

# Gerak Robot Berkaki Dua menggunakan ROS dan RViz sebagai Visualisasi Interaktif

ADI SUCIPTO, RADEN SANGGAR DEWANTO, DADET PRAMADIHANTO

Teknik Elektro Politeknik Elektronika Negeri Surabaya, Indonesia  
Email: adisucipto19950824@gmail.com

*Received* 22 Mei 2020 | *Revised* 18 Juni 2020 | *Accepted* 6 Juli 2020

## ABSTRAK

*Pengembangan sistem operasi pada bidang robotika telah menjadi fokus utama pada era ini. Salah satu perkembangan sistem operasi pada teknologi robot saat ini adalah Robot Operating System (ROS) dengan RViz. ROS merupakan sistem operasi berbasis library dan beberapa tools untuk mengembangkan suatu program pada robot, sedangkan RViz merupakan visualisasi tiga dimensi yang dapat digunakan untuk memvisualisasikan robot dan data sensor dynamixel. Pada Penelitian kali ini, peneliti membuat simulasi beberapa gerakan yang dilakukan pada RViz dan kemudian diimplementasikan pada robot. Tingkat keberhasilan dari perencanaan gerakan ini memiliki rata-rata error sebesar 1.8%. Gerakan condong ke kiri memiliki rata-rata error sebesar 0.83%. Gerakan condong ke kanan memiliki rata-rata error sebesar 0.84%. Gerakan mengangkat satu kaki memiliki rata-rata error sebesar 1.71%. Gerakan kaki kanan ke depan memiliki rata-rata error sebesar 3.83%.*

**Kata kunci:** Robot Berkaki Dua, Robot Operating System (ROS), RViz (ros-visualization), Dynamixel Controller, Data Sensor Dynamixel.

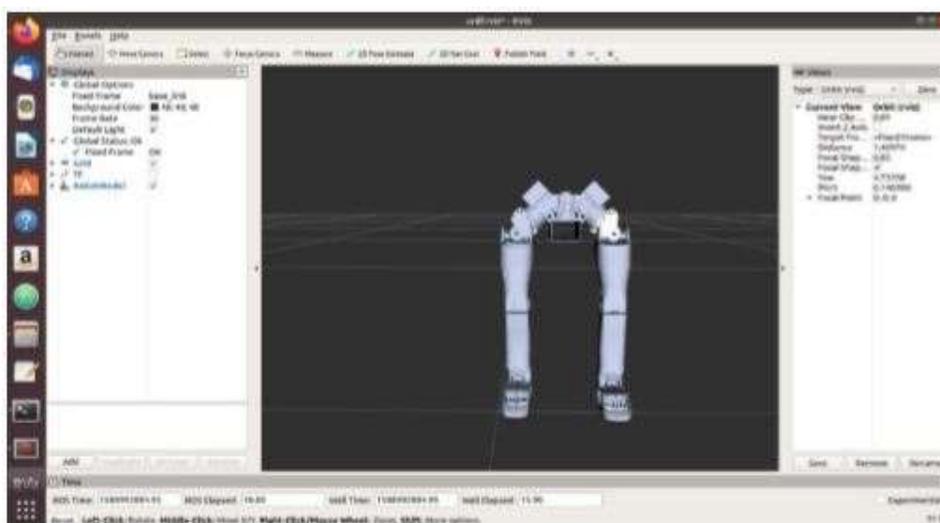
## ABSTRACT

*The development of operating systems in the field of robotics has become the main focus of this era. One of the operating system developments in robot technology today is the Robot Operating System (ROS) with RViz. ROS is a library-based operating system and several tools for developing a program on robots, while RVIZ is a three-dimensional visualization that can be used to visualize robots and dynamixel sensor data. In this study, researchers made a simulation of some of the movements carried out on RViz and then implemented on robots. The success rate of planning this movement has an average error of 1.8%. Leaning to the left has an average error of 0.83%. Leaning to the right has an average error of 0.84%. One leg lift has an average error of 1.71%. The movement of the right foot forward has an average error of 3.83%.*

**Keywords:** Biped Robot, Robot Operating System (ROS), RViz (Ros-Visualization), Dynamixel Controller, Sensor Dynamixel Data.

## 1. PENDAHULUAN

Perkembangan dunia robotika pada era revolusi industri 4.0 telah berkembang pesat. Salah satu perkembangan yang populer dilakukan saat ini adalah pembuatan sistem operasi pada robot. Beberapa pengembangan sistem operasi yang dikembangkan saat ini adalah robot operating sistem (**Cashmore, dkk, 2015**) Selain itu, beberapa software pendukung simulasi seperti V-REP juga dilakukan untuk mempermudah sistem pembuatan robot (**Arfaq, dkk, 2018**). Sistem tersebut dapat berupa perencanaan gerakan menggunakan algoritma dan metode tertentu (**Suhendra, dkk, 2017**). Akan tetapi robot operating sistem memiliki keunggulan yaitu dapat mensimulasikan semua keadaan lingkungan dengan lebih detail (**Santos, dkk, 2013**). Oleh karena itu, dibutuhkan sistem operasi yang dapat meringankan kinerja dalam pembuatan robot dengan tingkat kepresisian yang tinggi (**Balakirsky, dkk, 2012**). Sistem operasi yang digunakan pada penelitian kali ini adalah *Robot operating system (ROS)* dengan RViz sebagai media visualisasi interaktif. Penelitian yang dilakukan kali ini menggunakan robot berkaki dua dengan tujuan perencanaan gerakan robot. Ilustrasi dari salah satu perencanaan gerakan tersebut dapat dilihat pada Gambar 1 berikut.



