Modelling and Analysis of a Free-floating Robot with Flexible Links in SIMPACK

Daichi Hirano

DLR

Institut für Robotik und Mechatronik	BJ.: 2015
	IB. Nr.: 572-2015/03

MODELING AND ANALYSIS OF A FREE-FLOATING ROBOT WITH FLEXIBLE LINKS IN SIMPACK

Freigabe:	Der Bearbeiter:	Unterschriften
	Daichi Hirano	Daichi
	Betreuer:	
	Roberto Lampariello	Lampariello
	Der Institutsdirektor	
	Prof. Dr. A. Albu-Schäffer	Albu-Schäffer

Dieser Bericht enthält 46 Seiten, 46 Abbildungen und 1 Tabellen

Ort: Oberpfaffenhofen	Datum:	Bearbeiter:	Zeichen:

List of Contents

1. Introduction

1.1 Purpose of this document

This document describes the technical manual and results of dynamic simulation for the DEOS project using SIMPACK, which is a software to emulate the dynamic motion of multibody system including flexible/deformable bodies. While this document explains how to set up the simulation, the original user document of SIMPACK should be referred for understanding how to use the user interface and fundamental functions at the first step. In this document, technical manual to explain how to set up a dynamic model and how to perform dynamic simulation in SIMPACK. In addition, several simulations are performed to verify the developed model and to analyze the extent of disturbance in the manipulator motion due to the link flexibility. In the first simulation, the dynamic model is compared with the Nastran model that ASTRIUM made. The second simulation compares the motions in cases that the base is fixed or free-floating. The third simulation verifies the effect of link flexibility in the docking motion of a target. In addition, the resonance behavior is verified though the simulation. Finally, the conclusion of these verifications is given.

The one of main purposes of this document is to provide technical supports for performing dynamic simulation of flexible multi-body system in SIMPACK. The original user document of SIMPACK does not include the details and is not easy to understand them. This document provides the details and additional information that the original user document lacks. Readers would be able to create a proper dynamic model and perform simulation after reading this document.

1.2 General Information for SIMPACK

<u>Start up SIMPACK</u>

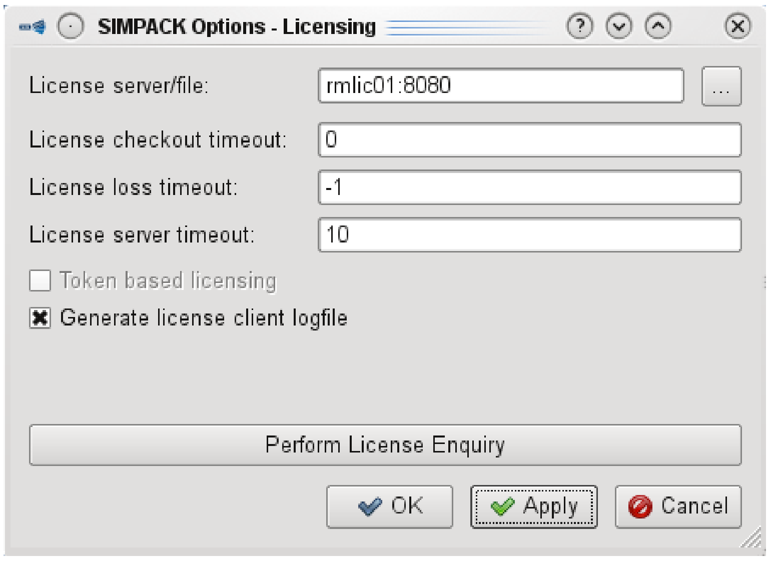
The command to start up SIMPACK in Linux is as follows.

/opt/simpack/s_97_64/run/bin/linux64/simpack-gui

Note: The software version should be changed on the basis of your computer.

When you start the SIMPACK at the first time, you need to fill out the license information in accordance with the procedure as below.

- Click on “Extras” ⇒ “Licensing…” in the menu bar.
- Put “rmlic01:8080” in the license server/file as shown in the following figure.
- Click “Apply”.


SIMPACK Options - Licensing

License server/file: rmlic01:8080

License checkout timeout: 0

License loss timeout: -1

License server timeout: 10

☐ Token based licensing

☒ Generate license client logfile

Fig. 1.2.1 SIMPACK license dialog.

User Reference Document of SIMPACK

After starting up SIMPACK, you can find the original user document of SIMPACK by pushing the "F1" key button or going "Help" ⇒ "Documentation" in the tool bar. It is recommended to refer the user document with this technical report, because this technical report skips some details that can be found in the user document. If you are a beginner or have never used SIMPACK, it is recommended to learn how to use SIMPACK at the first step with the tutorial training, which is contained in the user document (Section E.1).

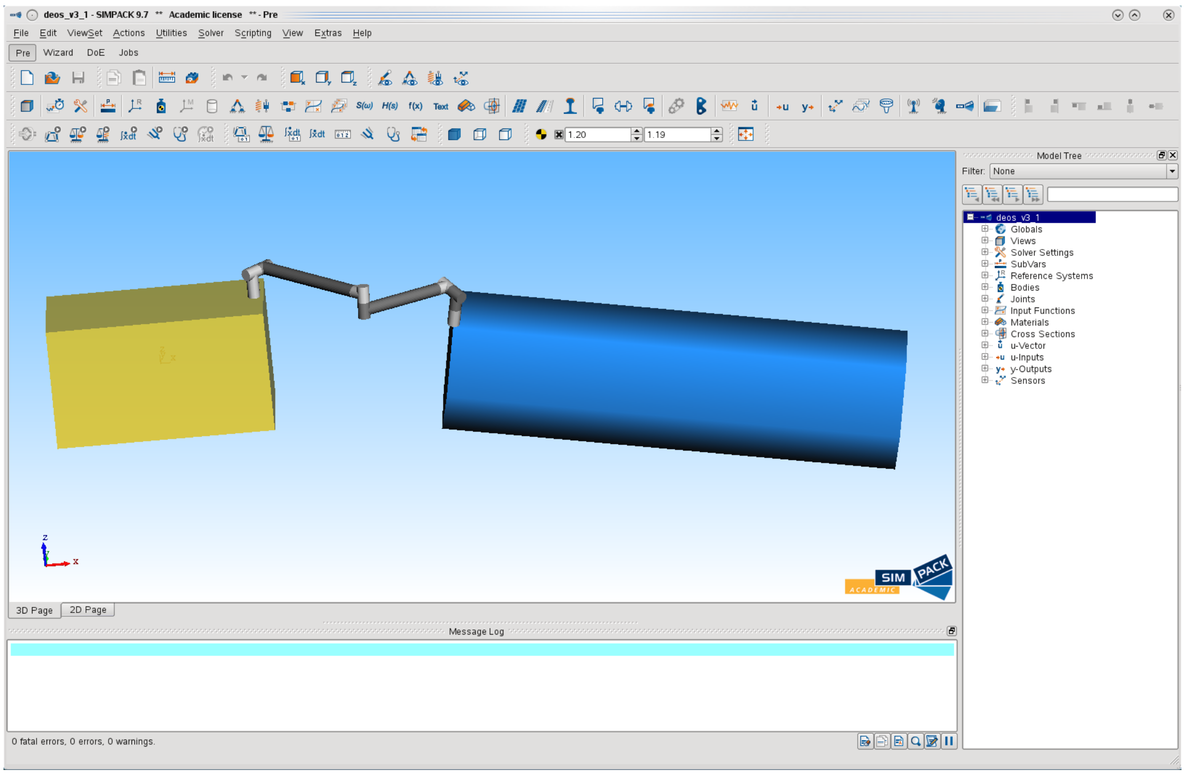
Sample Files

The sample files and simulation results that this report refers are located at the following folder.

report/data

User Interface

The user interface in SIMPACK can be classified into main three parts: the tool bar in the upper part, the graphic window in the middle, and the model tree in the right part. The tool bar has many functions for modeling and simulation. The graphic window displays the model you create. The model tree shows the parameters used in the model, and you can access and change the parameters from this tree. For more details, please see the user document of SIMPACK.



```
0 fatal errors, 0 errors, 0 warnings.
```

Fig. 1.2.2 SIMPACK user interface with graphics.

Import Files

Several structure files can be imported into SIMPACK from other software to create a multi-body model. Although CAD data can be imported using several supported file's formats such as .stl file, these files do not include the structural properties and cannot be used as flexible/deformable bodies (These files can be used as only rigid body). In SIMPACK, the .fbi (Flexible Body Input) file is required for importing flexible/deformable bodies. The .fbi file can be created in SIMPACK from the specific files, which can be exported from structural analysis software such as ANSYS and Nastran. For more detailed information, please see the user document of SIMPACK.

License of SIMPACK in DLR

DLR has the SIMPACK license. But the license for a flexible beam called SIMBEAM is only one, which is bought by another institute (not RMC). To check the license information and who uses the license, command the following text in your command line.

/opt/simpack/s_93/partners/cosin/ftire/bin/linux32/olixtool -OLicenseServer rmlic01:8080

2. SIMPACK Manual

2.1 Dynamic Model

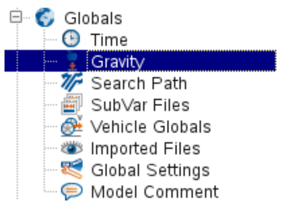
This section describes how to set up a dynamic model of multibody system. The following first section explains how to build a "**rigid**" dynamic model with joint constraints. The second section describes the setting for "**flexible**" bodies.

Rigid Model (General Model)

This section describes the general procedure to create a rigid multibody system. Each body is created and connected by a joint constraint with geometric points called "Marker" that you can define on the created bodies. The Sensors are defined at the appropriate place to output the simulation result (e.g. Base position and joint angle). The y-Outputs are set to output the Sensor's value into the simulation result file.

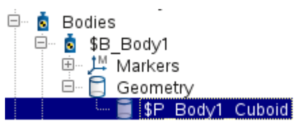
1) Create new model
   - Click on "File" ⇒ "New" or click the icon in the tool bar.
   - Choose "General" and click "OK" in the appeared dialog (Create New Model).
   - Save the model file (.spck file) with an appropriate name in a folder.

2) Set "Gravity"
   - Double click the following gravity icon in the model tree, and set the gravity values in accordance with your simulation.



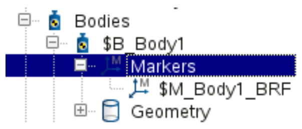
Note:
The default value of the gravity is -9.81 [m/s^2]. So, when you perform simulation of free-floating motion in microgravity, it should be zero.

3) Create new "Bodies"
   - Double click the icon Bodies in the model tree, and enter a name for the new body.
   - Set body properties (Mass, Center of Gravity, and Inertia) in the appeared dialog (Body Properties), and click "OK".
   - Double click the following geometry icon to open the dialog (Primitive Properties), and chose the geometry (Type, Position, Angles, Size, etc.).



4) Create “Markers”

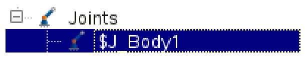
- Double click the following marker icon in the model tree, and name the new marker.



- In the appeared dialog (Marker Properties), set the marker at the appropriate position and angle that you want to create the joint on the selected body.

5) Create “Joint Constraints”

- Double click the following joint icon in the model tree.



- In the appeared dialog (Joint Properties), choose “From Marker” and “To Marker” as the connecting markers from the different bodies that you created in the above step, and choose the constraint type (e.g. Revolute Joint ga).

Note:
When you perform simulation of <u>free-floating</u> system, the joint type of the base’s constraint with respect to the inertial coordinate should be “6 DOF al-be-ga” (free motion without any constraints).

6) Create “Sensors”

- Right click on the model tree and choose “Create” ⇒ “Sensors” or click the icon in the tool bar, name the sensor, and click “OK”.
- Choose “From Marker” and “To Marker” from the created markers, and select the reference coordinate.

7) Create “y-Outputs”

- Right click on the model tree and choose “Create” ⇒ “y-Outputs” or click the icon in the tool bar, name the y-output, and click “OK”.
- Choose the type of sensor (e.g. Sensor Position and Sensor Angular Position), and select the appropriate sensor from the created sensors and choose the direction/magnitude.

<u>Flexible Model (SIMBEAM)</u>

While a FEM model can be imported from structure analysis software (e.g. ANSYS and Nastran), this section describes the setting of flexible bodies using SIMBEAM, which is a beam model provided by SIMPACK. To create SIMBEAM, you need to input the material information and the cross section of the flexible beam. The following section explains the procedure to define the flexible body.

Note:
DLR has only one license of SIMBEAM, and this license is bought by another institute (not RMC). When using SIMBEAM, keep in touch with them.

1) Create "Material"

- Right click on the model tree and choose "Create" ⇒ "Material" or click the icon in the tool bar, name the material, and click "OK".
- In the appeared dialog, set the material information (e.g. Density, Young's modulus, and Poisson ratio).

Example:
In the case of Aluminum, the parameters are as follows.

- Density: 2700 [kg/m^3]
- Young's modulus: 69*10^9 [N/m^2]
- Poisson's Ratio: 0.334 [-]

2) Create "Cross Sections"

- Right click on the model tree and choose "Create" ⇒ "Cross Sections" or click the icon in the tool bar, name the cross section, and click "OK".
- In the appeared dialog (Cross Section Properties), choose the appropriate type and relevant parameters (e.g. Diameter). Additionally, choose the created material in the above step.

Example:
DEOS arm is designed as a hollow cylinder (tube), and their parameters are as follows.

