Kendali Formasi Mobile Robot Berdasarkan Jarak Menggunakan Algoritma Cosinus

by Anggoro Dwi Nur Rohman

Submission date: 30-Jul-2021 12:01PM (UTC+0700)
Submission ID: 1625703978
File name: ANGGORO_DWI_NUR_ROHMAN_4.pdf (435.29K)
Word count: 4807
Character count: 27718

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Kendali formasi adalah topik penelitian kendali multi-robot untuk memecahkan permasalahan koordinasi pergerakan. Kendali formasi bertujuan untuk mengendalikan sekelompok robot dalam mencapai formasi tertentu dan dapat mempertahankan formasi tersebut ketika bermanuver menuju arah yang diinginkan. Penjabaran oleh Guanghua, dkk (2013), pengembangan kendali formasi dilakukan dari beberapa algoritma strategi. Seperti yang dikembangkan oleh Wang, dkk (2014) menggunakan strategi leader-follower, menggunakan Fuzzy-Logic sebagai tingkah laku robot oleh Ferik, dkk (2016) dan menggunakan struktur virtual dimana sekelompok robot memiliki titik referensi sebagai satu robot oleh Li, dkk (2015).

Penjabaran oleh Oh, dkk (2015), bahwa dari berbagai pengembangan kendali formasi dapat ambil garis besar menjadi tiga bagian, yaitu berdasarkan posisi, perpindahan, dan jarak. Ketiga bagian tersebut tertuju pada jawaban dari pertanyaan, ”variabel apa yang digunakan sebagai sensor” dan ”variabel apa yang aktif dikendalikan oleh sistem multi-robot untuk mencapai formasi yang diinginkan”. Kendali formasi berdasarkan posisi adalah metode kendali formasi dimana robot diharuskan memiliki kemampuan untuk mengetahui kordinatnya sendiri berdasarkan koordinat global. Kerena itu kendali berdasarkan posisi membutuhkan sensor posisi seperti GPS. Kendali formasi berdasarkan perpindahan adalah kendali formasi dimana setiap robot tidak mengetahui koordinatnya berdasarkan koordinat global dikarenakan variabel yang digunakan sebagai sensor adalah kecepatan terhadap tetangganya. Metode berdasarkan posisi, robot membutuhkan kemampuan penyesuaian orientasi koordinat global sehingga setiap robot dibutuhkan sensor kompas untuk menyearahkannya. Formasi berdasarkan jarak adalah kendali formasi dimana variabel yang dikendalikan adalah variabel jarak antar robot yang terhubung sehingga koordinat yang digunakan tidak mengacu pada koordinat global. Penerapan formasi berdasarkan jarak menggunakan sensor yang lebih sedikit dibanding dengan posisi dan perpindahan. Namun pembahasan model yang lebih nyata untuk diterapakan kendali formasi berdasarkan jarak masih sedikit. Pengembangan formasi berdasarkan jarak telah dikembangkan menggunakan teori graph dengan model single dan double integrator sederhana oleh Oh, dkk (2014), menggunakan informasi

jarak untuk mengendalikan model sederhana double integrator oleh Cai, dkk (2014), menggunakan konsensus antara robot untuk mencapai bentuk formasi dengan model single integrator sederhana oleh Deghat, dkk (2016), mengendalikan bentuk formasi menggunakan adaptive control untuk mengestimasi kecepatan tetangga dari model robot yang sederhana oleh Kang, dkk (2014), menggabungkan kendali formasi berdasarkan jarak dan perpindahan untuk mengendalikan model robot yang sederhana oleh Park, dkk (2015), dan menggunakan kendali Proportional-Integral (PI) untuk mengendalikan jarak setiap model robot yang sederhana oleh Rozenheck, dkk (2015).

Kendali PI pada penelitian oleh Rozenheck, dkk (2015) tidak dapat langsung diterapkan menggunakan sensor jarak karena kendali tersebut mengambil informasi jarak menggunakan selisih dari koordinat kartesian global setiap robot. Sedangkan dalam praktiknya robot hanya bisa mengukur jarak dan tidak mengetahui koordinat dari robot tetangganya. Sudah umum penelitian dibidang lokalisasi menggunakan sensor jara seperti yang dibahas oleh Guo, dkk (2020) yang menggantikan sensor berbasis vision untuk mendapatkan koordinat dari beberapa robot, penelitian oleh Qiang, dkk (2017) membutuhkan beberapa sensor jarak yang terpasang statis digunakan untuk mengetahui koordinat sekelompok robot dan mengendalikannya secara terpusat, dan pengembangan Qiang, dkk (2018) dengan memasang dua sensor jarak di salah satu dari sekelompok robot, lalu mendistribusikan koordinat ke robot tetangganya. Lokalisasi menggunakan dua sensor jarak tersebut memanfaatkan rumus segitiga untuk mendapatkan koordinat robot tetangganya. Penelitian ini akan mengembangkan algoritma memanfaatkan rumus segitiga yang dikembangkan oleh Qiang, dkk (2018) dan menggantikan salah satu robot yang memiliki dua sensor dengan algoritma tersebut sehingga tidak mengharuskan salah satu robot memiliki dua sensor. Penelitian oleh Rozenheck, dkk (2015) mengembangkan kendali formasi menggunakan model holonomic sederhana dimana robot bergerak ke suatu arah tidak bergantung dari kondisi awal arah robot dan juga model sederhana tidak mempertimbangkan parameter fisik pada modelnya. Percobaan akan menggunakan tiga model robot holonomic dengan harapan menjadi langkah awal untuk mengembangkan kendali formasi berdasarkan jarak menggunakan model robot yang lebih nyata.

1.2 Identifikasi dan Perumusan Masalah

Pembahasan dari latar belakang dapat ditentukan pontensi permasalahan sebagai identifikasi masalah, yaitu kendali formasi berdasarkan jarak tidak dapat langsung

diterapkan menggunakan sensor jarak karena kendali formasi mengambil informasi jarak menggunakan selisih dari koordinat kartesian global setiap robot. Dalam penelitian ini akan digunakan batasan-batasan permasalahan sebagai berikut :

1. Kendali formasi berdsarkan jarak akan menggunakan tiga robot yang di-jalankan secara simulasi.
2. Sensor jarak bekerja secara ideal.
3. Strategi yang akan dikembangkan hanya untuk mengetahui koordinat kondisi awal saja.

