Implementasi Inverse Kinematics Untuk Pola Jalan Robot Penari Humanoid

Riyan Fadlillah
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
riyanfadlillah@student.telkomuniversity.ac.id

Angga Rusdinar
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
anggarusdinar@telkomuniversity.ac.id

Simon Siregar
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
siregar@telkomuniversity.ac.id

***Abstrak*— Kontes Robot Seni Tari Indonesia (KRSTI) adalah ajang kompetisi perancangan, pembuatan, dan pemrograman robot yang memfokuskan pada penggunaan unsur-unsur seni dan budaya Indonesia dalam pembuatan robot humanoid. Dalam kontes ini, kinematika memainkan peran penting dalam analisis gerak, perhitungan, dan kontrol robot. Namun, sistem kontrol robot Badaya_SAS yang sebelumnya menggunakan software R+ Motion tidak kompatibel lagi dengan konfigurasi aktuator terbaru yang ada pada robot, sehingga diperlukan solusi alternatif untuk menjalankan sistem kontrol inverse kinematics dalam membuat pola jalan robot. Berdasarkan masalah tersebut, dibuat sebuah desain sistem kontrol inverse kinematics yang menggunakan Robot Operating System (ROS) sebagai framework. ROS memiliki banyak library dan modul yang tersedia untuk memecahkan masalah perhitungan inverse kinematics dan memastikan bahwa solusi yang diterapkan dapat diterima dengan baik oleh robot. Framework ini juga dirancang untuk menjadi mudah dikembangkan dan diprogram untuk fitur-fitur tambahan. Hasil dari penelitian ini adalah sebuah rancangan sistem kontrol inverse kinematics pada bagian kaki robot humanoid menggunakan kinematic solver untuk membuat pola jalan robot. Framework sistem kontrol juga dirancang menggunakan ROS yang dapat diterapkan dan dikembangkan secara modular untuk fitur-fitur tambahan selanjutnya. Implementasi ini memastikan bahwa sistem kontrol robot dapat berfungsi dengan baik dan memenuhi tujuan dari penelitian ini, yaitu merancang sistem kontrol yang fleksibel dan mudah dikembangkan.**

***Kata kunci*— inverse kinematics, robot humanoid, pola jalan, ROS**

I. PENDAHULUAN

Dalam Kontes Robot Indonesia (KRI) terdapat dua buah kategori lomba yang memiliki fokus terhadap robot *humanoid*, salah satunya adalah Kontes Robot Seni Tari Indonesia (KRSTI). Tujuan dari kontes robot ini adalah untuk menumbuh kembangkan kreatifitas dan minat para mahasiswa dalam teknologi, khususnya teknologi robotika yang selain diperuntukkan bagi industri juga diharapkan dapat membantu kegiatan manusia sehari-hari dan seni budaya khususnya seni tari [1].

Kinematika memainkan peran dasar dalam analisis gerak, perhitungan, dan kontrol sebuah robot *humanoid*. Kontrol gerak berbasis kinematika cukup sering digunakan dalam algoritma kontrol untuk gerakan tangan atau kaki [2]. *Inverse kinematics* adalah penggunaan persamaan kinematik untuk menentukan gerakan robot dalam mencapai posisi yang diinginkan. Penerapan persamaan *inverse kinematics* akan mendapatkan kombinasi sudut pada joint robot yang dapat bergerak bebas atau *Degrees of Freedom* (DoF).

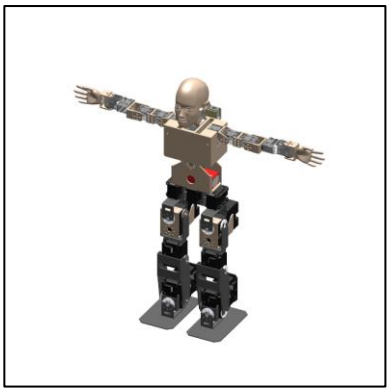
Pada implementasi sebelumnya, sistem kontrol yang digunakan pada robot menggunakan bantuan *software* R+ Motion. Namun, dengan adanya perkembangan konfigurasi aktuator robot dengan servo seri baru mengakibatkan konfigurasi robot yang baru tidak kompatibel dengan *software* ini. Oleh karena itu, diperlukan sebuah solusi alternatif yang dapat mengatasi masalah tersebut dan memastikan bahwa sistem kontrol robot dapat berfungsi dengan baik.

