Distributed Control with Web-based 3D Visualization using Kinematics Analysis for IoRT

Z Iklima[1], T M Kadarina[1*]

[1] *Department of Electrical Engineering, Universitas Mercu Buana, Jl. Meruya Selatan No.31, Jakarta 11610*
[*] *Email: trie.maya@mercubuana.ac.id*

Article Information:

Received:
16 October 2020

Received in revised form:
17 November 2020

Accepted:
5 December 2020

Volume 2, Issue 2, December 2020
pp. 107 – 114

http://dx.doi.org/10.23960/jesr.v2i2.62

Abstract

The use of graphical user interface in web platform has been increasing in the recent decades. The concept of Internet of Robotic Things (IoRT) integrates heterogenous intelligent devices into a distributed architecture of platforms. This research is conducted to deliver proof of concept of distributed control of 2 DOF robot manipulator with web-based 3D visualization. The 3D modeling was developed using Three.js and the WebGL library. Synchronization was made using WebSocket communication on TCP network layer. This protocol allowed 3D mesh data broadcasted from web interface to manipulator with average response 7.86 ms. Moreover, using Denavit-Hartenberg convention, kinematic model facilitates the convention of end-effector position data into angular form and vice versa. The error percentage at a joint 1 is 4% and joint 2 of 2.96%. The error occurs because of the characteristic of inverse function called singularity in which the application value in each joint has the input values of and respectively to get the end-effector position on the 2 DoF Manipulator Robot.

Keywords: *WebGL, IoRT, Kinematics Analysis, 2DoF Manipulator Robot, WebSocket*

I. INTRODUCTION

The role of robotics in industry nowadays has increased and continue to grow to achieve higher quality and productivity goals. Robots are performing tasks that need to be done remotely or in dangerous location. With more general applications, engineers need to develop systems that can gather and integrate data to provide better control and enable the operators to work in a collaboration [1].

Robotic technology has been applied to the industry widely to improve the performance of the factories. On the other hand, researchers now hope that such technology can also improve other sectors, for example education, health care, medical aides, and household services [2][3].

The concept of Internet of Robotic Things (IoRT) integrates heterogenous intelligent devices into a distributed architecture of platforms. These platforms can be run in the cloud or at the edge. This concept leverages the present IoT technologies and the convergence of robotic devices to provide better robotic capabilities. It also aggregated IoT functionalities with new applications and provide new opportunities for business and ivestment in many sectors [4] [5].

Factories and medical or pharmaceutical facilities are now using robotic manipulators widely [6]. It is very common for robotic manipulators to be used in industrial and biomedical applications [7].

In a process of kinematic analysis of robotic manipulators, first we have to determine the Degree of Freedom (DOF) of the robot before choosing the reference frames and calculate the corresponding homogeneous matrix. To choose the reference frames in robotic applications, the Denavit-Hartenberg convention is frequently used [8][9].

Simulator-based analysis in the area of robotics is also becoming a trend. It is very useful for introductory level course in robotics since it gives initial familiarization of the system. Web Graphic Library (WebGL) is a JavaScript Application Programming Interface (API) for rendering 2D or 3D graphics in a browser supporting HTML 5. This can be use to develop web-based interfaces to visualize 3D simulations with interactive application. WebGL has been implemented in many ways for standard industrial robot simulations. Many JavaScript libraries such as Three.js, SceneJS, and BabylonJS uses WebGL and provide many features for 3D object visualization. Three.js is a high-level utility library to make a WebGL program. Combined with other libraries, it can be used to develop interactive application which has camera control, light control, and animations. The use of JavaScript in WebGL implementation is going on in

many fields of study [10].

It is also said that it is becoming more common today that the way of working is more distributed and there is a growing interest in collaborative remote 3D modeling systems. A similar application has been built by utilizing cross-platform WebGL rendering API to visualize 3D models on many devices. Synchronization was made on WebSocket based communication over a Node.js collaboration server. The Transmission Control Protocol (TCP) based WebSocket protocol have proved to be very efficient and suitable. This application was developed to help geographically dispered people in a real-time collaborative work [11].

This paper describes the implementation of distributed control of 2 DOF manipulator with web-based 3D visualization. The 3D modeling was developed using Three.js and the WebGL library. Synchronization was made using WebSocket communication on TCP network layer. This protocol allowed 3D mesh data broadcasted from web interface to manipulator. Moreover, using Denavit-Hartenberg convention, kinematic model facilitates the convention of end-effector position data into angular form and vice versa.

II. Materials and Methods

A. *Three.js and WebGL*

Three.js is a JavaSript library we can use to create and display 3D scene using WebGL technology. This open source library has some predefined commands for creating some basic 3D geometries needed to assemble and render a model of simple robotic configuration [10].

