Perancangan Jaringan Saraf Tiruan untuk Menyelesaikan Kinematika Balik Manipulator Robot Denso 6-DoF

Zulkifli Hidayat[*1], Mochamad Sahal[2], Yusuf Bilfaqih[3], Rusdhianto EAK[4], Daniel Cristover Sirait[5]

[12345]Departemen Teknik Elektro, Institut Teknologi Sepuluh Nopember
Kampus ITS Sukolilo, Surabaya 60111
[1]zulkifli@ee.its.ac.id

Intisari — **Pada artikel ini, dibahas tentang kinematika balik (inverse kinematics) untuk manipulator robot denso 6-DoF. Kinematika balik adalah metode menentukan konfigurasi sendi dengan mengetahui posisi end-effector yang diinginkan. Kesulitan dari kinematika balik adalah penyelesaiannya yang tidak tunggal. Untuk satu titik end-effector, ada banyak konfigurasi sendi yang mungkin sehingga tidak ada penyelesaian yang unik. Untuk menyelesaikan kinematika balik ini, diusulkan penyelesaian menggunakan jaringan saraf tiruan untuk mendapatkan salah satu penyelesaian berupa posisi dan orientasi dari end-effector.**
Kata kunci — **Kinematika balik, kinematika maju, jaringan saraf buatan, lengan robot**

Abstract — **In this article, inverse kinematics problem and solution of the 6-DoF denso manipulator is presented. Inverse kinematics is a method to compute the configuration of joins of a manipulator given the position and orientation of the end-effector. For a given end-effector position and orientation, the configuration is not unique as there are many configurations are possible. Here, the use of artifical neural network is proposed to solve the inverse kinematics problem.**
Keywords — **Inverse kinematics, forward kinematics, artificial neural networks, robot manipulator.**

I. PENDAHULUAN

Manipulator robot berperan penting dalam industri, khususnya dalam perakitan produk elektronik atau mekanik. Manipulator robot adalah sistem elektromekanik yang dirancang berfungsi seperti lengan manusia. Karena itu, manipulator robot umum juga disebut dengan lengan robot. Lengan robot umum digunakan pada industrik manufaktur karena kemampuannya untuk melakukan suatu pekerjaan yang berulang dengan tingkat ketelitian yang tinggi, misalnya untuk pengelasan badan mobil atau penyolderan komponen elektronika.

Lengan robot terdiri dari beberapa sendi rotasi atau linier yang masing-masing diprogram untuk memiliki nilai tertentu sehingga ujung dari lengan robot berada pada posisi dan orientasi tertentu, dengan kata lai. Posisi setiap saat *end-effector* dapat dihitung dari konfigurasi posisi semua sendi manipulator. Analisis ini disebut dengan kinematika maju (*forward kinematics*)[1].

Sebaliknya jika diketahui posisi *end-effector*, konfigurasi semua sendi manipulator memiliki banyak kemungkinan. Dengan kata lain, konfigurasi sendi robot tidak unik ketika diinginkan posisi *end-effector* tertentu. Problem ini disebut dengan kinematika balik (*inverse kinematics*)[1].

Persoalan kinematika balik masih menjadi topik yang mendapat perhatian sampai saat ini. [2] mendapatkan penyelesaian kinematika balik robot denso dalam bentuk tertutup. [3] meneliti bentuk tertutup dari solusi kinematika maju dan balik dari manipulator denso. [4] menggunakan pembelajaran mesin (machine learning) untuk menyelesaikan problem kinematika balik. [5] menggunakan convolutional neural netoworks untuk menyelesaikan kinematika balik robot dengan 7 derajat kebebasan. [6] mendapatkan formulasi kinematika balik secara analitik untuk manipulator redundan.

Persoalan kinematika balik masih menjadi topik yang mendapat perhatian sampai saat ini. [2] mendapatkan penyelesaian kinematika balik robot denso dalam bentuk tertutup. [3] meneliti bentuk tertutup dari solusi kinematika maju dan balik dari manipulator denso. [4] menggunakan pembelajaran mesin (machine learning) untuk menyelesaikan problem kinematika balik. [5] menggunakan convolutional neural netoworks untuk menyelesaikan kinematika balik robot dengan 7 derajat kebebasan. [6] mendapatkan

formulasi kinematika balik secara analitik untuk manipulator redundan.

