Vol. 02, No. 1, Oktober 2019, pp. 13-26

Pemodelan Mekanisme *Docking Smart Warehouse Robot* Menggunakan Sistem Kontrol Penjejakan Lintasan dan Stabilisasi Titik

Docking Smart Warehouse Robot Mechanism Modelling using Lane Stepping System Control and Point Stabilization

Prasetyo Wibowo Laksono Sanjaya[a], Annisa Zulfa Hidayah[a], Augie Widyotriatmo*[b,1]

[a] Program Studi Teknik Fisika, Fakultas Teknologi Industri, ITB, Bandung, 40132, Indonesia
[b] Kelompok Keahlian Instrumentasi & Kontrol, Fakultas Teknologi Industri, ITB, Bandung, 40132, Indonesia
[1] augie@tf.itb.ac.id*
* penulis yang sesuai (corresponding author)

Abstrak. Pada makalah ini dilakukan pemodelan mekanisme *docking* sebuah robot beroda untuk aplikasi pengisian daya pada robot *smart warehouse*. Diasumsikan terdapat dua belas lokasi *docking* dengan tujuan postur akhir robot yang siap keluar dari tempat *docking*. Diusulkan sistem pengontrolan pergerakan robot menggunakan penjejakan lintasan dalam tiga segmen *piecewise* yang dibangkitkan dengan polinomial orde tiga dilanjutkan dengan stabilisasi titik. Dilakukan pula pengontrolan penggerakan roda kanan-kiri robot untuk mencapai pergerakan yang diinginkan menggunakan pengontrol PI (*Proportional-Integral Controller*). Diperoleh hasil simulasi pengontrolan yang stabil dari berbagai postur robot awal menuju berbagai lokasi *docking* yang telah didefinisikan.

Kata Kunci: *robot smart warehouse*, penjejakan lintasan, stabilisasi titik, kontrol mekanisme *docking.*

Abstract. *The modelling of the docking mechamism of a wheeled robot for power charging application on smart warehouse robots was conducted. It was assumed that there were twelve docking locations with the purpose of the last robot posture being ready to exit the docking spot. It was suggested for the robot movement control system to use lane stepping in three piecewise segments which were activated with third order polynomial continued with point stabilization. The left-right robot wheel movement control was applied to attain the desired movement using Proportional-Integral Controller (PI controller). The result showed a stable control simulation of various initial robot posture to the defined various docking locations.*

Keywords**:** *robot smart warehouse, lane stepping, point stabilization, docking control mecchanism.*

1 Pendahuluan

Penggunaan robot dalam berbagai aplikasi industri telah terbukti cenderung membuat kehidupan manusia menjadi lebih baik. Manfaat penggunaan robot dalam industri sangat terasa dalam peningkatan produktivitas, keamanan, dan penghematan uang serta waktu [1]. Hal ini membuat aplikasi robot semakin berkembang dan merambah sektor yang semakin luas dengan permasalahan-permasalahan yang semakin spesifik.

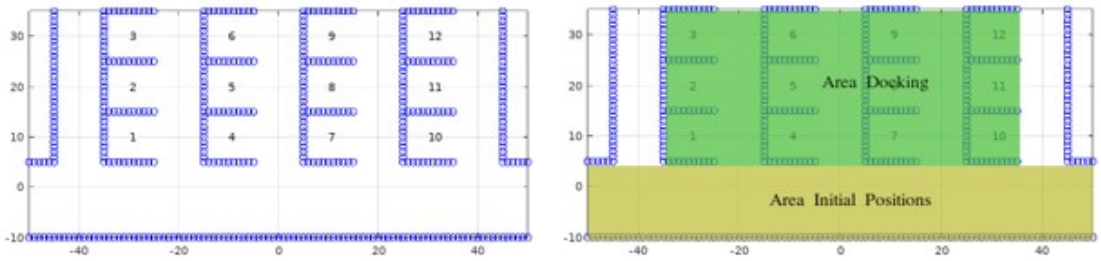
Salah satu sektor yang mulai menggandrungi penggunaan robot adalah otomasi pergudangan atau *smart warehouse*. Pergudangan merupakan salah satu bagian kunci dalam manajemen rantai pasok di industri. Pergudangan yang baik akan menjadi indikator daya saing perusahaan, mengingat biaya logistik merupakan bagian penting dalam biaya produksi keseluruhan [2]. Selain untuk rantai pasok pabrik, sistem *smart warehouse* juga mulai banyak diimplementasikan

Dikirimkan 2 Agustus 2019, Revisi 28 Agustus 2019, Diterima 23 September 2019

dalam industri ritel daring atau *e-commerce*. Robot beroda untuk otomasi gudang telah banyak dikembangkan dan bahkan diimplementasikan di perusahaan *e-commerce* raksasa tingkat dunia, misalnya Amazon dan Alibaba. Menurut laporan Bloomberg [3], Amazon telah memiliki lebih dari 30.000 Kiva robot untuk membantu mengurus pergudangannya.

Salah satu sorotan permasalahan dalam pengembangan robot beroda adalah terkait daya. Karena penggunaannya yang *mobile*, daya dalam robot beroda umumnya dibuat *portable* dan dapat diisi ulang [4]. Pengisian ulang daya suatu robot beroda dapat dilakukan dalam skema *charging dock* atau pangkalan pengisian daya. Dalam skema ini, robot beroda yang mendeteksi diri kehabisan daya akan menjalankan mekanisme pergerakan menuju pangkalan pengisian daya yang tidak terisi robot lain.