- Outer diameter: 127 [mm]
- Inner diameter: 123 [mm]

3) Set "SIMBEAM"

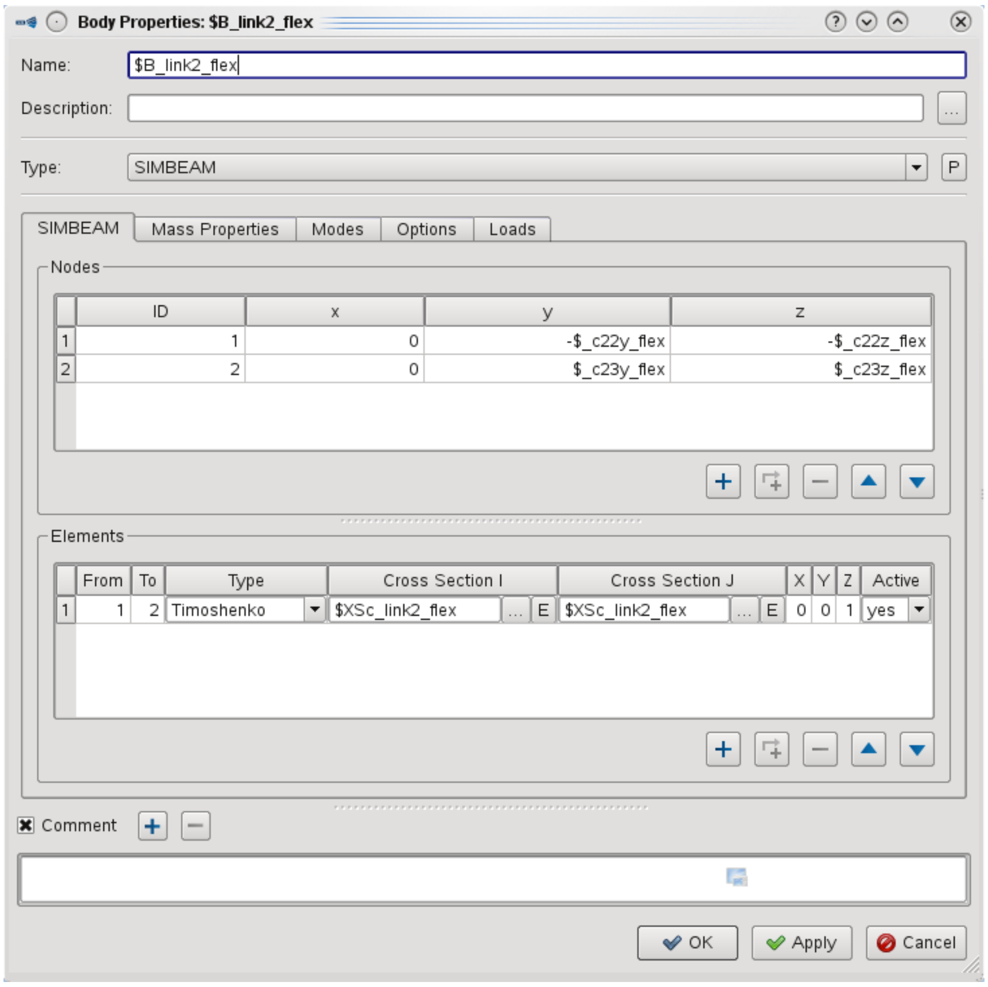
- Click the body properties and change the type into "SIMBEAM".
- In the "SIMBEAM" tab, define "Nodes" for markers with the ID number
  - Nodes can be added by clicking "+".

- ID should be assigned as 1, 2, 3…
- Each Node position is defined in the “Nodes” table.

- In the “SIMBEAM” tab, define the beam in the “Elements” table.
  - Select “Node ID” in “From” and “To”.
  - Choose beam type in “Type” (e.g. Euler-Bernoulli or Timoshenko).
  - Select “Cross Section I” and “Cross Section J” from the created “Cross Sections”.
  - Set reference line (e.g. X=0, Y=0, Z=1).

- In the “Modes” tab, define frequencies you consider in the simulation.
  - Select “Eigenmode” as you need (e.g. “f-min, n-modes”, f-min =0, n-modes = 3).
  - Select “IRM” (Inertia Relief Mode) or “FRM” (Frequency Response Mode) as you need.
  - Set damping type and values.

Note:
Small damping can cause the simulation error. So, the damping value should be carefully chosen.



Name: $B_link2_flex

Description:

Type: SIMBEAM

Nodes

	ID	x	y	z
1	1	0	-$_c22y_flex	-$_c22z_flex
2	2	0	$_c23y_flex	$_c23z_flex

Elements

	From	To	Type	Cross Section I	Cross Section J	X	Y	Z	Active
1	1	2	Timoshenko	$XSc_link2_flex … E	$XSc_link2_flex … E	0	0	1	yes

Comment

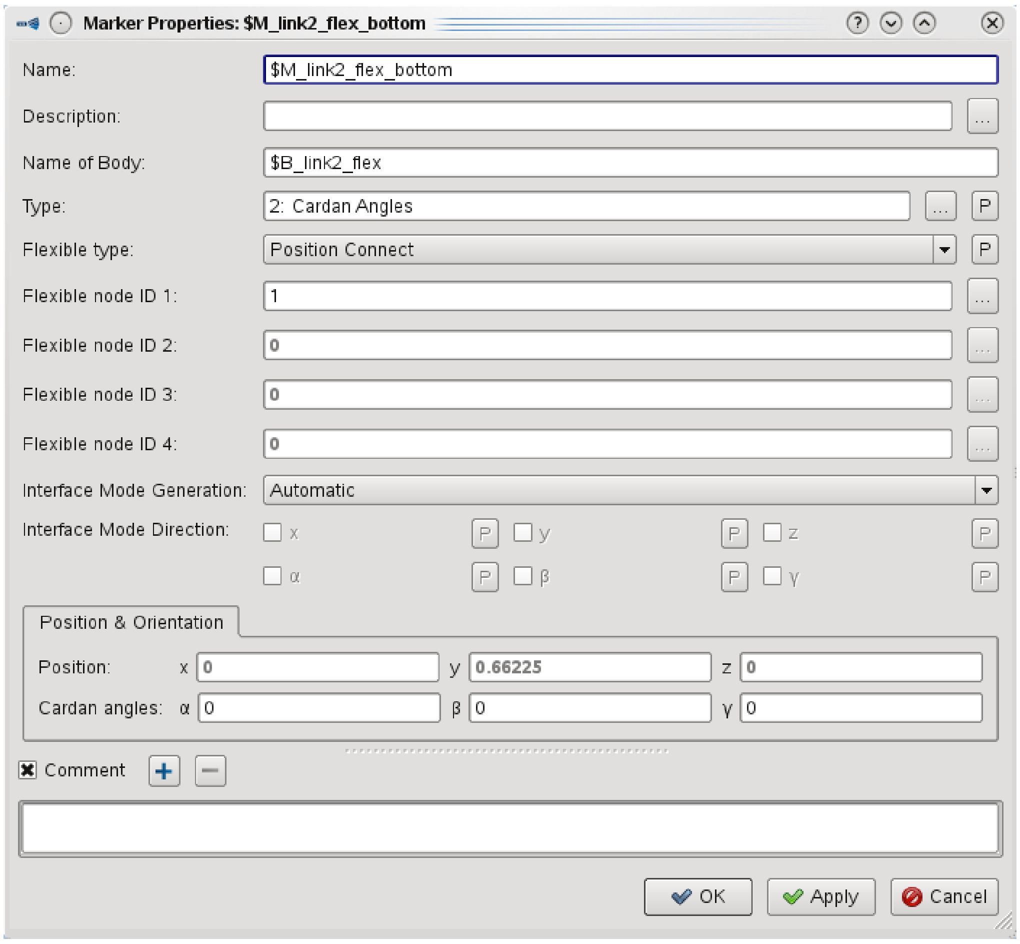
Fig. 2.1.1 Body properties dialog.

Note:
SIMBEAM can emulate multiple modes in different directions (e.g. bending, torsion, and extension) of the beam's vibrations, but it seems to not calculate higher vibrational modes such as second and third modes. If you need to analysis such higher vibrational modes, it might be better to use structural analysis software such as Nastran and ANSYS.

4) Set "Markers" on "Nodes"

- Double click on the maker of flexible body (or create new a maker) and open the property dialog.
- In the dialog, select "Position Connect" in "Flexible type", and choose "Flexible node ID" from the created nodes.

Note:
The flexible type "Position Connect" is recommended for joints of flexible bodies. For more details, see the user reference document of SIMPACK (Section D.1.4.3).



Marker Properties: $M_link2_flex_bottom

Name:	$M_link2_flex_bottom
Description:	
Name of Body:	$B_link2_flex
Type:	2: Cardan Angles
Flexible type:	Position Connect
Flexible node ID 1:	1
Flexible node ID 2:	0
Flexible node ID 3:	0
Flexible node ID 4:	0
Interface Mode Generation:	Automatic
Interface Mode Direction:	☐ x ☐ y ☐ z ☐ α ☐ β ☐ γ

Position & Orientation

Position:	x 0	y 0.66225	z 0
Cardan angles:	α 0	β 0	γ 0

☒ Comment

Fig. 2.1.2 Marker properties dialog.

5) Create “Joint Constraints”

- Double click the joint icon in the model tree, and open the dialog (Joint Properties).
- Choose “From Marker” and “To Marker” as the connecting markers from the different bodies that you created in the above step, and choose the constraint type (e.g. “Revolute Joint ga”).

2.2 Simulation Setting

This section describes two different simulation methods: (a) simulation in SIMPACK and (b) simulation in MATLAB/Simulink. In (a), the simulation is performed using time integrators that SIMPACK provides, and the animation of robot motion can be displayed during the simulation. In (b), a dynamic model for Simulink is exported from SIMPACK, and the simulation is performed using the Simulink integrator. While both software (SIMPACK and Simulink) need to be used at the same time, this method enables us to conduct feedback control in Simulink. The properties of these simulation methods are summarized in the following table.

Table 2.2 Simulation method summary.

	Animation	Result Output	Feedback Control	Input Type
(a) SIMPACK - Online Solver - Offline Solver	 ○ ×	 × ○	 × ×	Manual or Input Function
(b) Simulink	×	○	○	u-Input

2.2.1 (a) Simulation in SIMPACK

SIMPACK provides two solvers: (a-1) online solver and (a-2) offline solver. The online solver displays the animation of simulation, but doesn't output the simulation results. In contrast, the offline can output the simulation results, while it cannot display the animation. The dynamic model and setting for input used in the both solver are same. The following section describes how to set inputs and time integration and how to perform the simulation in SIMPACK. In addition, a method to export the simulation result into MATLAB to see the result is described.

<u>How to set the inputs</u>

The input can be chosen from a force/torque or a set of position, velocity, and acceleration. In general, torque or set of angular position, velocity, and acceleration is used for joint input in the simulation. These inputs are given by using a function called "Input Function" in two methods: (i) manual setting and (ii) import external file (.afs file) that contains input profile into SIMPACK.

(i) Manual setting

1) Create "Input Function"

- Right click on the model tree and choose "Create" ⇒ "Input Function" or click the icon in the tool bar, name the Input Function, and click "OK".
- Insert function values in the Points table by hand, and check the profile in the graph.
- Select the inter- and extrapolation. (e.g. "Liner").
- Click "OK".

(ii) Import external file

1) Create .afs file

- Use the sample file and modify it.

> **Note:**
> You can access the sample file and find the details more for .afs file in the user reference document of SIMPACK (Section H.4.1.2).

2) Set a path to .afs file

- Go to "Extras" ⇒ "Options" from the tool bar.
- In the appeared dialog, select "Search Path" tab.
- Add the path to the folder that contains the created .afs file by clicking "+" button.
- Click "OK".

3) Create "Input Function"

- Right click on the model tree and choose "Create" ⇒ "Input Function" or click the icon in the tool bar, name the Input Function, and click "OK".
- In the dialog (Input Function Properties), choose the created .afs file in "Filename".
- Check the input profile in the plot.
- Select the inter- and extrapolation. (e.g. "Liner").
- Click "OK".

> **Note:**
> When applying inputs with the Input Function, it is recommended to add the input profile before and after motion (not only during motion) in the .afs file. For example, in case that the motion starts at 0 [s] and stops at10 [s], you should add the input values before 0 [s] (ex. -10~0 [s]) and after 10 [s] (ex. 10~20 [s]), otherwise the SIMPACK automatically interpolate/extrapolate the input profile and lead to a failure of the simulation.

The following section describe how to use/connect the Input Function to apply the joint torque and the set of angular position, velocity, and acceleration.

< Input of torque >

1) Create "Excitation".

- Right click on the model tree and choose "Create" ⇒ "Excitation" or click the icon in the tool bar, name the Excitation, and click "OK".
- Select "15: Value or Derivative from IFctn" or "2: From Input Function" in the "Type".

- Create “u-Vector” by clicking “create” at Value in u-Vector Assignments.
- Select the Input Function and its type in Parameter tab.

2) Create “Force Element”.
   - Right click on the model tree and choose “Create” ⇒ “Force Element” or click the icon in the tool bar, name the Force Element, and click “OK”.
   - In the appeared dialog (Force Element Properties), choose “From Marker” and “To Marker” from the markers on joint.
   - Select “93: Force/Torque by u(t) Cmp” in the “Type”.
   - Select the “u-Vector” of torque in the proper parameters (e.g. “nr_u for I_z”).

> **Note:**
> When applying the torque inputs, the joint type of the joint to be applied the torque must be a free-motion type. For example, select “3: Revolute Joint ga” in the Joint Properties dialog.

< Input of angular position, velocity, and acceleration >

1) Double click the appropriate joint in the model tree, and open the dialog (Joint Properties).
2) Select “34: Single Axis by IFctn of Time” in the “Type”.
3) Select the appropriate rotation axis in the “1: Axis of motion”.
4) Select “From IFctns, not recommended” in the “5: Calculate sd, sdd”.
5) Choose Input Functions s(t), sd(t), and sdd(t) from the imported .afs file.
   - s(t): Angular position.
   - sd(t): Angular velocity.
   - sdd(t): Angular acceleration.