Pada artikel ini dibahas penyelesaian kinematika balik dari robot denso dengan menggunakan metode Jaringan Saraf Tiruan (JST) atau *neural networks* dengan metode pembelajaran propagasi balik. Pembelajaran dilakukan menggunakan model robot denso secara simulasi.

Sistematika dari artikel ini adalah: setelah pengantar pada Seksi 1, pada Seksi 2 dibahas mengenai model matematika robot denso dan dilanjutkan dengan pembahasan tentang jaringan saraf tiruan pada Seksi 3. Seksi 4 menjelaskan metode penyelesaian kinematika balik dengan menggunakan jaringan saraf tiruan. Hasil simulasi dan analisis ditampilkan pada Seksi 5 dan ditutup dengan kesimpulan pada Seksi 6.

II. MODEL ROBOT DENSO

Robot denso yang digunakan merupakan robot denso 6-DoF VP-6242G. Robot tipe ini hanya menggunakan enam sendi rotasi. Posisi dan orientasi robot ditentukan oleh masing-masing tiga sendi yang berbeda. Posisi ditentukan oleh tiga sendi terbawah dan orientasi ditentukan oleh tiga sendi di atasnya. Ilustrasi robot denso 6-DoF VP-6242G ditampilkan pada Gambar 1.

Gerakan pada robot terdiri dari gerakan posisi dan orientasi. Deskripsi dari gerakan ini dinyatakan dengan menggunakan transformasi homogen berupa matriks yang terdiri dari bagian rotasi, translasi, perspektif, dan skala. Matriks transformasi homogen yang dinyatakan oleh Persamaan (1) ini, digunakan pada perhitungan kinematika maju [1]:

$$H_1^0 = \left[\begin{array}{c|c} R_{3x3} & P_{3x1} \\ \hline f_{1x3} & S_{1x1} \end{array}\right] = \left[\begin{array}{c|c} Rotation & Translation \\ \hline Perspective & Scale \end{array}\right] \quad (1)$$

Persamaan kinematika maju didapatkan dengan menetapkan paramater Denavit-Hartenberg (DH). Parameter DH terdiri dari menjadi θ_i (sudut sendi), α_i (putaran *link*), a_i (panjang *link*), dan d_i (*link offset*). Untuk manipulator robot Denso 6-DoF, parameter-parameter DH-nya ditampilkan pada Tabel 1 dan 2.

Persamaan kinematika maju dihitung menggunakan Persamaan (2) yang menghasilkan matriks transformasi berupa posisi dan orientasi *end-effector*.

$$T_0^6 = A_1A_2A_3A_4A_5A_6 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Transformasi homogen A_i adalah kombinasi dari empat transformasi dasar yang dinyatakan dengan perkalian dari tiap-tiap matriks transformasi. Transformasi ini dapat dilihat pada Persamaan (3), dimana c = cos dan s = sin.

$$A_i = Ro_{z,\theta_i}\ Tr_{z,d_i}\ Tr_{x,a_i}\ Ro_{x,\alpha_i} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_ic_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -s_{\theta_i}s_{\alpha_i} & a_is_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

Pada matriks transformasi (2), variabel *n*, *s*, dan *a* menyatakan orientasi *end-effector* dan nilai *d* menyatakan posisi x, y, dan z. Nilai *roll*, *pitch*, dan *yaw* dari matriks orientasi didapatkan (4) - (6)

$$pitch = \sin^{-1}(a_x) \quad (4)$$

$$roll = \sin^{-1}\left(\frac{s_x}{-\cos(pitch)}\right) \quad (5)$$

$$yaw = -\sin^{-1}\left(\frac{a_y}{\cos(pitch)}\right) \quad (6)$$

III. JARINGAN SARAF TIRUAN

Untuk penyelesaian kinematika balik di artikel ini, digunakan *multilayer perceptron* (MLP). MLP terdiri dari *tiga* lapisan: masukan, lapisan *hidden*, dan lapisan keluaran. Lapisan masukan adalah lapisan yang langsung mendapatkan sinyal dari variabel masukan. Lapisan hidden merupakan lapisan transformasi yang keluarannya dikirim ke lapisan keluaran menjadi keluaran sistem. Secara prinsip, JST adalah suatu sistem kotak hitam yang perilakunya ditentukan oleh nilai bobot antar neuron dan fungsi aktifasinya.