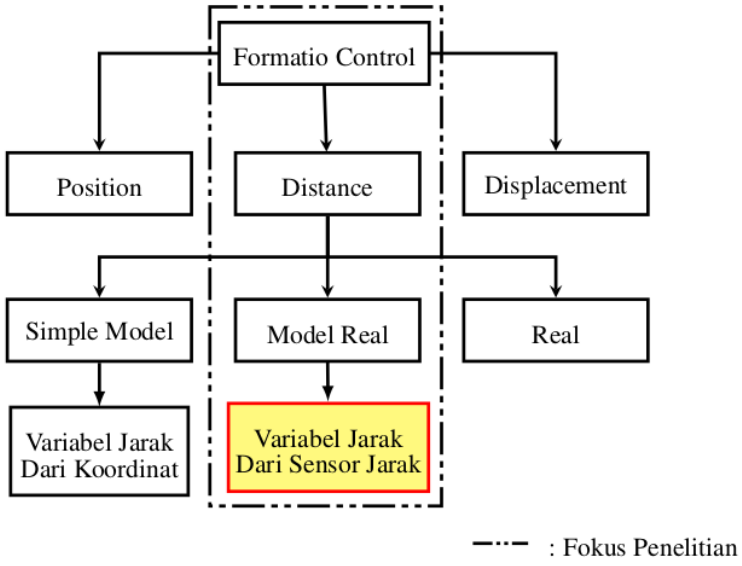
Dapat ditentukan permasalaah untuk penelitian ini, yaitu ”bagaimanakah strategi untuk mendapatkan koordinat kondisi awal kendali formasi apabila variabel yang dikendalikan adalah jarak antar robot?.

1.3 Tujuan dan Manfaat

Tujuan dari penelitian ini adalah mengetahui strategi untuk mendapatkan koordi-nat kondisi awal kendali formasi apabila variabel yang dikendalikan adalah jarak antar robot. Manfaat dari penelitian ini adalah memberikan referensi untuk permasalahan kendali multi-robot, kususnya pada permasalhaan kendali formasi, terhadap model yang lebih nyata. dan membuka peluang penelitian dibidang kendali mengenai kendali formasi pada kendali multi-robot dilingkungan Fakultas Teknik Elektro, Universitas Brawijaya.

BAB 3
KERANGKA KONSEP PENELITIAN

Kerangka konsep penelitian akan dibahas mengenai potensi permasalahan yang timbul dalam topik kendali formasi. Kerangka penelitian ini berdasarkan literatur oleh Oh, dkk (2015), dimana didalam literatur tersebut, peneliti menguraikan berbagai metode yang digunakan dalam bidang kendali multi-robot, khususnya dalam kendali formasi. Kerangka penelitian akan digabungkan dengan permasalahan penerapan kendali formasi berdasarkan jarak oleh Rozenheck Rozenheck, dkk (2015), dimana kendali formasi tidak dapat langsung diterapkan menggunakan sensor jarak secara langsung.



Gambar 3.1: Kerangka Penelitian

Kendali formasi dikategorikan menjadi 3 bagian, yaitu berdasarkan posisi, perpindahan, dan jarak.

Pada formasi berdasarkan posisi, dimana agent diharuskan memiliki kemampuan untuk mengetahui koordinatnya sendiri berdasarkan koordinat global. Sehingga, koordinat tujuan didistribusikan kepada setiap agent dan agent bekerja untuk mencapai koordinat tersebut. Karena itu, kebutuhan individu untuk berinteraksi dengan individu lain sangat kecil. Metode formasi ini pada praktiknya, interaksi

antar individu dilakukan untuk menangani masalah disturbance, saturasi akselerasi, dan lain-lain. Karena metode ini membutuhkan kemampuan untuk mengetahui koordinat global, dibutuhkan biaya yang lebih dibanding metode lain dalam perangkat sensor yang *advance*, seperti sensor GPS;

Pada formasi kendali berdasarkan perpindahan, secara individu agent tidak mengetahui koordinatnya berdasarkan koordinat global. Akan tetapi, individu agent memiliki koordinatnya sendiri terhadap individu agent tetangganya dan harus dilakukan penyearahan terhadap koordinat setiap robot dengan koordinat global. Koordinat relatif itulah yang menjadi variable yang dikendalikan oleh agent. Oleh karena itu agent diharuskan memiliki kemampuan untuk mengetahui perpindahan dari individu lain berdasarkan koordinat agent itu sendiri, dan semua agent harus menyearahkan koordinatnya berdasarkan koordinat global, serta dibutuhkan interaksi antara individu lain untuk mencapai formasi yang dinginkan. Permasalahan pada metode ini ditujukan pada kendali formasi pada agent yang bersifat heterogent, pemeliharaan dalam komunikasi, dan kemampuan dalam menghindari rintangan;

Pada formasi berdasarkan jarak, dimana setiap individu agent memiliki koordinatnya masing-masing dan tidak perlu disearahkan dengan koordinat global. Variable yang dikendalikan pada meteode ini adalah variabel jarak antar agent yang terhubung, sehingga dibutuhkan kemampuan untuk agent saling berkomunikasi antar agent lain. Permasalah pada metode ini ditujukan pada analisa stabilitas secara general; Pentingnya dilakukan investigasi pada penerapan model yang lebih nyata. Pemeliharaan komunikasi juga menyumbang dalam permasalahan secara praktik, dan kemampuan untuk menghindari rintangan juga dibutuhkan.

Penelitian oleh Rozenheck Rozenheck, dkk (2015) mengembangkan kendali formasi berdasarkan jarak dengan bagus. Kendali formasi yang dikembangkan menggunakan kendali PI untuk mengendalikan beberapa robot menggunakan variabel jarak. Akan tetapi pada kendali formasinya mengubah informasi koordinat menjadi jarak dimana pada penerapannya robot tidak dapat mengetahui koordinat tetangganya.

3.1 Definisi Permasalahan Kendali Formasi

Kendali formasi adalah kendali multi-agent untuk mencapai suatu formasi yang diinginkan. Dapat diperhatikan dalam Gambar 3.1, dari berbagai metode teresebut dapat disimpulkan secara umum dalam 3 kategori. Yaitu berbasis posisi, pergerakan, dan jarak. Pembagian tersebut berdasarkan kemampuan sensor yang digunakan dan penggunaan komunikasi dalam metodenya. Dari ketiga kategori tersebut,

kendali formasi berbasis jarak sangat dibutuhkan pembahasan mengenai penerapan metode tersebut pada model yang nyata. Pembagian antara model yang simple, model yang nyata, dan secara praktik berdasarkan tingkat analisisnya. Pada model yang simple, analisis dilakukan untuk mengembangkan strategi kendali formasi saja. Dalam kenyataannya model yang digunakan memiliki kekurangan, seperti batas kerja kecepatan, batasan sensor, komunikasi dan batasan lain-lain yang mempengaruhi kendali formasi tersebut.