Berdasarkan permasalahan tersebut, dirancanglah sebuah sistem kontrol robot dengan menggunakan *Robot Operating System* (ROS) sebagai *framework inverse kinematics*. ROS adalah sistem operasi open source yang dikhususkan untuk mengembangkan robot. Dengan menggunakan ROS, sistem kontrol robot akan menjadi lebih fleksibel dan mudah diprogram [3]. ROS juga memiliki banyak *library* dan modul yang tersedia untuk memecahkan masalah pada perhitungan *inverse kinematics*.

II. KAJIAN TEORI

A. Robot Humanoid

Robot *humanoid* adalah robot yang bentuk dan struktur tubuhnya dibuat sedemikian rupa sehingga menyerupai tubuh manusia [4]. Sebagian besar teknologi yang ada merupakan implementasi manusia, mulai dari struktur fisik hingga kecerdasan. Robot *humanoid* terdiri dari DoF dalam jumlah yang relatif besar. Robot yang redundan secara kinematis dimodelkan sebagai sistem yang tidak dapat ditentukan. Karakterisasi seperti itu, bagaimanapun, bergantung pada jumlah tugas yang seharusnya dilakukan robot [2].

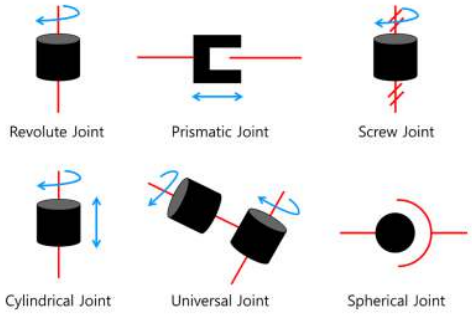


GAMBAR 1
Desain Robot Badaya_SAS

Robot *humanoid* pada umumnya memiliki karakteristik sebagai sistem multi-tubuh [2]. Dalam membuat pola jalan, pergerakan joint robot hanya dilakukan pada tubuh bagian kaki dan perut.

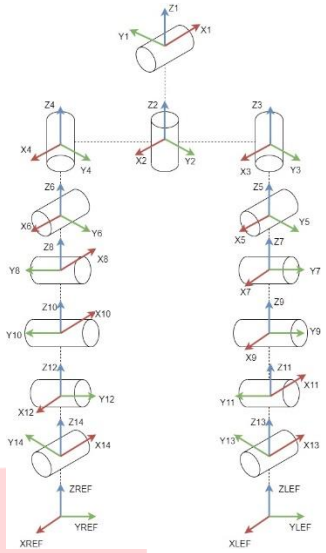
B. Pemodelan Robot

Pemodelan robot adalah hal terpenting untuk dapat menghitung *forward kinematics* dan *inverse kinematics* [5]. Robot *humanoid* terdiri dari beberapa link dan joint, semuanya terhubung pada sebuah base robot yang biasanya terletak pada bagian badan robot. Setiap link pada umumnya memiliki satu joint, tetapi bisa memiliki lebih dari satu. Tergantung pada karakteristik pergerakannya, joint dapat diklasifikasikan menjadi beberapa jenis, seperti Revolute, Prismatic, Screw, Cylindrical, Universal, dan Spherical [3].



GAMBAR 2
Tipe *Joint [3]*

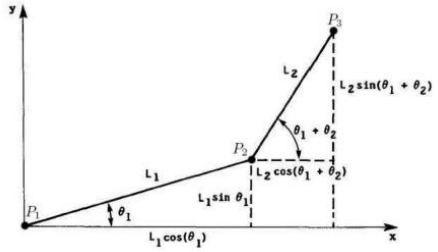
Bagian bawah robot Badaya_SAS memiliki 14 joint. Jumlah joint pada bagian bawah robot Badaya_SAS dapat dibagi menjadi 3 bagian yaitu: pada kaki terdapat 12 joint, dengan kaki kanan sebanyak 6 joint dan kaki kiri sebanyak 6 joint. Sedangkan, pada bagian badan terdapat 2 joint. Acuan perputaran joint pada robot Badaya_SAS dapat dibagi kedalam 3 sumbu, yaitu : roll (x), pitch (y), dan yaw (z).



GAMBAR 3
Konfigurasi Joint Bagian Bawah Robot Badaya_SAS

C. Forward Kinematics

Forward kinematics merupakan analisa gerak ketika yang diketahui adalah nilai variabel joint, sedangakan nilai yang dicari adalah posisi dan orientasi dari *end-effector*. Permasalahan ini dapat mudah diselesaikan menggunakan geometri dan kerangka koordinat pada robot yang ditentukan dalam Denavit-Hartenberg (DH) parameter [5]. Parameter DH memberikan tuntunan yang sistematis untuk analisa kinematika robot meskipun sebenarnya dapat pula dilakukan analisis tanpa menggunakan aturan ini.