> **Note:**
> As an option, you can choose “As derivative of s(t)” in *5: Calculate sd, sdd*. In this option, SIMPACK automatically calculates sd(t) and sdd(t) by differentiating the given s(t), but these values are not calculated correctly when selecting “Linear” and “Step” as *Interpolation type* in the “Input Function Properties”. So, it’s recommended to add all profile (not only position, but also velocity and acceleration) in this setting.

<u>How to set the time integration</u>

SIMPACK offers several integration methods for simulation. The following section explains how to set and change the time integration.

1) Go to “Solver Setting” ⇒ “SLV_SolverSettings” in the model tree.
2) In the dialog (Solver Settings Properties), select “Time Integration” in the left subwindow.
3) Input the simulation time and output steps.
4) Select the appropriate integration method as you want.

- "SODASRT 2" (SIMPACK Optimized DASSL Integrator with Root function handling)

  In this method, the step size is automatically changed. While this method makes it possible to reduce the simulation time, it can cause the wrong simulation results. So, it is recommend to set the maximum step size to be a small value. (You can set the maximum step size in the "General" tab.)
- "Fixed step size"

  This method is recommended because the step size is constant.
- Others…

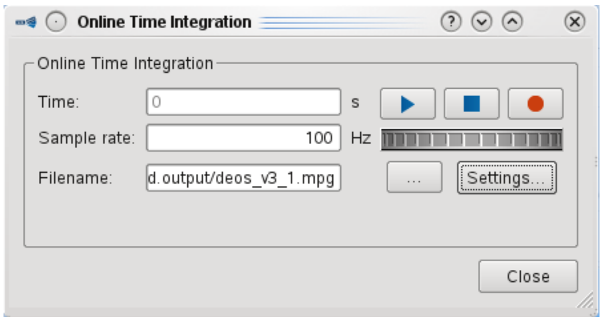
Note:
A big step size of time integration can lead to a simulation error. So, the step size should be small (recommended less than 0.001 [s]).

How to perform the simulation

As mentioned above, there are two different simulations: (a-1) online solver and (a-2) offline solver. The following section describes how to perform the simulation each.

(a-1) Online solver

1) Click the icon ∫xdt in the tool bar.
2) In the appeared dialog, set the sample rate and the video properties by clicking "Settings".
3) Click the start button to run the simulation, and push the pose button to stop it.
   (When you save the video, push the recording button before the simulation starts.)


Online Time Integration

Online Time Integration

Time: 0 s

Sample rate: 100 Hz

Filename: d.output/deos_v3_1.mpg

Fig. 2.2.1 Online simulation dialog.

(a-2) Offline solver

1) Click the icon ∫xdt in the tool bar. After that, the simulation automatically starts and finishes in the "Jobs" window.
2) Check that the status become "Finished", and see the result in the assigned folder.

How to export the simulation result into MATLAB

SIMPACK can export the simulation results in a readable format for MATLAB. The following description explains how to export the result file and read in MATLAB

1) Set the output file in SIMPACK
   - Go to “Solver Setting” ⇒ “SLV_SolverSettings” in the model tree.
   - In the dialog (Solver Settings Properties), select “Result File” in the left subwindow.
   - Set the output directory as you want.
   - Check the “.mat file” button.

2) Set the output results in SIMPACK
   - Go to “Solver Setting” ⇒ “SLV_SolverSettings” in the model tree.
   - In the dialog (Solver Settings Properties), select “Measurement” in the left subwindow.
   - Set the output results in the “Result Configuration” tab.

3) Perform the simulation using the offline solver

   After the simulation, the output file (.mat file) is generated in the chosen directory.

4) Read the output file in MATLAB
   - Load the .mat file in the manuscript or command line window.

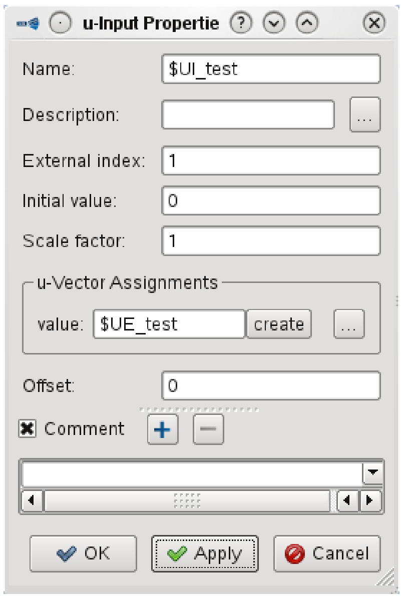
Sample code:

```
data = load(‘XXX/YYY.mat’); % XXX/YYY is the path to the .mat file
time = data.timeInt.time.values;
zzz = data.timeInt.yout.SY_ZZZ.values; % ZZZ is the name of y-Output
```

2.2.2 (b) Simulation in Simulink

SIMPACK can export a model block for MATLAB/Simulink. When performing this simulation, SIMPACK and Simulink must be run simultaneously using an interface connecting them, called SIMAT (Co-simulation interface between SIMPACK and MATLAB/Simulink). As a preparation for this simulation, you need to create “u-Input” and “y-Output” in SIMPACK, which are used as input/output ports in Simulink block. The following section describes the procedure to perform the simulation in Simulink using a SIMPACK model. In the description below, the settings of two different inputs (torque or a set of position, velocity, and acceleration) are explained.

1) Create “u-Input” and “u-Vector” in SIMPACK
   - Right click on the model tree and choose “Create” ⇒ “u-Input” or click the icon →u in the tool bar, name the u-Input, and click “OK”.
   - In the dialog (u-Input Properties), insert an integer number in “External index”.
   - Click the “create” button in the “u-Vector Assignments”.
     (After clicking this button, u-Vector is automatically created in the model tree.)
   - Click the “Apply” and “OK” button.


u-Input Propertie

Name: $UI_test

Description:

External index: 1

Initial value: 0

Scale factor: 1

u-Vector Assignments

value: $UE_test

Offset: 0

Comment

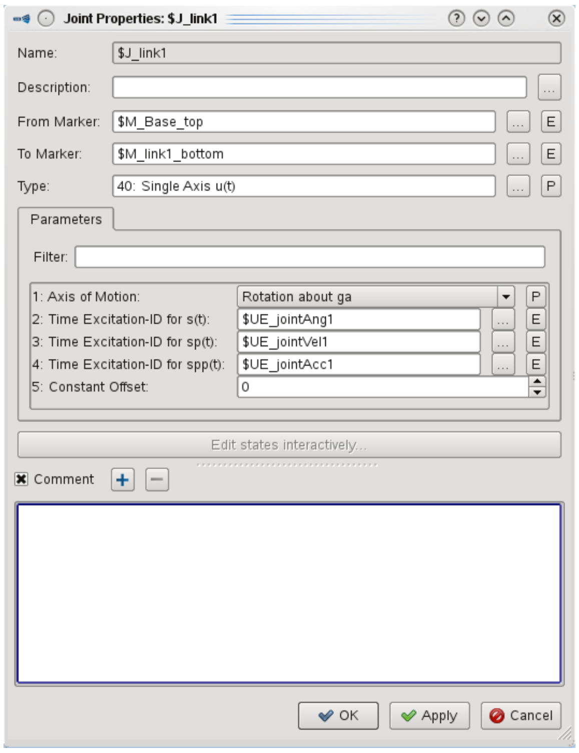
Fig. 2.2.2 u-Input properties dialog.

Note:
Assign the different integer number in "External index" from other u-Inputs (ex. 1, 2, 3…), otherwise the input ports are not correctly appeared in the Simulink block.

2) Set the constraints in SIMPACK

< For inputs of a set of position, velocity, and acceleration >

- Double click the joint icon that you want to create a constraint in the model tree.
- In the appeared dialog (Joint Properties), select "40: Single Axis u(t)" in the "Type".
- Select the rotational axis in the "Axis of motion".
- Select the appropriate u-Vectors in "Time Excitation-ID for s(t)" and "Time Excitation-ID for sp(t)", and "Time Excitation-ID for spp(t)".
- Click the "OK" button.


Joint Properties: $J_link1

Name: $J_link1

Description:

From Marker: $M_Base_top

To Marker: $M_link1_bottom

Type: 40: Single Axis u(t)

Parameters

Filter:

1: Axis of Motion:	Rotation about ga
2: Time Excitation-ID for s(t):	$UE_jointAng1
3: Time Excitation-ID for sp(t):	$UE_jointVel1
4: Time Excitation-ID for spp(t):	$UE_jointAcc1
5: Constant Offset:	0

Fig. 2.2.3 Joint properties dialog.

Note:
Make sure to create the u-Inputs and the u-Vectors of all the angular position, velocity, and acceleration, and assign the all u-Vectors in the proper parameters as the figure above. Otherwise, the simulation result will be wrong.

< For inputs of torque >

- Right click on the model tree and choose “Create” ⇒ “Force Element” or click the icon in the tool bar, name the Force Element, and click “OK”.
- In the appeared dialog (Force Element Properties), choose “From Marker” and “To Marker” from the markers on joint.
- Select “93: Force/Torque by u(t) Cmp” in the “Type”.
- Select the “u-Vector” of torque in the proper parameters (e.g. “nr_u for I_z”).

3) Add the path of SIMAT in MATLAB

- Start up MATLAB and move to the working folder.
  The command to start up SIMPACK in Linux is as follows.
  /opt/matlab/2010b/bin/matlab

  Note: The MATLAB version should be changed on the basis of your computer.
- Go to “File” ⇒ “Set Path” ⇒ “Add Folder” from the tool bar in MATLAB.

- Add the following folder in the path.
  /opt/simpack/s_97_64/partners/mathworks/simat

> **Note**: The SIMPACK version in this path (ex. "s_97_64") should be changed on the basis of the version you are using.

- Click "OK", "Save" and "Close".

4) Add SIMAT block into Simulink
   - Type the command "simat" in the MATLAB command window.
   - Copy and paste the appeared block into another Simulink.

5) Perform the simulation
   - Go to "Solver" ⇒ "Co-simulation" ⇒ "Start Co-simulation" from the tool bar in SIMPACK.
   - Double click the imported model block in the Simulink.
   - Check the parameters (e.g. sampling period).
   - Click the "OK".
   - Confirm that the state "Connected to SIMPACK" on the Simulink block was changed from "NO" to "YES". (If it's not correctly connected to SIMPACK, see the error messages.)
   - Click "Simulation" ⇒ "Start" in the Simulink tool bar or button the key "Ctrl" and "T" at the same time in the Simulink. (The simulation will start immediately.)

2.3 Eigenfrequency

SIMPACK also has a function to calculate the eigenfrequencies of whole system. The following section describes how to measure the eigenfrequencies.

1) Click on "Solver" ⇒ "Eigenvalues" ⇒ "Online" or click the icon in the tool bar.
2) Click "Perform eigenvalue calculation"

You can see and record the animation of each mode after the above procedure.

3. Simulation

This section describes the results of three different simulations: the two benchmark tests and the docking motion simulation. In the first benchmark test, vibrations is induced in the arm by applying a constant acceleration on a joint, and the simulation result is compared with the simulation result that was performed by ASTRIUM using a finite element model. The second benchmark test compares the dynamic behaviors in cases that the base is fixed and free-floating. In the docking motion simulation, a joint trajectory for docking motion after capturing a large target (see DEOS phase B D2C) is given in the robot, and the effects of link flexibility such as vibrations are accessed. The both simulations are performed using the time integrator that SIMPACK provides.

3.1 Simulation Model

The same dynamic model of the chaser is used in both the benchmark test and motion simulation, while the target mass and inertia are different. The dynamic model of the chaser is simplified in accordance with the following assumptions.

<u>Assumptions</u>

a) Base is rigid.

b) Structural flexibilities of robotic arm can be lumped on the long parts of the links (Link 2 and 4), and they can be modeled by the beam theory.

i.e.) - No joint flexibility.
- Other links (Link 1, 3, 5, 6, 7) are rigid.
- Short parts of Link 2 and 4 are also rigid
- Long parts of Link 2 and 4 are modeled as flexible beams

c) The flexible long parts of Link 2 and 4 are hollow cylinders (tube), and their parameters are as follows.

- Outer diameter: 127 [mm]
- Inner diameter: 123 [mm]

These geometric parameters of the arm can be seen in the CAD data that Erich Krämer (RM) made in the past. To see the CAD data, you need a CAD software called Creo Parametric.

Note: DLR has the license of Creo Parametric, but this software is available only for Windows computers. To install the software, please contact to a system administer.

d) The CAD data also indicate that the material of flexible long parts of Link 2 and 4 is Aluminum. The following material parameters are used as the properties of general Aluminum.

- Density: 2700 [kg/m^3]
- Young's modulus: 69*10^9 [N/m^2]
- Poisson's Ratio: 0.334 [-]

e) The mass of the rigid parts of Link 2 and 4 are set so that the total mass of flexible part and rigid part are match to the assumed value in the .def file, which is included in the following folder.
Report/data/sample/def_file

The other parameters used in the simulation are imported from the above .def file. The reference configuration and each link's coordinate are shown in the following figure.