A browser-based user interface with 3D visualization can be built on Three.js to provide multiuser environment. A WebSocket-based communication is provided by a server to accommodate data exchange between clients. The web server stores and dispatches files of 3D model and host the web pages while the client side utilized web workers to optimize the communication between the client and the server. When a client sends a message to the server other client can receive the message and respond if needed [12].

B. *ESP8266 and SG90 Mini Servo*

The design of two degree of freedom (DOF) robotic manipulator in this research required ESP8266 Wi-Fi module which can be found in a NodeMCU v.3 microcontroller. The Wi-Fi module will enable the robotic manipulator to communicate with the server through a wireless local netowrk.

The robotic manipulator also needs two mini servos SG90 to create movements. These servos are attached to the NodeMCU microcontroller as depicted in Fig. 1 while the higher-level connection between the servos, NodeMCU, web server and the web-based client for 3D visualization can be seen in Fig. 2.

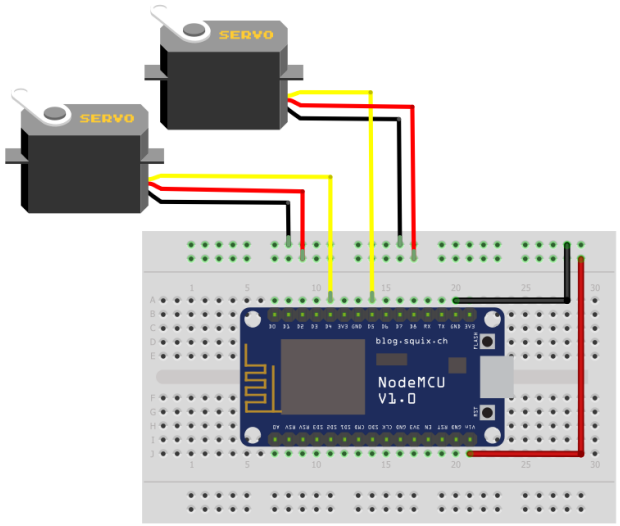

Figure 1. Wiring diagram for 2DoF robotic manipulator

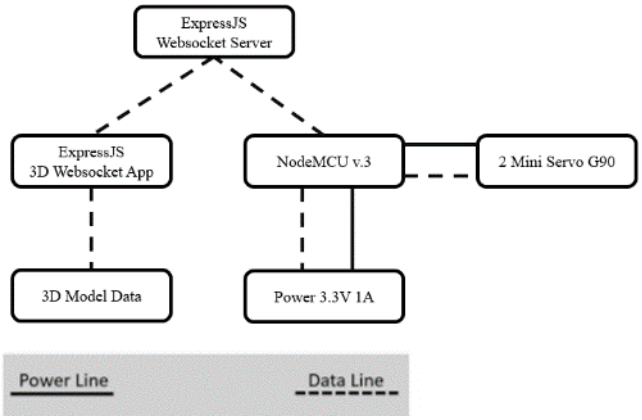

Figure 2. Block diagram of the system

As seen in Fig. 2 data in 3D model was sent using Websocket communication built using ExpressJS and ThreeJS framework therefore the 3D model can be accessed through the web platform. Using this platform, users can send data to server and then received by NodeMCU. The two SG90 servos linked to NodeMCU will work as actuator receiving the translated data from microcontroller and move the end effector to the desired position.

C. *Express.js and WebSocket*

The server is written in JavaScript and utilizes the WebSocket for communication. We use Express.js framework for relaying messaging to robotic manipulators. Therefore, 3D model can be accessed through the web platform and giving a new position data to the 3D object on the browser triggers a message transmission containing the new position to the remote

system's object.

Fig. 3 shows the importance of kinematic analysis to represent the position of the end-effector to get the actual value of the angle. Forward Kinematic (FK) and Inverse Kinematic (IK) is used to convert the data which will be sent to clients through the WebSocket server and received by WebSocket client in microcontroller.