GAMBAR 4
Kinematika 3DoF *[7]*

Mengacu pada Gambar 2.7, perhitungan *forward kinematics* secara umum dapat dituliskan dalam persamaan berikut:

$$\mathbf{P_2} = [\mathbf{L_1} * \boldsymbol{cos}(\boldsymbol{\theta_1}), \mathbf{L_1} * \boldsymbol{sin}(\boldsymbol{\theta_1})] \quad (1)$$

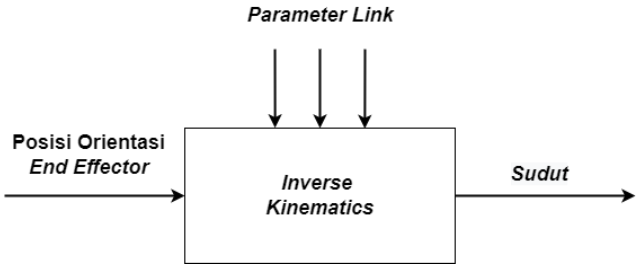
$$\mathbf{P_3} = [\mathbf{L_2} * \boldsymbol{cos}(\boldsymbol{\theta_1} + \boldsymbol{\theta_2}), \mathbf{L_2} * \boldsymbol{sin}(\boldsymbol{\theta_1} + \boldsymbol{\theta_2})] \quad (2)$$

Nilai posisi x dan y dari P3 dapat dicari dengan menambahkan persamaan (1) dan persamaan (2) yang akan menghasilkan persamaan sebagai berikut:

$$\mathbf{P_{3x}} = \mathbf{L_1} * \boldsymbol{cos}(\boldsymbol{\theta_1}) + \mathbf{L_2} * \boldsymbol{cos}(\boldsymbol{\theta_1} + \boldsymbol{\theta_2}) \quad (3)$$

$$\mathbf{P_{3y}} = \mathbf{L_1} * \boldsymbol{sin}(\boldsymbol{\theta_1}) + \mathbf{L_2} * \boldsymbol{sin}(\boldsymbol{\theta_1} + \boldsymbol{\theta_2}) \quad (4)$$

D. Inverse Kinematics



GAMBAR 5
Proses Inverse Kinematics

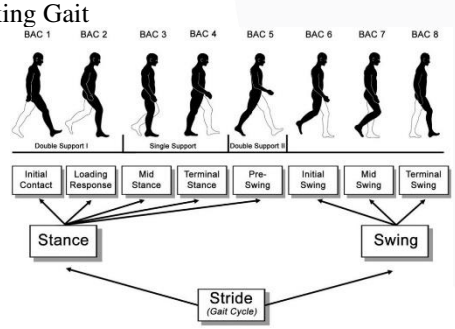
Inverse kinematics adalah metode untuk menentukan satu set sudut bersama yang akan memenuhi *end-effector* yang diberikan berpose di ruang euklides [8]. Penggunaan *inverse kinematics* lebih banyak digunakan pada robot, karena untuk menggerakkan *end-effector* hanya diperlukan koordinat yang ingin dituju dan fungsi *inverse kinematics* akan menghitung nilai variabel joint. Nilai variabel joint diperlukan oleh pengontrol motion robot untuk memindahkan *end-effector* robot ke titik yang diinginkan di ruang kartesian, atau melalui beberapa titik jika ada lintasan. Dengan mengacu pada model kinematik yang terdapat pada Gambar 2.7, persamaan *inverse kinematics* dapat dituliskan sebagai berikut:

$$\boldsymbol{cos}(\boldsymbol{\theta_2}) = \frac{(P_{3x}^2 + P_{3y}^2) - (L_1^2 + L_2^2)}{2L_1L_2} \quad (5)$$

$$\boldsymbol{tan}(\boldsymbol{\theta_1}) = \frac{-L_2 sin(\theta_2)P3x + (L_1 + L_2 cos(\theta_2))P3y}{L_2 sin(\theta_2)P3y + (L_1 + L_2 cos(\theta_2))P3x} \quad (6)$$

Karena persamaan-persamaan dari sistem ini dalam bentuk sinus dan kosinus, mereka dapat diselesaikan secara sistematis. Namun solusi umum untuk kinematika yang memiliki lebih banyak link dan joint tidak layak menggunakan persamaan diatas, karena kompleksitas meningkat dengan cepat ketika variabel tersebut bertambah [7].