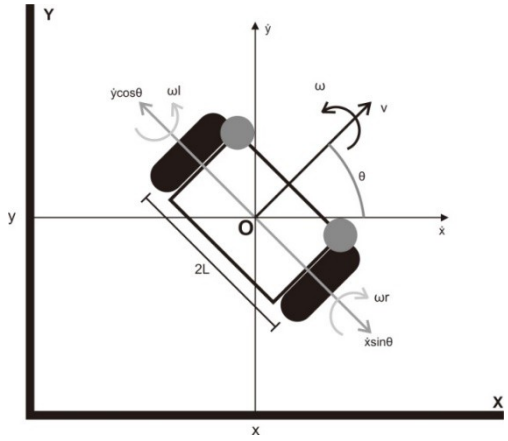
Pada makalah ini akan dibahas pengimplementasian sistem kontrol terkait mekanisme pergerakan *docking robot* beroda atau *wheeled mobile robot* (WMR) pada aplikasi *Smart Warehouse*. Permasalahan utama sistem kontrol pada robot beroda dapat dibagi dalam tiga jenis, yaitu stabilisasi, generasi lintasan, dan pelacakan lintasan [5], [6]. Untuk itu, akan dibahas dua algoritma sistem kontrol pergerakan *docking* robot beroda, yaitu penjejakan lintasan dan stabilisasi titik. Diasumsikan terdapat dua belas lokasi *docking* seperti pada Gambar 1 dan WMR yang telah berada pada area tertentu di dekat area *docking* memperoleh informasi lokasi *docking* yang kosong, kemudian mampu memilih lokasi *docking* yang akan ditempati. Akan dirancang sistem kontrol general yang mampu mengontrol gerak WMR dari berbagai posisi dan orientasi dalam area tertentu menuju ke salah satu sisi dari dua belas lokasi *docking*, kemudian menstabilkan orientasi WMR untuk siap dikeluarkan.



Gambar 1. Skema area *docking*

2 Pemodelan Robot WMR

Robot WMR yang digunakan memiliki jumlah roda dua, dengan sensor yang mampu mendeteksi sudut hadap robot serta posisi relatif robot x dan y terhadap suatu kerangka acuan. Seperti yang diusulkan pada literatur [6], akan dilakukan pemodelan kinematika dan dinamika dari badan robot beroda pada koordinat dua dimensi yang diilustrasikan pada Gambar 2.



Gambar 2. Skematik robot beroda dengan mekanisme roda differensial

Diasumsikan sebuah robot beroda dengan jarak antar titik tengah roda sebesar 2L, titik tengah O pada suatu koordinat x-y, dan memiliki kecepatan v pada arah θ. Kondisi robot pada setiap waktu dapat dinyatakan dengan vektor $q = [x \quad y \quad \theta]^T$ dengan batasan nonholonomik

$$\dot{x}sin(\theta) - \dot{y}cos(\theta) = 0 \tag{1}$$

Batasan nonholonomik ini membatasi gerakan robot ke arah samping secara langsung, sehingga robot harus melakukan gerak memutar terlebih dahulu untuk bergerak ke arah samping. Dengan mendefinisikan suatu vektor $J(q) = [sin\theta \quad -cos\theta \quad 0]^T$, terdapat suatu vektor S yang unik memenuhi $S^T(q)J(q) = 0$, atau

$$S(q) = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \tag{2}$$

Dengan mendefinisikan kecepatan linier v dan kecepatan putar ω dari badan robot, persamaan kinematika dari robot beroda dengan batasan nonholonomik diperoleh sebagai

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3}$$

Pergerakan suatu robot beroda diferensial dengan batasan nonholonomik ditentukan oleh kecepatan linier v dan kecepatan putar ω dari robot tersebut. Kedua parameter ini ditentukan lebih lanjut oleh kecepatan roda kanan ω_r dan kecepatan roda kiri ω_l robot dalam hubungan

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \frac{R}{2} \begin{bmatrix} 1 & 1 \\ 1/L & -1/L \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \tag{4}$$

Apabila didefinisikan suatu matriks *T* sebagai

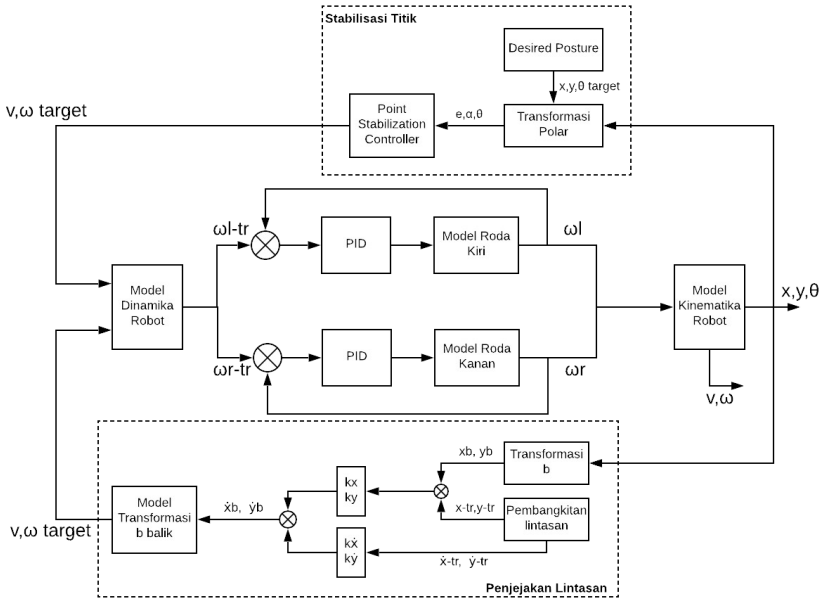
$$T = \frac{R}{2} \begin{bmatrix} 1 & 1 \\ 1/L & -1/L \end{bmatrix} \tag{5}$$