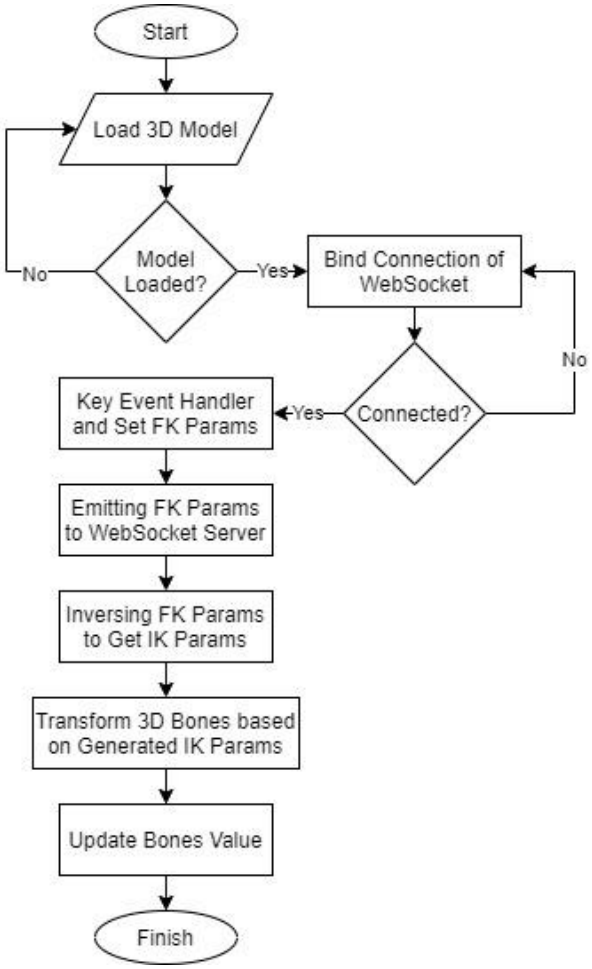

Figure 3. Flow chart for 3D application

Flow chart in Fig.4 shows Inverse Kinematic data obtained from the server has triggered change in servo's PWM based on data transmitted using WebSocket between ESP8266 (WebSocket client) and WebSocket server.

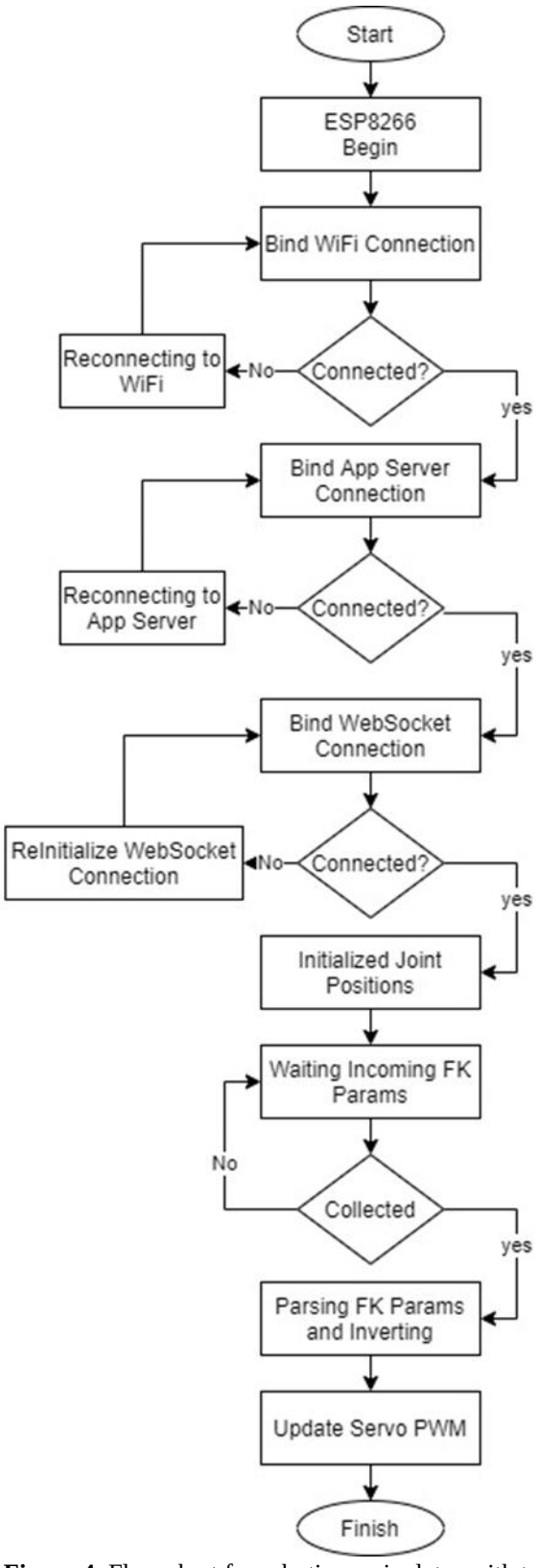

Figure 4. Flow chart for robotic manipulator with two degree-of-freedom

Architecture of 3D distributed control system is shown in Fig.5. An ExpressJS based server is responsible actively (WebSocket transmission) and passively (HTTP request/response).