**Gambar 1. Ilustrasi Robot Berkaki Dua Berdiri pada ROS Menggunakan RViz**

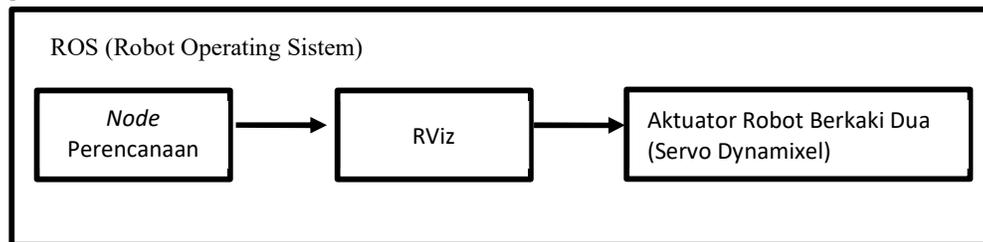
Beberapa penelitian yang terkait dengan perencanaan gerak pada robot berkaki dua antara lain perancangan sistem kendali gerak lengan robot pengikut gerak lengan manusia berbasis mikrokontroler (**Syafrudin, dkk, 2012**). Penelitian lainnya adalah pengendalian robot *line follower* berbasis mikrokontroler AT-8535 menggunakan PID (**Prabowo, dkk, 2013**). Penggunaan perancangan dan pengendalian lengan robot dan *line follower* tersebut menandakan bahwa kurangnya efisien kinerja dari sistem robot yang menggunakan mikrokontroler.

Pengembangan sistem operasi menggunakan ROS juga telah dilakukan pada penelitian robot humanoid NimRo-OP (**Allgeuer, dkk, 2018**). Penelitian tersebut mencakup keseluruhan sistem yang diadaptasi pada robot menggunakan ROS. Penelitian yang serupa pada robot berkaki dua yaitu *Darwin*. Penelitian tersebut menggunakan ROS untuk melakukan sinergi pada aktuatornya (**Cunha, dkk, 2016**). Penelitian yang dilakukan kali ini mengadopsi dari jurnal yang membahas tentang sistem operasi pada robot antara lain *Robot operating system (ROS)* dan Gazebo sebagai media pembelajaran interaktif (**Jalil, 2018**). Selain itu, penelitian ini juga mengadopsi dari jurnal lain yang berjudul *RViz: a toolkit for real domain data*

*visualization* (Kam, dkk, 2015) yang berfungsi sebagai visualisasi data pada model. Penelitian yang lain diadopsi untuk visualisasi adalah menggunakan gazebo (Takaya, dkk, 2016). Penelitian yang dilakukan saat ini menggunakan implementasi perencanaan gerak pada robot berkaki dua menggunakan ROS dan RViz sebagai media visualisasi interaktif. Pada penelitian ini dilakukan simulasi pada robot berkaki dua dengan melakukan beberapa gerakan. Gerakan tersebut adalah gerakan condong ke kanan, gerakan condong ke kiri, gerakan mengangkat satu kaki dan gerakan melangkahkan kaki ke depan. Pada simulasi tersebut digunakan ROS dan RViz untuk membangun perencanaan gerakan robot pada komputer. Hasil dari perencanaan tersebut kemudian diimplementasikan pada robot yang sesungguhnya.

## 2. METODE PENELITIAN

Penelitian yang digunakan kali ini menggunakan metode perencanaan gerak menggunakan robot operating sistem menggunakan RViz sebagai media visualisasi interaktif. Sebuah *node* yang berisi perintah menggerakkan model yang ada didalam RViz telah dirancang dan kemudian diimplementasikan pada robot dengan menggunakan *feature* dari beberapa *packages*.

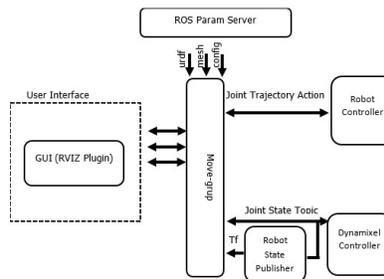


Gambar 2. Blok Diagram Sistem

Pada Gambar 2 dijelaskan bahwa perencanaan gerakan dibuat pada sebuah *node*. *Node* merupakan suatu aplikasi *client* pada sistem operasi yang terhubung dengan server ROS. Di dalam *node* ini dirancang rencana pergerakan dari masing-masing *joint*. *Node* tersebut kemudian dikirim pada media visualisasi yaitu RViz. Media visualisasi ini menampilkan gerakan dari robot berkaki dua sesuai dengan *node* yang telah dibuat.

### 2.1 Perancangan Sistem Robot Berkaki Dua

Perencanaan gerakan robot berkaki dua menggunakan ROS memiliki 2 sistem utama yaitu *ROS Parameter Server* dan *ROS Node Client* (Megalingam, dkk, 2018). Gambar 3 merupakan ilustrasi tentang hubungan kedua sistem tersebut untuk membentuk gerakan pada visualisasi dan robot sesungguhnya secara sama. Penelitian yang dilakukan kali ini menggunakan *robot operating system* (ROS) dengan menggunakan beberapa *packages* seperti *dynamixel controller* dan *robot controller*. Masing-masing *packages* tersebut memiliki *node*. *Node* tersebut kemudian dikirim pada media visualisasi yaitu RViz (Sokolov, dkk, 2016). Dalam pembuatan media visualisasi interaktif kali ini, digunakan tampilan model robot yang telah diatur pada *packages urdf*.

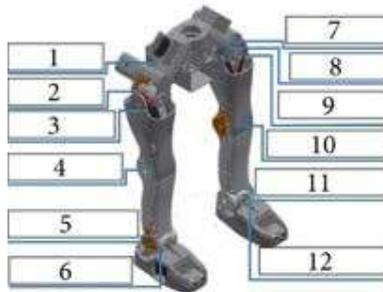


**Gambar 3. Perancangan Sistem pada ROS**

Pada Gambar 3 menunjukkan ros parameter server mengirimkan *urdf*, *mesh* dan *config* sebagai *node* yang berfungsi untuk menampilkan desain robot pada media visualisasi RViz. *Dynamixel controller* merupakan sebuah *packages* yang berisi tentang *library* dari aktuator servo *dynamixel*. *Packages* ini dapat mengakses *feature* data sensor *dynamixel*. *Feature* tersebut dapat mengetahui kondisi aktuator seperti posisi, beban, temperature dan tegangan saat ini. Selain itu, pada *packages* ini berfungsi interface antara visualiasi RViz ke aktuator secara langsung. Sedangkan *packages robot controller* berfungsi untuk mengirimkan *node* dari program editor kepada RViz.