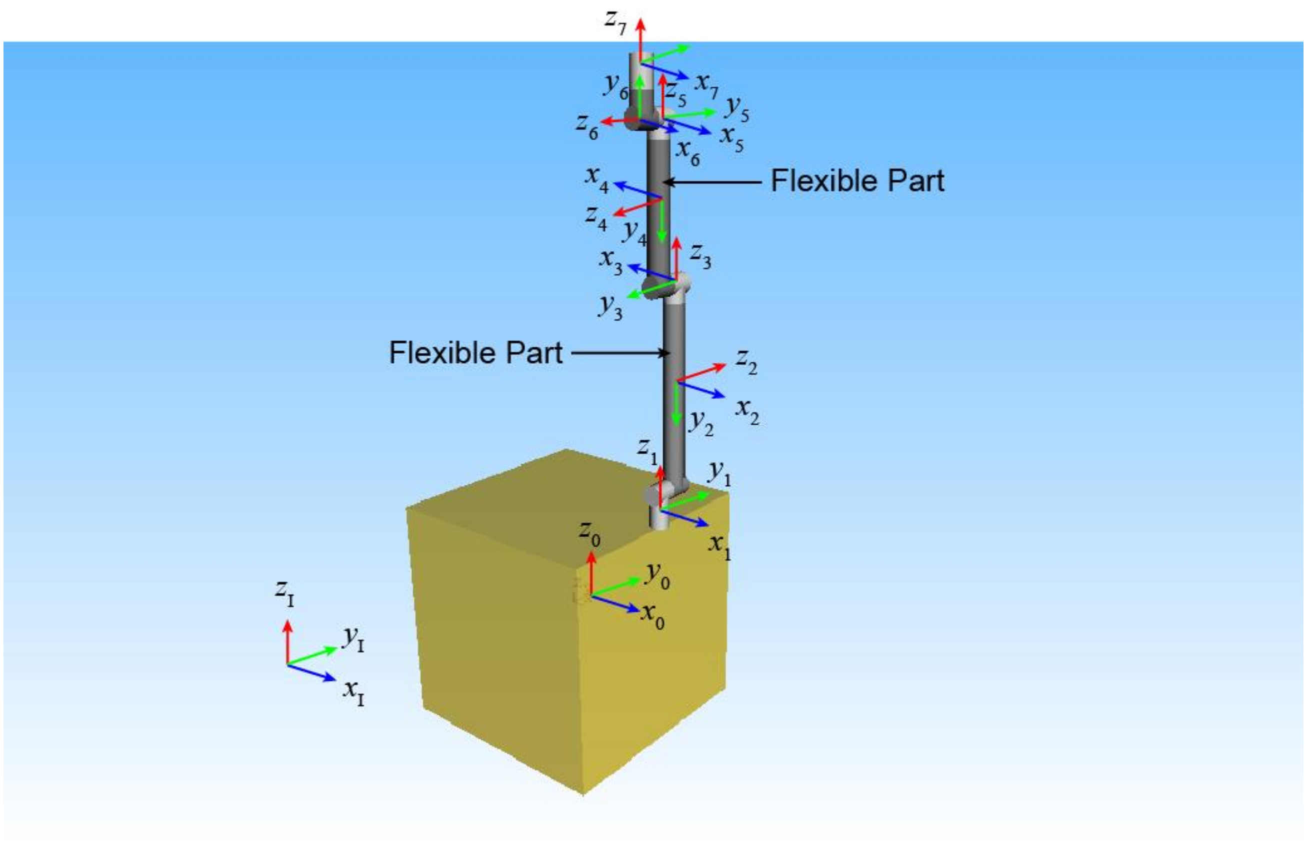


Fig. 3.1.1 Dynamic model for DEOS in SIMPACK.

3.2 Benchmark Test 1

In this benchmark test, the dynamic responses are compared in the simulation results obtained using the SIMPACK model and the finite element model that ASTRIUM made. The vibration of the arm is induced by applying a constant acceleration (0.0365 [rad/s^2]) on Joint 2 during 1 [sec]. Unfortunately, because there is no information about the exact conditions and the input profile after 1 [sec] in the ASTRIUM documents, the initial configuration and the whole input profile in this simulation are assumed as follows. (The simulation conditions and results of ASTRIUM are shown in the PDF file in the folder [*report/data/benchmark_test1/astrium*].)

Assumptions

a) Base is fixed (not free-floating).

b) Target mass is 350 [kg].

Initial configuration

- Joint angles

  [q_1 q_2 q_3 q_4 q_5 q_6 q_7] = [180 -90 0 30 0 0 0] (deg)
- Relative coordinate of target with respect to end-effector (Link 7)

  [alpha beta gamma] = [0 180 180] (deg)
- Grasping point on the target (Relative position from the target's CoM)

  [430.831 454.094 787.062] (mm)

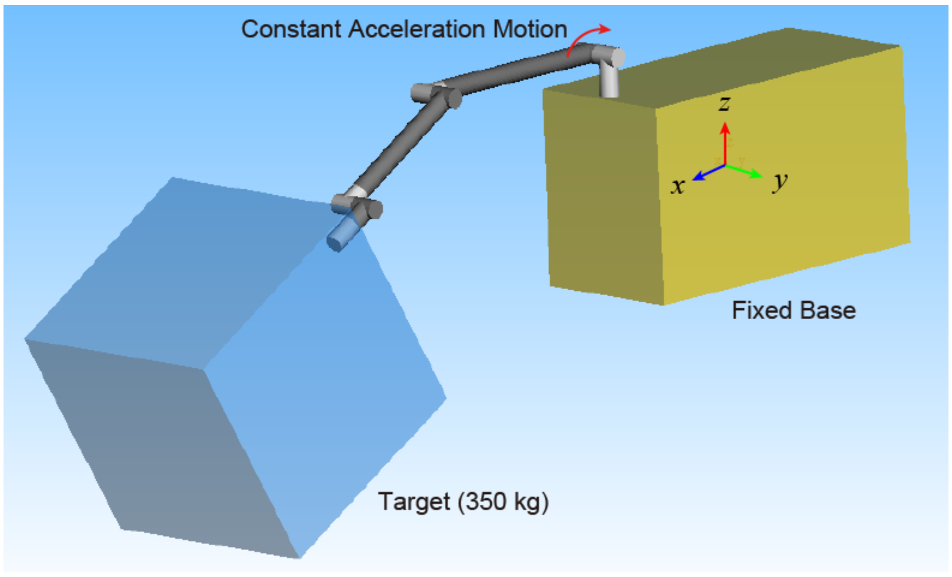


Fig. 3.2.1 Initial configuration for benchmark test 1.

Inputs

The constant acceleration (0.0356 [deg/s^2]) is applied on Joint 2 during 1 [sec]. After that, the joint motion is stopped. The profiles of angular position, velocity, and acceleration on Joint 2 are shown in the following graphs.

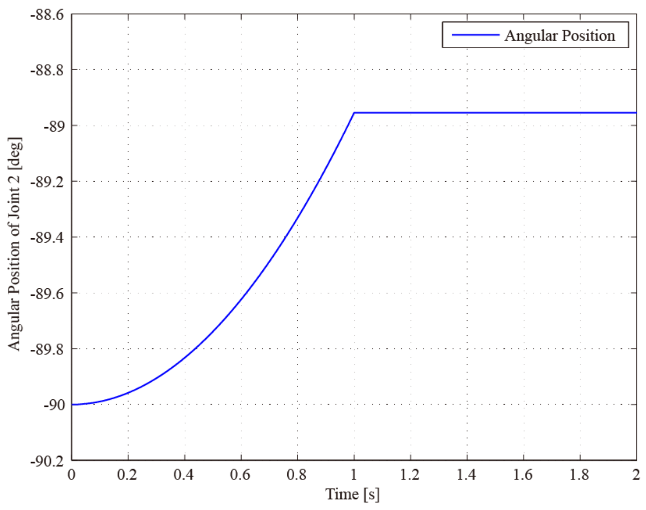


Fig. 3.2.2 Input of angular position on Joint 2 for benchmark test 1.

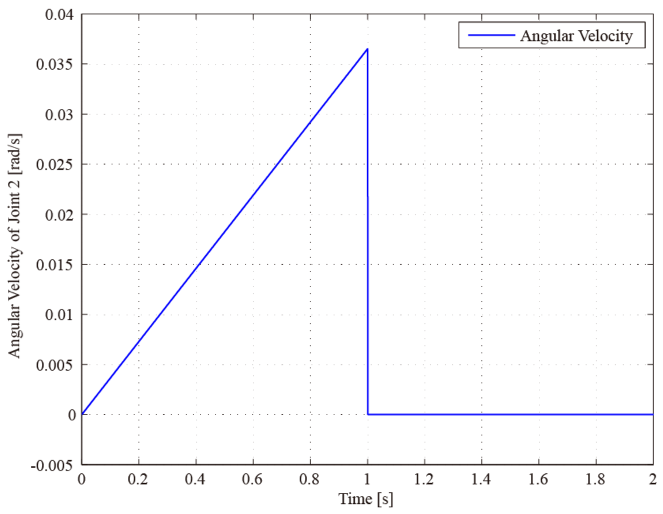


Fig. 3.2.3 Input of angular velocity on Joint 2 for benchmark test 1.

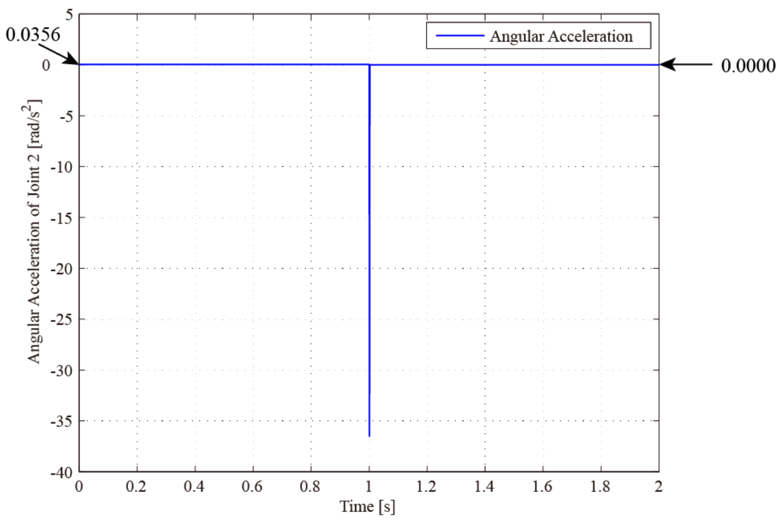


Fig. 3.2.4 Input of angular acceleration on Joint 2 for benchmark test 1.

Results and discussions

The simulation results including MATLAB figures and a video are stored in the folder [*report/data/benchmark_test1/results*]. The motions of target position in Z-axis of the inertial coordinate are compared here. The first figure shows the simulation result of SIMPACK using the simplified model. The second figure shows the simulation result of target motion ("maybe" in Z-axis) obtained using the finite model of ASTRIUM.

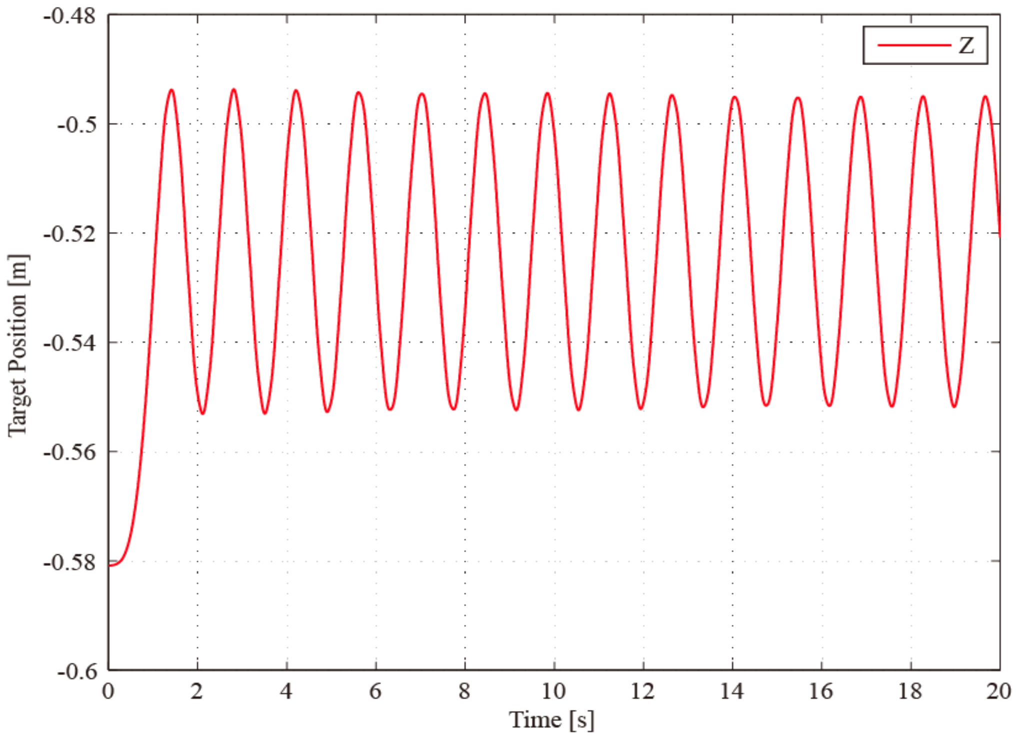


Fig. 3.2.5 Target position in Z axis in SIMPACK simulation.

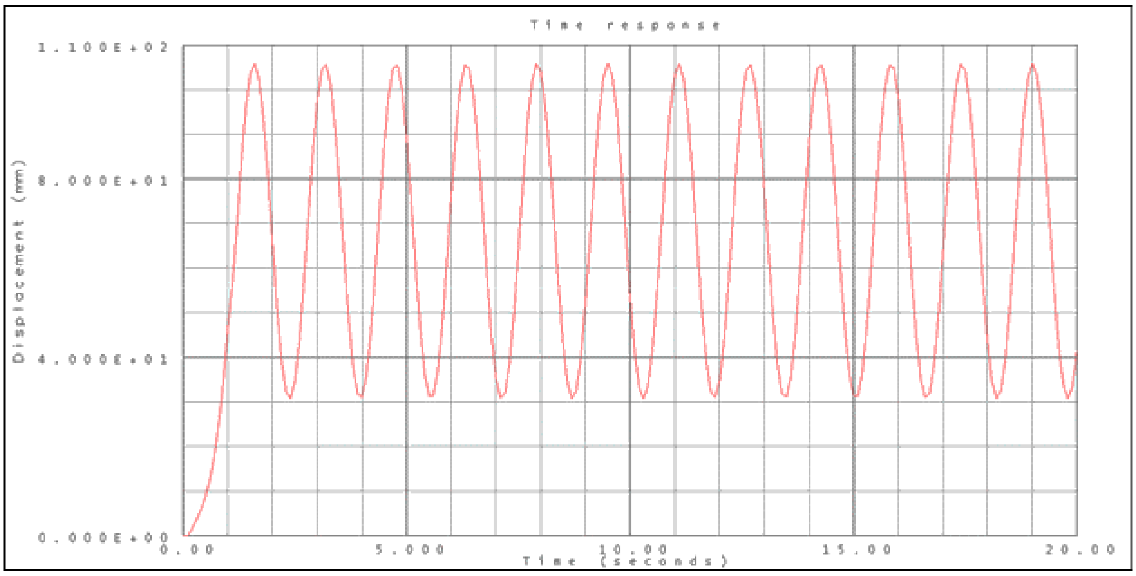


Fig. 3.2.6 Target Position in Z axis in ASTRIUM's simulation.