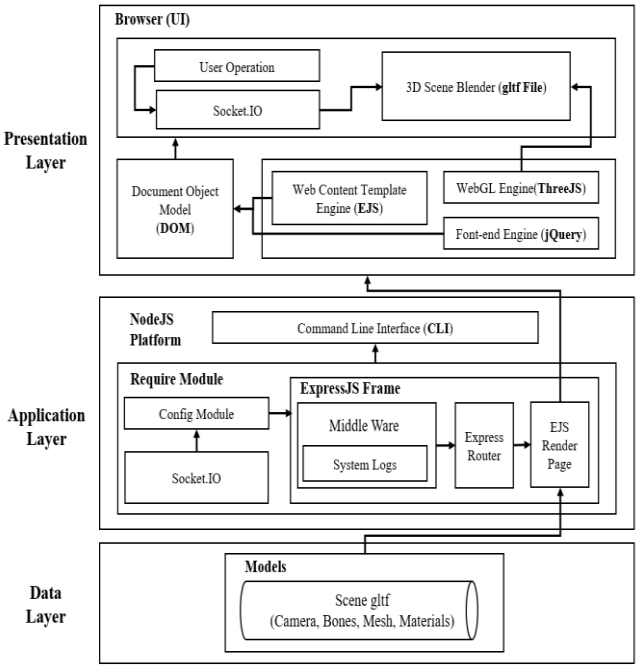

Figure 5. Architecture of 3D simulator application

D. Kinematic Analysis

There are two major aspects in the study of kinematic analysis, they are forward and inverse kinematic analysis of any manipulator. Forward kinematics analysis is carried out by assigning Denavit-Hartenberg or D-H parameters to all links to form the desired position and orientation. D-H parameters are the four parameters used to describe the connection between frames in a kinematic spatial network as follows:

- Joint offset (d): length of junctions of ordinary on the joint axis.
- Joint angle (θ): angle between the orthogonal projections of the common normal to the plane normal to the joint axes.
- Link length (a): the range among the common normal to the axis.
- Twist angle (α): the angle between the orthogonal projections of the joint axes onto a plane normal to the common normal. θ is variable if a joint is revolute, α is variable if a joint is prismatic.

Fig. 6 shows relationship between joint and length of 2 DoF manipulator used in this research. The schematic is used to get DH parameter as shown in Table 1.

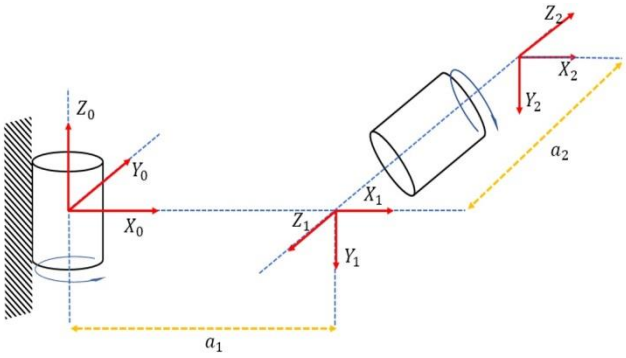

Figure 6. Schematic of DH parameters on robotic manipulator 2DoF

Table 1. DH parameter

n	θ	d	α	r
1	θ_1	0	$\frac{\pi}{2}$	L_1
2	θ_2	0	0	L_2

Using DH parameter shown in Table 1 we can obtain transformation formula as in (1) and (2).

$$ {}^{i-1}_{i}T = Trans(Z_{i-1}, d_i)Rot(Z_{i-1}, \theta_i)Trans(x_i, a_i)Rot(x_i, \alpha_i) \quad (1) $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} c(\theta_i) & -c(\alpha_i).s(\theta_i) & s(\alpha_i).s(\theta_i) & a_i.c(\theta_i) \\ s(\theta_i) & c(\alpha_i).c(\theta_i) & -s(\alpha_i).c(\theta_i) & a_i.s(\theta_i) \\ 0 & s(\alpha_i) & c(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2) $$

Where ${}^{i-1}_{i}T$ denotes as a Homogeneous Transformation Matrices of DH parameter from joint $i-1$ to joint i, which contains $Trans(Z_{i-1}, d_i)$ denotes as a translation in z-axis given by d_i, $Trans(x_{i-1}, a_i)$ denotes as a translation in x-axis given by a_i, $Rot(Z_{i-1}, \theta_i)$ denotes as a rotation in x-axis given by θ_i, and $Rot(x_{i-1}, \alpha_i)$ denotes as a rotation in z-axis given by α_i. Hereinafter, $c(\theta_i)$ declared as $cos(\theta_i)$, $s(\theta_i)$ declared as $sin(\theta_i)$, $c(\alpha_i)$ declared as $cos(\alpha_i)$, and $sin(\alpha_i)$ declared as $sin(\alpha_i)$. Based on values in Table 1 and (2) transformation value of joint 0 against joint 1 and transformation of joint 1 against joint 2 as in (3).

$$ {}^{0}_{1}T = \begin{bmatrix} c_1 & 0 & s_1 & L_1c_1 \\ s_1 & 0 & -c_1 & L_1s_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{1}_{2}T = \begin{bmatrix} c_2 & -s_2 & 0 & L_2c_2 \\ s_2 & c_2 & 0 & L_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

Where ${}^{0}_{1}T$ denotes as Homogeneous Transformation Matrices of DH parameter from joint 0 to joint 1, ${}^{1}_{2}T$ denotes as Homogeneous Transformation Matrices of DH parameter from joint 1 to joint 2, L_1 equal to a_1, and L_2 equal to a_2. Having transformation value for each join then Forward Kinematic and Inverse Kinematic translation can be done.