### 2.2 Konfigurasi Robot Berkaki Dua

Pada penelitian ini digunakan robot berkaki dua dengan 12 aktuator. Aktuator yang digunakan menggunakan servo *dynamixel* MX-106T. Aktuator jenis ini dapat memberikan *feedback* berupa posisi, *temperature*, beban dan tegangan *input*. Berikut merupakan konfigurasi dari robot berkaki dua tersebut:



Keterangan :

- |                       |                       |
|-----------------------|-----------------------|
| 1. Hip Yaw kanan      | 9. Hip Pitch Kiri     |
| 2. Hip Roll Kanan     | 10. Lutut Pitch Kiri  |
| 3. Hip Pitch Kanan    | 11. Engkel Pitch Kiri |
| 4. Lutut Pitch Kanan  | 12. Engkel Roll Kiri  |
| 5. Engkel Pitch Kanan |                       |
| 6. Engkel Roll Kanan  |                       |
| 7. Hip Yaw Kiri       |                       |
| 8. Hip Roll Kiri      |                       |

**Gambar 4. Konfigurasi Robot Berkaki Dua**

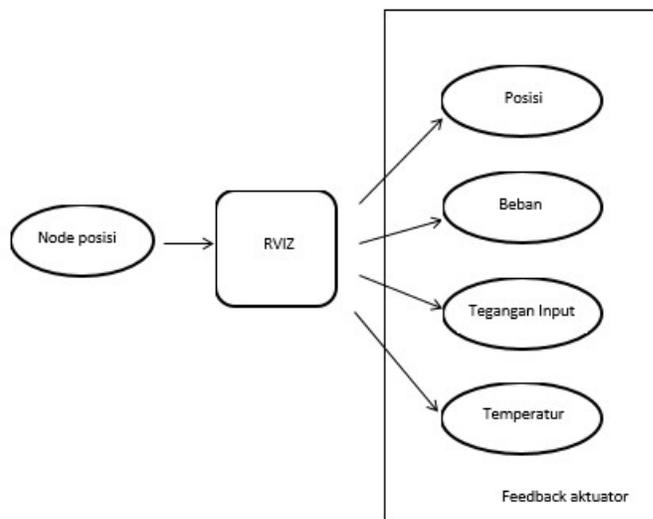
Pada Tabel 1 di bawah ini merupakan spesifikasi yang digunakan pada penelitian kali ini. Sistem operasi yang digunakan adalah *robot operating system melodic, Ubuntu 18.04*. Salah satu alasan digunakan ROS Melodic adalah ketersediaan *library* yang lengkap untuk mengakses aktuator servo *dynamixel* MX-106T.

**Tabel 1. Tabel Spesifikasi Robot**

No	Item	Spesifikasi
	Mini-PC	Intel (R) Core (TM) i5-6260U CPU@1.80 Ghz
1.	Sistem operasi	Robot Operating Sistem Melodic, Ubuntu 18.04
2.	Baterai	3 cell @12.6V 5000 mAh
3.	Aktuator (Servo <i>Dynamixel</i> MX-106T)	ARM CORTEX-M3 (72 [MHz], 32 bit) Coreless (Maxon motor) Baud Rate: 8,000 [bps] ~ 4.5 [Mbps] Torsi 8.4 [Nm] (pada 12 [V], 5.2 [A]) Feature: Posisi, Temperatur, Beban, Tegangan Input, dst.

### 2.3 RViz sebagai Visualisasi Interaktif

RViz merupakan visualisasi tiga dimensi yang dapat digunakan menampilkan robot pada program simulasi. Terdapat 2 sistem utama pada penelitian perencanaan gerakan robot ini. Sistem ini meliputi pengiriman *node* dengan pengaturan masing-masing sudut tiap *joint* pada RViz dan Pengiriman perintah pada aktuator servo *dynamixel*. Kedua sistem ini dapat divisualisasikan melalui RViz. Gambar 5 merupakan ilustrasi tentang hubungan kedua sistem tersebut untuk membentuk gerakan pada visualisasi dan robot sesungguhnya secara sama.



**Gambar 5. Ilustrasi Skema Pengiriman *Node* pada RViz dengan Aktuator**

Pada Gambar 5 menunjukkan bahwa RViz merupakan media visualisasi robot yang dapat menampilkan gerakan sesuai dengan pengaturan *node* yang telah dibuat. Isi dari *node* tersebut adalah pengaturan nilai-nilai sudut tiap *joint* yang kemudian dikirim sebagai perintah untuk menggerakkan visualisasi robot pada tampilan RViz. Media visualisasi ini menampilkan gerakan dari robot berkaki dua sesuai dengan *node* yang telah dibuat. Setelah *node* ini dijalankan, terdapat perintah *rostopic* yang berfungsi untuk menampilkan sistem pada ROS yang sedang berjalan. *Rostopic* yang digunakan pada penelitian ini bertujuan untuk mengetahui *feedback* dari aktuator seperti nilai posisi sudut, nilai beban, tegangan *input* dan *temperature* aktuator pada saat itu.



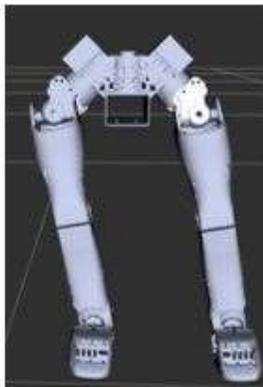
untuk melakukan perencanaan gerakan selanjutnya. Pada Tabel 2 berikut ini merupakan hasil dari kalibrasi pembacaan nilai berdiri dari masing-masing aktuator.