E. Walking Gait



GAMBAR 6
Gaya Berjalan Manusia

Sebagai sistem kinematik yang redundan, perencanaan walking gait dari sebuah robot *humanoid* harus dilakukan secara tepat. Dalam perencanaan berjalan diperlukan koordinat untuk menggerakan *end-effector* berada pada telapak kaki dan susunan beberapa koordinat yang disebut gait [9]. Untuk manusia normal, ada 4 kriteria utama yang penting untuk berjalan [10].

1. Kemampuan manusia dalam menjaga keseimbangan dan stabilitas.
2. Kemampuan manusia dalam memulai gerakan dan mendukung pola berjalan berulang.
3. Integritas muskuloskeletal: Untuk menjaga keseimbangan berjalan yang tepat, tulang, otot, dan sendi berkolaborasi dan berfungsi bersama.
4. Kontrol neurologis yang memungkinkan komunikasi antara otak dan otot untuk menyampaikan instruksi dan memberi tahu tubuh kapan harus bergerak dan bagaimana melanjutkannya.

Periode antara satu kaki yang melakukan kontak dengan permukaan dan kaki lainnya melakukan kontak dengan permukaan disebut sebagai satu siklus gaya berjalan. Langkah manusia dapat dibagi menjadi dua, tahap pertama yang dikenal sebagai *stance* adalah ketika kaki benar-benar bersentuhan dengan permukaan. Tahap kedua disebut tahap *swing* di mana kaki tidak bersentuhan dengan permukaan [10].

F. *Robot Operating System* (ROS)

ROS adalah *Software Development Kit* (SDK) open source untuk aplikasi robot [11]. ROS menyediakan tools dan *library* untuk filtering, mapping, komunikasi, *Kinematic and Dynamic Library* (KDL) dll. Selain itu, ROS dapat menggambarkan robot dalam *Unified Robot Description Format* (URDF), yang diproyeksikan sebagai model 3D yang masing-masing model dapat dipindahkan atau dioperasikan sesuai dengan DoF yang aslinya, sehingga dapat digunakan untuk simulasi atau kontrol [3]. ROS menyediakan alat yang cukup untuk proses data dan analisis seperti visualisasi 3D (RViz), mencatat robot waktu nyata eksperimen dan memainkannya secara offline dengan (rosbag/rxbag), merencanakan data (rxplot) dan memvisualisasikan seluruh jaringan ROS struktur (rxgraph).

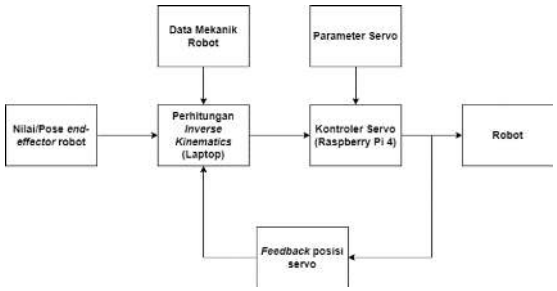
G. TRAC-IK

Implementasi sistem kontrol *inverse kinematics* untuk pola jalan robot *humanoid* di penelitian ini menggunakan *kinematic solver* TRAC-IK metode iteratif. TRAC-IK menyediakan *kinematic solver* alternatif untuk metode Jacobian yang populer pada KDL. Secara khusus, algoritma konvergensi KDL didasarkan pada metode Newton, yang tidak bekerja dengan baik dengan adanya batasan joint untuk banyak platform robot [13]. Metode iteratif pada TRAC-IK bekerja dengan memulai prediksi awal untuk sudut joint, kemudian berulang kali memperbarui estimasi sudut joint hingga solusi yang memuaskan ditemukan. Hal ini dilakukan dengan menggunakan persamaan *forward kinematics* untuk menghitung posisi dan orientasi *end-effector* berdasarkan estimasi sudut joint saat ini, dan kemudian menggunakan persamaan *inverse kinematics* untuk menghitung estimasi sudut sendi yang diperbarui berdasarkan perbedaan antara posisi dan orientasi *end-effector* yang diinginkan dengan posisi dan orientasi *end-effector* yang sebenarnya [7].