Therefore, from the value of θ_1 and θ_2 we have the formula for forwad kinematic as shown in (4), (5), (6),

(7) and (8) where L is link length (a).

$$ {}^{0}_{2}\boldsymbol{T} = {}^{0}_{1}\boldsymbol{T}\,{}^{1}_{2}\boldsymbol{T} \tag{4} $$

$$ {}^{0}_{3}T = \begin{bmatrix} \dots & \dots & \dots & L_2c_1c_2 + L_1c_1 \\ \dots & \dots & \dots & L_2s_1c_2 + L_1s_1 \\ \dots & \dots & \dots & L_2s_2 \\ \dots & \dots & \dots & 1 \end{bmatrix} = \begin{bmatrix} \dots & \dots & \dots & P_x \\ \dots & \dots & \dots & P_y \\ \dots & \dots & \dots & P_z \\ \dots & \dots & \dots & 1 \end{bmatrix} \tag{5} $$

$$ \boldsymbol{P_x = L_2c_1c_2 + L_1c_1} \tag{6} $$

$$ \boldsymbol{P_y = L_2s_1c_2 + L_1s_1} \tag{7} $$

$$ \boldsymbol{P_z = L_2s_2} \tag{8} $$

Where $P_x, P_y, and\ P_z$ express the end-effector coordinate distributed in x-axis, y-axis, and z-axis. Using the formula in (6), (7), and (8) we can calculate data from clients sent through WebSocket communication to get the value of inverse kinematic (θ_1 and θ_2) as in (10), (11), (12), (13), (14), (15).

$$ s_2 = p_z/L_2 \tag{10} $$

$$ c_2 = \sqrt{1 - {s_2}^2} \tag{11} $$

$$ \boldsymbol{\theta_2 = Atan2(s_2, c_2)} \tag{12} $$

$$ c_1 = (p_x)/(L_1 + L_2c_2) \tag{13} $$

$$ s_1 = (p_y)/(L_1 + L_2c_2) \tag{14} $$

$$ \boldsymbol{\theta_1 = Atan2(s_1, c_1)} \tag{15} $$

Those values of θ_1 and θ_2 are used to update the angle of the arm model in web-based application and in NodeMCU.

E. Design of 3D Model of 2 DOF Robot Manipulator

The 3D model of 2 DOF robot manipulator has been designed based on schematic in Figure 6 with values L1 = 2.75 cm and L2 = 4.67 cm. Therefore, 3D design of the model can be shown as in Fig. 7.

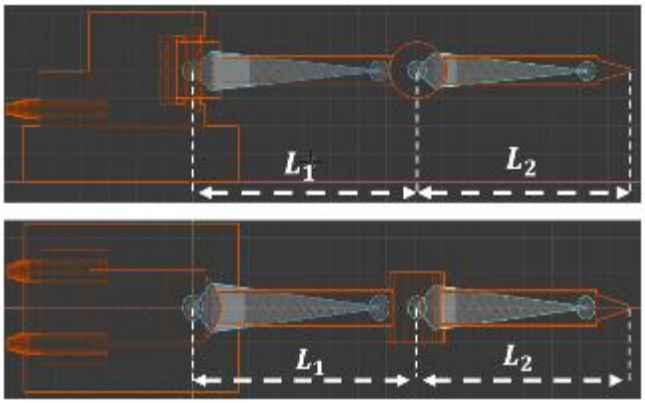

Figure 7. 3D Model of 2 DOF Robot Manipulator

The 3D model is designed with two bones as actuator of 2 DOF manipulator. The model is exported as gltf format where contain data of bones, material, mesh, geometries, etc. The exported file can be used to render a 3D image as depicted in Fig. 8.

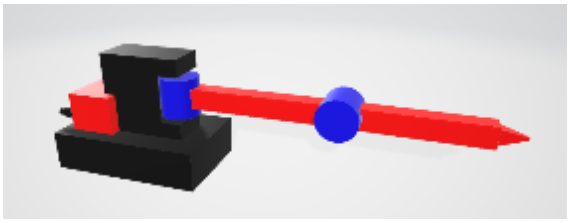

Figure 8. Rendered 3D Model of 2 DoF Robot Manipulator

In order to define the trajectory of 2 DOF robot manipulator we first defined minimum value, central value and maximum angular value (T_p) also called triangular trajectory. The triangular trajectory is illustrated in Fig. 9.