**Tabel 2. Hasil Kalibrasi Nilai Resolusi Saat Robot Berdiri**

No	Bagian Robot	Resolusi	Sudut (°)
1.	<i>Hip Yaw</i> Kanan	2440	0
2.	<i>Hip roll</i> Kanan	1228	0
3.	<i>Hip Pitch</i> Kanan	1631	0
4.	Lutut Pitch Kanan	1076	0
5.	Engkel Pitch Kanan	674	0
6.	Engkel Roll Kanan	461	0
7.	<i>Hip Yaw</i> Kiri	1441	0
8.	<i>Hip roll</i> Kiri	3207	0
9.	<i>Hip Pitch</i> Kiri	3012	0
10.	Lutut Pitch Kiri	3690	0
11.	Engkel Pitch Kiri	970	0
12.	Engkel Roll Kiri	268	0

Resolusi tersebut merupakan nilai satuan posisi yang terdapat pada data sensor *dynamixel*. Nilai satuan tersebut adalah 4096 untuk mewakili sudut 360° (**Robotis\_MX-106T , 2020**). Sedangkan pada Gambar 7(b) merupakan implementasi dari pengiriman *node* dari RViz menuju robot secara langsung. Implementasi tersebut menggunakan data dari RViz menggunakan *packages robot controller* dan kemudian dikirim ke aktuator menggunakan *packages dynamixel controller*. Nilai dari setiap resolusi yang didapat dari aktuator *dynamixel* digunakan sebagai titik acuan 0 dari robot.

Setelah mendapatkan kalibrasi berdiri robot, kemudian dilakukan perencanaan gerakan lainnya seperti condong ke kanan, condong ke kiri, mengangkat satu kaki dan kaki kanan robot ke depan. Pada Gambar 8 berikut ini merupakan hasil dari simulasi visualisasi dan implementasi dari gerakan condong ke kiri pada robot:



(a) Visualisasi Condong ke Kiri



(b) Implementasi Condong ke Kiri

**Gambar 8. Robot Kondisi Condong ke Kiri**

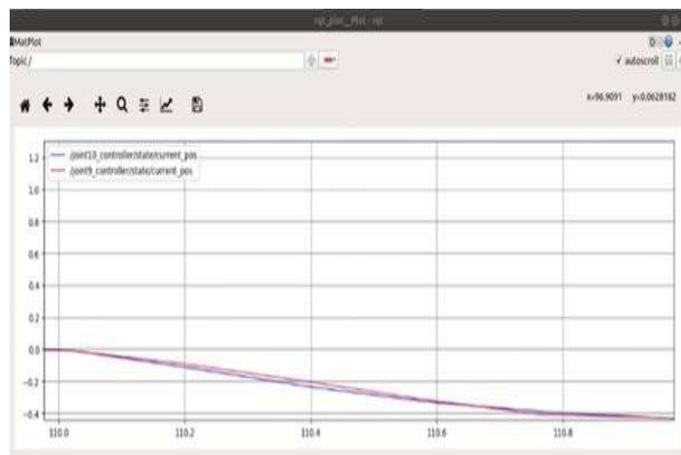
Pada Gambar 8(a) merupakan visualisasi dari robot condong ke kiri yang ada di dalam RViz. Pemberian nilai-nilai sudut pada setiap join yang diberikan diatur pada *node* yang terdapat pada *packages robot controller*. Pembuatan posisi robot condong ke kiri, dilakukan dengan mengatur nilai pada bagian *hip roll* kanan dan *hip roll* kiri sebesar 25°. Sedangkan pada gambar 8(b) merupakan implementasi dari gerakan condong ke kiri. Pemberian nilai-nilai sudut

pada aktuator ini didapatkan dari nilai yang diberikan pada RViz. Berikut merupakan tabel 3 yang menunjukkan nilai posisi sudut pada simulasi visualisasi pengiriman *node* dengan robot yang sudah diimplementasikan.

**Tabel 3. Nilai Sudut pada Gerakan Robot Condong ke Kiri**

No	Bagian Robot	Sudut Pada Visualisasi (°)	Sudut Pada Implementasi (°)	Error (%)
1.	Hip Yaw Kanan	0	0.625	0.9982
2.	<i>Hip roll</i> Kanan	25	25.14	0.0056
3.	Hip Pitch Kanan	0	0.877	0.9975
4.	Lutut Pitch Kanan	0	0.439	0.9987
5.	Engkel Pitch Kanan	0	0.275	0.9992
6.	Engkel Roll Kanan	0	0.375	0.9989
7.	Hip Yaw Kiri	0	0.352	0.9990
8.	<i>Hip roll</i> Kiri	25	24.94	0.0024
9.	Hip Pitch Kiri	0	0.439	0.9987
10.	Lutut Pitch Kiri	0	0.375	0.9989
11.	Engkel Pitch Kiri	0	0.439	0.9987
12.	Engkel Roll Kiri	0	0.352	0.9990
Rata-Rata Error				0.83

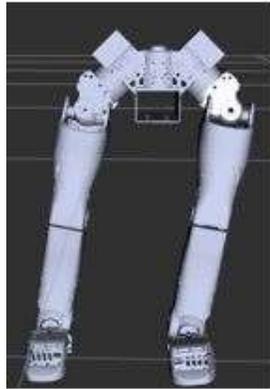
Pada Tabel 3 menunjukkan nilai pengaturan sudut yang dilakukan pada media visualisasi dan implementasi. Dari perencanaan gerakan robot condong ke kiri, dibuat pengaturan sudut pada bagian *joint hip roll* kanan dan *hip roll* kiri (Gambar 4). Hal ini bertujuan untuk membuat *pose* dari robot berkaki dua menjadi condong kiri. Pada percobaan kali ini dilakukan pemberian nilai sudut 25° pada bagian *hip roll* kanan dan *hip roll* kiri. Proses pengukuran sudut implementasi robot dilakukan dengan cara membaca *feedback* posisi. Nilai error yang didapatkan untuk pengaturan gerakan robot condong ke kiri pada *hip roll* kiri adalah 0.0024 %. Sedangkan nilai error yang didapatkan untuk gerakan robot condong pada pada *hip roll* kanan adalah 0.0056%. Sehingga didapatkan nilai error rata-rata pada gerakan ini sebesar 0.83%. Pada Gambar 9 berikut ini merupakan grafik dari perubahan sudut gerakan condong ke kiri.