Maka nilai ω_r dan ω_l dapat diperoleh dari kecepatan linier v dan kecepatan putar ω sebagai

$$\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = T^{-1} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{6}$$

3 Perancangan Sistem Pengontrolan

Pengontrolan mekanisme *docking* dilakukan dengan menggunakan dua sistem kontrol (Gambar 3) yaitu penjejakan lintasan dan dilanjutkan stabilisasi titik, agar diperoleh posisi dan postur robot pada lokasi *docking* tertentu dari lokasi awal robot tertentu. Kedua sistem kontrol tersebut memberikan kecepatan linier *v* dan kecepatan sudut ω referensi yang harus dicapai oleh sistem robot utama. Sistem robot kemudian mengejar nilai *v* dan ω dengan mengatur kecepatan roda kanan dan kiri robot ($\omega_r \, dan \, \omega_l$) menggunakan skema pengontrolan PI (*Proportional-Integral Controller*).



Gambar 3. Diagram Blok Skema Pengontrolan Keseluruhan

3.1 Pengontrolan Kecepatan Sudut Roda Robot

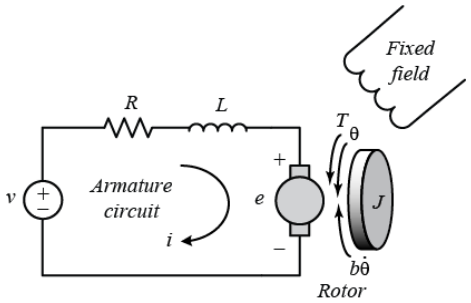
Telah dibuktikan bahwa pengontrolan kecepatan sudut roda kanan dan kiri robot (ω_r dan ω_l) dapat digunakan untuk mengontrol kecepatan linier dan kecepatan sudut robot (v dan ω) yang lebih lanjut dapat mengontrol posisi dan sudut hadap robot (x, y dan θ). Maka dari itu, apabila diketahui orientasi target robot (x_{target}, y_{target}, dan θ_{target}) dapat ditentukan kecepatan sudut roda kanan dan kiri target ($\omega_{rtarget}$ dan $\omega_{ltarget}$), dan apabila sistem kontrol berhasil mengontrol kecepatan sudut roda kanan-kiri robot menuju kecepatan sudut roda kanan-kiri target, maka orientasi robot akan menuju orientasi target. Pada bagian ini akan dibahas pengontrolan motor penggerak roda kanan-kiri pada robot agar tercapai kecepatan sudut roda yang diinginkan menggunakan pengontrol PI (*Proportional-Integral Controller*).

3.1.1 Pemodelan Motor DC

Pertama-tama didefinisikan terlebih dahulu fungsi transfer dari kecepatan sudut roda ω terhadap tegangan masukan motor V dari model motor pada Gambar 4 dengan persamaan *input output* sebagai berikut:

$$\frac{\omega}{V} = \frac{K}{(Js+b)(Ls+R)+K^2} \tag{7}$$

Nilai-nilai parameter sistem bergantung pada motor DC yang digunakan. Pada kasus ini dipilih motor DC pabrikan PITTMAN – *Brush Commutated DC Servo Motors* ID 23000 yang memiliki nilai parameter seperti pada Tabel 1.



Gambar 4. Model Umum Motor DC

Tabel 1 Parameter Motor DC ID23000

Nama	Definisi	Nilai	Satuan
K	Faktor Proporsional Motor	0.055	Nm/A atau V/rad/s
J	Mome Inersia Motor	0.0000268	Kgm^2
b	Konstanta Gesekan	0.0001	Nms
R	Resistansi Internal	1.16	Ohm
L	Induktansi Internal	0.0014	H

Adapun diasumsikan penggunaan motor *driver "Pololu High Power Motor Driver"* dengan batasan arus maksimal 12 Ampere dan batasan tegangan maksimal 24 Volt.

3.1.2 Pengontrol PI (*Proportional Integral Controller*)

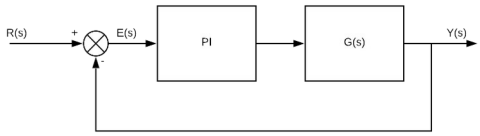
Pengontrol PI umum dapat dinyatakan dalam bentuk

$$PI = K_p\, e(t) + K_i \int e(t)dt \tag{8}$$

Atau apabila dinyatakan sebagai fungsi transfer dapat ditulis sebagai

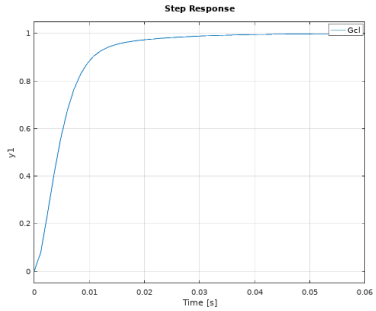
$$PI = K_p + \frac{K_i}{s} \tag{9}$$

Dengan K_p dan K_i merupakan penguatan masing-masing pengontrol proporsional dan integral, serta e(t) merupakan galat nilai luaran terhadap nilai referensi. Pengontrol PI ini dapat diimplementasikan dalam skema pengontrolan kalang tertutup seperti pada Gambar 5.