Pembelajaran JST terdiri dari dua tahap: tahap maju atau feedforward dan tahap perbaikan bobot penghubung neuron atau *backward* berdasarkan kesalahan atau selisih antara keluaran JST dan data.

Pada tahap feedforward, lapisan masukan mendapat masukan berupada posisi dan

orientasi dari end-effector, yaitu variabel *x,y,z* dan orientasi *roll* dan *pitch*. Dengan kata lain ada lima neuron pada lapisan masukan. Keluaran dari JST adalah nilai dari sendi-sendi dari lengan robot.

Tabel 1. Parameter Denavit-Hartenberg

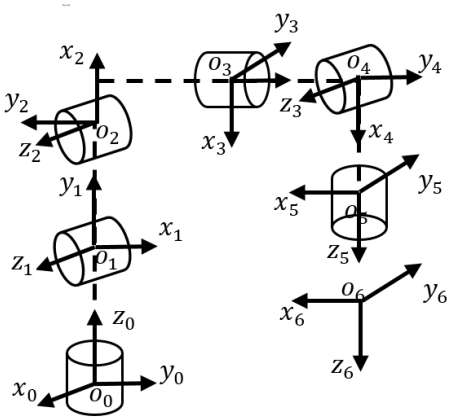
Sendi	a_i	d_i	α_i,	θ_i	*Range (deg)*
1	0	280	90	θ_1^*	-160 s/d 160
2	210	0	0	θ_2^*	-120 s/d 120
3	75	0	-90	θ_3^*	20 s/d 160
4	0	210	90	θ_4^*	-160 s/d 160
5	0	0	-90	θ_5^*	-120 s/d 120
6	0	70	0	θ_6^*	-360 s/d 360

Tabel 2. Penyederhanaan Parameter DH

Sendi	a_i	d_i	α_i,	θ_i	Range (deg)
1	0	d_1	90	θ_1^*	-160 s/d 160
2	a_2	0	0	θ_2^*	-120 s/d 120
3	a_3	0	-90	θ_3^*	20 s/d 160
4	0	d_4	90	θ_4^*	-160 s/d 160
5	0	0	-90	θ_5^*	-120 s/d 120
6	0	d_6	0	θ_6^*	-360 s/d 360

Langkah-langkah untuk tahap *feedforward* adalah sebagai berikut [7]:

1. Nilai awal bobot adalah 0 untuk semua bobot penghubung
2. Setiap neuron masukan ($x_i, i = 1, \dots, 5$) menerima masukan nilai orientasi *roll* dan *pitch*, dan posisi *x,y,z*. Data masukan dipropagasi ke lapisan *hidden* melalui link-link berbobot antar neuron.
3. Menghitung keluaran dari setiap neuron pada lapisan hidden. Pada setiap neuron, sinyal berbobot dari lapisan sebelumnya dijumlahkan:
   $Z_{in_j} = \sum_{i=1}^{n} x_i v_{ij}$ (7)
   Jumlahan berbobot tersebut menjadi masukan pada fungsi aktivasi *sigmoid*:
   $z_j = f(Z\_in_j)$
   $z_j = \frac{1}{1+e^{-(Z\_in_j)}}$ (8)
4. Keluaran dari lapisan hidden menjadi masukan berbobot pada lapisan keluaran. Pada tiap neuron lapisan keluaran, nilai yang masuk dijumlahkan
   $y\_in_k \sum_{j=1}^{p} z_j w_{jk}$ (9)
   dan menjadi masukan pada fungsi aktivasi *linier*.
   $y_k = f(y\_in_k)$
   $y_k = y\_in_k$ (10)



Gbr 2.Pemberian Label Sumbu o_n, x_n, y_n, z_n

Keluaran dari tahap feedforward adalah nilai-nilai sudut dari semua sendi lengan denso. Jika nilai-nilai besar sudut ini tidak sesuai dengan posisi dan orientasi yang diingikan, maka dilakukan proses belajar yang disebut dengan *backpropagation*.

