gambar 5 di bawah. Perangkat lunak dikembangkan dengan berbasis animasi, sehingga proses verifikasi dapat dilakukan tidak hanya melalui data statistik yang diperoleh, tapi juga dapat diuji secara visual untuk melihat pergerakan robot dalam menemukan konfigurasi jaringan sebagai efek dari algoritma yang dikembangkan. Gambar 6 adalah contoh tampilan perangkat lunak yang dikembangkan dengan visualisasi kondisi robot saat menemukan konfigurasi jaringan komunikasi *multihop* yang sedang dicari.

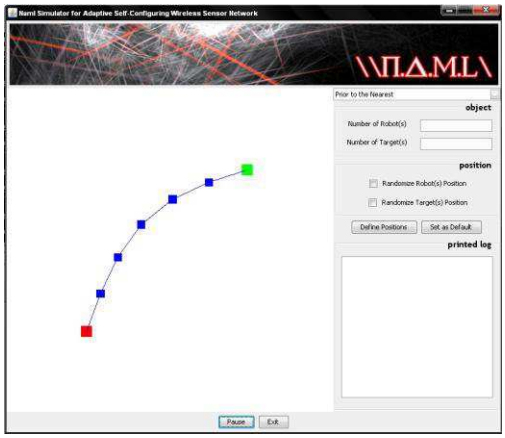


Gambar 5. Hubungan antar class pada implementasi simulator

3.2 Penerapan Berbasis Perangkat Keras

Setelah melakukan implementasi di tahap perangkat lunak (dan dilakukan verifikasi algoritma seperti yang akan dijelaskan pada bagian selanjutnya), tahap pengembangan dilanjutkan kepada implementasi perangkat keras. Pada tahap ini, pengujian dan simulasi dilakukan dengan menggunakan komponen-komponen sebagai berikut:

- 3 (tiga) unit robot yang dilengkapi modul komunikasi. Robot yang digunakan adalah robot Lego Mindstorm NXT.
- 3 (unit) web-kamera sebagai alat untuk tracking posisi robot atau pengganti fungsi GPS. Kamera yang digunakan adalah Logitech Quickcam E3500.
- 1 (unit) komputer sebagai simulasi server atau host dari sistem GPS.
- 1 (satu) unit lapangan yang menjadi lingkungan pergerakan robot.



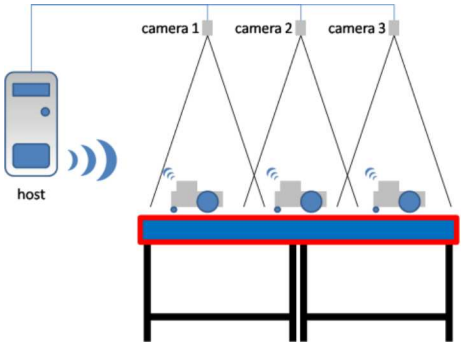
Gambar 6. Tampilan pengujian berbasis perangkat lunak dengan visualisasi solusi konfigurasi jaringan

Desain dari implementasi perangkat keras ini mengikuti Gambar 7 sebagai berikut. Dari gambar tersebut dapat dilihat bahwa metode *localization* yang sebelumnya diasumsikan dengan menggunakan sistem GPS, sekarang digantikan oleh fungsi web-kamera yang terhubung dengan sistem *computer*/PC sebagai modul komputasinya. Robot dan PC kemudian akan berkomunikasi dengan menggunakan Bluetooth.

Penggunaan sistem kamera untuk pemetaan posisi robot pada lapangan simulasi ini merupakan salah satu perbedaan terbesar antara penerapan berbasis perangkat lunak dengan penerapan berbasis perangkat keras. Oleh karena itu, pada penerapan berbasis perangkat keras ini akan dibagi menjadi dua

tahapan implementasi yaitu: pengembangan sistem pemetaan posisi atau koordinat robot dengan menggunakan kamera untuk *covering* lapangan simulasi, serta pengembangan sistem pengaturan pergerakan robot.