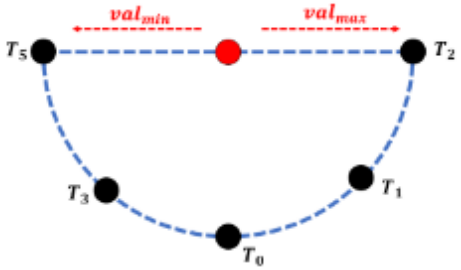

Figure 9. Triangular Trajectory

The value of triangular trajectory is defined to integrate the value in virtual environment (3D application) against PWM value in microcontroller (2 DOF robot manipulator) which is used to move the servo to the desired position.

F. Design 2 DOF Robot Manipulator

The wiring diagram of the 2 DOF robot manipulator can be seen in Fig. 1. There are two SG90 mini servo act as linked actuator with value L1 and L2. There is a PWM pin in NodeMCU v.3 therefore the servos can be operated using PWM signal.

III. RESULTS AND DISCUSSIONS

A. Kinematic Analysis Test Results

Velocity representation of each link and joint of 2 DOF manipulator are simulated using RoboAnalyzer Software. Output representation of forward kinematic analysis where it has 3 variables (p_x, p_y and p_z) can be seen in Fig. 11.

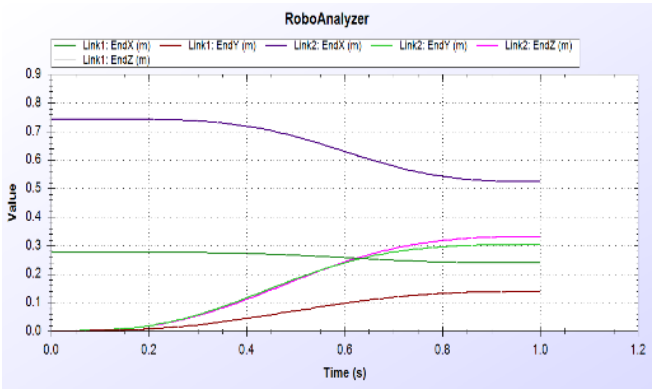

Figure 11. Representation of end-effector values

This simulation used input values $q_1 = 30^o$ and $q_2 = 60^o$ within 1 second. The simulation representation of each joint both velocity and angular values can be seen in Fig.12.

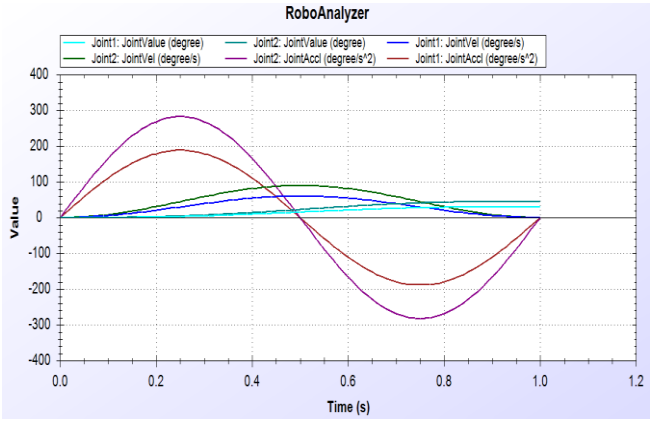

Figure 12. Representation of joint velocity and joint angular values

Fig.13 represent an imperceptible comparison between desired degree and generated degree (θ_1 and θ_2).

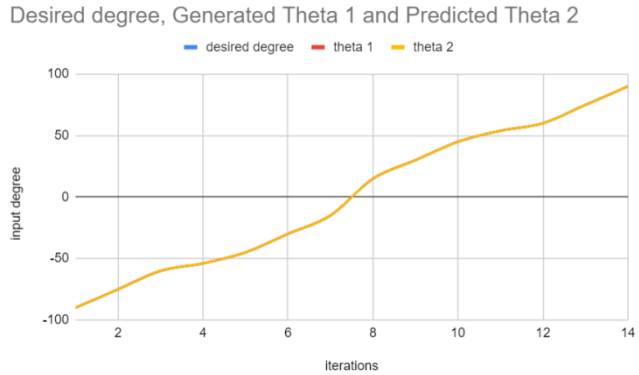

Figure 13. IK Result

Fig. 14 clearly shows the value of the error feedback to the inverse kinematic function. Error value at joint 1 is 0.037% and an error value at joint 2 is 0.075%. The error value is caused by the characteristic inverse function called singularity where in its application the value of each joint has an input value of each equal to q_1 and q_2 to get the end-effector position on the 2 DoF manipulator.