Pembelajaran dilakukan dengan memperbaiki bobot semua link penghubung neuron berdasarkan kesalahan atau selisih antara posisi dan orientasi yang seharusnya dengan posisi dan orientasi yang didapat dari keluaran JST. Langkah-langkah perhitungan *backpropagation* adalah sebagai berikut [7]:

1. Untuk setiap keluaran ($y_k, k = 1, \dots, 5$) yang merupakan besar sudut $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$, dan dengan nilai θ_6 sama dengan nol, dihitung kinematika maju sehingga didapatkan keluaran dari JST berupa $roll_n, pitch_n, x_n, y_n, z_n$.
2. Menghitung *gradient error* sendi satu sampai sendi lima
   $\delta_k = (t_k - y_k)$ (11
3. Formulasi target θ_1 sampai θ_5 yang diharapkan:
   $t_1 = tan^{-1}\left(\frac{y}{x}\right)$ (12)
   $t_2 = \sqrt{x^2 + y^2}$ (13)
   $t_3 = z$ (14)
   $t_4 = roll$ (15)
   $t_5 = pitch$ (16)
   Formulasi target θ_1 sampai θ_5 keluaran JST
   $t_1 = tan^{-1}\left(\frac{y_n}{x_n}\right)$ (17)
   Mendapatkan nilai perbaikan bobot yang digunakan untuk mengkoreksi bobot w_{jk}:
   $\Delta w_{jk} = \alpha \delta_k z_j$ (22)
4. Setiap pembobot link neuron pada lapisan keluaran dengan neuron pada lapisan *hidden* dikalikan dengan delta lalu

dijumlahkan, dan nilainya menjadi masukan.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_j w_{jk} \quad (23)$$

Nilai error didapatkan dari persamaan diatas, dikalikan dengan turunan dari fungsi aktivasinya

$$\delta_j = \delta\_in_j f'(y\_in_j)$$

$$\delta_j = \delta\_in_j z_j(1 - z_j) \quad (24)$$

Nilai error diatas digunakan untuk menghitung nilai koreksi bobot dan selanjutnya digunakan untuk memperbarui nilai bobot v_{ij}:

$$\Delta v_{ij} = \alpha \delta_j x_i \quad (25)$$

5. Setiap neuron keluaran $(y_k, k = 1, \dots, m)$ memperbarui nilai bias dan bobot $(j, k = 1, \dots p)$:

$$w_{jk}(new) = w_{oj}(old) + \Delta w_{jk} \quad (26)$$

Setiap neuron tersembunyi $(z_j, j = 1, \dots, m)$ memperbarui nilai bias dan bobot ($k = 0, \dots n$)

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \quad (27)$$

IV. KINEMATIKA BALIK DENGAN JARINGAN SARAH TIRUAN

JST untuk kinematika balik menggunakan lima masukan, (x_1, x_2, x_3, x_4, x_5) , yang merepresentasikan variabel masukan x, y, z, *roll*, dan *pitch* dari manipulator robot. Banyaknya neuron pada lapisan *hidden* adalah 50. Nilai ini bukan nilai yang pasti, dan ditentukan secara coba-coba. Pada kasus ini, nilai yang dipilih sudah cukup untuk menyelesaikan masalah kinematika balik lengan robot denso. Fungsi aktifasi yang digunakan pada lapisan ini adalah sigmoid biner.

Lapisan keluaran memiliki 5 keluaran $(z_1, z_2, z_3, z_4, z_5)$, yang merupakan nilai sudut untuk masing-masing sendi, mulai dari sendi 1 hingga sendi 5 pada manipulator robot. Fungsi aktivasi yang digunakan pada lapisan ini adalah fungsi aktivasi *linier*. Struktur dari JST untuk menyelesaikan kinematika balik dari manipulator robot dapat dilihat pada Gambar 3.

Sebelum melakukan simulasi penyelesaian kinematika balik dengan JST, dilakukan simulasi kinematika maju untuk mendapatkan data pembelajaran. Pada progam kinematika maju dimasukkan nilai acak seperti pada Tabel 3 sehingga didapatkan orientasi *roll*, *pitch*, dan *yaw*, serta posisi *x*, *y*, dan *z*.