**Gambar 9. Grafik Perubahan Sudut Gerakan Condong ke Kiri**

Salah satu keunggulan menggunakan ROS adalah terdapat *rqt\_plot*. *Rqt\_plot* merupakan sebuah *plugin* yang dapat memvisualisasikan nilai numerik dalam plot 2D menggunakan *backend plot* yang berbeda. Pada Gambar 9 merupakan grafik yang menunjukkan perubahan

sudut yang terbaca pada *hip roll* kanan dan *hip roll* kiri. Pada sumbu X menunjukkan respon terhadap waktu dan sumbu y merupakan nilai sudut dalam satuan radian. Grafik tersebut terlihat perubahan dari kondisi robot diam menjadi bergerak condong ke kiri ( $1 \text{ rad} = 57.29^\circ$ ). Nilai pembacaan pada *hip roll* kanan adalah 0.4384 rad. Sedangkan nilai pembacaan pada *hip roll* kiri adalah 0.4352 rad. Pada Gambar 10 berikut merupakan gambar yang menunjukkan visualisasi dan implementasi gerakan robot condong ke kanan.



(a) Visualisasi Condong ke Kanan



(b) Implementasi Condong ke Kanan

Gambar 10. Kondisi Robot Condong ke Kanan

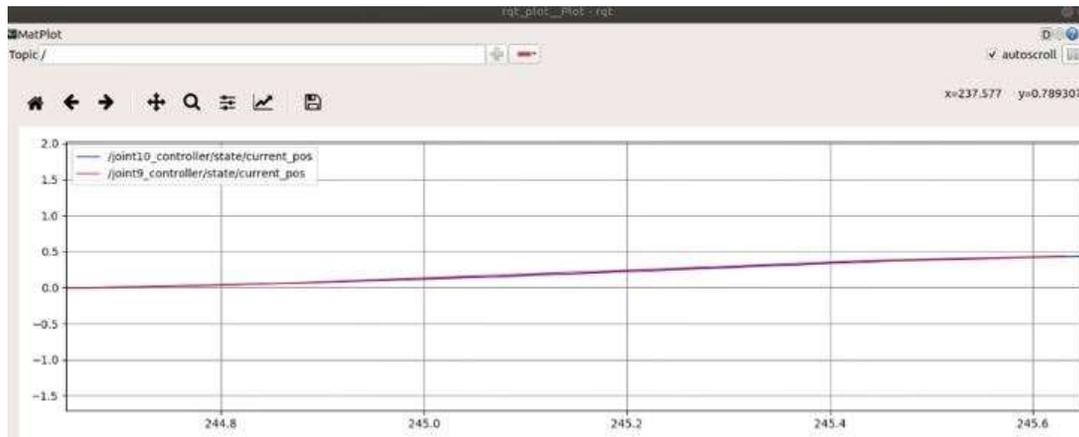
Pada Gambar 10(a) merupakan visualisasi dari robot condong ke kanan yang ada di dalam RViz. Pemberian nilai-nilai sudut pada setiap *joint* yang diberikan diatur pada *node* yang terdapat pada *packages robot controller*. Sedangkan pada Gambar 10(b) merupakan implementasi dari gerakan condong ke kanan. Pemberian nilai-nilai sudut pada aktuator ini didapatkan dari nilai yang diberikan pada RViz. Berikut merupakan Tabel 4 yang menunjukkan nilai posisi sudut pada simulasi visualisasi pengiriman *node* dengan robot yang sudah diimplementasikan.

Tabel 4. Nilai Sudut pada Gerakan Robot Condong ke Kanan

No	Bagian Robot	Sudut Pada Visualisasi ( $^\circ$ )	Sudut Pada Implementasi ( $^\circ$ )	Error (%)
1.	<i>Hip Yaw</i> Kanan	0	0.625	0.9982
2.	<i>Hip roll</i> Kanan	-25	-25.31	0.0120
3.	<i>Hip Pitch</i> Kanan	0	0.877	0.9975
4.	Lutut Pitch Kanan	0	0.439	0.9975
5.	Engkel Pitch Kanan	0	0.275	0.9992
6.	Engkel Roll Kanan	0	0.375	0.9989
7.	<i>Hip Yaw</i> Kiri	0	0.352	0.9990
8.	<i>Hip roll</i> Kiri	-25	-24.53	0.0188
9.	<i>Hip Pitch</i> Kiri	0	0.439	0.9987
10.	Lutut Pitch Kiri	0	0.375	0.9895
11.	Engkel Pitch Kiri	0	0.439	0.9987
12.	Engkel Roll Kiri	0	0.352	0.9990
Rata-Rata Error				0.84