Gambar 5. Skema Sistem Kontrol PI

Dengan R(s) merupakan nilai referensi yang hendak dicapai yang Y(s) merupakan nilai luaran sistem, atau dalam hal ini kecepatan sudut roda. Apabila pada G(s) digunakan fungsi transfer motor DC serta dipilih nilai K_p = 0.1 dan K_i = 10 diperole respon sistem kalang tertutup terhadap input step seperti terlihat pada Gambar 6.



Gambar 6. *Step Response* Kalang Tertutup dengan PD

Teramati respon sistem yang cukup cepat dengan settling time 0.058800 detik tanpa mengalami *overshoot* maupun galat tunak. Diputuskan bahwa pengontrol PI parameter Kp dan Ki yang ditentukan tadi telah cukup untuk memperoleh respon sistem yang diinginkan tanpa masalah berarti, sehingga pada akhirnya akan dapat diperoleh kecepatan sudut roda kanan-kiri yang diinginkan.

3.2 Pengontrolan Pergerakan Robot

Pengontrolan pergerakan robot dilakukan dengan dua sistem kontrol, yaitu penjejakan lintasan – yang digunakan untuk membawa posisi robot menuju tempat *docking* – dan dilanjutkan dengan sistem kontrol stabilisasi titik – yang digunakan untuk mempersiapkan postur robot agar kelak mudah melakukan pergerakan keluar dari tempat *docking*. Penggunaan sistem kontrol penjejakan lintasan dibanding lajur lebih dipilih karena memberikan kepastian waktu tempuh dan posisi pada setiap waktu tertentu. Diinginkan pengontrolan penjejakan lintasan yang selurus mungkin agar benturan dengan objek sekitar dapat diminimalkan.

3.2.1 Pembangkitan Lintasan

Untuk membangkitkan lintasan yang akan dijejaki oleh robot, dapat digunakan metode polinomial orde tiga [7]. Metode ini mendeskripsikan lintasan pada suatu sumbu terhadap waktu sebagai

$$x(t) = c_0 + c_1(t - t_0) + c_2(t - t_0)^2 + c_3(t - t_0)^3 \tag{10}$$

$$\dot{x}(t) = c_1 + 2c_2(t - t_0) + 3c_3(t - t_0)^2 \tag{11}$$

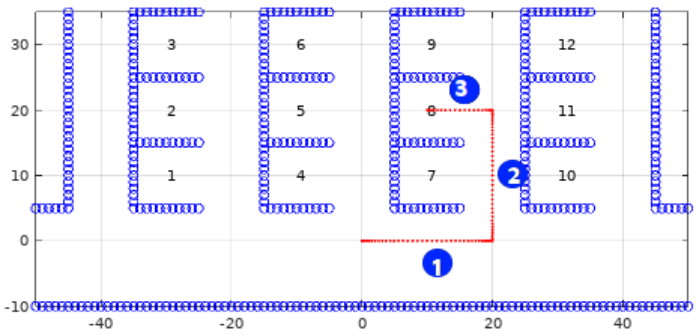
Persamaan ini dapat diselesaikan dengan mengetahui kondisi awal (saat $t = t_0$) dan kondisi akhir (saat $t = t_f$) untuk posisi dan kecepatan. Dipilih kondisi awal dan akhir untuk posisi sebagai x_0 dan x_f, serta diinginkan kecepatan awal maupun akhir $v_0 = v_f = 0$. Sehingga sistem polinomial menjadi

$$\begin{cases} x(t_0) = & c_0 & = x_0 \\ x(t_f) = & c_0 + c_1 t_f + c_2 {t_f}^2 + c_3 {t_f}^3 & = x_f \\ \dot{x}(t_0) = & c_1 & = 0 \\ \dot{x}(t_f) = & c_1 + 2c_2 t_f + 3c_3 {t_f}^2 & = 0 \end{cases} \tag{12}$$

Dengan menyelesaikan sistem tersebut, diperoleh nilai parameter

$$c_0 = x_0\,; \quad c_1 = 0 \quad ; \quad c_2 = -\frac{3}{2} t_f \frac{(x_f - x_0)}{-\frac{{t_f}^3}{2}} \quad ; \quad c_3 = \frac{(x_f - x_0)}{-\frac{{t_f}^3}{2}} \tag{13}$$

Sebuah lintasan dua dimensi dapat didefinisikan dengan dua buah lintasan satu dimensi pada sumbu-sumbu yang terkait dengan pergerakan (sumbu x dan y). Ide inilah yang akan digunakan pada sistem kontrol penjejakan lintasan ini. Pertama-tama didefinisikan tiga buah segmen lintasan seperti pada Gambar 7 – dimana untuk tiap segmen memiliki lintasan x dan y masing-masing. Karena untuk setiap sumbu diketahui kondisi awal dan akhir yang ingin dituju ($[x_0 \quad x_f \quad y_0 \quad y_f \quad t_0 \quad t_f]_i$, i = 1,2,3), maka dengan memilih waktu tempuh setiap segmen yang sesuai, dapat dibentuk persamaan polinomial orde tiga yang merepresentasikan lintasan tiap sumbu untuk setiap segmen.



Gambar 7. Tiga segmen lintasan

Segmen pertama memberikan lintasan untuk robot bergerak dari posisi awal robot menuju posisi bukaan suatu kolom tempat *docking*, segmen kedua memberikan lintasan untuk bergerak dari posisi bukaan kolom *docking* menuju baris *docking* yang dituju, dan segmen ketiga memberikan lintasan untuk robot bergerak masuk ke posisi *docking* yang diinginkan. Pada segmen pertama, kedua, dan ketiga dipilih kondisi-kondisi awal dan akhir seperti terlihat pada Tabel 2.