To understand the structural difference between the SIMPACK model and the ASTRIUM model, the Young's modulus used in the simulations are analyzed on the basis of the cantilever theory as below. This analysis can provide how much they are different each other. Of course, this robot arm is a different system from the cantilever, but they are similar structures (the base is fixed on the ground, joints are locked, and arm tip is free).

< Frequency >
SIMPACK: f_S = 0.714 [Hz]
ASTRIUM: f_A = 0.636 [Hz]

The frequency of cantilever is described as follows:

$$f = \frac{\lambda}{2\pi L}\sqrt{\frac{E}{\rho}}$$

where $\lambda = \frac{1}{2}\pi, \frac{3}{2}\pi, \frac{5}{2}\pi \cdots$. Supposing that the density is constant and the only first mode is considered, the proportion of the Young's modulus can be calculated from the proportion of the obtained frequencies in the simulation as follows.

$$\frac{f_A}{f_S} = \sqrt{\frac{E_A}{E_S}} = \frac{0.636}{0.714} = 0.89$$

$$E_A = 0.79E_S$$

< Amplitude >
SIMPACK: δ_S = 60 [mm]
ASTRIUM: δ_A = 75 [mm]

The amplitude of cantilever when the static load applies on the tip is expressed as follows.

$$\delta = \frac{PL^3}{3EI}$$

where P is the force exerting on the cantilever tip. The proportion of the Young's modulus can be calculated from the proportion of the obtained amplitudes in the simulation as follows.

$$\frac{\delta_A}{\delta_S} = \frac{E_S}{E_A} = \frac{75}{60}$$

$$E_A = 0.8E_S$$

Although the above analyses are based on the cantilever model, the similar results are obtained in the both analyses: the Young's modulus of the SIMPACK model is approximately 80 [%] of one of the ASTRIUM model.

3.3 Benchmark Test 2

This second benchmark test compares the dynamic motions in cases of free-floating base and fixed base. The initial configuration in this test is same with the first benchmark test.

Assumptions

a) Base mass is 1082.1 [kg]

b) Target mass is 350 [kg].

Initial configuration

The initial configuration is same with the first benchmark test.

Inputs

During first 1 [s], the constant positive angular acceleration (0.0365 [rad/s^2]) is applied on Joint 2. After that, the negative angular acceleration (-0.0365 [rad/s^2]) is applied for 1 [s] so as to the angular velocity become zero at 2 [s]. The input profiles of angular position, velocity, and acceleration on Joint 2 are shown as follows.

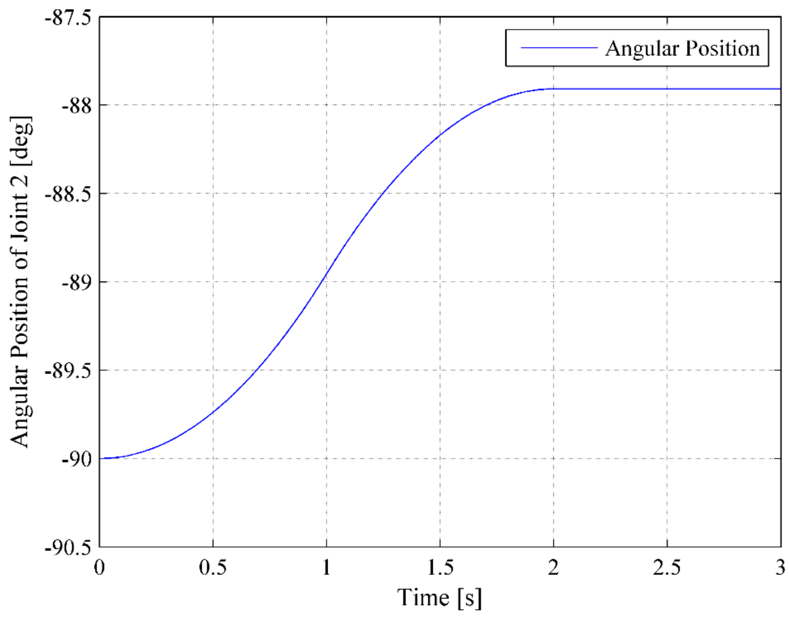


Fig. 3.3.1 Input of joint position for benchmark test 2.

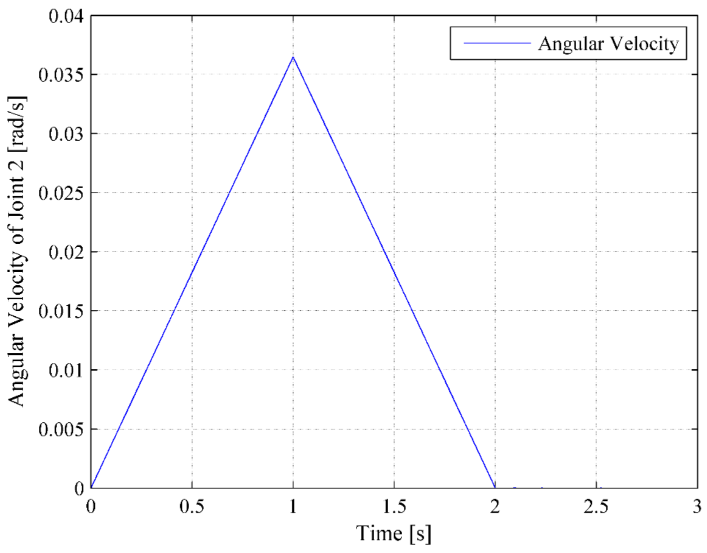


Fig. 3.3.2 Input of joint velocity for benchmark test 2.

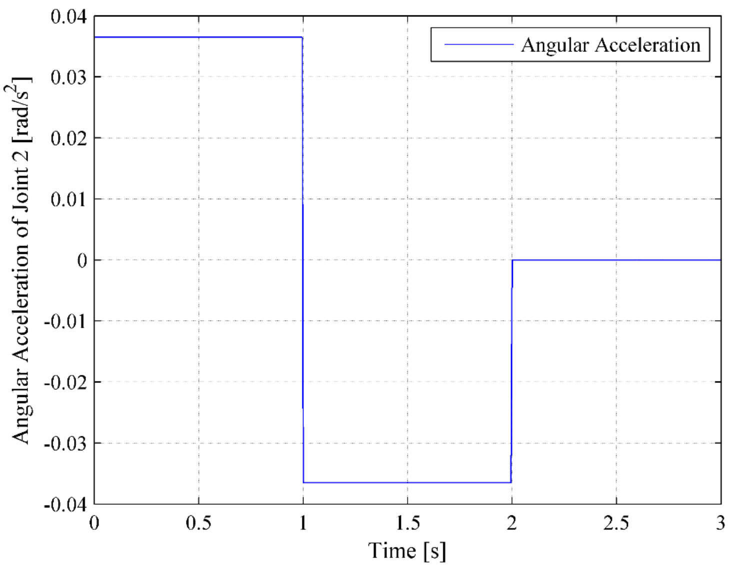


Fig. 3.3.3 Input of joint acceleration for benchmark test 2.

<u>**Results and discussions**</u>

< Input Torque >

The input torque profile is shown in the figure below. In this figure, the blue solid line shows torque input in case that the base is free-floating, and the blue dot line indicates one in case that the base is fixed. Additionally, the red line, which shows the torque input in the case of rigid links and free-floating base, is inserted in this figure as a reference. In the case of rigid links, the input torque is step profile as well as the joint acceleration. In contrast, in the cases of flexible links, the input toque profiles are oscillated due to the link flexibility. In the case of free-floating base, the torque around the limitation (80 [Nm]) is applied to realize the given joint motion. In contrast, in the case that the base is fixed, higher torque is required for implementing the given joint motion. In this simulation test, the maximum torque 550 [Nm] was applied around 1.6 [s].

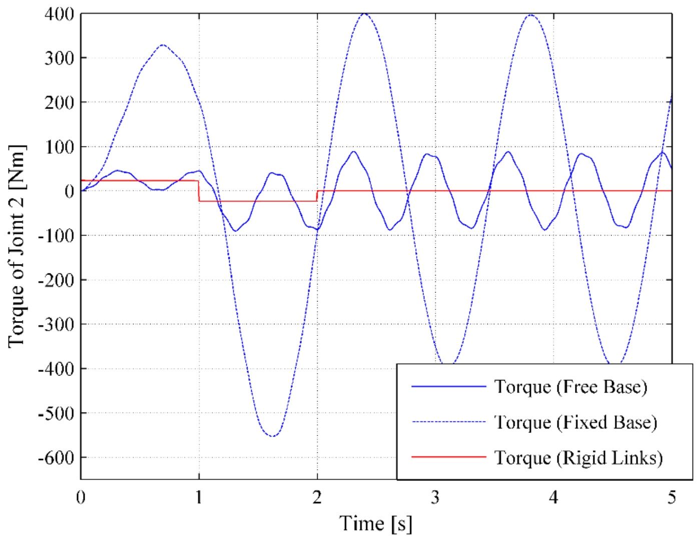


Fig. 3.3.4 Torque profile applied on Joint 2 in benchmark test 2.

< Base Motion >

The following figures show the base position and attitude motions with respect to the inertia frame. It can be confirmed that the base position and attitude does not change in the case that the base is fixed. In contrast, the base motion including vibrations due to the link flexibility can be seen in the figures. In particular, the base transnationally moved around 10 [mm] in the X-axis and was oscillated. The base attitude was also changed, and the bigger motion and vibrations were observed in the beta-axis.

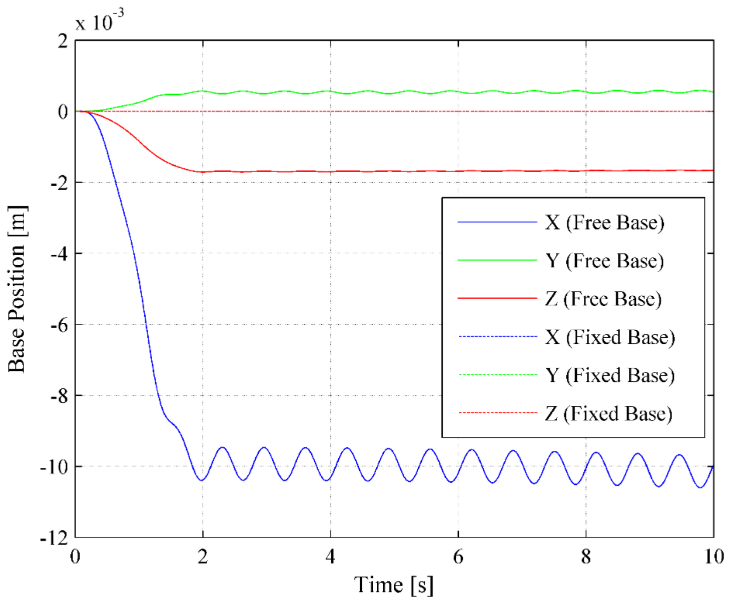


Fig. 3.3.5 Base position w.r.t. inertial coordinate in benchmark test 2.

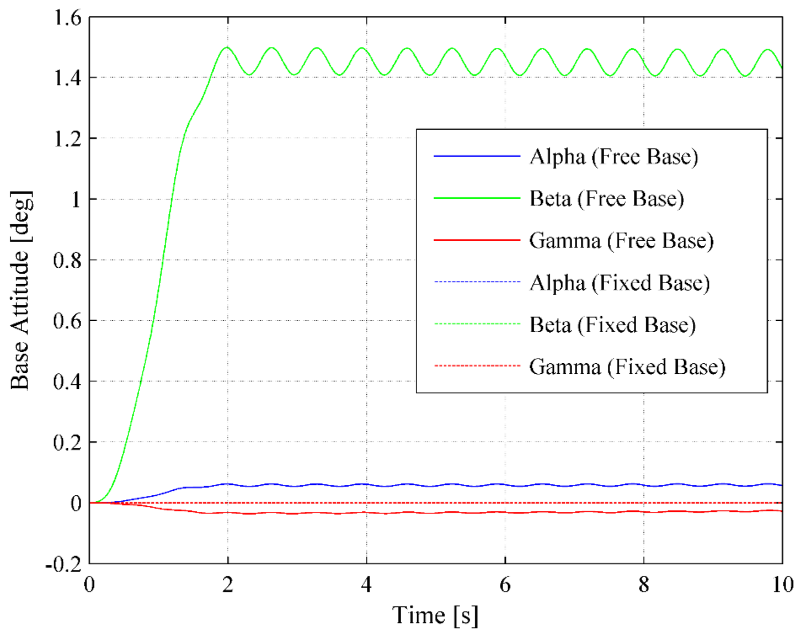


Fig. 3.3.6 Base attitude w.r.t. inertial coordinate in benchmark test 2.

< Relative Position & Attitude to Target from Base >

The relative position and attitude to the target from the base are shown in the following figures.