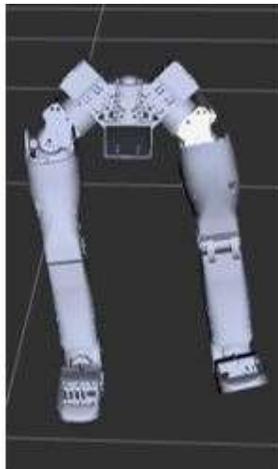
Pada Tabel 4 menunjukkan nilai pengaturan sudut yang dilakukan pada media visualisasi dan implementasi. Dari perencanaan gerakan robot condong ke kiri, dibuat pengaturan sudut pada bagian *joint hip roll* kanan dan *hip roll* kiri (Gambar 4). Hal ini bertujuan untuk membuat *pose* dari robot berkaki dua menjadi condong kiri. Pada percobaan kali ini dilakukan pemberian nilai

sudut  $-25^\circ$  pada bagian *hip roll* kanan dan *hip roll* kiri. Nilai (-) menunjukkan arah perputaran motor searah dengan jarum jam. Proses pengukuran sudut implementasi robot dilakukan dengan cara membaca *feedback* posisi. Nilai error yang didapatkan untuk pengaturan gerakan robot condong ke kiri pada *hip roll* kiri adalah 0.0188%. Sedangkan nilai error yang didapatkan untuk gerakan robot condong pada pada *hip roll* kanan adalah 0.012%. Sehingga didapatkan nilai error rata-rata pada gerakan ini sebesar 0.84%. Pada Gambar 11 berikut ini merupakan grafik dari perubahan sudut gerakan condong ke kanan:



**Gambar 11. Grafik Perubahan Sudut Gerakan Condong ke Kanan**

Pada Gambar 11 merupakan grafik yang menunjukkan perubahan sudut yang terbaca pada *hip roll* kanan dan *hip roll* kiri. Pada sumbu X menunjukkan respon terhadap waktu dan sumbu y merupakan nilai sudut dalam satuan radian. Grafik tersebut terlihat perubahan dari kondisi robot diam menjadi bergerak condong ke kanan ( $1 \text{ rad} = 57.29^\circ$ ). Nilai pembacaan pada *hip roll* kanan adalah 0.4417 rad. Sedangkan nilai pembacaan pada hiproll kiri adalah 0.4281 rad. Pada Gambar 12 berikut merupakan gambar yang menunjukkan visualisasi dan implementasi gerakan robot mengangkat satu kaki:



**(a) Visualisasi Mengangkat Satu Kaki**



**(b) Implementasi Mengangkat Satu Kaki**

**Gambar 12. Kondisi Robot Mengangkat Satu Kaki**

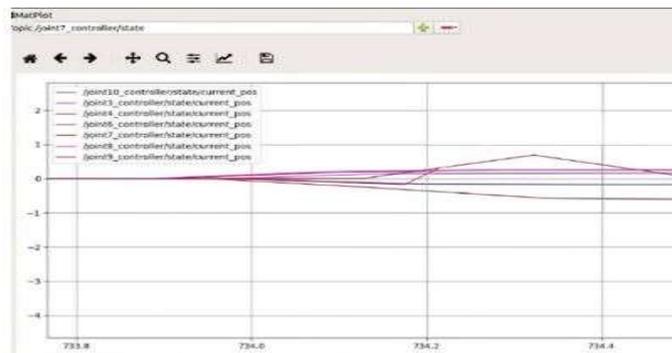
Pada Gambar 12(a) merupakan visualisasi dari robot mengangkat satu kaki yang ada di dalam RViz. Pemberian nilai-nilai sudut pada setiap joint yang diberikan diatur pada *node* yang

terdapat pada *packages robot controller*. Sedangkan pada Gambar 12(b) merupakan implementasi dari gerakan robot mengangkat satu kaki. Pemberian nilai-nilai sudut pada aktuator ini didapatkan dari nilai yang diberikan pada RViz. Berikut merupakan Tabel 5 yang menunjukkan nilai posisi sudut pada simulasi visualisasi pengiriman *node* dengan robot yang sudah diimplementasikan.

**Tabel 5. Nilai Sudut pada Gerakan Robot Angkat Satu Kaki**

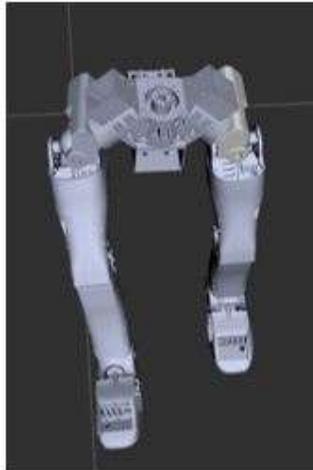
No	Bagian Robot	Sudut Pada Visualisasi (°)	Sudut Pada Implementasi (°)	Error (%)
1.	Hip Yaw Kanan	0	0.0878	0.9975
2.	Hip roll Kanan	10	10.2832	2.8320
3.	Hip Pitch Kanan	0	0.0878	0.9975
4.	Lutut Pitch Kanan	10	10.4062	4.0625
5.	Engkel Pitch Kanan	10	9.7273	2.7265
6.	Engkel Roll Kanan	0	0.8789	0.9975
7.	Hip Yaw Kiri	0	0.8789	0.9975
8.	Hip roll Kiri	10	10.2832	2.8320
9.	Hip Pitch Kiri	45	44.8760	0.2753
10.	Lutut Pitch Kiri	45	45.3109	0.6909
11.	Engkel Pitch Kiri	10	9.7832	2.1679
12.	Engkel Roll Kiri	0	0.8789	0.9975
Rata-Rata Error				1.71

Pada Tabel 5 menunjukkan nilai pengaturan sudut yang dilakukan pada media visualisasi dan implementasi dari perencanaan gerakan robot mengangkat satu kaki, dibuat pengaturan sudut pada bagian *joint hip roll* kanan dan *hip roll* kiri (Gambar 4). Hal ini bertujuan untuk membuat *pose* dari robot berkaki mengangkat satu kaki. Pada percobaan kali ini dilakukan pemberian nilai sudut 10° pada bagian *hip roll* kanan. Nilai error yang muncul antara visualisasi dan implementasi pada bagian ini adalah 2.83%. Nilai error yang muncul pada pemberian 10° lutut pitch kanan adalah 4.06%. Nilai error yang muncul pada pemberian 10° engkel pitch kanan adalah 2.72%. Nilai error yang muncul pada pemberian 45° hip pitch kiri adalah 0.27%. Nilai error yang muncul pada pemberian 45° lutut pitch kiri adalah 0.69%. Proses pengukuran sudut implementasi robot dilakukan dengan cara membaca *feedback* posisi. Perbedaan pembacaan nilai implementasi 0° pada aktuator disebabkan karena factor mekanik seperti *backlash* pada servo. Sehingga didapatkan nilai error rata-rata pada gerakan ini sebesar 1.71%. Pada Gambar 13 berikut ini merupakan grafik dari perubahan sudut gerakan angkat satu kaki.