Oh, dkk (2015) menyatakan bahwa mayoritas dari hasil penelitian yang menggunakan pendekatan ini (*distance-based*) berfokus pada model agent dengan integrator-tunggal di suatu bidang datar. Gagasan model yang nyata memiliki manfaat ketika menginvestigasi karakteristik kendali secara mendasar, model agent yang lebih realistik (model yang nyata) perlu untuk dipelajari lebih lanjut untuk menambah kepraktisan metode kendali multi-robot, kususnya pada kendali formasi berdasarkan jarak. Dengan bertambahnya kepraktisan diharapkan dapat diterapkan dalam model yang real. Pada penelitian oleh Rozenheck Rozenheck, dkk (2015), kendali formasi berdasarkan jarak dikendalikan menggunakan kendali PI dan menghasilkan pergerakan yang baik. Dapat diperhatikan pada persamaan (2.3) bahwa peneliti menggunakan model yang simpel untuk mengembangkan kendali multi-robotnya. Penelitian oleh Rozenheck Rozenheck, dkk (2015) memperoleh variabel jaraknya menggunakan koordinat tetangganya dimana pada penerapannya robot tidak mengetahui koordinat akan tetapi berbentuk jarak yang diperoleh dari sensor jarak.

3.2 Permasalah dan Solusi

Dapat diperhatikan pada persamaan (2.7), state yang digunakan membutuhkan koordinat relatif dari tetangganya. Akan tetapi pada batasan penelitian ini, sensor yang digunakan hanya memberikan jarak terhadap tetangganya. Sedangkan koordinat relatif yang digunakan adalah kartesian. Apabila yang diketahui adalah jarak maka koordinat yang bisa digunakan adalah polar. Sedangkan koordinat polar membutuhkan sudut antara agent dan tetangganya. Oleh karena itu dibutuhkan algoritma khusus untuk mendapatkan sudut tersebut.

Untuk mengembangkan algoritma tersebut, dapat menggunakan hukum *cosinus* segitiga untuk menentukan sudutnya. Dengan memanfaatkan komunikasi antar robot, maka robot dapat mengirimkan informasi state kecepatan kepada tetangganya. Sehingga informasi tersebut dapat digunakan untuk memantau koordinat relatif terhadap tetangganya. Akan tetapi state kecepatan tersebut membutuhkan ni-

lai inisialisasi. Nilai inisialisasi ini akan diperoleh menggunakan algoritma *cosinus*. Sehingga, harapanya adalah kendali multi-robot menggunakan kendali PI dapat digunakan sesuai batasan penelitian.

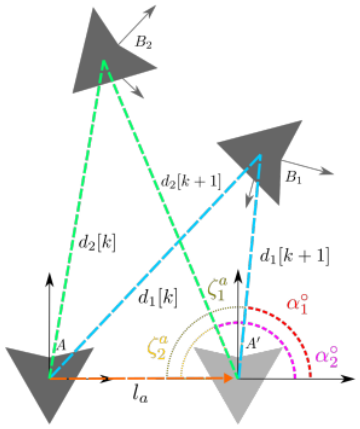
BAB 4
METODE PENELITIAN

Metode penelitian akan menerangkan tentang bagaimana strategi untuk menemukan koordinat tetangga menggunakan variabel jarak saja. Dapat diperhatikan pada persamaan (2.7) bahwa *state* yang dibutuhkan adalah kecepatan. Maka kendali robot diharuskan dapat mencapai kecepatan tertentu. Strategi pencarian koordinat mengharuskan juga robot untuk mencapai koordinat relatif tertentu. Pada bab ini akan menjelaskan bagaimana robot menemukan koordinat, mencapai kecepatan dan koordinat tertentu, dan membentuk formasi menggunakan model robot *holonomic*.

4.1 Strategi Penentuan Koordinat Tetangga

Tujuan dari strategi penentuan koordinat tetangga ini adalah menemukan koodinat menggunakan variabel jarak saja. Mengadopsi dari pengembangan oleh Qiang (Qiang, dkk (2018)) meletakkan dua sensor jarak pada salah satu robot, menggunakan kaidah rumus segitiga untuk mendapatkan sudut diantara robot tetangganya dan mengubahnya menjadi koordinat polar. Di penelitian ini menggunakan kaidah rumus segitiga tersebut akan tetapi robot hanya terpasang satu sensor jarak saja dan mengharuskan melakukan langkah tertentu untuk mendapatkan sudut diantara robot tetangganya.

Penentuan koordinat tentangga dapat ditemukan dengang mengubah koordinat polar menjadi koordinat kartesian. Koordinat polar membutuhkan panjang d_a, dan sudut α. Panjang d_a adalah variable yang didapat dari sensor yang memberikan nilai jarak dari robot A ke robot B, akan tetapi untuk mendapatkan koordinat polar, pengukuran sudu α tidak tersedia. Algoritama yang ditawarkan memanfaatkan hukum *cosinus* pada segitiga untuk mendapatkan sudut tersebut.