3.2.1 Pemetaan Posisi Robot dengan Menggunakan Sistem Covering Kamera



Gambar 7. Rancangan arsitektural implementasi berbasis perangkat keras

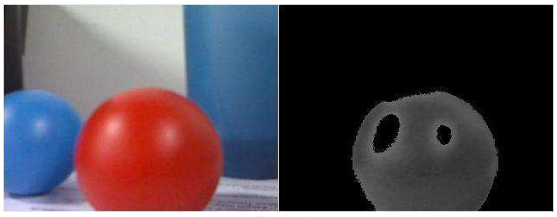
Dengan adanya keterbatasan perangkat keras yang tersedia untuk menerapkan hasil penelitian yang dilakukan, sistem kamera yang dibangun akan membutuhkan bantuan suatu komponen PC sebagai modul untuk melakukan proses komputasi. Jadi, dalam rangka mendapatkan informasi posisinya masing-masing, robot nantinya dapat melakukan *query* kepada komponen PC tersebut melalui bentuk komunikasi yang ditentukan. Akan tetapi perlu diperhatikan karena penggunaan PC di sini hanya dimaksudkan untuk keperluan *localization* bagi masing-masing robot, setiap robot tidak dapat mengakses posisi robot lain secara bebas walaupun hal tersebut mungkin saja dilakukan. Kemampuan robot untuk mengenali robot lain dalam rentang area *communication* dan area *sensing* tertentu tetap akan mengikuti aturan dan asumsi yang telah ditentukan sebelumnya.

Selanjutnya karena keterbatasan rentang yang dapat dijangkau oleh satu unit kamera, untuk dapat melakukan *covering* terhadap seluruh lapangan simulasi, digunakan 3 (tiga) unit kamera dengan spesifikasi sama untuk menggantikan fungsi GPS ini. Ketiga unit kamera ini kemudian secara bersama-sama akan terhubung pada 1 (satu) perangkat lunak yang dikembangkan untuk memetakan posisi robot dalam suatu sistem koordinat yang bersifat global.

Kemudian, dalam pengembangan perangkat lunak yang akan memetakan posisi robot ini, tentunya sistem kamera harus dapat menentukan objek-objek mana saja yang akan dikenali sebagai robot, serta objek-objek mana yang akan dikenali sebagai bukan robot. Hal ini dapat dilakukan dengan memberikan suatu karakteristik tertentu pada robot yang bersifat khas, yang dapat membuat robot dibedakan dari setiap objek lainnya. Pemberian karakteristik ini dilakukan melalui pembedaan warna pada setiap robot. Hal ini disebabkan karena pengenalan objek berdasarkan perbedaan warna yang dimilikinya merupakan suatu metode yang paling mudah untuk dilakukan, sehingga diharapkan tidak akan menemui banyak kesulitan pada pengembangan sistem kamera ini.

Karena pada penerapan berbasis perangkat keras ini akan digunakan 3 (tiga) unit robot di dalam lapangan simulasi, maka dipilih warna merah, biru dan hijau sebagai warna yang dipasangkan untuk setiap robot karena ketiga warna ini memiliki perbedaan kontras yang besar. Diasumsikan bahwa pada lapangan simulasi tidak terdapat objek lain yang memiliki warna serupa, sehingga proses pengenalan robot dapat dilakukan dengan baik.

Selanjutnya, setelah dilakukan pembedaan warna pada masing-masing robot, proses pengenalan robot dapat dilakukan dengan menerapkan *filtering* warna pada sistem kamera yang digunakan. Dalam hal ini, proses *filtering* hanya akan mengenali objek dengan warna merah, biru ataupun hijau; warna selain itu akan dipetakan menjadi hitam yang berarti tidak ada objek. Adapun proses *filtering* warna ini dapat dilihat seperti pada Gambar 8 berikut. Pada contoh tersebut, *filtering* hanya dilakukan pada objek yang berwarna merah.



Gambar 8. Proses filtering warna merah pada objek

Adapun proses pengenalan objek dengan menggunakan sistem kamera ini, mengikuti alur sebagai berikut.

1. Mengambil gambar/*frame* dari kamera.

   Gambar/*frame* akan diambil dari masing-masing kamera setiap 250 milidetik atau empat kali dalam satu detiknya. Gambar tersebut kemudian akan diteruskan kepada komponen PC untuk diproses pada langkah selanjutnya.
2. Mendeteksi objek pada gambar/*frame*.

Setelah gambar/*frame* diteruskan kepada PC, akan dilakukan proses pengenalan objek dengan menggunakan *filtering* warna dan algoritma *blob detection*. *Filtering* warna dilakukan dengan cara melakukan penyaringan pada setiap *pixel* gambar / *frame*. Apabila suatu *pixel* berada di luar rentang yang diinginkan maka *pixel* tersebut akan diubah menjadi warna hitam. Algoritma *blob detection* adalah metode pendeteksian suatu *pixel* yang berwarna bukan hitam kemudian akan dilakukan pengecekan terhadap *pixel* tetangga di sebelahnya. Setiap *pixel* yang memiliki tetangga yang juga *pixel* bukan hitam akan dimasukkan sebagai calon *blob*. Dari calon *blob* ini kemudian akan diseleksi sesuai ukuran yang diinginkan. *Pixel-pixel* yang kemudian lolos proses penyeleksian akan menjadi *blob* yang diteruskan pada langkah selanjutnya.