**Gambar 13. Grafik Perubahan Sudut Gerakan Angkat Satu Kaki**

Pada Gambar 13 merupakan grafik yang menunjukkan perubahan sudut yang terbaca pada *hip roll* kanan, *hip roll* kiri, engkel pitch kanan, lutut pitch kanan, hip pitch kiri, engkel pitch kiri dan lutut pitch kiri. Pada sumbu X menunjukkan respon terhadap waktu dan sumbu y merupakan nilai sudut dalam satuan radian. Grafik tersebut terlihat perubahan dari kondisi robot diam menjadi bergerak condong ke kiri ( $1 \text{ rad}=57.29^\circ$ ). Nilai pembacaan pada hip pitch kiri adalah 0.78 rad. Sedangkan nilai pembacaan pada lutut pitch kiri adalah 0.79 rad.



(a) Visualisasi Kaki Kanan ke Depan



(b) Implementasi Kaki Kanan ke Depan

Gambar 14. Kondisi Kaki Kanan Robot ke Depan

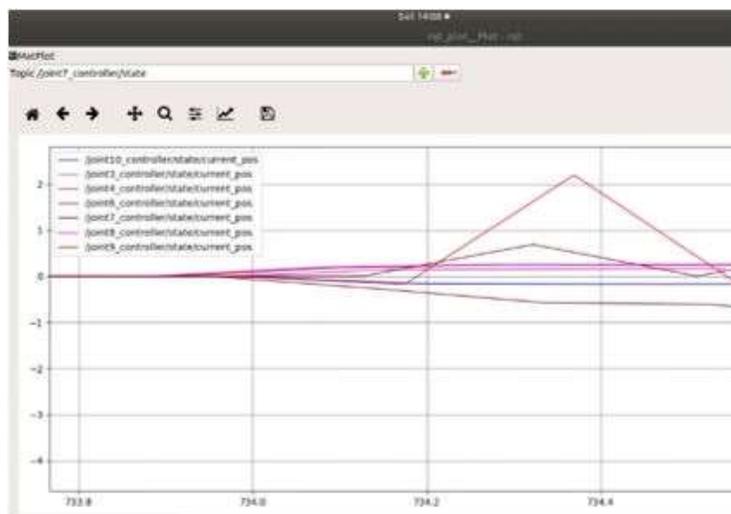
Pada Gambar 14(a) merupakan visualisasi dari kaki kanan robot ke depan yang ada di dalam RViz. Pemberian nilai-nilai sudut pada setiap *joint* yang diberikan diatur pada *node* yang terdapat pada *packages robot controller*. Sedangkan pada Gambar 14(b) merupakan implementasi dari kaki kanan robot ke depan. Pemberian nilai-nilai sudut pada aktuator ini didapatkan dari nilai yang diberikan pada RViz. Berikut merupakan Tabel 5 yang menunjukkan nilai posisi sudut pada simulasi visualisasi pengiriman *node* dengan robot yang sudah diimplementasikan.

Tabel 6. Nilai Sudut pada Gerakan Kaki Kanan Kedepan

No	Bagian Robot	Sudut Pada Visualisasi (°)	Sudut Pada Implementasi(°)	Error (%)
1.	<i>Hip Yaw</i> Kanan	0	0.5273	0.9985
2.	<i>Hip roll</i> Kanan	10	10.9296	9.2968
3.	<i>Hip Pitch</i> Kanan	35	35.2050	0.5859
4.	Lutut Pitch Kanan	0	0.7910	0.9978
5.	Engkel Pitch Kanan	15	16.2609	8.4062
6.	Engkel Roll Kanan	0	0.3730	0.9989
7.	<i>Hip Yaw</i> Kiri	0	0.2636	0.9992
8.	<i>Hip roll</i> Kiri	10	9.3878	6.1210
9.	<i>Hip Pitch</i> Kiri	10	11.1757	11.7578
10.	Lutut Pitch Kiri	0	0.8789	0.9975
11.	Engkel Pitch Kiri	15	14.8828	0.7812
12.	Engkel Roll Kiri	0	0.8789	0.9975
Rata-rata Error				3.58

Pada Tabel 6 menunjukkan nilai pengaturan sudut yang dilakukan pada media visualisasi dan implementasi. Dari perencanaan gerakan kaki kanan robot ke depan, dibuat pengaturan sudut

pada bagian *hip roll* kanan, *hip pitch* kanan, *engkel pitch* kanan, *hip roll* kiri, *hip pitch* kiri dan *engkel pitch* kiri (Gambar 4). Hal ini bertujuan untuk membuat *pose* dari gerakan robot kaki kanan ke depan. Proses pengukuran sudut implementasi robot dilakukan dengan cara membaca *feedback* posisi. Nilai error yang didapatkan untuk pengaturan *hip roll* kanan adalah 9.28%. Sedangkan untuk nilai error yang didapatkan untuk pengaturan *hip pitch* kanan 0.58%. Nilai error yang didapatkan untuk pengaturan *engkel pitch* kanan 8.4% Sehingga didapatkan nilai error rata-rata pada gerakan ini sebesar 3.58%. Pada Gambar 15 berikut ini merupakan grafik dari perubahan sudut gerakan kaki kanan ke depan.