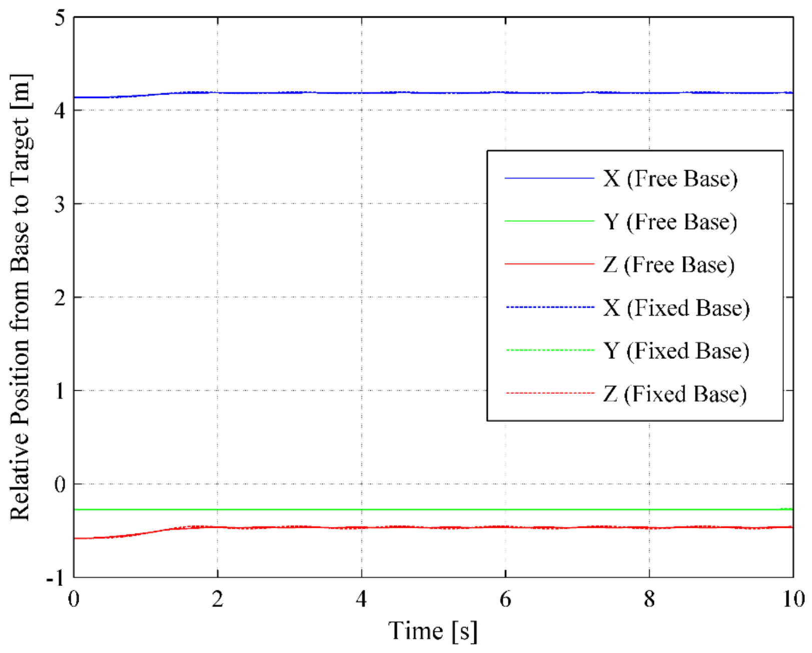


Fig. 3.3.7 Relative position from base to target in benchmark test 2.

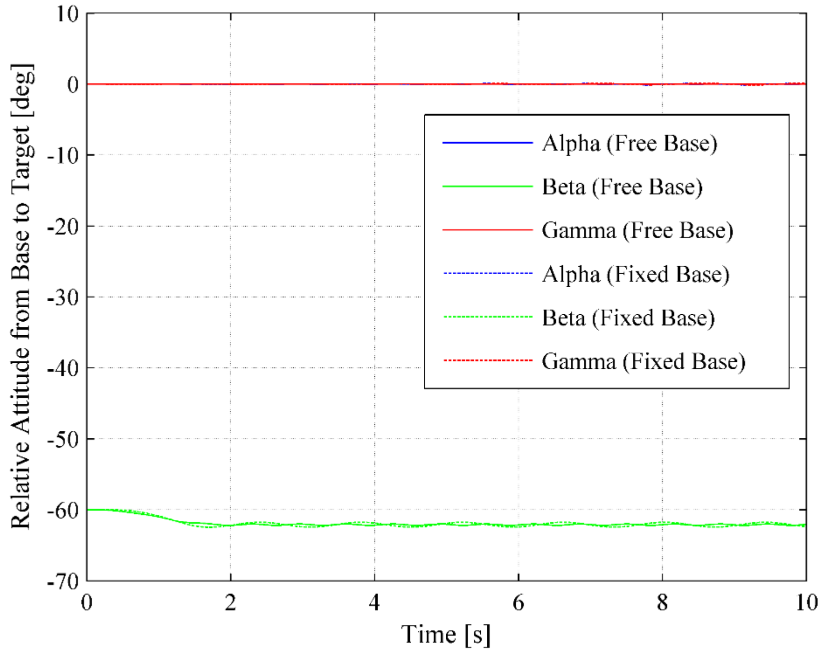


Fig. 3.3.8 Relative attitude from base to target in benchmark test 2.

In addition to the above figures, the following figures shows the precise comparisons of the relative position in X-axis and Z-axis and the relative attitude in Beta-axis. In the all axes, the vibration amplitude of the fixed base is bigger than those of the free-floating base, and the vibration frequency of the fixed base is smaller than those of the free-floating base.

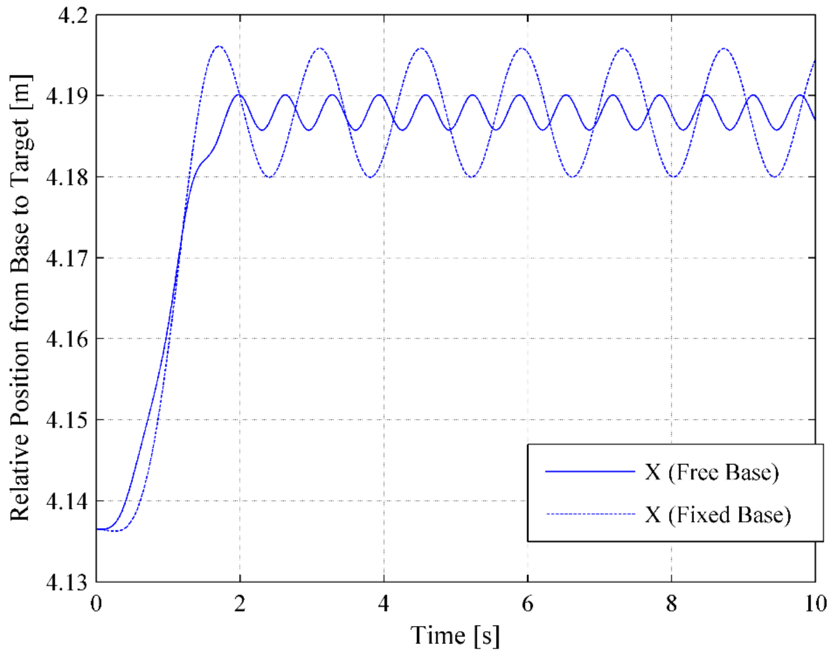


Fig. 3.3.9 Relative position from base to target in X axis in benchmark test 2.

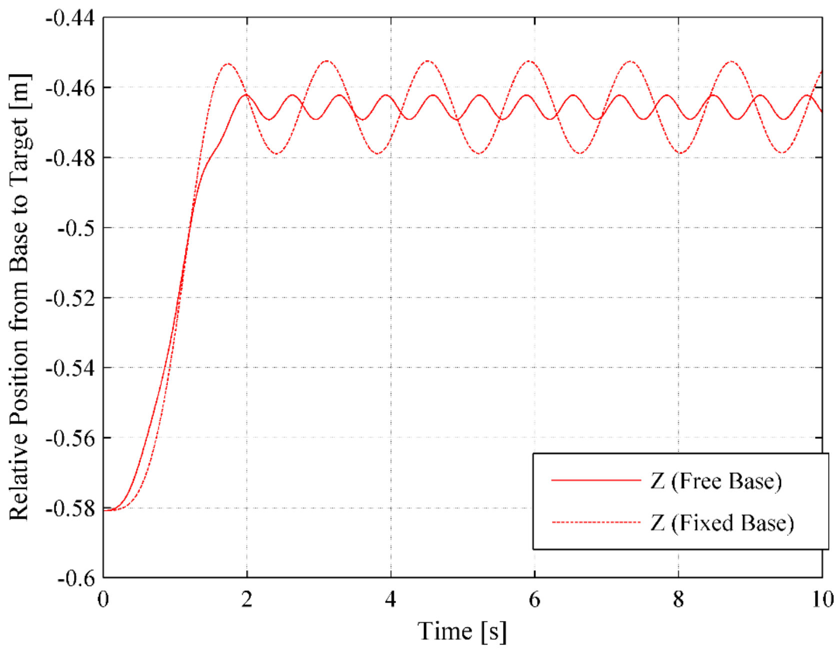


Fig. 3.3.10 Relative position from base to target in Z axis in benchmark test 2.

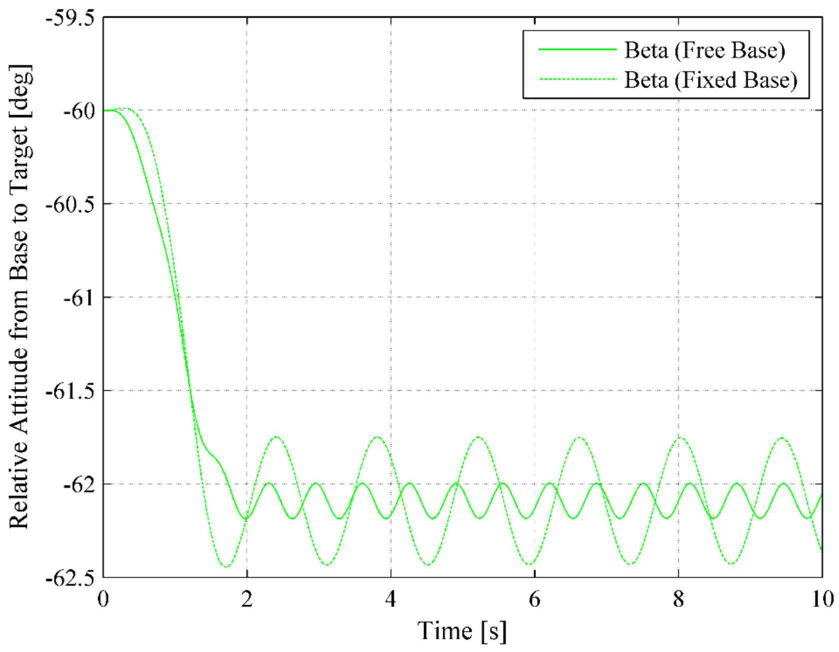


Fig. 3.3.11 Relative attitude from base to target in beta axis in benchmark test 2.

3.4 Docking Motion Simulation

This simulation analyzes the dynamic response in the motion to transfer the target into the docking position after capturing it. The simplified model of SIMPACK is used in this simulation.

<u>**Assumptions**</u>

a) Base is free-floating (The base mass is 1082.1 [kg]).

b) Target mass is 1398.52 [kg].

<u>**Inputs**</u>

A set of angular position, velocity, and acceleration to transfer the grasped target into the docking position is given in this simulation. This set was calculated in the prior simulation. The input of the arm has been given during 30 [sec], and the inputs become zero after the motion (10 [sec]). The profile of the robot configuration and the angular position of each joint are shown in the following figures.

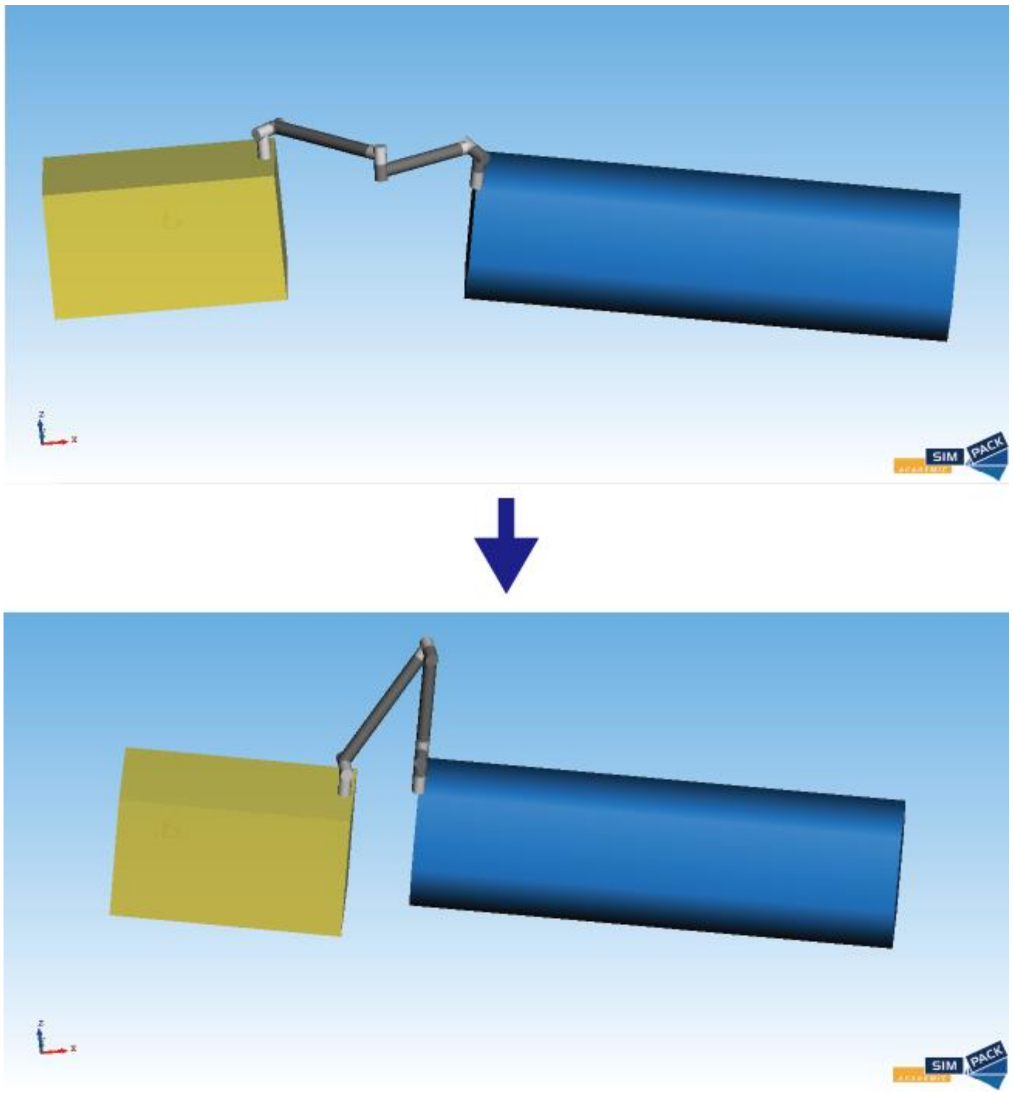


Fig. 3.4.1 Initial and final configuration for docking motion simulation.