Gambar 4.1: Strategi Penentuan Koordinat Langkah Pertama

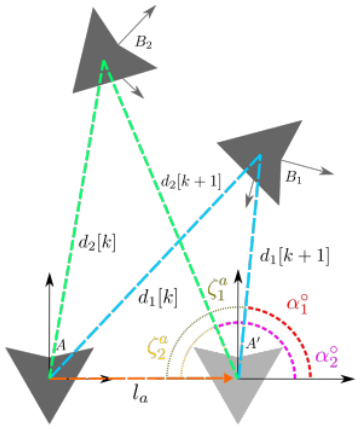
Langkah pertama. Gambar 4.1 adalah Ilustrasi langkah pertama algoritma cosinus dimana robot A diharuskan berpindah sepanjang la atau ke koordinat $A' = (0, la)$ menggunakan kendali mode dua di Persamaan (4.9) akan tetapi robot harus menyimpan jarak di $[k]$ terlebih dahulu. Setelah berpindah robot A mendapatkan jarak di $[k+1]$ digunakan untuk menentukan sudut α_i° menggunakan rumus segitiga cosinus. Berikut adalah Persamaan α_i°.

$$\zeta_i^a = cos^{-1}\left(\frac{l_a^2 + d_a[k+1]^2 - d_a[k]^2}{2d_a[k+1]l_a}\right) \tag{4.1}$$

$$\alpha_i^\circ = 180^\circ \pm \zeta_i^a \tag{4.2}$$

Variabel α_i° dan $d_i[k+1]$ adalah nilai dari koordinat polar dari setiap robot tetangga A. Diubah menjadi koordinat kartesian untuk dapat dimasukkan dalam state kendali formasi.

$$x_{B_i}^A = \begin{bmatrix} x_{B_i} = d_i[k]\cos\alpha_i^\circ \\ y_{B_i} = d_i[k]\sin\alpha_i^\circ \end{bmatrix} \tag{4.3}$$



Gambar 4.2: Strategi Penentuan Koordinat Langakah Kedua

Langkah kedua. Koordinat di Persamaan (4.3) akan menghasilkan bias dikarenakan Persamaan (4.2) tidak mengetahui letak kuadran sudutnya. Menggunakan ilustrasi di Gambar 4.1, langkah pertama menghasilkan dua kemungkinan koordinat robot B_1 dan B_1' . Apabila di Gambar 4.1, Sudut ζ_i^a adalah sudut segitiga $\angle AA'B_1$ atau $\angle AA'B_2$ sehingga dimungkinkan koordinat yang dihasilkan Persamaan (4.2) bisa berada pada kuadran 1 atau kuadran 4. Oleh karena itu di Persamaan 4.2 terdapat operasi $\pm$ dimana operasi tersebut akan dilakukan berdasarkan letak kuadran B_i.

$$\alpha_i^\circ = \begin{cases} 180^\circ - \zeta_i^a & ,Robot berada pada kuadran 1 dan 2 \\ 180^\circ + \zeta_i^a & ,Robot berada pada kuadran 3 dan 4 \end{cases} \tag{4.4}$$

Langkah kedua ini bertujuan untuk menentukan kejadian di Persamaan (4.4) dimana robot harus berpindah ke koordinat $A'' = (x*_a, y*_a)$ (Gambar 4.2). Sebelum robot berpindah, kondisi robot telah mendapatkan koordinat dari langkah pertama. Koordinat tersebut akan diubah menjadi jarak dan akan dibandingkan jarak tersebut dengan informasi jarak dari sensor setelah berpindah ke A'' . Apabila terdapat perbedaan maka kejadian di Persamaan (4.2) diubah kejadian selanjutnya dan mengkoreksi koordinat sebelumnya.

4.2 Analisa Algoritama Dengan Tetangga Statis

Telah dijelaskan pada Bab 4.1 bahwa robot bergerak kearah yang random dengan jarak tertentu untuk mengetahui koordinat tetangga. Analisa akan dilakukan de-

ngan membandingkan berbagai jarak dari tinggat rendah, sedang, dan tinggi untuk mengetahui respon algoritma yang sesuai dan optimal untuk mendapatkan koordinat tetangga.

Dari hasil analisa ini akan menghasilkan jarak terbaik untuk algoritma menentukan koordinat tetangga. Pembuktian akan dilakukan secara menampilkan grafik respon.

4.3 Kendali Robot *Holonomic*

Kendali robot *holonomic* akan dibagi menjadi dua mode. Perbedaan kedua mode tersebut adalah *setpoint* kendalinya, dimana mode satu akan memiliki setpoint kecepatan robot sedangkan mode dua memiliki *setpoint* koordinat kerangka robot. Penjelasan lebih lengkap kegunaan dari kedua mode tersebut akan dijelaskan di pembahasan metode strategi penentuan koordinat.

Mode Satu. Kendali robot mode satu bertujuan untuk robot mencapai kecepatan yang diinginkan. Untuk mencapai tujuan tersebut akan menggunakan metode *state-feedback* seperti yang telah dijabarkan sebelumnya. Mengadopsi persamaan *state-space* dari robot.

$$\dot{x_c}(t) = A_c x_c(t) + B_c u_c(t) + k_c sgn(x_c(t)), \tag{4.5}$$

$$y_c(t) = C_c x_c(t), \tag{4.6}$$

Dimana $A_c, B_c, k_c, C_c \in \mathbb{R}^{3\times 3}$ adalah matriks parameter robot yang telah dijelaskan sebelumnya.

$$A_c = \begin{bmatrix} -\frac{3.l^2.K_t^2}{2.M.R_a.r^2} - \frac{B_{\dot{x}_r}}{M} & 0 & 0 \\ 0 & -\frac{3.l^2.K_t^2}{2.M.R_a.r^2} - \frac{B_{\dot{y}_r}}{M} & 0 \\ 0 & 0 & -\frac{3.l^2.K_t^2}{2.I.R_a.r^2} - \frac{B_{\dot{\theta}_r}}{I} \end{bmatrix},$$

$$B_c = \begin{bmatrix} 0 & \frac{l.K_t}{R_a.r}.\frac{\cos(30^\circ)}{M} & -\frac{l.K_t}{R_a.r}.\frac{\cos(30^\circ)}{M} \\ \frac{l.K_t}{R_a.r}.\frac{-1}{M} & \frac{l.K_t}{R_a.r}.\frac{\cos(60^\circ)}{M} & \frac{l.K_t}{R_a.r}.\frac{\cos(60^\circ)}{M} \\ \frac{l.K_t}{R_a.r}.\frac{b}{I} & \frac{l.K_t}{R_a.r}.\frac{b}{I} & \frac{l.K_t}{R_a.r}.\frac{b}{I} \end{bmatrix}, k_c = \begin{bmatrix} -\frac{C_{\dot{x}_r}}{M} & 0 & 0 \\ 0 & -\frac{C_{\dot{y}_r}}{M} & 0 \\ 0 & 0 & -\frac{C_{\dot{\theta}_r}}{M} \end{bmatrix},$$

$$C_c = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Dimana $y_c(t) = x_c(t) = \begin{bmatrix} v & v_n & w \end{bmatrix}^T$ adalah vektor kecepatan. Lalu ditentukan

fungsi kendali $u_c(t)$ menggunakan *state-feedback*.