III. METODE

A. Desain Sistem

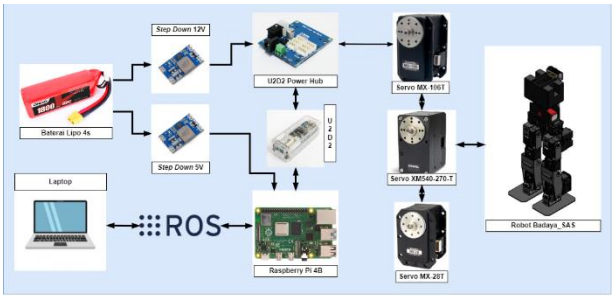
Desain ini digunakan sebagai gambaran umum yang mendefinisikan cara kerja sistem yang akan dirancang.



GAMBAR 7
Diagram Blok

Pada penelitian ini rancangan sistem gerak kaki robot menggunakan laptop untuk melakukan perhitungan *inverse kinematics* dan SBC Raspberry Pi 4 sebagai kontroler servo. Perancangan sistem terdiri dari perancangan sistem secara umum, perancangan perangkat keras dan perancangan perangkat lunak pada robot. Perancangan perangkat keras membahas spesifikasi komponen robot yang digunakan serta keterkaitan komponen satu sama lain agar sistem dapat berjalan sesuai yang diinginkan. Perancangan perangkat lunak menggunakan ROS sebagai *framework* robot dan TRAC-IK sebagai *kinematic solver* untuk proses perhitungan *inverse kinematics* robot.

B. Desain Perangkat Keras

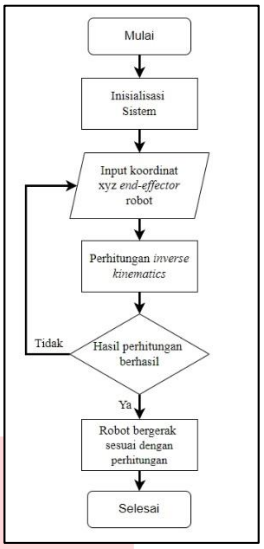


GAMBAR 8
Gambaran Umum Sistem

Berikut merupakan konfigurasi sistem umum untuk robot *humanoid* yang terdapat pada Gambar 8.

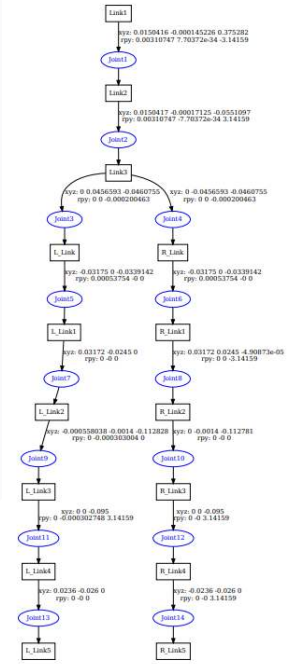
- Baterai Lipo 4S digunakan sebagai sumber power utama dalam sistem ini yang dihubungkan ke step down 12v dan step down 5v. Keluaran step down 12v digunakan sebagai power utama servo dynamixel sedangkan step down 5v digunakan sebagai power utama SBC Raspberry Pi 4.
- Komponen U2D2 digunakan untuk mengirimkan data dari servo dynamixel yang dihubungkan secara paralel pada power hub menuju Raspberry Pi 4 yang bekerja sebagai kontroler servo.
- Laptop pada sistem digunakan sebagai ROS Master yang menjalankan perhitungan *inverse kinematics*. Komunkasi data antara Laptop dan Raspberry Pi 4 dilakukan melalui protokol ROS Master Uniform Resource Identifier (URI).

C. Desain Perangkat Lunak



GAMBAR 9
Diagram Alir Sistem

Gambar 9 menggambarkan Diagram alir dari sistem robot saat proses perhitungan *inverse kinematics* menggunakan plugin RViz pada ROS. Proses input nilai koordinat *end-effector* robot dilakukan melalui GUI RViz, setelah input diterima makan akan diteruskan menuju proses perhitungan. Hasil dari perhitungan dapat dilihat pada terminal atau RViz, jika pergerakan robot sudah sesuai maka hasil perhitungan dapat diteruskan ke setiap servo dari robot.



GAMBAR 10
KDL TREE URDF

Gambar 10 menunjukan tampilan dari URDF yang dibuat menggunakan *software* solidwork. Nilai data tersebut akan tergenerasi secara otomatis berdasarkan konfigurasi yang dilakukan saat proses assembly robot menggunakan plugin URDF Exporter.