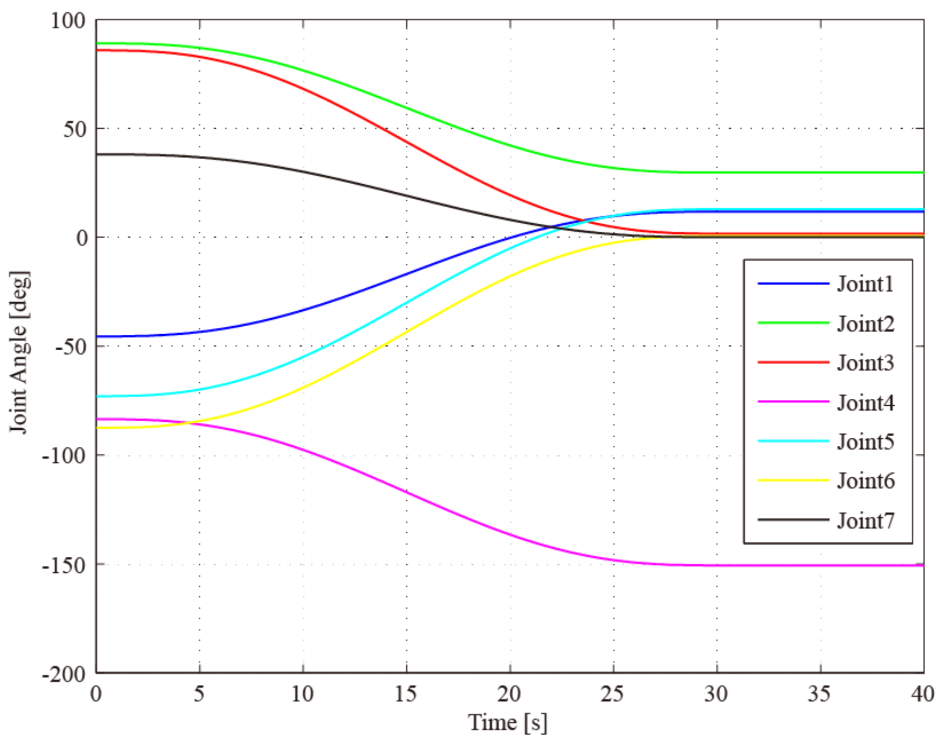


Fig. 3.4.2 Joint angle input for docking motion simulation.

<u>Results and discussions</u>

The simulation results including MATLAB figures and a video are stored in [*report/data/docking_motion_simulation/results*]. The detailed analysis is described as below.

< Input Torque >

The following figure shows the input torque required for the given motion. The oscillated torque was observed on the joints due to the vibrations of flexible links. The maximum absolute value of the input torque is seen on Joint 4 around 15 [sec], whose value is approximately 40 [Nm]. Therefore, the trajectory given in this simulation can be implemented without exceeding the torque limitation (80 [Nm]).

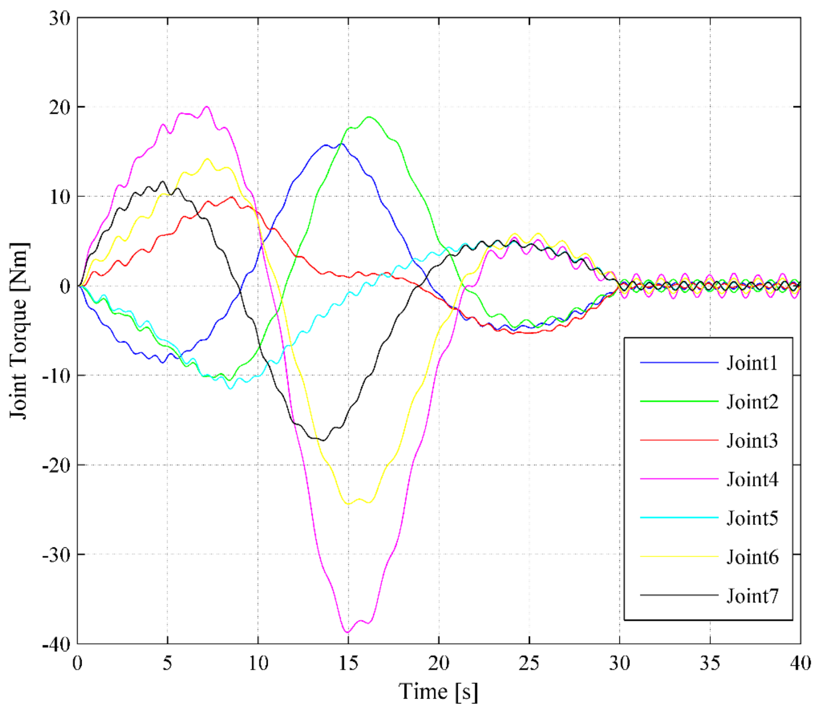


Fig. 3.4.3 Joint torque applied on each joint for docking motion.

< Relative Position & Attitude to Target from Base >

The following figures show the relative position and attitude of the target from the free-floating base with respect to the base coordinate.

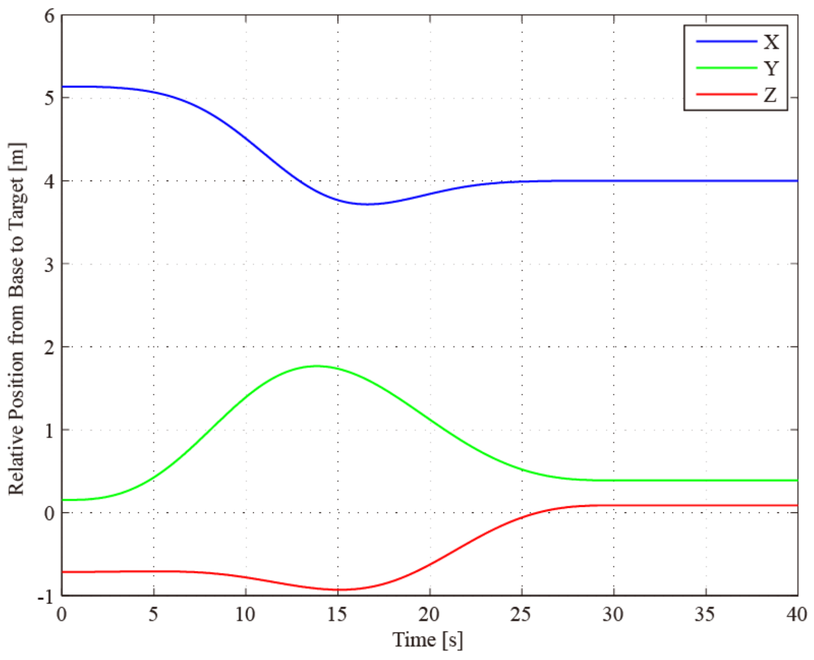


Fig. 3.4.4 Relative position to target from base in docking motion.

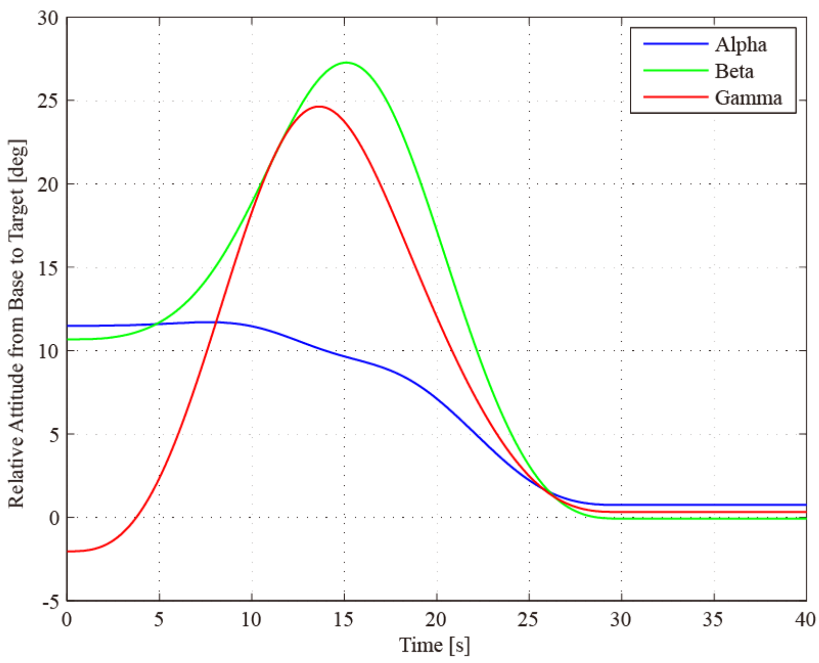


Fig. 3.4.5 Relative attitude to target from base in docking motion.

The following figures show the difference of the target's relative position and attitude from the equilibrium points.

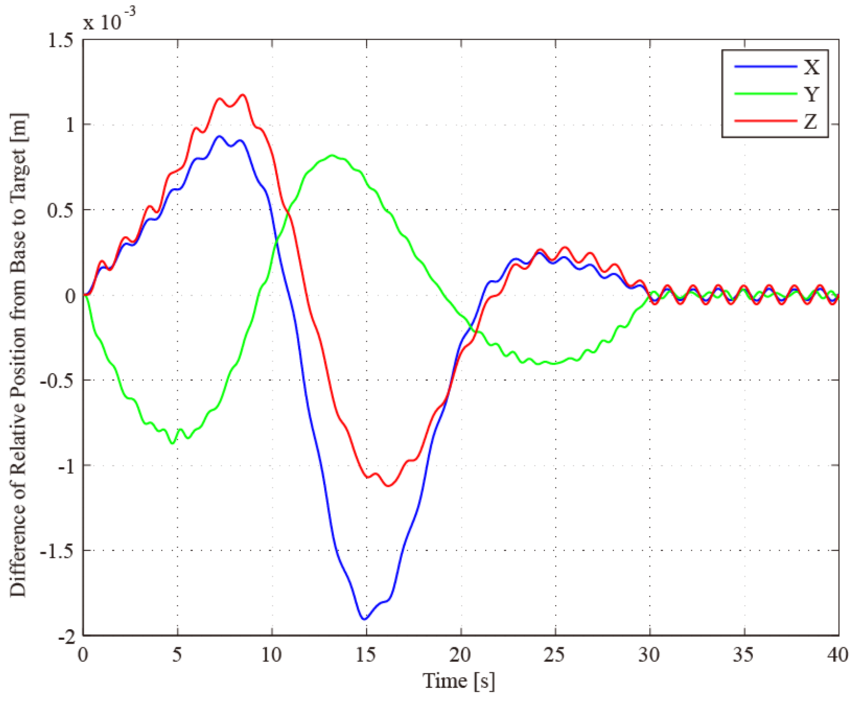


Fig. 3.4.6 Difference of relative position from equilibrium point in docking motion.

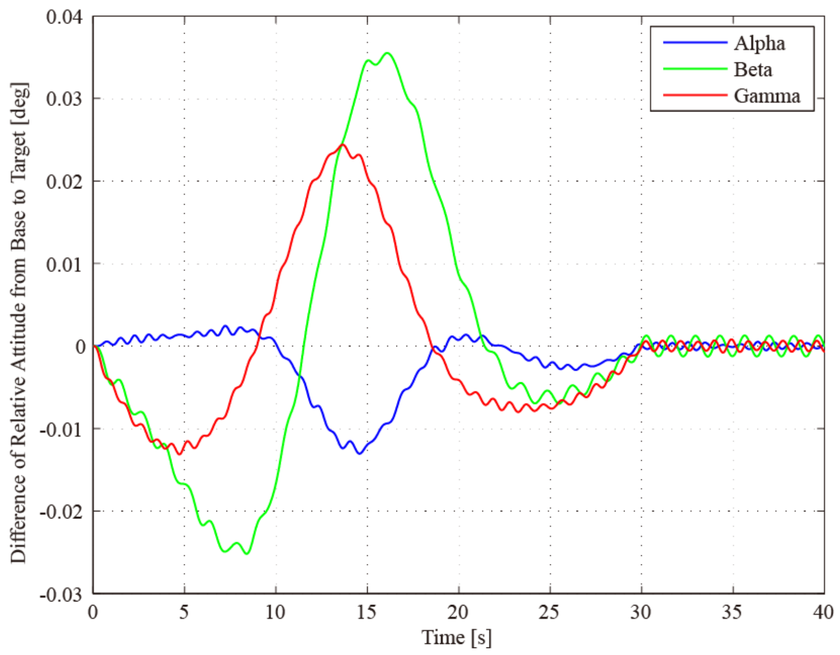


Fig. 3.4.7 Difference of relative attitude from equilibrium point in docking motion.

The maximum deformation of the target was less than 2 [mm] during the motion, and the vibration after the motion was very small. This result indicates that the effect of link flexibility is not critical for the docking motion.

3.5 Resonance Simulation

This simulation is performed to analyze the resonance behavior using the simplified model in SIMPACK. Two different inputs based on cosine waves with the eigenfrequencies and its 80 percent frequency are applied on Joint 2. This simulation compares the responses when applying these frequencies on the flexible link system and the response when applying the eigenfrequency on the rigid system.

<u>**Assumptions**</u>

a) Base is free-floating (The base mass is 1082.1 [kg]).

b) Target mass is 1398.52 [kg].