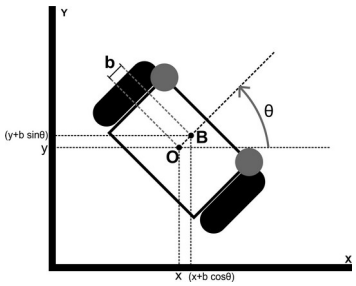
Tabel 2 Kondisi awal-akhir tiap segmen

Segmen	*X awal (x_0)*	*X akhir (x_f)*	*Y awal (-y_0)*	*Y akhir (y_f)*	*t awal (t_0)*	*t akhir (t_f)*
1	x_0	x_{lr}	y_0	*0*	*0*	$\frac{\lvert(x_0 - x_{lr})\rvert}{f_v}$
2	x_{lr}	x_{lr}	*0*	y_{br}	$\frac{\lvert(x_0 - x_{lr})\rvert}{f_v}$	$\frac{y_{br}}{f_v}$
3	x_{lr}	x_{br}	y_{br}	y_{br}	$\frac{y_{br}}{f_v}$	$\frac{\lvert(x_{lr} - x_{br})\rvert}{f_v}$

Dengan x_{lr} merupakan posisi x lorong kolom *docking* yang ingin dituju, x_{br} dan y_{br} merupakan posisi x dan y baris docking yang ingin dituju, serta f_v merupakan faktor kecepatan. Dapat dilihat bahwa pemilihan waktu tempuh setiap segmen lintasan disesuaikan dengan posisi awal robot dan lokasi *docking* yang diinginkan. Pada pengontrolan ini dipilih nilai f_v sebesar 5 agar pergerakan robot tidak memakan waktu terlalu lama.

3.2.2 Penjejakan Lintasan

Telah diperoleh lintasan yang diinginkan melalui pembangkitan lintasan dengan polinom orde tiga (x_{tr} dan y_{tr}) untuk setiap t. Selanjutnya akan dibahas skema pengontrolan agar lintasan yang telah ditetapkan dapat dicapai.



Gambar 8. Pemindahan titik referensi robot beroda differensial dari titik O ke titik B

Literatur [6] mengusulkan pemecahan permasalahan penjejakan lintasan dengang terlebih dahulu melakukan pemindahan titik referensi dari titik O(x,y) ke titik B(x_b,y_b) dengan penetapan jarak b>0 seperti pada Gambar 8. Relasi titik B dengan O ini ditunjukkan sebagai

$$\begin{bmatrix} x_b \\ y_b \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} bcos\theta \\ bsin\theta \end{bmatrix} \tag{14}$$

Sehingga diperoleh persamaan diferensial terhadap waktu untuk (x_b, y_b) terhadap (v,ω) sebagai

$$\begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} = \begin{bmatrix} cos\theta & -bsin\theta \\ sin\theta & bcos\theta \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{15}$$

Apabila ditentukan input kontrol u1 dan u2 sebagai

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} = R \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{16}$$

$$R = \begin{bmatrix} cos\theta & -bsin\theta \\ sin\theta & bcos\theta \end{bmatrix} \tag{17}$$

Maka diperoleh v dan ω robot sebagai

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = R^{-1} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ \frac{-sin\theta}{b} & \frac{cos\theta}{b} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{18}$$

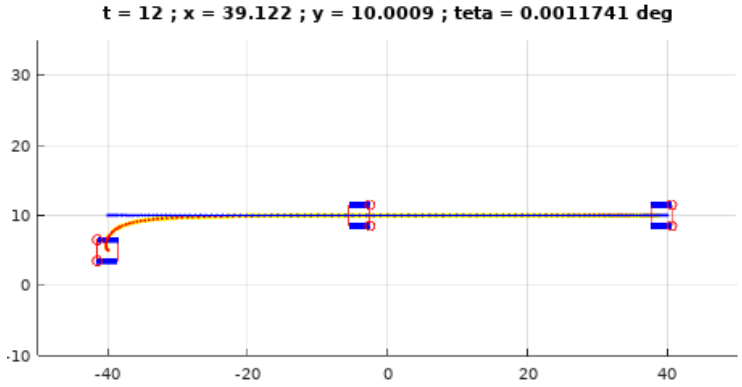
Input kontrol u1 dan u2 selanjutnya dirancang dengan algoritma pengontrol sebagai

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} k_{\dot{x}}\dot{x}_{tr} + k_x(x_{tr} - x_b) \\ k_{\dot{y}}\dot{y}_{tr} + k_y(y_{tr} - y_b) \end{bmatrix} \tag{19}$$

Dengan $k_{\dot{x}}, k_{\dot{x}}, k_x, k_y > 0$. Perlu diingat bahwa ($u_1$,$u_2$) = ($\dot{x}_b$.$\dot{y}_b$) sehingga persamaan di atas dapat ditulis ulang

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} k_{\dot{x}}(\dot{x}_{tr} - \dot{x}_b) + k_x(x_{tr} - x_b) \\ k_{\dot{y}}(\dot{y}_{tr} - \dot{y}_b) + k_y(y_{tr} - y_b) \end{bmatrix} \tag{20}$$

Atau dengan kata lain dapat ditunjukkan bahwa galat (x_b, y_b) dan ($\dot{x}_b, \dot{y}_b$) terhadap lintasan yang diinginkan ($x_{tr}, y_{tr}, \dot{x}_{tr}, \dot{y}_{tr}$) akan menuju O seiring waktu dengan adanya penguatan positif ($k_{\dot{x}}, k_{\dot{x}}, k_x, k_y$). Menggunakan ($k_{\dot{x}}, k_{\dot{x}}, k_x, k_y$) = (1,1,1,1) serta lintasan asal yang telah dibentuk menggunakan polinom orde tiga, dapat diperoleh hasil simulasi seperti pada Gambar 9. Terlihat bahwa robot telah mampu menjejaki lintasan yang diinginkan dengan baik. Selanjutnya akan dibahas mengenai stabilisasi titik untuk menutup galat posisi sekaligus mempersiapkan orientasi robot agar kelak dapat dikeluarkan dengan mudah.