3. Menentukan koordinat objek pada gambar/*frame*.

   Selanjutnya dari *blob* yang diperoleh pada langkah sebelumnya dapat ditentukan suatu titik yang akan menjadi nilai koordinat dari suatu objek dalam gambar/*frame* tersebut. Hal ini dilakukan dengan cara menemukan titik tengah dari *blob* yang ditemukan sebagai posisi koordinat yang dicari.

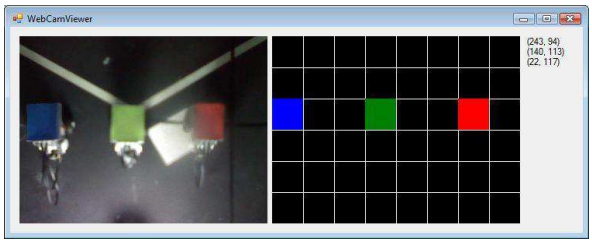
4. Menentukan posisi pada sistem koordinat global.

   Langkah terakhir yang diperlukan sebelum posisi objek dapat dipetakan dalam suatu sistem koordinat yang bersifat global adalah penentuan sistem koordinat itu sendiri. Walaupun dari langkah sebelumnya sudah diperoleh suatu angka yang dapat merepresentasikan posisi suatu objek dalam gambar/frame, nilai posisi tersebut masih berupa angka dengan rentang yang sangat besar dan tidak terbatas. Oleh karena itu, nilai-nilai posisi ini akan lebih mudah untuk diproses jika direpresentasikan dalam bentuk grid atau kotak-kotak pada sistem koordinat global, sehingga bersifat lebih diskrit dan mudah untuk dipresentasikan secara visual. Hasil akhir dari proses pemetaan dengan menggunakan sistem kamera (dengan menggunakan satu kamera) ini kemudian dapat dilihat pada Gambar 9 berikut. Tampilan sebelah kiri merupakan gambar/*frame* yang ditangkap oleh kamera sedangkan tampilan sebelah kanan merupakan posisi robot yang sudah dipetakan pada sistem koordinat global.

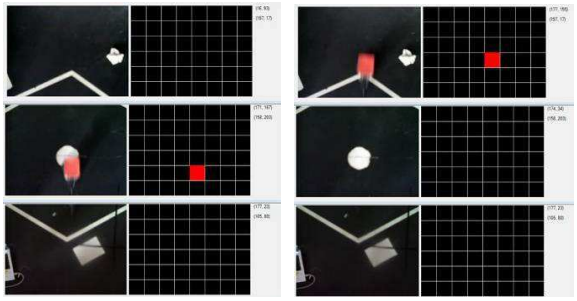
Selanjutnya, contoh pemetaan posisi robot dengan menggunakan sistem tiga unit kamera pada saat robot bergerak pun dapat dilihat pada Gambar 10 berikut.

3.2.2 Pengaturan Pergerakan Robot

Setelah posisi masing-masing robot berhasil dipetakan dalam suatu sistem koordinat yang bersifat global, langkah selanjutnya adalah pengembangan sistem untuk mengatur pergerakan robot dalam rangka menemukan konfigurasi jaringan komunikasi-robot multihop terdistribusi. Adapun penerapan pada tahap ini tidak jauh berbeda dengan penerapan berbasis perangkat lunak, hanya saja eksekusi perintah langsung dilakukan terhadap perangkat keras robot Lego Mindstorm NXT yang digunakan pada penelitian. Kemudian, karena perangkat robot Lego Mindstorm NXT tersebut hanya mendukung beberapa fungsi komputasi yang terbatas, keberadaan komponen PC juga dapat dimanfaatkan sebagai modul komputasi, layaknya bantuan yang juga dimanfaatkan pada sistem kamera sebelumnya.



Gambar 9. Contoh tampilan pengenalan 3 robot yang berbeda dengan menggunakan sistem kamera



Gambar 10. Contoh tampilan sistem 3 unit kamera secara bersama-sama menangkap pergerakan robot

Gambar 11 berikut di bawah merupakan contoh *setting* yang ditemukan pada saat dilakukan pengujian berbasis perangkat keras pada lapangan dan perangkat keras yang telah disebutkan spesifikasinya sebelumnya.