$$u_c(t) = -K_c x_c(t) + N_c r_c \tag{4.7}$$
$$N_c = -[C(A_c - B_c K_c)^{-1} B_c]^{-1} \tag{4.8}$$

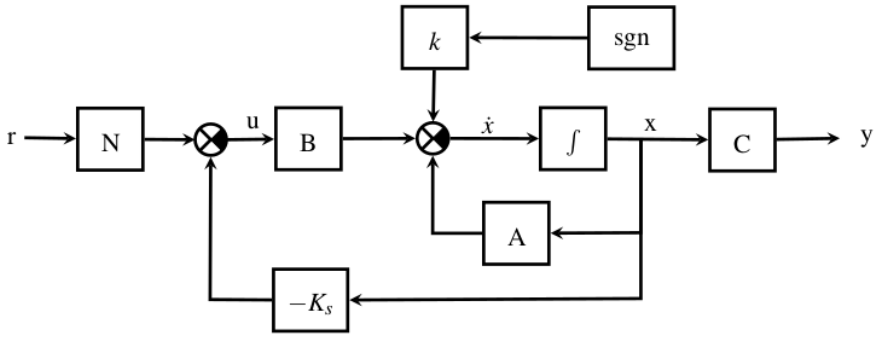
Dimana vektor $r_c = \begin{bmatrix} v* & v_n* & w* \end{bmatrix}$ adalah *set-point* kendali mode satu dan matriks K_c diperoleh dari solusi persamaan *Riccati*.

Mode dua. Kendali robot mode bertujuan untuk robot mencapai koordinat relatif yang diinginkan. Untuk mencapai tujuan tersebut akan menggunakan metode *state-feedback* seperti yang telah dijabarkan sebelumnya. Mengadopsi persamaan *state-space* dari robot Persamaan(2.18)-(2.19). Lalu ditentukan fungsi kendali $u(t)$ menggunakan *state-feedback*.

$$u(t) = -K_r x_c(t) + N_r r_r \tag{4.9}$$
$$N_r = -[C(A_r - B_r K_r)^{-1} B_r]^{-1} \tag{4.10}$$

Dimana vektor $r_c = \begin{bmatrix} \dot{x*}_r & \dot{y*}_r & \dot{\theta*}_r & v* & v_n* & w* \end{bmatrix}$ adalah *set-point* kendali mode dua dan matriks K_r diperoleh dari solusi persamaan *Riccati*.

4.4 State Feedback



Gambar 4.3: State-feedback Sistem

Pada persamaan (2.18) diketahui bahwa state memiliki dimensi 6×1. Dimensi tersebut tidak menunjukan sistem memiliki orde 6. Apabila diperhatikan orde dari sistem adalah orde 2. Dengan membaginya kedalam 3 persamaan state-space akan lebih mudah dalam analisis parameter kendalinya. Berikut adalah persamaannya.

$$\begin{bmatrix} \dot{x}_p \\ \ddot{x}_r \end{bmatrix} = \begin{bmatrix} 0 & A_{14} \\ 0 & A_{44} \end{bmatrix} \begin{bmatrix} x_p \\ \dot{x}_r \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ B_{11} & B_{12} & B_{13} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + k_{44} sgn(\dot{x}_r) \tag{4.11}$$

$$\begin{bmatrix} \dot{y}_p \\ \ddot{y}_r \end{bmatrix} = \begin{bmatrix} 0 & A_{25} \\ 0 & A_{55} \end{bmatrix} \begin{bmatrix} y_p \\ \dot{y}_r \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ B_{21} & B_{22} & B_{23} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + k_{55} sgn(\dot{y}_r) \tag{4.12}$$

$$\begin{bmatrix} \dot{\theta}_p \\ \ddot{\theta}_r \end{bmatrix} = \begin{bmatrix} 0 & A_{34} \\ 0 & A_{66} \end{bmatrix} \begin{bmatrix} \theta_p \\ \dot{\theta}_r \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + k_{66} sgn(\dot{\theta}_r) \tag{4.13}$$

State feedback membutuhkan kembalian nilai state dari sistem dan mengkalikanya dengan besaran tertentu agar nilai karakteristik sistem tetap dalam keadaan stabil atau sesuai ketentuan. Secara umum, state tidak dapat diperoleh langsung dari sistem. Kemampuan untuk memperoleh state dari sistem langsung disebut dengan kemampuan Observablity. Apabila sebuh sistem tidak Observable, maka dalam kendalinya dibutuhkan Observer. Dimana tugasnya adalah mengestimasi state pada sistem dengan membandingkan keluaran dan masukan. Syarat untuk dapat diterapkan state feedback, sistem harus observable dan controlable. Berikut adalah rumus untuk menguji apakah sistem bersifat controlable atau tidak (Dorf, dkk (2010)).

$$P_c = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix}$$

$$rank[P_c] = n \tag{4.14}$$

Apabila hasil dari $rank(P_c) \neq n$ maka sistem tidak *fully controlable*. Sedangkan untuk menguji observabilitas dapat menggunakan rumus berikut.

$$P_o = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

$$rank[P_o] = n \tag{4.15}$$

Apabila sistem observable, rank dari matriks Observablity P_o sama dengan besar orde sistem. Menggunakan parameter robot pada Tabel 4.1 untuk diimplementasi pada persamaan (2.18)-(2.19) akan menghasilkan $rank[P_c] = 6$ dan $rank[Po] = 6$. Maka dari itu dapat disimpulkan sistem robot adalah controlable dan observable. Karena pengukuran pada setiap *state* dapat dilakukan, maka *observer* tidak dibutuhkan dalam desain kendali robot.