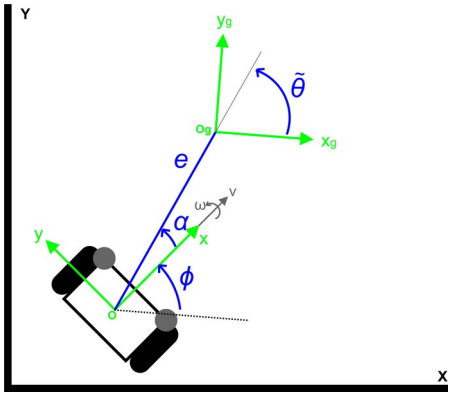


Gambar 9. Simulasi penjejakan lintasan sederhana

3.2.3 Stabilisasi Titik

Pengontrolan stabilisasi titik bertujuan untuk mengontrol posisi WMR dari suatu postur awal O (x_i, y_i, θ_i) menuju postur akhir O_g(x_f, y_f, θ_f). Permasalahan ini dijawab oleh Aicardi, dkk. (1995) [8] dengan terlebih dahulu mentransformasikan sistem yang ada dari representasi

koordinat kartesian menjadi koordinat polar seperti pada Gambar 10. Dengan O_g merupakan kerangka postur tujuan dan ϕ merupakan sudut hadap robot terhadap kerangka postur tujuan.



Gambar 10. Representasi masalah stabilisasi titik dalam koordinat polar

Persamaan kinematika robot sekarang direpresentasikan dengan jarak galat terhadap postur tujuan *e* dan orientasi robot $\tilde{\theta}$ terhadap kerangka tujuan O_g sebagai

$$\begin{aligned} \dot{e} &= -v \cdot cos(\tilde{\theta} - \phi) \\ \dot{\tilde{\theta}} &= v\frac{sin\alpha}{e} \\ \dot{\phi} &= \omega \end{aligned} \tag{21}$$

Dengan memilih $\alpha = \tilde{\theta} - \phi$ sebagai sudut yang dibentuk postur awal terhadap vektor *e*, akan diperoleh

$$\begin{aligned} \dot{e} &= -v \cdot cos\alpha \\ \dot{\alpha} &= -\omega + v\frac{sin\alpha}{e} \\ \dot{\tilde{\theta}} &= v\frac{sin\alpha}{e} \end{aligned} \tag{22}$$

Persamaan ini akan terdefinisi di seluruh titik kecuali saat $e = 0$. Dengan kata lain, algoritma hanya dirancang ketika terdapat galat posisi antara postur awal dengan postur akhir (saat $e \neq 0$). Perancangan algoritma kontrol untuk kecepatan linear *v* dan kecepatan putar ω dapat dilakukan menggunakan langkah sebagai berikut. Pertama, didefinisikan suatu fungsi kandidat *Lyapunov* definit positif sebagai berikut

$$V = V_1 + V_2 = \frac{1}{2}e^2 + \frac{1}{2}(\alpha^2 + h\tilde{\theta}^2); h > 0 \tag{23}$$

Terlebih dahulu akan dirancang algoritma kontrol untuk kecepatan linear *v*, yaitu dengan membuat energi V_1 dapat dituliskan

$$\dot{V_1} = e\dot{e} = -e \cdot v \cdot cos\alpha \tag{24}$$

Kecepatan linier *v* dapat dirancang sebagai

$$v = K_1 \cdot e\, cos\alpha\,, K_1 > 0 \tag{25}$$

Sehingga akan diperoleh nilai $\dot{V}_1$ yaitu

$$\dot{V}_1 = -K_1(e \cdot cos\alpha)^2 \leq 0 \tag{26}$$

Kemudian dilakukan perancangan pengontrol untuk kecepatan putar ω, yaitu dengan membuat energi V_2 meluruh ke nol atau dengan kata lain derivatif V_2 harus kurang dari nol. Derivatif dari fungsi V_2 dapat dituliskan

$$\begin{aligned} \dot{V}_2 &= \alpha\dot{\alpha} + h \cdot \tilde{\theta}\dot{\tilde{\theta}} \\ &= \alpha\left[-\omega + K_1\frac{cos\alpha\, sin\alpha}{\alpha}(\alpha + h\tilde{\theta})\right] \end{aligned} \tag{27}$$

Kecepatan putar ω dapat dirancang sebagai

$$\begin{aligned} \omega = K_2\alpha + K_1\frac{cos\alpha\, sin\alpha}{\alpha}(\alpha + K_3\tilde{\theta}) \\ K_1, K_2 > 0; K_3 = h > 0 \end{aligned} \tag{28}$$