Tabel 3. Masukan Kinematika Maju

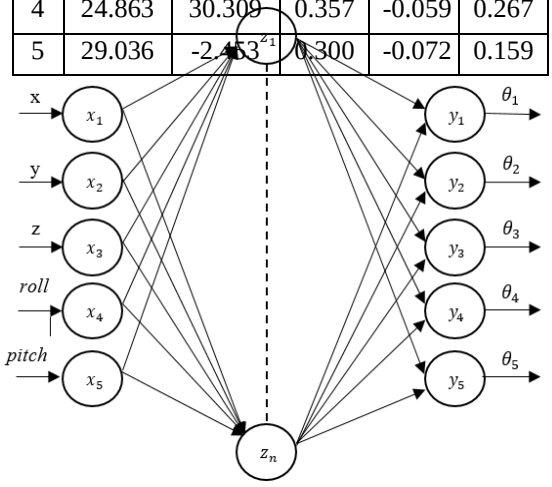
No	Besar sudut sendi ($derajat$)					
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	180	140	30	20	20	0
2	185	145	25	10	30	0
3	175	135	15	10	15	0
4	160	120	10	10	5	0
5	170	130	10	15	10	0
6	165	145	15	15	25	0

Tabel 4: Keluaran Kinematika Maju

No	Orientasi ($derajat$)			Posisi (m)		
	$roll$	$pitch$	yaw	x	y	z
1	19.928	-8.820	-173.201	0.260	0.008	0.152
2	5.158	-19.927	-176.503	0.257	0.028	0.140
3	14.145	15.375	-178.697	0.335	-0.026	0.216
4	36.535	42.100	162.110	0.342	-0.123	0.334
5	24.863	30.309	177.121	0.357	-0.059	0.267
6	29.036	-2.453	-172.829	0.300	-0.072	0.159

Tabel 5. Posisi dan Orientasi Keluaran 50 *Neuron* Lapisan *Hidden*

No	Orientasi ($derajat$)		Posisi (m)		
	$roll$	$pitch$	x	y	z
1	5.158	-9.927	0.257	0.028	0.140
2	14.145	15.375	0.335	-0.026	0.216
3	36.535	42.100	0.342	-0.123	0.334
4	24.863	30.309	0.357	-0.059	0.267
5	29.036	-2.453	0.300	-0.072	0.159



Gbr 3. Struktur Kinematika Balik Dengan JST

Data orientasi dan posisi *end-effector* yang didapatkan dari kinematika maju menjadi masukan pada kinematika balik dengan JST. Dalam pengujian kinematika balik juga terdapat pengujian manipulator robot denso untuk membuat kurva segitiga dan lingkaran.

Tabel 6. *Error* Posisi dan Orientasi 50 *Neuron* Lapisan *Hidden*

No	Orientasi (*derajat*)		Posisi (m)		
	roll	***pitch***	***x***	***y***	***z***
1	2.55e-05	2.94e-05	2.35e-06	1.84e-07	4.82e-05
2	-3.13e-05	-2.73e-05	7.38e-07	2.09e-07	-1.06e-05
3	2.12e-05	1.42e-05	1.14e-06	-2.82e-07	-3.28e-06
4	-2.70e-05	-4.99e-05	1.81e-06	-4.99e-07	3.34e-06
5	1.05e-05	2.68e-06	1.32e-06	-4.97e-07	2.99e-05
RMSE	2.42e-05	2.93e-05	1.57e-06	3.61e-07	2.59e-05

V. SIMULASI DAN ANALISIS

Pada bagian ini dijelaskan tentang simulasi dari program kinematika maju dan kinematika balik menggunakan JST. Simulasi dilakukan menggunakan Matlab. Pengujian pertama adalah pengujian kinematika maju dengan menggunakan nilai sudut tiap sendi dari manipulator robot denso seperti pada Tabel 3 pada kinematika maju. Dari tahap kinematika maju didapatkan keluaran posisi dan orientasi lengan robot yang ditunjukkan pada Tabel 4. Keluaran posisi dan orientasi dari kinematika maju ini digunakan menjadi data pembelajaran untuk penyelesaian kinematika balik dengan JST.