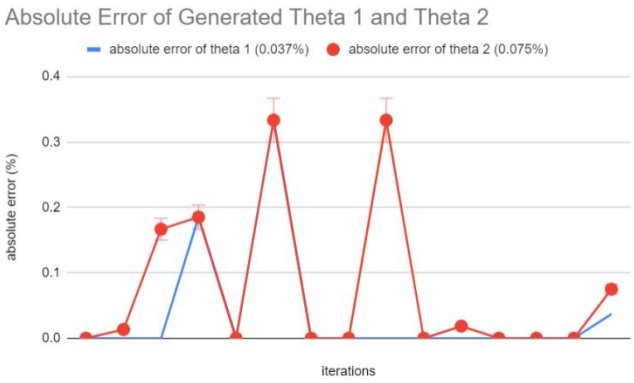

Figure 14. Absolute Error of IK

B. Network and Data Test Results

WebSocket has an important role for the server to broadcast data to all connected clients. Fig. 14 shows how FK data is received by the WebSocket Server and then broadcasts to all WebSocket Clients with an event (update_pos). This event consists of 2 types, the first is when the WebSocket Client or Server waits for the data to be sent to the WebSocket room called the listening event or on_listening event. The second is when the WebSocket Client or Server triggers or sends data to the WebSocket room called the emit event or the on_emit event.

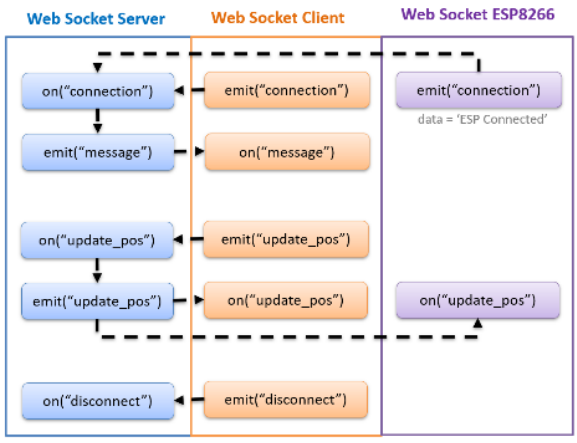

Figure 15. WebSocket Sequence

The service for the sequence (Fig.15) is created on the server, client app and ESP8266 clients. So that the data traffic in the WebSocket room can be seen on these 3 sides as shown in Fig. 16.

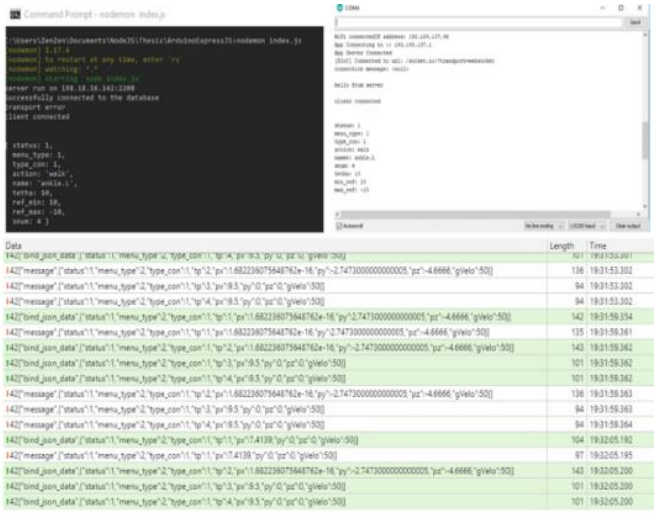

Figure 16. WebSocket Data Traffic

Data traffic clearly shows 3 parameters (p_x, p_y and p_z) that are set based on forward kinematic analysis calculations. The data for this parameter has an average capacity of 122B with an average response time of 7.86 ms as shown in Table 2.

Table 2. WebSocket Client Data Traffic

Event	Data Length (B)	Response Time (ms)
Emit	131	12
Listening	111	6
Emit	131	12
Listening	111	3
Emit	131	12
Listening	111	6
Emit	129	4
Mean	**122**	**7.86**

WebSocket Loader on the client side is 332 B (actual size 103 B) with a 132ms server response as shown in Table 3.

Table 3. WebSocket Loader

Name	Status	Type	Initiator	Size ▼	Time
?EIO=3&transport=poll...	200	xhr	index.js:83	332 B	3 ms
?EIO=3&transport=poll...	200	xhr	index.js:83	230 B	129 ms

WebSocket Loader used in the 3D App is a pooling made by WebSocket Handshake (pooling) on both the server and client sides. This concept is referred to as distributed where the server can actively broadcast via WebSocket network.

C. Implementation Results

The 3D model visualization of the system has succesfully implemented (Fig.17). Three JS (WebGL engine) can runs 3D models in the browser that contain data of mesh, materials, geometries and bones.