Tabel 4.1: Parameter model oleh Correia, dkk (2012)

Simbol	Deskripsi	Nilai
$B_v(N/m/s)$	viscous friction coefficient related to v	0.94
$B_{vn}(N/m/s)$	viscous friction coefficient related to vn	0.96
$B_\omega(N/rad/s)$	viscous friction coefficient related to ω	0.01
$C_v(N)$	coulomb friction coefficient related to v	2.2
$C_{vn}(N)$	coulomb friction coefficient related to vn	1.5
$C_\omega(N.m)$	coulomb friction coefficient related to ω	0.099
$b(m)$	radius of the robot	0.1
$M(kg)$	mass of the robot	1.5
$In(kg.m^2)$	inertia moment of the robot	0.025
δ	angle	30°
$r_1, r_2, r_3(m)$	radius of the wheels	0.035
l_1, l_2, l_3	reduction of the motors	19:1
$L_{a1\ldots3}(H)$	motor's armature inductance	0.00011
$R_{a1\ldots3}(\Omega)$	motor's armature resistance	1.69
$K_v(Volts/rad/\mathrm{s})$	motor's emf constant	0.0059
$K_{t1\ldots3}$ (N.m/A)	motor's torque constant	0.0059

4.5 Desain Kendali

Berdasarkan Dorf, dkk (2010), bahwa kendali optimal berdasarkan indeks kinerja sistem. Indeks tersebut adalah hasil dari meminimalisasi pada integral kuadrat error atau *integration square error* (ISE). Kendali optimal dilakukan oleh komputer untuk mengkalkulasi minimal indeks tersebut. Apabila sebuah sistem *state space*

$$\dot{x} = Ax + Bu$$
$$u = -K_s x$$

maka indeks kinerja

$$J = \int_0^\infty (x^T Qx + u^T Ru)dt \tag{4.16}$$

dimana Q adalah matriks diagonal $n \times n$, R adalah matriks diagonal $m \times m$ dan keduanya adalah matriks pembobot terhadap state sistem dan input. Ketika indeks terminimalisasi, maka

$$K_s = R^{-1}B^T P \tag{4.17}$$

dengan matriks P $n \times n$ ditentukan dari solusi persamaan *Riccati*.

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{4.18}$$

Kalkulasi konstanta K_s akan dikalkulasi menggunakan persamaan-(4.11) (4.13). Sehingga konstanta K_s akan terbagi dalam sub matriks K_{sx}, K_{sy}, dan $K_{s\theta}$. Berikut adalah hasil kalkulasi.

$$K_s^x = \begin{bmatrix} 0.00000 & 0.00000 \\ 18.25742 & 3.25168 \\ -18.25742 & -3.25168 \end{bmatrix} = \begin{bmatrix} K_s^{x1} & K_s^{x2} \end{bmatrix}$$

$$K_s^y = \begin{bmatrix} -21.08185 & -3.74993 \\ 10.54093 & 1.87496 \\ 10.54093 & 1.87496 \end{bmatrix} = \begin{bmatrix} K_s^{y1} & K_s^{y2} \end{bmatrix}$$

$$K_s^\theta = \begin{bmatrix} 3.33333 & 0.99738 \\ 3.33333 & 0.99738 \\ 3.33333 & 0.99738 \end{bmatrix} = \begin{bmatrix} K_s^{\theta 1} & K_s^{\theta 2} \end{bmatrix}$$

Apabila diintegrasi terhadap persamaan (2.18) terhadap diagram 4.3

$$K_s = \begin{bmatrix} K_s^{x1} & K_s^{y1} & K_s^{\theta 1} & K_s^{x2} & K_s^{y2} & K_s^{\theta 2} \end{bmatrix} \tag{4.19}$$

Setelah mendapatkan konstanta K_s, sistem sudah dalam keadaan stabil. Akan tetapi sistem tidak mencapai set poin yang diinginkan. Maka permasalahan tersebut dapat diselesaikan dengan *input refrence*. Dimana refrence tersebut akan dikalikan dengan konstanta N. Berikut adalah persamaan *input refrence* sebagai penambah dari *state feedback*.

$$u(t) = -Kx(t) + Nr \tag{4.20}$$

Sehingga persamaan *state space* menjadi berikut.

$$\begin{cases} \dot{x} &= (A - BK_s)x + BNr \\ y &= Cx \end{cases} \tag{4.21}$$

Untuk mendapatkan nilai N maka dapat diasumsikan bahwa sistem dalam keadaan *steady state*, yaitu $\dot{x} = 0$, sehingga persamaan state space menjadi berikut.

$$x = -(A - BK_s)^{-1}BNr \tag{4.22}$$

Dalam keadaan *steady state*, harapannya adalah nilai refrence sama dengan nilai keluaran, $y = r$. Sehingga dapat diperoleh persamaan N.

$$N = -[C(A - BK_s)^{-1}B]^{-1} \tag{4.23}$$

Berikut adalah hasil kalkulasi dari rumus N menggunakan matriks pada persamaan (4.11),(4.12), dan (4.13).

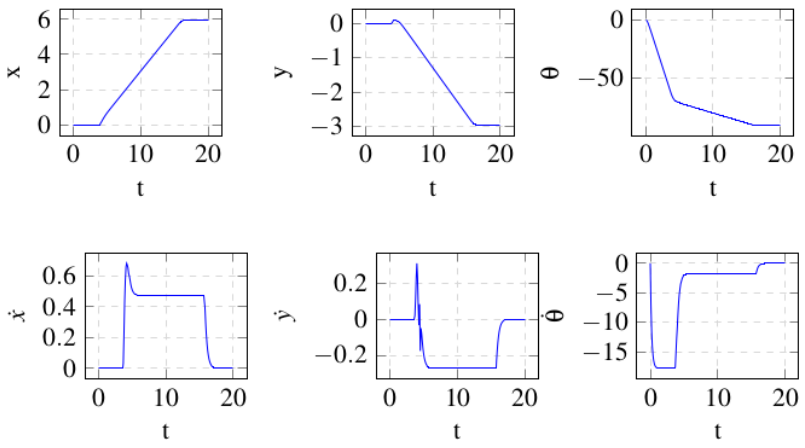
$$N^x = \begin{bmatrix} 0.00000 & 0.00000 \\ 18.25742 & 0.00000 \\ -18.25742 & 0.00000 \end{bmatrix} = \begin{bmatrix} N^{x1} & N^{x2} \end{bmatrix}$$

$$N^y = \begin{bmatrix} -21.08185 & 0.00000 \\ 10.54093 & 0.00000 \\ 10.54093 & 0.00000 \end{bmatrix} = \begin{bmatrix} N^{y1} & N^{y2} \end{bmatrix}$$

$$N^\theta = \begin{bmatrix} 3.33333 & 0.00000 \\ 3.33333 & 0.00000 \\ 3.33333 & 0.00000 \end{bmatrix} = \begin{bmatrix} N^{\theta 1} & N^{\theta 2} \end{bmatrix}$$

Apabila diintegrasi terhadap persamaan (2.18) terhadap diagram 4.3

$$N = \begin{bmatrix} N^{x1} & N^{y1} & N^{\theta 1} & N^{x2} & N^{y2} & N^{\theta 2} \end{bmatrix} \tag{4.24}$$