Gambar 11. Contoh tampilan *setting* lapangan pada saat simulasi berlangsung

4. HASIL EKSPERIMEN

Berikut akan ditampilkan hasil yang diperoleh melalui kegiatan uji coba serta simulasi perangkat lunak dan perangkat keras pada penelitian ini. Tabel 1-3 di bawah ini menunjukan hasil pengujian pada simulasi dengan memanfaatkan parameter jumlah robot, rentang *sensing area*, dan *communication area* (*range*) yang berbeda.

Tabel 1. Waktu simulasi dengan jumlah robot=5

Radius Sensing	Range	Time (in ms)				
		Ave	Min	Max	Std Dev	Range
0.25	0.0005	9045.73	6047	14438	2426	8392
	0.001	4776.65	2360	8469	1462.8	6110
0.5	0.0005	3196.15	562	8234	1673.32	7673
	0.001	1939.8	422	3578	775.1	3157

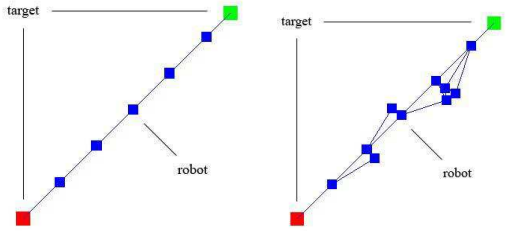
Tabel 2. Waktu simulasi dengan jumlah robot=7

Radius Sensing	Range	Time (in ms)				
		Ave	Min	Max	Std Dev	Range
0.25	0.0005	6165.65	3844	9812	1514.28	5969
	0.001	3456.95	1641	6969	1193.04	5329
0.5	0.0005	3261.6	672	6969	1389.2	6298
	0.001	1449.35	547	2531	552.63	1985

Tabel 3. Waktu simulasi dengan jumlah robot=10

Radius Sensing	Range	Waktu (dalam ms)				
		Ave	Min	Max	Std Dev	Range
0.25	0.0005	4903.1	2969	8266	1199.39	5298
	0.001	2762.55	1282	6234	1087.06	4953
0.5	0.0005	2320.2	485	4015	1017.21	3531
	0.001	1437.4	828	2532	463.35	1705

Gambar 12 berikut ini menunjukkan kondisi simulasi pada saat berhasil menemukan solusi jaringan komunikasi-robot multihop dengan jumlah robot yang berbeda. Pada Gambar 12.a dapat dilihat bahwa solusi konfigurasi jaringan multihop ditemukan melalui hubungan langsung antara kedua target. Oleh karena itu, pada saat jumlah robot lebih banyak, solusi yang paling optimum tetap saja merupakan hubungan langsung antara kedua target tersebut.



(a) **(b)**

Gambar 12. Contoh solusi jaringan komunikasi multihop berbasis robot bergerak dengan jumlah penggunaan robot yang berbeda

5. KESIMPULAN

Penelitian dengan topik jaringan komunikasi-robot multihop terdistribusi pada lingkungan statis dan terbatas ini telah berhasil dilaksanakan. Adapun hasil yang diperoleh berupa algoritma dan metodologi untuk menyelesaikan permasalahan pada kasus dengan dua lokasi atau target yang dihubungkan oleh jaringan komunikasi atau disebut juga dengan istilah permasalahan dengan dua sumber atau *Pair-source*.

Selain itu juga dibahas metode penyelesaian pada kasus dengan lebih dari dua lokasi atau target yang dihubungkan oleh jaringan komunikasi atau kemudian dikenal dengan sebutan permasalahan dengan banyak sumber atau *multi-source*. Pada kedua permasalahan ini diharapkan solusi yang ditemukan dapat bersifat optimal, artinya robot dapat menemukan solusi konfigurasi dengan jarak terpendek antar target, pada jumlah robot tertentu yang terbatas. Pada permasalahan *pair-source* hal ini mudah untuk dilakukan karena solusi konfigurasi jaringan yang diperlukan adalah jarak langsung antara kedua target. Sedangkan pada permasalahan *multi-source* menjadi sedikit lebih sulit karena harus mempertimbangkan adanya *trade-off* antara nilai *cost* dan *distance* untuk membentuk suatu konfigurasi jaringan komunikasi.