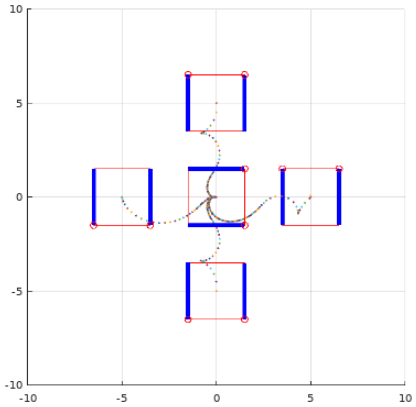
Sehingga akan diperoleh nilai $\dot{V}_2$ yaitu

$$\dot{V}_2 = -K_2\alpha^2 \leq 0 \tag{29}$$

Diperoleh persamaan sistem kontrol untuk stabilisasi titik sebagai

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} K_1\, e\, cos\alpha \\ K_2\, \alpha + K_1\, \frac{cos\alpha\, sin\alpha}{\alpha}(\alpha + K_3\tilde{\theta}) \end{bmatrix} \tag{30}$$

Dengan K_1, K_2 dan K_3 merupakan faktor penguatan umpan balik sistem. Perlu diingat bahwa $\lim_{\alpha\to 0}\frac{sin\alpha}{\alpha} = 1$. Pengontrolan umpan balik yang diusulkan ini akan mampu membawa nilai *e* dan α menuju 0 sehingga diperoleh postur robot yang diinginkan. Hasil simulasi dengan nilai [K_1, K_2, K_3] = [1, 4, 3] terdapat pada Gambar 11. Terlihat bahwa postur robot telah berhasil distabilkan menuju postur target dari berbagai postur awal.



Gambar 11. Hasil simulasi stabilisasi titik

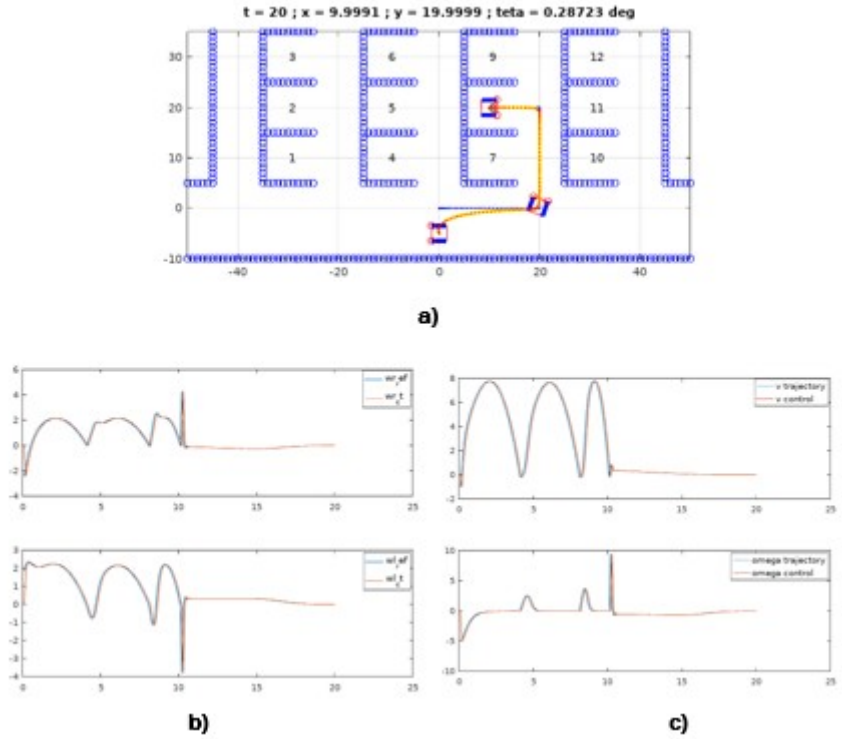
Algoritma kontrol stabilisasi titik ini akan diimplementasikan pada sistem robot utama setelah algoritma penjejakan lintasan berakhir untuk menutup kesalahan posisi dari penjejakan lintasan serta memberikan sudut hadap robot ke arah luar tempat *docking*, sehingga kelak robot siap untuk diperintahkan ke luar dengan mudah. Untuk menjalankan sistem kontrol stabilisasi titik, diberikan pengalokasian waktu hingga 10 detik.

4 Hasil Simulasi

Simulasi dilakukan dengan menggunakan skema pengontrolan penjejakan lintasan dilanjutkan stabilisasi titik dan juga pengontrol PI seperti yang terdapat pada Gambar 3. Sistem kontrol stabilisasi titik dan penjejakan lintasan memberikan kecepatan linier v dan kecepatan sudut ω referensi yang harus dicapai oleh sistem robot utama. Sistem robot kemudian mengejar nilai v dan ω dengan mengatur kecepatan roda kanan dan kiri robot ($\omega_r \, dan \, \omega_l$) menggunakan skema pengontrolan PI. Nilai ω_r dan ω_l yang didapat dari pengotrolan PI kemudian dikonversikan kembali menjadi nilai v dan omega aktual pengontrolan – yang kemudian digunakan untuk memperoleh x, y, dan θ aktual hasil pengontrolan. X, y, dan θ aktual hasil pengontrolan inilah yang akan digunakan pada visualisasi. Terdapat 12 area *docking* yang direncanakan dengan koordinat posisi (x,y) sebagai berikut