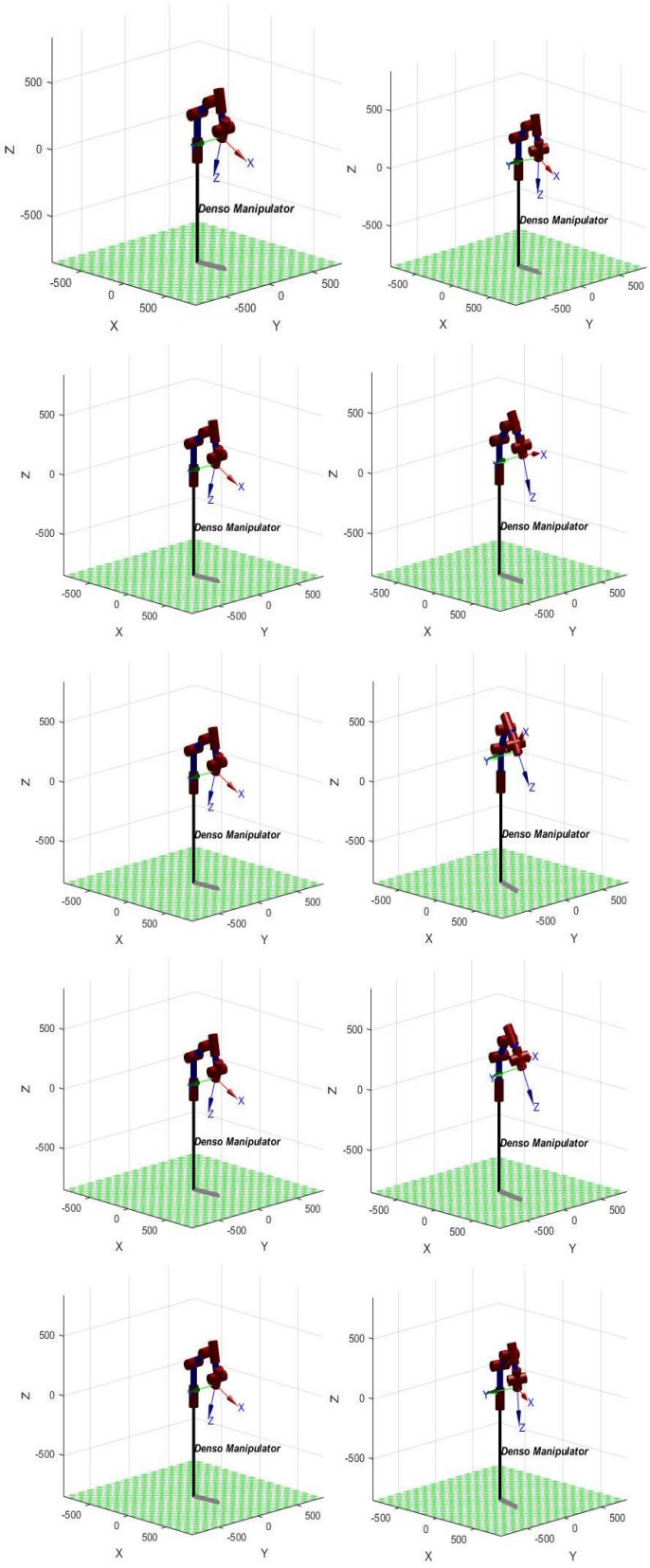
Selanjutnya dilakukan pengujian kinematika balik dengan JST. Hasil keluaran kinematika maju yang telah didapatkan melalui pengujian kinematika maju dijadikan data uji, atau masukan pada program kinematika balik, dan dilihat besar error antara data uji dengan keluaran JST. Pengujian kinematika balik dengan JST dilakukan dengan 50 *neuron* lapisan *hidden*, *learning rate* 0.00025 dan ketelitian sebesar 0.00005. Data posisi dan orientasi manipulator dari keluaran dari JST dapat dilihat pada Tabel 5. Sedangkan data *error* antara orientasi dan posisi yang diinginkan dengan orientasi dan posisi keluaran JST dapat dilihat pada Tabel 6. Titik berangkat manipulator robot adalah dari data uji nomor 1 dengan orientasi [*roll, pitch, yaw*] = [19.9289 -8.8202 -173.201] dan posisi [*x,y,z*] = [0.2604 0.0081 0.1525], dengan besar sudut masing-masing sendi [$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$] = [180 140 30 20 20 0]. Manipulator robot akan bergerak menuju masing-masing data uji, mulai data uji 2 hingga data uji 6 yang terdapat pada Tabel 4. Rata-rata iterasi dan lama waktu melakukan pembelajaran berturut-turut adalah 43162 dan 2227 detik.