**Gambar 15. Grafik Aktuator Robot Kaki Kanan di Depan**

Pada Gambar 15 merupakan grafik yang menunjukkan perubahan sudut yang terbaca pada *hip roll* kanan dan *hip roll* kiri. Pada sumbu X menunjukkan respon terhadap waktu dan sumbu y merupakan nilai sudut dalam satuan radian. Grafik tersebut terlihat perubahan dari kondisi robot diam menjadi bergerak condong ke kiri ( $1 \text{ rad} = 57.29^\circ$ ). Nilai pembacaan pada engkel pitch kanan adalah 0.28 rad. Grafik tersebut menunjukkan implementasi pada masing-masing aktuator ketika terjadi perubahan dari kondisi robot diam menjadi bergerak kaki kanan maju.

#### 4. KESIMPULAN

Berdasarkan pada pengujian yang telah dilakukan terhadap perencanaan sistem gerakan pada robot berkaki dua menggunakan ROS dan RViz dapat disimpulkan bahwa: Tingkat keberhasilan dari perencanaan gerakan ini memiliki rata rata error sebesar 1.8%. Pada gerakan condong ke kiri memiliki rata-rata error sebesar 0.83%. Kemudian untuk gerakan condong ke kanan memiliki rata-rata error sebesar 0.84%. Sedangkan untuk gerakan mengangkat satu kaki memiliki rata-rata error sebesar 1.71%. Selain itu, gerakan kaki kanan kedepan memiliki rata-rata error sebesar 3.83%. Error tersebut dapat terjadi karena faktor mekanik yang terdapat pada robot seperti *backlash* pada aktuator. Untuk selanjutnya sistem ini dapat diterapkan untuk pengembangan penelitian robot *humanoid* seperti meloncat, berlari dan sebagainya.

#### UCAPAN TERIMA KASIH

Terimakasih kepada Politeknik Elektronika Negeri Surabaya yang telah memberikan penulis beasiswa *freshgraduate* untuk program Pascasarjana Terapan. Terima kasih kepada Member

Laboratorium *Robotic and Intelligent Systems Center* (RoISC) yang selalu memberikan dukungan dan doa kepada penulis untuk dapat menyelesaikan penulisan makalah ini.

## DAFTAR RUJUKAN

- Cashmore, M., Fox, M., Long, D., Magazzeni, D., Ridder, B., Carrera, A., Palomeras, N., Hurtós, N., & Carreras, M. (2015). Rosplan: Planning in the robot operating system. *IEEE Proceedings International Conference on Automated Planning and Scheduling*, (pp. 333–341).
- Arfaq, M., Dewanto, R. S., & Pramadihanto, D. (2018). Fall Detection in T-FLoW Humanoid Robot:V-REP Simulation. *IEEE International Electronics Symposium on Engineering Technology and Applications*, (pp. 224–228). <https://doi.org/10.1109/ELECSYM.2018.8615463>
- Suhendra, T., & Priyambodo, T. K. (2017). Analisis Perbandingan Algoritma Perencanaan Jalur Robot Bergerak Pada Lingkungan Dinamis. *Indonesian Journal of Computing and Cybernetics Systems*, *11*(1), 21. <https://doi.org/10.22146/ijccs.15743>
- Santos, J. M., Portugal, D., & Rocha, R. P. (2013). An evaluation of 2D SLAM techniques available in Robot Operating System. In *IEEE International Symposium on Safety, Security, and Rescue Robotics*, (pp. 1-6). <https://doi.org/10.1109/SSRR.2013.6719348>
- Balakirsky, S., & Kootbally, Z. (2012). Usarsim/ros : A combined framework for robotic control and simulation. *IEEE International Symposium on Flexible Automation*. (pp. 101-108). [https://doi.org/10.11509/isciesci.56.12\\_635](https://doi.org/10.11509/isciesci.56.12_635)
- Fitri, N., & Syafrudin, R. M. (2012). Perancangan Sistem Kendali Gerak Lengan Robot STMIC: STMIC GI MDP.
- Prabowo, Y., Wisjhnuadji, T. W., & Wibowo, A. A. (2013). Aplikasi pid pada robot line follower berbasis mikrokontroler at-8535. *Jurnal Arsitron4*, *4* (1), 15–22.
- Allgeuer, P., Schwarz, M., Pastrana, J., Schueller, S., Missura, M., & Behnke, S. (2018). A ROS-based Software Framework for the Nimbro-OP Humanoid Open Platform. In *IEEE International Conference on Humanoid Robots (Humanoids)*.
- Tomás Cunha, Pedro M Vieira<sup>2</sup>, Kevin Costa, and C. P. S. (2016). Looking for motor synergies in Darwin-OP biped robot. In *IEEE International Conference On Robotics and Automation*. (pp. 1776-1781).
- Jalil, A. (2018). Robot Operating System (Ros) Dan Gazebo Sebagai Media Pembelajaran Robot Interaktif. *ILKOM Jurnal Ilmiah*, *10* (3), 284–289. <https://doi.org/10.33096/ilkom.v10i3.365.284-289>

- Kam, H. R., Lee, S. H., Park, T., & Kim, C. H. (2015). RViz: a toolkit for real domain data visualization. *Telecommunication Systems*, *60*(2), 337 - 345.
- Takaya, K., Asai, T., Kroumov, V., & Smarandache, F. (2016). Simulation environment for mobile robots testing using ROS and Gazebo. *IEEE International Conference on System Theory, Control and Computing*, (pp. 96–101).
- Kannan Megalingam, R., Ravi Teja, C., Sreekanth, S., & Raj, A. (2018). ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm. *International Journal of Pure and Applied Mathematics*, *118*(7), 199–205.
- Sokolov, M., Lavrenov, R., Gabdullin, A., Afanasyev, I., & Magid, E. (2016). 3D modelling and simulation of a crawler robot in ROS/Gazebo. *IEEE International Conference on Control, Mechatronics and Automation*, (pp. 61-65). <https://doi.org/10.1145/3029610.3029641>
- Robotis\_MX-106T*. (2020, May 20). Robotis e-Manual. Retrieved from <Http://Emanual.Robotis.Com/Docs/En/Dxl/Mx/Mx-106/>.