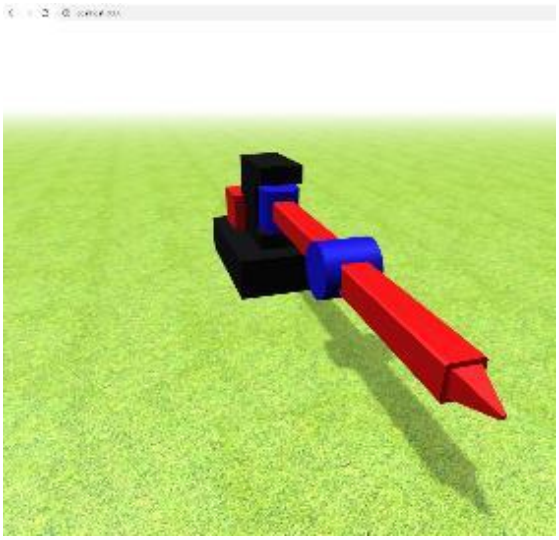

Figure 17. 3D Model Visualization

In Fig. 18, the 3D app is successfully combined with WebSocket technology so that this system can support the IoRT platform as a realtime system.

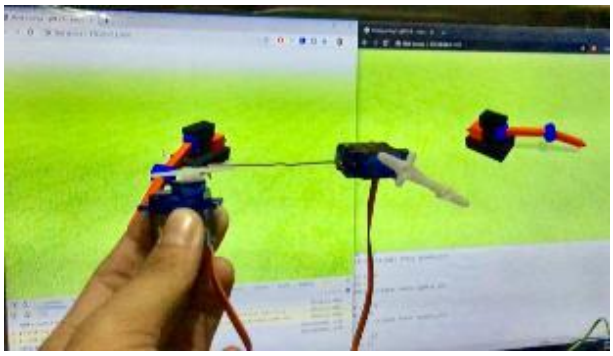

Figure 18. Results of 2DoF robot manipulator control with web-based 3D App

The control system is listening to key events of ASCII number for key A (65), key D (68), key W (87), and key Z (88). The triggered keys correspond to a change in angle value of +1 or -1. WebSocket technology was successfully implemented by testing data transmisison from 3D app client through server and received by NodeMCU client.

IV. CONCLUSIONS

This paper illustrates the utilization of Three.js library for developing the 3D model of the 2DoF robot manipulator which is controlled using kinematic analysis and distributed using Express.js and Websocket based application server. It uses the technology of WebGL, which provides a very efficient platform for 3D object visualization. Using the library of Three.js helps in the direct rendering of the 3D object in Cartesian space. This environment can be further used for Forward and Inverse Kinematics analysis

calculated in a JavaScript. Here multiple robots motion simulation can also be incorporated into the same environment, which is a unique feature in this type of methodology. Since the whole analysis is carried out considering actual dimensions and kinematic parameters, it is almost similar to real system operation. It can be further programmed to connect this simulation model and (distributed) real-time control system of the actual robot for remote operation.

ACKNOWLEDGMENT

This research is supported by Research Institute of Universitas Mercu Buana contract ID 02-5/234/B-SPK/III/2020.

REFERENCES

[1] A. Dünser, M. Lochner, U. Engelke, and D. R. Fernández, "Visual and manual control for human-robot teleoperation," *IEEE Comput. Graph. Appl.*, 2015.

[2] R. C. Luo, M. J. Hong, and P. C. Chung, "Robot artist for colorful picture painting with visual control system," 2016.

[3] N. Ferdana, A. Adriansyah, S. Budiyanto, and J. Andika, "Design of a telemedicine robot using behavior-based control architecture," 2020.

[4] O. Vermesan *et al.*, "Internet of robotic things-converging sensing/actuating, hyperconnectivity, artificial intelligence and IoT platforms," in *Cognitive hyperconnected digital transformation: Internet of things intelligence evolution*, 2017.

[5] O. Vermesan and J. Bacquet, *Cognitive hyperconnected digital transformation: Internet of things intelligence evolution*. 2017.

[6] M. A. Adly and S. K. Abd-El-Hafiz, "Inverse kinematics using single-and multi-objective particle swarm optimization," 2016.

[7] M. M. U. Atique and M. A. R. Ahad, "Inverse Kinematics solution for a 3DOF robotic structure using Denavit-Hartenberg Convention," 2014

[8] A. N. Barakat, K. A. Gouda, and K. A. Bozed, "Kinematics analysis and simulation of a robotic arm using MATLAB," 2017.

[9] L. Qingsheng and J. Andika, "Analysis of kinematic for legs of a hexapod using denavit-hartenberg convention," *Sinergi*, 2018.

[10] U. Dey and K. Cheruvu Siva, "Kinematic Analysis and Simulation of a 6 DOF Robot in a Web-Based Platform Using CAD File Import," 2018.

[11] M. Wenzel, A. Klinger, and C. Meinel, "Tele-board prototyper - Distributed 3D modeling in a web-based real-time collaboration system," 2016

[12] B. Chen and Z. Xu, "A framework for browser-based Multiplayer Online Games using WebGL and WebSocket," 2011.