Selanjutnya adalah membuat plot masing-masing data pengujian yang dapat dilihat pada Gambar 4. Pada gambar tersebut, gambar sebelah kiri adalah bentuk manipulator robot pada posisi dan orientasi awal, sedangkan gambar sebelah kanan adalah bentuk manipulator robot pada posisi dan orientasi masing-masing data uji mulai dari data uji yang pertama hingga data uji yang kelima.

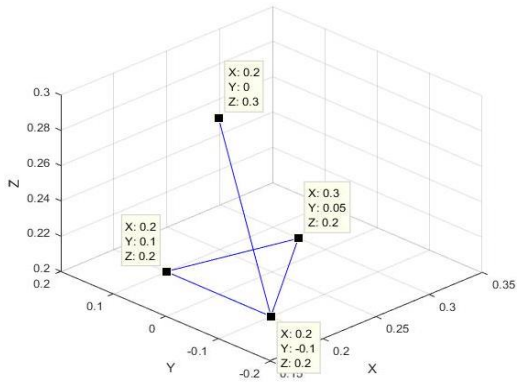
Pengujian selanjutnya adalah uji gerakan end-effector manipulator denso untuk membuat kurva segitiga dengan nilai *roll* dan *pitch* yang konstan dan posisi *x*, *y*, dan *z* nya yang berubah. Manipulator robot denso bergerak dari orientasi [*roll pitch*] = [0 0], dan posisi [*x,y,z*] = [0.2 0 0.3], dengan sudut [$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$] = [180 93.23 48.87 0 37.88 0] menuju konfigurasi titik-titik tujuan dalam bentuk kurva segitiga seperti pada Gambar 5.

Plot kurva segitiga keluaran dari kinematika balik dengan JST ditunjukkanpada Gambar 6. Titik-titik kotak pada gambar merupakan lintasan pergerakan manipulator robot denso saat membentuk kurva segitiga pada bidang datar.

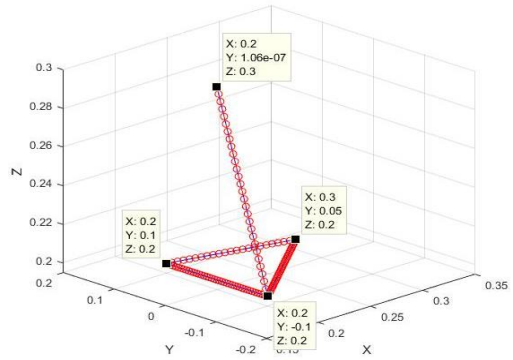
Pengujian yang berikutnya adalah membentuk kurva lingkaran dengan masukan orientasi *roll* dan *pitch* serta posisi *x*, *y*, dan *z*. Serupa dengan pengujian sebelumnya, pada pembuatan kurva lingkaran ini orientasi *roll* dan *pitch* dibuat konstan [0 0] dengan titik *x*, *y*,dan *z* bervariasi. Dalam proses pembuatan kurva lingkaran ini manipulator bergerak dari orientasi awal [*roll pitch*] = [0 0], dan posisi [*x,y,z*] = [0.2 0 0.2], dengan sudut [$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$] = [180 117.06 54.65 0 8.28]. Untuk pengujian ini, kurva lingkatan dibuat pada bidang datar yang dibagi menjadi 36 titik seperti pada Gambar 7.