Proses pengujian dan verifikasi algoritma kemudian dilakukan dalam dua tahap, yaitu tahapan

penerapan berbasis perangkat lunak, serta penerapan berbasis perangkat keras. Dari tahapan pertama, kinerja algoritma dinilai pada beberapa *setting* simulasi dengan menggunakan parameter jumlah robot yang berbeda, berikut dengan rentang *sensing* dan *communication area* robot yang berbeda-beda pula. Pada tahapan kedua, pengujian lebih dimaksudkan untuk melihat visibilitas metode untuk diterapkan pada kehidupan nyata. Pengujian dilakukan dengan menggunakan permasalahan *pair-source*, dengan menggunakan tiga unit robot serta sistem kamera untuk menangani masalah *localization*.

Selanjutnya, perlu diperhatikan bahwa pada penelitian kali ini masih terdapat berbagai batasan dan asumsi yang digunakan, di antaranya adalah batasan bahwa lingkungan yang digunakan bersifat ideal, statis dan terbatas. Dalam kehidupan di dunia yang sebenarnya lingkungan yang seperti ini hampir tidak pernah benar-benar dapat dijumpai. Oleh karena itu, diharapkan pada penelitian selanjutnya dapat dilakukan dengan memperhitungkan faktor keadaan topologi lingkungan, keberadaan objek atau daerah-daerah yang tidak dapat dijangkau oleh robot, ataupun kemungkinan robot mengalami kerusakan, target berpindah tempat, dan sebagainya. Dengan demikian diharapkan aplikasi ini dapat diterapkan untuk menyelesaikan permasalahan nyata yang sebenarnya.

REFERENSI

[1] Sugiharto, Agung, *Slide Bahan Kuliah Sistem Terdistribusi*, Yogyakarta: STMIK El Rahma, 2003.

[2] Jatmiko, Wisnu et al, "Evolutionary computation approach for odor source localization problem: practical review", *IEEE Transaction on Evolutionary Computation*, 2009.

[3] Jatmiko, Wisnu, Kosuke Sekiyama and Toshio Fukuda, "A PSO-based mobile robot for odor source localization in extreme dynamic advection-diffusion environment with obstacle: theory, simulation and measurement", *IEEE Computational Intelligence Magazine: Special Issue on Biometric*, Vol. 2, Issue 2, pp. 37-51, 2007.

[4] Jatmiko, Wisnu et al, "Traffic signal control modification based on self-organizing in Indonesia", *IEEE Transactions on Evolutionary Computation*, 2008.

[5] Jatmiko, Wisnu, Adila A Krisnadhi, Rizki Mardian, dan Nulad W Pambudi. "Pemulihan jaringan informasi dan komunikasi secara adaptif menggunakan teknologi robotika pada daerah bencana alam", Faculty of Computer Science – University of Indonesia, 2009.

[6] Takahashi, Junji, Kosuke Sekiyama dan Toshio Fukuda, "Cooperative object tracking with mobile robotic sensor network", Departemen Micro-Nano Systems Engineering Nagoya University, 2007.

[7] Mardian, Rizki. "Penanganan jaringan komunikasi multihop terkonfigurasi sendiri pada kasus dua atau banyak Sumber dengan menggunakan koloni robot otonom terdistribusi berdasarkan prinsip kecerdasan kolektif serta Pengembangan Simulator Naml", Faculty of Computer Science – University of Indonesia, 2007.

[8] Mardian, Rizki and Wisnu Jatmiko. "Approaching Distributed Mobile Robot Network in Dynamic Environment by Using Virtual Force and Artificial Potential Field Method for Optimized Configuration Solution", *International Seminar on QIR (Quality in Research)*, 2009.

[9] Siegwart, Roland dan Illah R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, London: MIT Press, 2004.

[10] Meger, David, Ioannis Rekleitis, and Gregory Dudek, *Simultaneous Planning, Localization, and Mapping in a Camera Sensor Network*. Minneapolis: Springer, 2006.

[11] Gossage, Mark, Ai P New, and Chee K Cheng, *Frontier-Graph Exploration for Multi-robot Systems in an Unknown Indoor Environment*. Minneapolis: Springer, 2006.

[12] Russell, J Stuart dan Peter Norvig, *Artificial Intelligence – A Modern Approach Second Edition*, New Jersey: Prentice Hall, 2003.

[13] Rossen, Kenneth H, *Discrete Mathematics and Its Applications Fifth Edition*, Boston: McGraw-Hill, 2003.