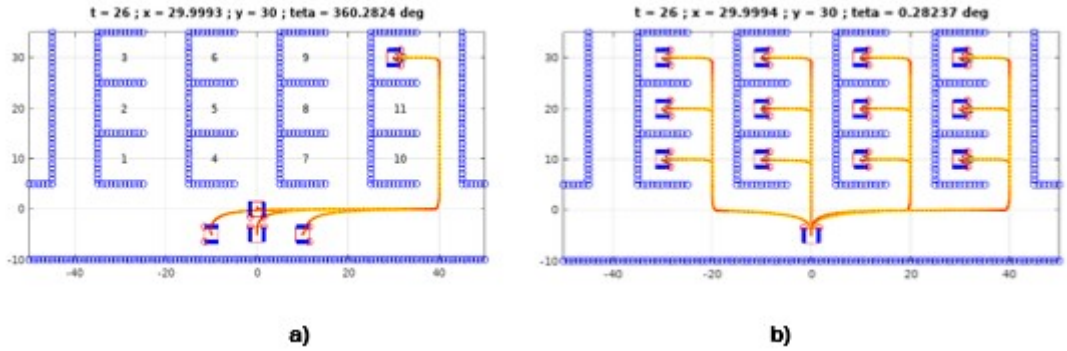
$$\begin{array}{cccc} (-30,30)_3 & (-10,30)_6 & (10,30)_9 & (30,30)_{12} \\ (-30,20)_2 & (-10,20)_5 & (10,20)_8 & (30,20)_{11} \\ (-30,10)_1 & (-10,10)_4 & (10,10)_7 & (30,10)_{10} \end{array}$$

Diinginkan orientasi sudut hadap akhir robot 0º atau menghadap arah kanan. Menggunakan pengontrol PI dan lintasan 3 segmen yang telah didefinisikan serta skema pengontrolan penjejakan lintasan dilanjutkan skema stabilisasi titik dengan parameter-parameter pengontrol yang telah ditetapkan sebelumnya, apabila dipilih suatu slot *docking* asal, dapat diperoleh hasil simulasi pengontrolan sebagai berikut



Gambar 12. Hasil simulasi sistem pengontrolan keseluruhan: a) Visualisasi pergerakan robot; b) ω_r dan ω_l lintasan vs pengontrol PI; c) v dan ω lintasan vs pengontrol

Gambar 12 menunjukkan hasil simulasi dengan sistem pengontrol keseluruhan. Dapat terlihat bahwa keberhasilan pengontrol PI untuk menggerakkan kecepatan roda kiri-kanan kontrol (warna merah) sesuai ω_r dan ω_l target (warna biru) pada Gambar 12b telah berhasil mengejar v dan ω dari sistem kontrol (Gambar 12c) dan pada akhirnya membawa robot menuju posisi *docking* dengan postur akhir yang diinginkan (Gambar 12a). Selanjutnya akan disimulasikan sistem tadi untuk berbagai posisi *docking* serta dari berbagai posisi maupun orientasi awal. Hal ini dilakukan untuk mengecek kestabilan sistem kontrol yang telah dibuat. Hasil simulasi terdapat pada Gambar 13. Teramati bahwa sistem mampu menangani variasi postur dan posisi awal robot (Gambar 13a) maupun variasi posisi *docking* tujuan (Gambar 13b). Dengan ini dapat dikatakan bahwa simulasi pengontrolan telah berhasil mencapai tujuan yang ditentukan – yaitu mencapai posisi *docking* yang dipilih dengan postur menghadap kanan.



Gambar 13. Simulasi pengecekan kestabilan sistem untuk: a) variasi postur awal; b) variasi posisi *docking* tujuan

5 Kesimpulan

Makalah ini menyajikan perancangan sistem kontrol untuk pemodelan mekanisme *docking* suatu robot beroda untuk keperluan *smart warehouse*. Sistem kontrol yang diajukan terbagi menjadi dua bagian kalang tertutup. Bagian pertama merupakan perancangan pergerakan robot menggunakan sistem kontrol penjejakan lintasan dan dilanjutkan stabilisasi titik. Penjejakan lintasan dilakukan dalam tiga segmen *piecewise* yang dibangkitkan dengan polinomial orde tiga untuk membawa posisi robot menuju lokasi *docking*, sedangkan stabilisasi titik dilakukan untuk menutup galat posisi dan mempersiapkan postu robot keluar lokasi *docking*. Bagian kedua merupakan perancangan penggerakan roda kanan-kiri robot menggunakan pengontrol PI. Menggunakan sistem kontrol tersebut dengan parameter-parameter yang telah diajukan, hasil simulasi menunjukkan robot dapat mencapai berbagai lokasi *docking* terdefinisi dari berbagai variasi postur awal.

Referensi

[1] Singh, B., Sellappan, N., & Kumaradhas, P., *Evolution of Industrial Robots and their Applications.* International Journal of Emerging Technology and Advanced Engineering, 3(5), 763-768, 2013.

[2] Kamali, A., *Smart Warehouse vs. Traditional Warehouse – Review.* International Journal of Automation and Autonomous System, 11(1), 9-16, 2019.

[3] Bhasin, K., Clark, P. (2016, June 29), *How Amazon Triggered a Robot Arms Race*. https://www.bloomberg.com/news/articles/2016-06-29/how-amazon-triggered-a-robot-arms-race , Retreived May 23, 2019.

[4] Tampubolon, M., Pamungkas, L., Chiu, H,-J., & Hsieh, Y.-C., *Dynamic Wireless Power Transfer for Logistic Robots*. Energies, 11(3), 527. Dio:10.3390/en11030527, 2018.

[5] Spong, M. W., Hutchinson, S., & Vidyasagar, M., *Robot modelling and control* (1st ed.). John Wiley & Sons, 2005.

[6] Widyotriatmo, A. Dasar-dasar Mekatronika dan Robotika. Bandung, Jawa Barat: ITB Press, 2018

[7] Niku, S. B., *Introduction to robotics: Analysis, control, applications* (2nd ed.). John Wiley & Sons, 2011.

[8] Aicardi, M., Casalino, G., Bicchi, A., & Balestrino, A. *Closed loop steering of unicycle like vehicles via Lyapunov techniques*. IEEE Robotics and Automation Magazine, 2(1), 27-35, March 1995.