IV. HASIL DAN PEMBAHASAN

Bab hasil dan analisis akan membahas hasil pengujian Implementasi *Inverse kinematics* untuk Pola Jalan Robot Penari *Humanoid*. Kemampuan dari sistem yang telah dirancang perlu dibahas pada pengujian sistem. Pengujian sistem dilakukan menggunakan Ubuntu 20.04 dengan ROS Noetic serta menggunakan CPU I7-7700HQ dan GPU GTX 1050. Berikut merupakan beberapa pengujian yang telah dilakukan :

1. Pengujian performa waktu & tingkat keberhasilan algoritma dalam mencari nilai sudut.
2. Pengujian error posisi kaki robot.
3. Pengujian implementasi pola jalan.

A. Hasil Pengujian Performa Package TRAC-IK

Hasil pengujian performa package TRAC-IK ditampilkan pada tabel berikut.

TABEL 1
TABEL PENGUJIAN PERFORMA PACKAGE TRAC-IK

		Package IK *Solver*					
		KDL		LMA		TRAC-IK	
Planning Group	DoF	Persentase	Rata-rata Waktu	Persentase	Rata-rata Waktu	Persentase	Waktu
Kanan	6	100%	0.249 s	100%	0.136 s	100%	0.222 s
Kiri	6	100%	0.187 s	96%	0.175 s	100%	0.094 s
Fullkanan	8	95%	0.348 s	99%	0.181 s	99%	0.075 s
Fullkiri	8	100%	0.055 s	98%	0.049 s	100%	0.056 s

Hasil pengujian menunjukkan TRAC-IK memiliki tingkat keberhasilan paling tinggi jika dibandingkan dengan algoritma lain. Sedangkan untuk rata-rata waktu yang dibutuhkan dalam melakukan perhitungan *inverse kinematics*, TRAC-IK juga memiliki hasil yang bagus jika dibandingkan pada metode IK lain yang diuji. Secara keseluruhan, hasil performa package TRAC-IK dapat dikatakan lebih unggul jika khususnya pada pengujian 8 DoF jika dibandingkan pada algoritma IK solver bawaan yang tersedia pada MoveIt!.

B. Hasil Pengujian Error Posisi Kaki Robot

Pengujian error posisi robot ini dilakukan dengan cara memberikan masukan posisi *end-effector* berupa nilai x,y,z atau pose robot melalui interface RViz dan keluaran yang dihasilkan berupa sudut joint yang menggerakan kaki robot. Output feedback posisi servo dari *forward kinematics* akan dibandingkan dengan masukan posisi yang diinginkan.

TABEL 2
NILAI POSISI ERROR KAKI KANAN (DS)

ID *Joint*	Input		Output		Absolut Error (°)
	Rad	Derajat (°)	Rad	Derajat (°)	
4	0.00003	0.00152	0.00307	0.17587	0.17435
6	0.00004	0.00218	-0.00153	-0.08794	0.09011
8	-0.77492	-44.42208	-0.77313	-44.31935	0.10274
10	1.95000	111.78343	1.96503	112.64500	0.86157
12	1.17496	67.35450	1.17810	67.53424	0.17974
14	0.00007	0.00404	-0.00767	-0.43968	0.44372

TABEL 3
NILAI POSISI ERROR KAKI KIRI (DS)

ID *Joint*	Input		Output		Absolut Error (°)
	Rad	Derajat (°)	Rad	Derajat (°)	
3	0.00009	0.00492	0.00307	0.17587	0.17095
5	-0.00008	-0.00479	-0.00920	-0.52761	0.52282
7	0.77496	44.42459	0.76546	43.87967	0.54492
9	-1.95001	-111.78406	-1.95889	-112.29326	0.50920
11	1.17496	67.35450	1.17810	67.53424	0.17974
13	0.00007	0.00404	-0.00767	-0.43968	0.44372

TABEL 4
NILAI POSISI ERROR KAKI KANAN (SS)

ID *Joint*	Input		Output		Absolut Error (°)
	Rad	Derajat (°)	Rad	Derajat (°)	
4	-0.05065	-2.90332	-0.05062	-2.90186	0.00146
6	0.23617	13.53820	0.23163	13.27822	0.25999
8	-0.83289	-47.74532	-0.83142	-47.66088	0.08444
10	2.11221	121.08185	2.10155	120.47123	0.61062
12	1.29170	74.04656	1.29621	74.30525	0.25868
14	0.30067	17.23610	0.29759	17.05943	0.17667

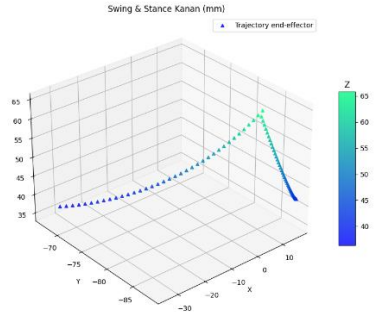
TABEL 5
NILAI POSISI ERROR KAKI KIRI (SS)