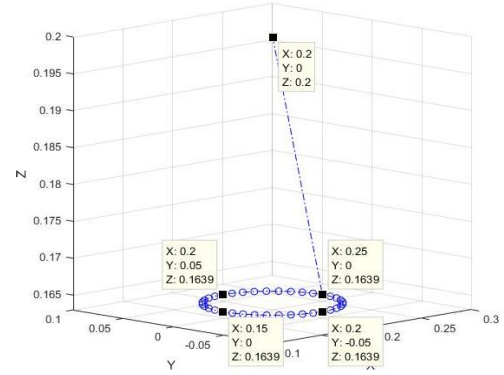
Gbr 4. Bentuk Manipulator Robot Denso pada masing-masing data uji



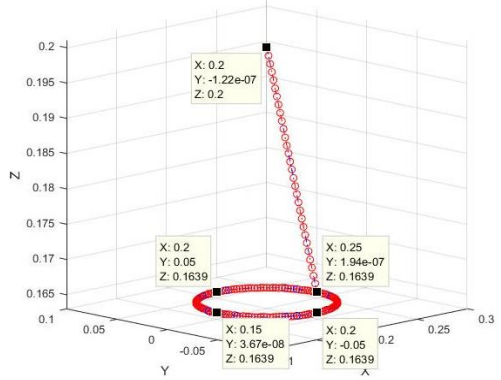
Gbr 5. Titik-Titik Pada Kurva Segitiga untuk Pengujian JST



Gambar 6.. Titik-Titik Pada Kurva Segitiga dari Keluaran JST



Gbr 7. Titik –Titik Pola Lingkaran Bidang Datar



Gbr 8. Titik –Titik Pola Lingkaran Keluaran JST

VI. KESIMPULAN DAN SARAN

Pada artikel ini telah dipaparkan metode penyelesaian kinematika balik menggunakan jaringan saraf tiruan dengan pembelajaran *back-propagation* yang diaplikasikan pada manipulator robot denso. Metode jaringan saraf tiruan ini mampu membuat model kinematika balik dari manipulator robot denso dan menemukan titik-titik data uji dengan benar secara simulasi. Pada simulasi, model kinematika balik dapat membuat pola kurva segitiga dan lingkaran dengan masukan *roll*, *pitch*, *x*, *y*, dan *z*, dengan rata-rata *error* orientasi dibawah 0.00005 derajat dan *error* posisi dibawah 0.00005 meter.

Untuk pengembangan selanjutnya akan dilakukan implementasi pada robot yang sebenarnya. Pengujian untuk gerakan yang relevan dengan aplikasi, misalnya pengelasan, juga menjadi bagian dari pengembangan penelitian ini. Selanjutnya juga akan diteliti pemilihan model JST yang optimal: banyaknya neuron hidden and learning rate, untuk manipulator denso dan manipulator-manipulator jenis lainnya. Konfigurasi JST yang adapatif juga menjadi rencana penelitian berikutnya.

REFERENSI

[1] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. John Wiley & Sons, 2020.

[2] I. E. Prasetia and T. Agustinah, “Inverse Kinematics With Closed Form Solution For Denso Robot Manipulator,” *JTITS*, vol. 4, no. 1, pp. A77–A82, Mar. 2015.

[3] M. E. Kütük, M. T. Daş, and L. C. Dülger, “Forward and Inverse Kinematics Analysis of Denso Robot,” presented at the The Second International Symposium Of Mechanism and Machine Science, Azerbaijan, 2017, p. 8.

[4] F. Aggogeri, N. Pellegrini, C. Taesi, and F. L. Tagliani, “Inverse kinematic solver based on machine learning sequential procedure for robotic applications,” *J. Phys.: Conf. Ser.*, vol. 2234, no. 1, p. 012007, Apr. 2022.

[5] H. A. Elkholy, A. S. Shahin, A. W. Shaarawy, H. Marzouk, and M. Elsamanty, “Solving Inverse Kinematics of a 7-DOF Manipulator Using Convolutional Neural Network,” in *Proceedings of the*

International Conference on Artificial Intelligence and Computer Vision (AICV2020), Cham, 2020, pp. 343–352.

[6] D. R. Isenberg, "Analytical Inverse Kinematic Solutions for End-Effector Positioning and Pointing Applied to Revolute Manipulators with Spherical Wrists," in *SoutheastCon 2021*, Mar. 2021, pp. 1–8.

[7] S. S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Prentice Hall, 2009.