Gambar 4.4: Grafik state terhadap t dengan state refrensi $r = \begin{bmatrix} 6 & -3 & -90 & 0 & 0 & 0 \end{bmatrix}$

4.6 Kendali Formasi Multi Robot

Pada sub bab 2.1.2 dijabarkan bagaimana kendali formasi menggunakan kendali-P dan menghasilkan persamaan (2.7). Persamaan tersebut adalah persamaan *state-space* kendali formasi. Apabila diperhatikan *state* yang digunakan adalah koordinat relatif dari robot. Akan tetapi dalam batasanya, robot hanya bisa mengetahui nilai jarak dari robot lain. Dengan kata lain, yang dibutuhkan dalam metode kendali formasi adalah jarak dalam bentuk koordinat, $x \in \mathbb{R}^2$. Sedangkan dalam kenyataanya yang diketahui adalah jarak, $r \in \mathbb{R}$. Apabila hanya variable jarak tersebut sebagai acuan kendali, maka robot tidak mengerti kearah mana harusnya robot itu bergerak untuk meminimalisasi error jaraknya.

Di Persamaan (2.7) adalah persamaan *state-space* yang menggunakan model sederhana di Persamaan (2.3). Apabila diperhatikan pada model tersebut masukan pada perseamaan tersebut berbentuk kecepatan. Menggunakan Persamaan (4.5) dan (4.9) akan diperoleh persamaan *state-space* baru

$$\dot{x_c}(t) = \Big(\big(A_c - B_c K_c + \big) \Big) x_c(t) K_{cf} sgn(x_c(t)) + (B_c N_c) r_c, \tag{4.25}$$

Lalu diterapkan di Persamaan (2.7) sebgai berikut.

$$\dot{p} = A_{cf}p(t) + B_{cf}v(t) + K_{cf}sgn(p(t)) \tag{4.26}$$

$$\dot{v} = -k_{p1}v_i(t) - R(p(t))^T k_{p2}(R(p(t))x_1(t) - d)) \tag{4.27}$$

$$A_{cf} = \begin{bmatrix} A_c - B_cK_c & 0 & 0 \\ 0 & A_c - B_cK_c & 0 \\ 0 & 0 & A_c - B_cK_c \end{bmatrix}; B_{cf} = \begin{bmatrix} (B_cN_c) \\ (B_cN_c) \\ (B_cN_c) \end{bmatrix};$$

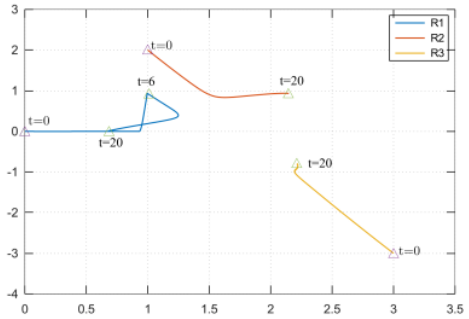
$$K_{cf} = \begin{bmatrix} k_c & 0 & 0 \\ 0 & k_c & 0 \\ 0 & 0 & k_c \end{bmatrix} \tag{4.28}$$

BAB 5
HASIL DAN PEMBAHASAN

Hasil pembahasan akan dilakukan dengan membahas hasil percobaan algoritma *cosinus*. Pembahasan hasil pertama akan membahas jalannya algoritma terhadap kendali formasi untuk menentukan koordinat kondisi awal dan mengetahui langkah-langkah nya. Pembahasa kedua akan membandingkan penerapan algoritma dengan tanpa algoritma sehingga mengetahui spesifikasi kecepatan algoritma untuk menemukan koordinat kondisi awal. Pemabahasan terakhir akan dilakukan percobaan dengan kuadran yang berbeda sehingga algoritma dapat bekerja dengan baik untuk menemukan koordinat.

5.1 Pergerakan Robot Terhadap Algoritma *Cosinus*

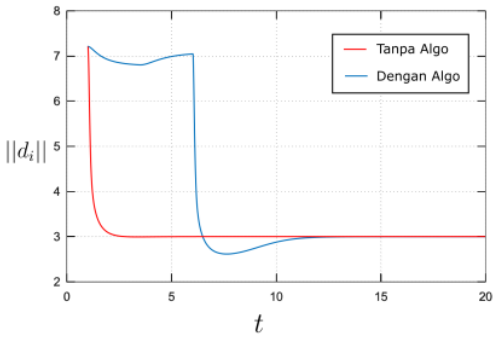
Kendali formasi berdasarkan jarak akan dijalankan secara simulasi menggunakan MATLAB/GNU Octave dan algoritma cosinus akan dijalankan pertama kali untuk mendapatkan state yang akan digunakan di Persamaan (16) dan (18). Simulasi akan menggunakan 3 robot dengan himpunan simpul $V = \{R_1, R_2, R_3\}$ dan himpunan sisi $E = \{(R_1, R_2), (R_3, R_2), (R_3, R1)\}$ sehingga variable jarak $d_1 = ||x_1 - x_2||$, $d_2 = ||x_3 - x_2||$ dan $d_3 = ||x_3 - x_1||$. Gambar 4 adalah pergerakan robot terhadap koordinat global robot R_1 dengan kendali formasi algoritma cosinus. dapat diperhatikan robot R_1 menjalankan algoritma cosinus langkah pertama dimana robot berpindah sepanjang $l_a = 1$ sehingga setpoint kendali Persamaan (13) adalah $r_2^c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$. Setelah setpoint tercapai, robot R_1 mendapatkan jarak yang dibutuh kan



Gambar 5.1: Grafik Pergerakan Robot Menggunakan Algoritma Cosinus danKendali Formasi

5.2 Perbandingan Penerapan Algoritma *Cosinus*

untuk menyelesaikan Persamaan (21) dan (22) lalu dilanjutkan ke langkah dua dengan berpindah ke koordinat (1,1) sehingga setpoint kendali Persamaan (13) adalah $r_2^c = [110000]^T$. Setelah setpoint tercapai maka dilakukan pengecekan kejadian Persamaan (23) dengan membandingkan jarak dari sensor dengan jarak dari koordinat yang dihasilkan dari Persamaan (22). Robot R_3 berada di kuadran 4 maka kejadian yang digunakan adalah dan robot R_2 berada di kuadran 1 maka kejadian yang digunakan adalah $\alpha_i^\circ = 180^\circ - \zeta_i^a$. Setelah koordinat ditemukan, maka kendali robot mulai berpindah menggunakan kendali formasi Persamaan (16) dan (18) dengan kondisi koordinat awal state berada di $t = 6$ dan akhir dari formasi di $t = 20$ pada Gambar 4.