ID *Joint*	Input		Output		Absolut Error (°)
	Rad	Derajat (°)	Rad	Derajat (°)	
3	0.04659	2.67099	0.04449	2.55012	0.12087
5	-0.23410	-13.41992	-0.23010	-13.19028	0.22964
7	0.59455	34.08275	0.58751	33.67918	0.40356
9	-1.73495	-99.45559	-1.71959	-98.57536	0.88023
11	-1.18087	-67.69319	-1.17963	-67.62217	0.07102
13	-0.23407	-13.41814	-0.23317	-13.36615	0.05199

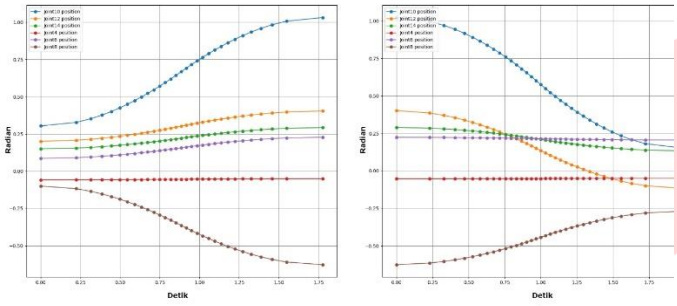
Berdasarkan nilai error diatas, dapat diketahui selisih terbesar nilai antara input dengan output adalah 0.88023°. Error tersebut masih dapat ditoleransi dan tidak mengakibatkan robot jatuh saat implementasi. Nilai error disebabkan karena adanya backlash dan slip pada servo yang digunakan.

C. Hasil Pengujian Implementasi Pola Jalan

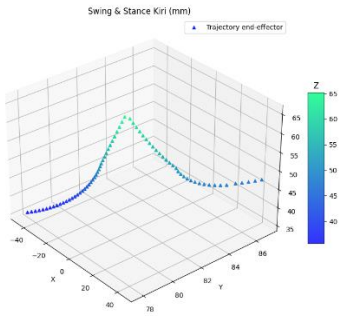
Pengujian implementasi pola jalan robot dilakukan untuk mengetahui hasil desain sistem *inverse kinematics* dalam menghasilkan trayektori pola jalan. Pada pengujian ini, akan diimplementasikan pola jalan pada robot Badaya_SAS. Perencanaan pola jalan didapat berdasarkan fase pergerakan jalan robot *humanoid* pada bab 2. Fase pergerakan yang diuji adalah saat robot berada difase *Single Support.*



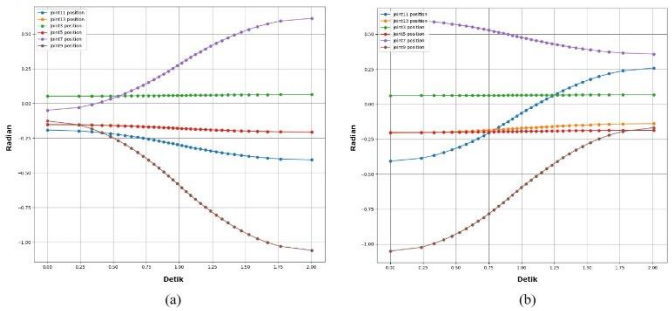
GAMBAR 11
Trayektori End-effector Swing & Stance Kaki Kanan



GAMBAR 12
(a) Swing Kanan (b) Stance Kanan

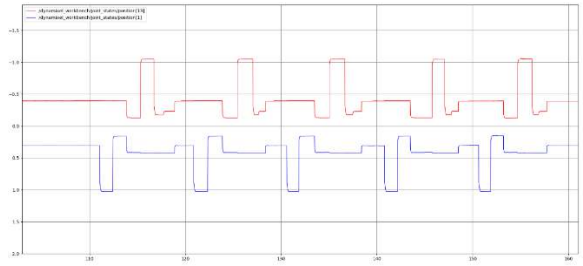


GAMBAR 13
Trayektori End-effector Swing & Stance Kaki Kiri



GAMBAR 14
(a) Swing Kiri (b) Stance Kiri

Berdasarkan Gambar 11 dan Gambar 12 pergerakan *swing* dan *stance* pada kaki kanan dapat dikatakan stabil karena tidak terjadi perubahan yang signifikan pada posisi kaki robot saat melakukan gerakan. Jumlah gerak langkah yang dihasilkan melebihi 60 gerak langkah yang dijalankan dalam kurun waktu 2 detik, yang menunjukkan bahwa robot dapat melakukan gerakan dengan cepat dan stabil. Hal tersebut juga sama seperti pergerakan *swing* dan *stance* yang dapat dilihat pada Gambar 13 dan Gambar 14 dimana perubahan posisi kaki robot termasuk stabil.



GAMBAR 15
Posisi Real-Time Robot

Gambar 15 merupakan grafik feedback posisi aktual robot saat pola jalan dijalankan. Perubahan posisi servo yang terjadi memiliki delay data yang disebabkan oleh komunikasi servo yang menggunakan sistem *half duplex*. Hasil dari grafik menunjukkan posisi servo yang stabil dalam melakukan gerakan dengan perbedaan yang tidak signifikan dengan posisi yang diharapkan. Perbedaan terjadi dikarenakan adanya absolut error pada servo yang digunakan. Hal ini menunjukkan bahwa robot dapat menjalankan pola jalan dengan stabil.

V. KESIMPULAN

Sistem kontrol *inverse kinematics* yang dikembangkan pada bagian kaki robot *humanoid* Badaya_SAS telah menghasilkan pola jalan yang stabil dan kompatibel dengan konfigurasi servo yang ada. Performa package TRAC-IK dalam menjalankan proses *inverse kinematics* pada robot Badaya_SAS mempunyai akurasi rata-rata sebesar 99,75% dan rata-rata waktu penyelesain sebesar 0,112 detik menggunakan GPU GTX 1050. Rancangan sistem menghitung absolut error pada posisi servo robot dengan nilai error terbesar 0.88023° dan dapat dikatakan dalam batas wajar karena error tersebut tidak memberikan dampak yang signifikan saat robot melakukan gerakan jalan. Pola jalan tersebut memenuhi tujuan untuk membuat pola jalan robot yang baik dan menjawab permasalahan dalam berjalan pada bidang datar.

REFERENSI

[1] Abdul Muis et al, (2021). PETUNJUK PELAKSANAAN KONTES ROBOT INDONESIA (KRI) TAHUN 2021. Jakarta: Pusat Prestasi Nasional, Kementrian Pendidikan dan Kebudayaan.

[2] David Coleman, Ioan A. Șucan, Sachin Chitta, Nikolaus Correll, Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study, Journal of Software Engineering for Robotics, 5(1):3–16, Mei 2014.

[3] YoonSeok Pyo, HanCheol Cho, RyuWoon Jung, TaeHoon Lim, ROS Robot Programming. ROBOTIS Co.,Ltd., 2017.

[4] G. Venture, J.-P. Laumond, dan B. Watier, Biomechanics of Anthropomorphic Systems. Springer International Publishing AG, 2019.

[5] Fabrice Noreils, "Inverse kinematics for a Humanoid Robot : a mix between closed form and geometric solutions" ResearchGate, 2017.
[6] Ilham Defra Nugraha, Putu Mahayana Santika, "Pendekatan Geometri untuk Perhitungan Inverse Kinematics Gerakan Lengan Robot 4 Derajat Kebebasan" JURNAL TEKNIK MESIN – ITI Vol. 5 No.1, 202.
[7] Rickard Nilsson, "Inverse Kinematics", 2009.
[8] Karan Khokar, "Implementation of KDL Inverse Kinematics Routine on the Atlas" ICICT (International Conference on Information and Communication Technologies), 2015.
[9] Akhmad Riko Kurniawan, "Perancangan Robot Bipedal Dengan Sistem Berjalan Berbasis Inverse Kinematic Dengan Sensor Mpu 6050 Sebagai Indikator Kemiringan" Transient, vol. 6 2017.
[10] H. A. Poonja, M. S. A. Shah, R. Uddin, S. M. H. Kazmi, H. Khan, A. H. Ali, and M. A. Shirazi, "Walking Algorithm Using Gait Analysis for Humanoid Robot", The 7th International Electrical Engineering Conference, 4 Agustus 2022.
[11] AmandaDattalo, "ROS/Introduction" (https://ros.org/ , Diakses pada 5 Januari 2022).
[12] Zeyang Xia,Yangzhou Gan, "Manipulation Task Simulation using ROS and Gazebo" International Conference on Robotics and Biomimetics, 2014.
[13] Patrick Beeson, Barrett Ames, "TRAC-IK: An Open-Source Library for Improved Solving of Generic Inverse Kinematics" IEEE RAS Humanoids Conference, 2015.