Gambar 5.2: Grafik Response Jarak Terhadap Waktu Pada Robot Dengan Kendali Formasi.

5.3 Algoritma *Cosinus* Di Seluruh Kuadran

Seperti yang telah dijelaskan di pembahasan algoritma cosinus (subbab 2.4), bahwa kendali formasi di Persamaan (16) dan (18) mengharuskan state dalam bentuk koordinat dimana ketika kondisi awal robot tidak mengetahuinya, maka peran utama dari algoritma cosinus untuk mendapatkan state tersebut. Gambar 5 adalah grafik perbandingan kendali formasi Persamaan (16) dan (18) dengan algoritma cosinus dan tanpa algoritma cosinus dengan asumsi ketika dilakukan percobaan tanpa algoritma, kondisi awal state telah diketahui. Hasil jarak $||d_i||$ pada kendali formasi tanpa algoritma cosinus menunjukan bahwa setiap robot mencapai jarak yang sama dalam waktu kurang lebih 4 detik. Sedangkan pada kendali formasi dengan algoritma cosinus terdapat tambahan waktu 6 detik. Tambahan waktu tersebut digunakan untuk menjalankan algoritmanya.

Tabel 5.1: Settling Time Dengan Konstanta Kp Yang Berbeda

Koordinat			**Settling Time Tanpa Algoritma cosinus (detik)**			**Settling Time Dengan Algoritma cosinus (detik)**			**Selisih (detik)**		
R1	R2	R3	A	B	C	A	B	C	A	B	C
(0,0)	(1,2)	(-2, 3)	6	5	3	14	11	9	8	6	6
(0,0)	(-2, -4)	(3, -2)	4	2	2	9	8	7	5	6	5
(0,0)	(1,2)	(3, -3)	7	4	2	16	13	11	9	9	9
(0,0)	(-2, -3)	(3,2)	10	7	4	16	11	9	6	4	5
Konstanta : (A) $K_{p1} = 50$; $K_{p2} = 3$; (B) $K_{p1} = 80$; $K_{p2} = 7$; (C) $K_{p1} = 100$; $K_{p2} = 15$;											

Percobaan di Gambar 5 adalah grafik settling time saat koordinat robot seperti di Gambar 4 yaitu R_2 berada di kuadran 1 dan R_3 berada di kuadran 4. Hasil percobaan Tabel 1 bertujuan untuk mengetahui kecepatan algoritma cosinus mencari koordinat tetangga dengan kombinasi kemungkinan dua robot dengan kejadian di Persamaan (23). Serta diberikan nilai K_{p1} dan K_{p2} di Persamaan (18) yang berbeda untuk mengetahui pengaruh parameter kendali formasi terhadap algoritma cosinus. Setelah dilakukan percobaan, selisih waktu ketika algoritma cosinus berjalan dengan perbedaan parameter menghasilkan waktu yang tidak jauh beda antara 0-2 detik. Maka, algoritma cosinus mendapatkan koordinat tetangga yang digunakan untuk nilai kondisi awal kendali hampir tidak berpengaruh terhadap kendali formasi Persamaan (18) dengan rata-rata waktu 6.5 detik, minimal waktu 4 detik dan maksimal 9 detik.

BAB 6
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Kendali formasi berdasarkan jarak telah dikembangkan fokus dibagian metode kendali dan masih perlu dibahas penerapannya. Telah dikembangkan di penelitian sebelumnya dengan fokus di metode kendali formasi berdasarkan jarak dengan penerapan metode PI dan analisa menghasilkan respon yang baik, akan tetapi penerapan metode tersebut dikenyataannya tidak berjalan dengan baik karena robot tidak mendapatkan kondisi awal yang digunakan untuk kendalinya. Penelitian ini telah mengusulkan algoritma *cosinus* sebagai solusi kondisi awal kendali. Dari hasil percobaan, algoritma *cosinus* dapat digunakan untuk mendapatkan koordinat kondisi awal kendali formasi berdasarkan jarak, algoritma dapat menemukan koordinat relatif tetangganya di semua kuadran, dan rata-rata waktu yang dibutuhkan adalah 6.5 detik.

6.2 Saran

Penelitian ini dimulai dengan pengembangan dibagian kendali formasi berdasarkan jarak dan penerapannya pada model yang nyata. Akan tetapi dalam proses penelitian terdapat permasalahan kondisi awal pada kendalinya sehingga pengembangan kendali formasi berdasarkan jarak masih jauh dianggap sempurna. Ditujuan kepada penelitian selanjutnya dibidang kendali formasi berdasarkan jarak terdapat peluang penelitian di bagian penerapan kendali tanpa menggunakan robot asli yaitu menggunakan metode HIL (Hardware In Loop). Saran ini disampaikan karena penelitian dibidang kendali formasi mengendalikan banyak robot dan juga pemodelan robot yang nyata telah banyak dikembangkan maka penggunaan metode HIL sangat bermanfaat untuk melakukan analisis kendali formasi. Peluang penelitian selanjutnya juga dapat dimulai di bagian analisis kendali menggunakan metode MPC (*Model Predictive Control*) karena metode MPC banyak digunakan untuk mengendalikan robot secara optimal dan *state feedback* adalah dasar dari metode MPC.

Kendali Formasi Mobile Robot Berdasarkan Jarak Menggunakan Algoritma Cosinus

ORIGINALITY REPORT

4%	2%	3%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Mariane Dourado Correia, André Gustavo, Scolari Conceição. "Modeling of a Three Wheeled Omnidirectional Robot Including Friction Models", IFAC Proceedings Volumes, 2012 Publication	2%
2	text-id.123dok.com Internet Source	<1%
3	Cao, J.. "Novel delay-dependent stability conditions for a class of MIMO networked control systems with nonlinear perturbation", Applied Mathematics and Computation, 20080401 Publication	<1%
4	www.meritnation.com Internet Source	<1%
5	library.binus.ac.id Internet Source	<1%
6	es.scribd.com Internet Source	<1%

7	issuu.com Internet Source	<1%
8	www.scribd.com Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		