<u>**Equilibrium configuration**</u>

- Joint angles
  $[q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7]$ = [-45 90 90 -90 -90 -90 45] (deg)
- Relative coordinate of target with respect to end-effector (Link 7)
  [alpha beta gamma] = [0 180 0] (deg)
- Grasping point on the target (Relative position from the target's CoM)
  [-2292 -375 662] (mm)

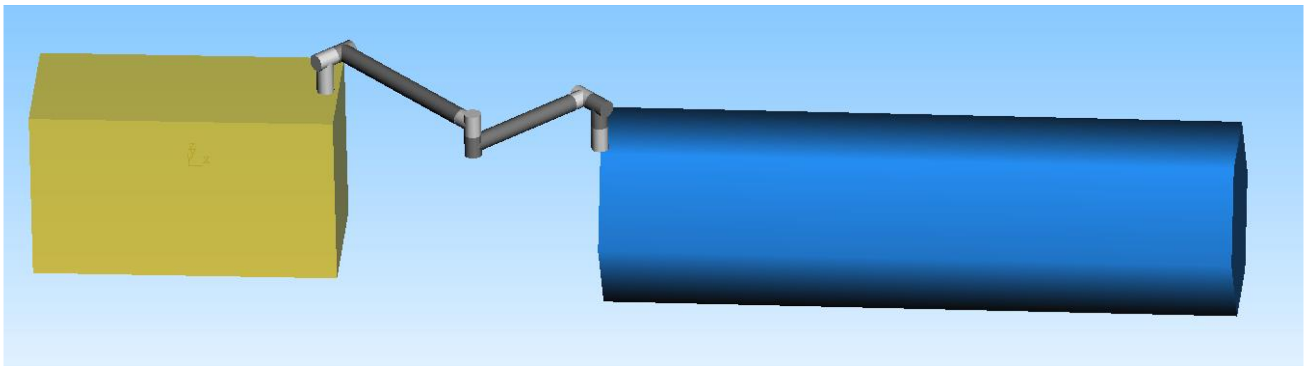


Fig. 3.5.1 Equilibrium configuration for resonance simulation.

<u>**Eigenfrequencies**</u>

The first three eigenfrequencies of whole system with the above configuration are as follows.

$f_1 = 0.8726$ [Hz]

$f_2 = 1.1953$ [Hz]

$f_3 = 1.4756$ [Hz]

<u>**Inputs**</u>

The following cosine waves with the first eigenfrequency and its 80% frequency are applied on Joint 2.

$$q_2(t) = A\cos\omega t + q_0$$

where

$$\mathrm{A} = 0.5 \ \ [\mathrm{deg}]$$
$$\omega = 2\pi f_{e1} \ \text{ or } \ 2\pi f_1$$
$$f_{e1} = 0.8726 \ \ [\mathrm{Hz}] \text{ (First eigenfrequency)}$$
$$f_1 = 0.6980 \ \ [\mathrm{Hz}] \text{ (Not eigenfrequency)}$$
$$q_0 = 90 \ \ [\mathrm{deg}].$$

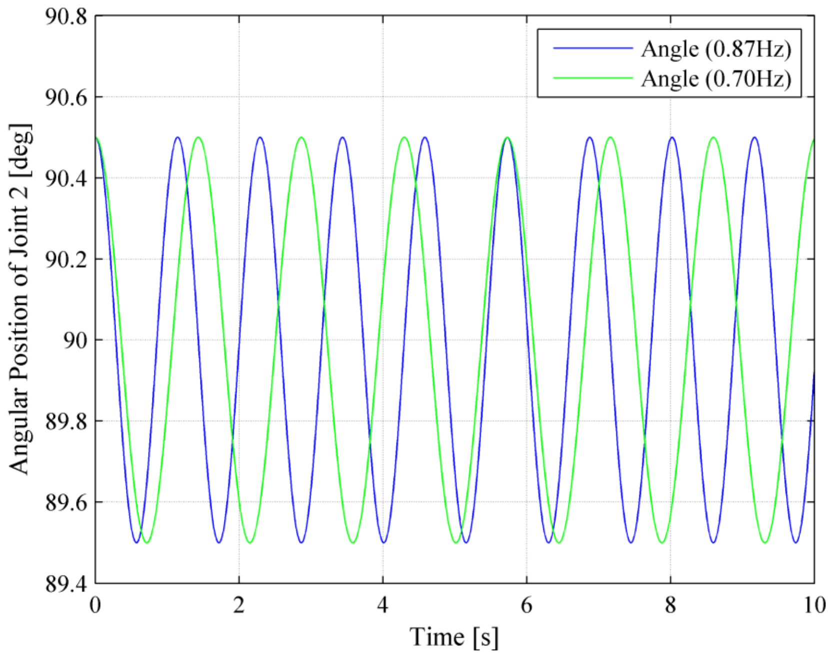


Fig. 3.5.2 Input of Joint 2 for resonance simulation.

<u>Results and discussions</u>

The simulation results including MATLAB figures and a video are stored in [*report/data/resonance_simulation/ results*]. The detailed analysis is described as below.

< Relative Position & Attitude to Target from Base >

The following figures show the relative position and attitude of the target from the free-floating base with respect to the base coordinate. In these figures, the blue line shows the result in the case of the first eigenfrequency with the flexible links, the green line indicates the results in the case of non-eigenfrequency with the flexible links, and the red line shows the result in the case of the first eigenfrequency with the rigid links.

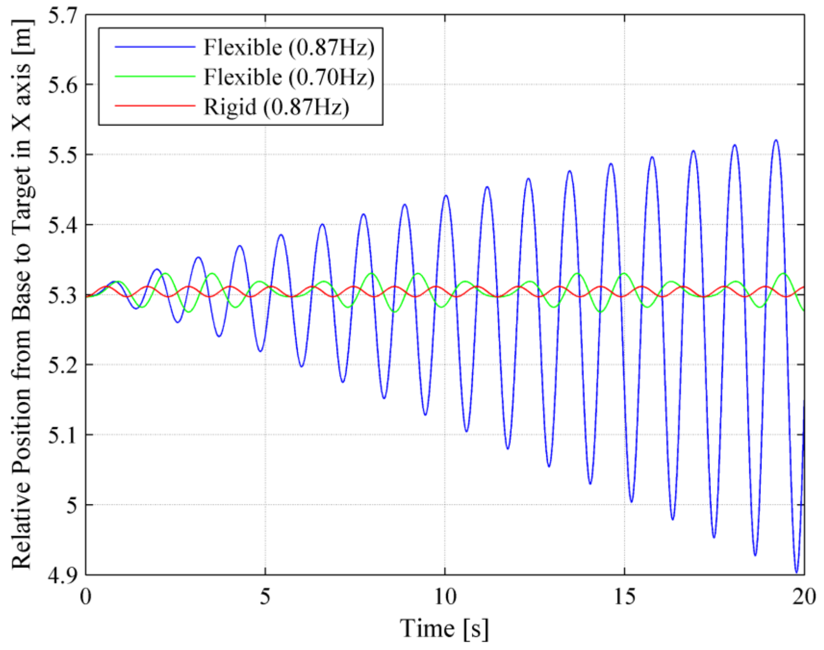


Fig. 3.5.3 Relative position from base to target in X axis.

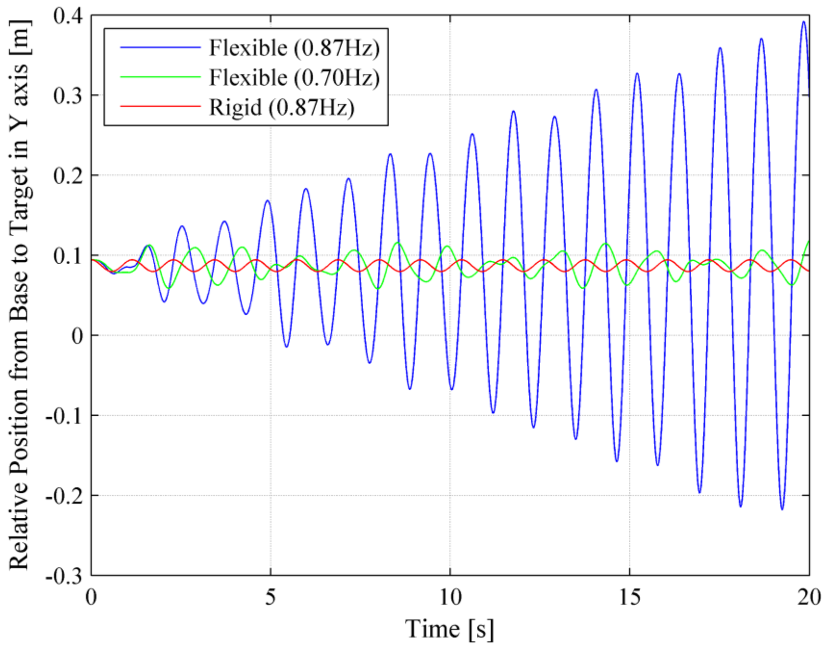


Fig. 3.5.4 Relative position from base to target in Y axis.

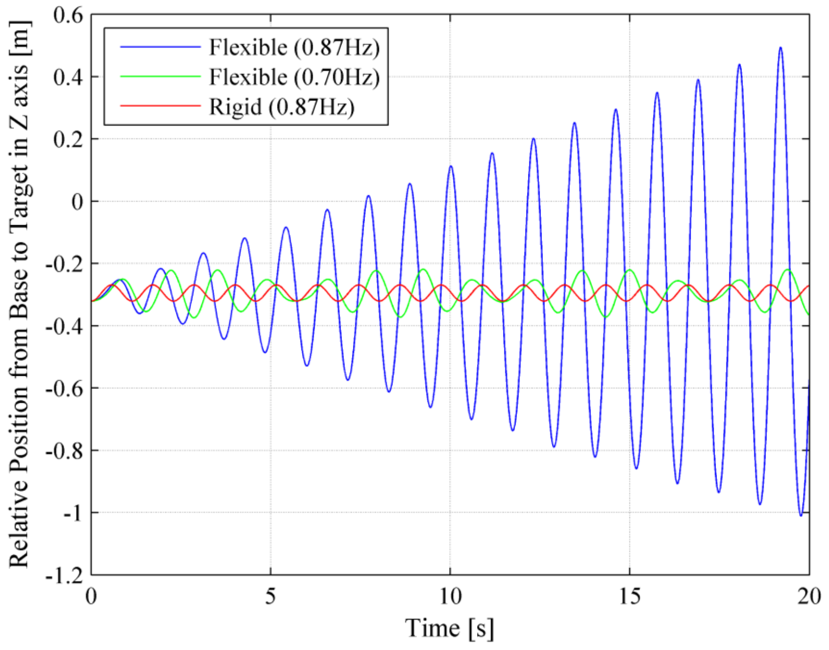


Fig. 3.5.5 Relative position from base to target in Z axis.

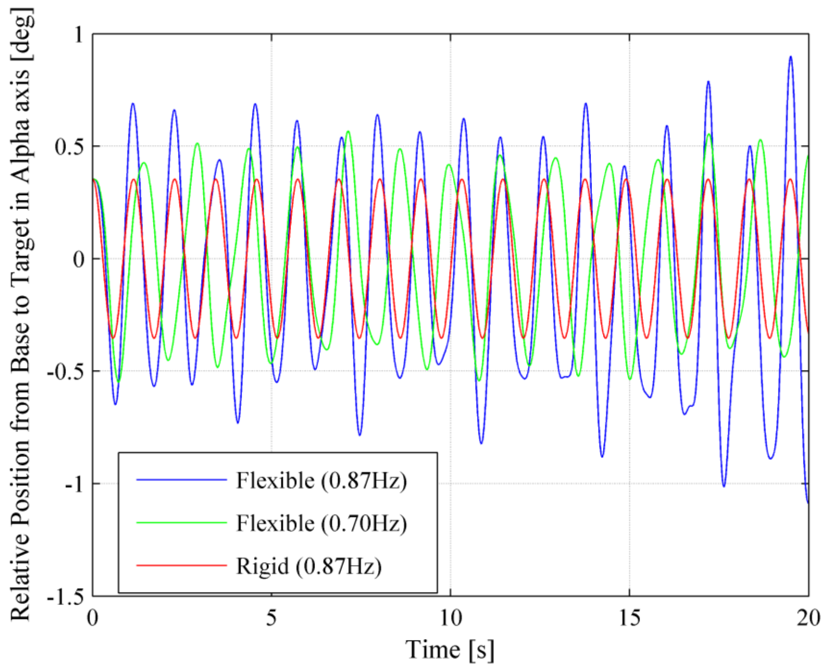


Fig. 3.5.6 Relative attitude from base to target in Alpha axis.

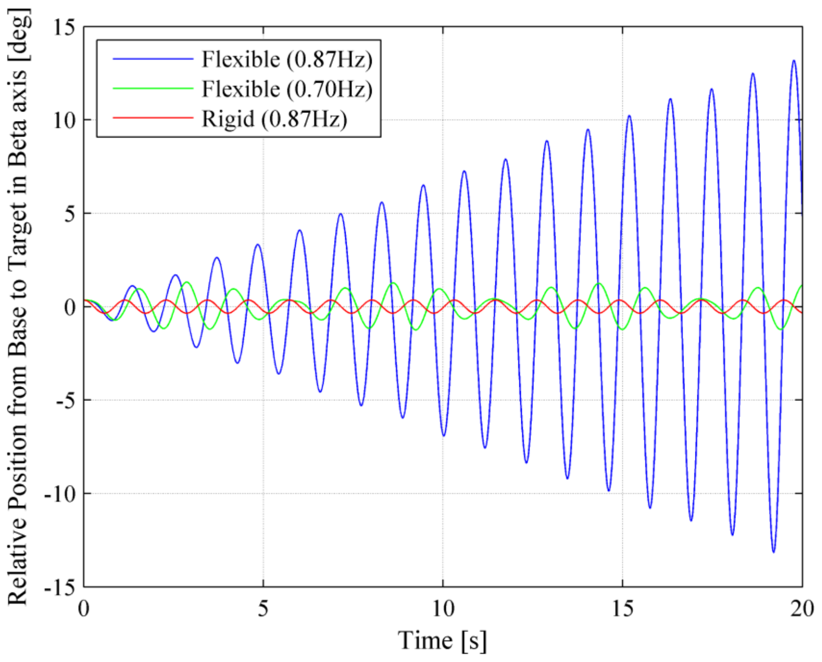


Fig. 3.5.7 Relative attitude from base to target in Beta axis.

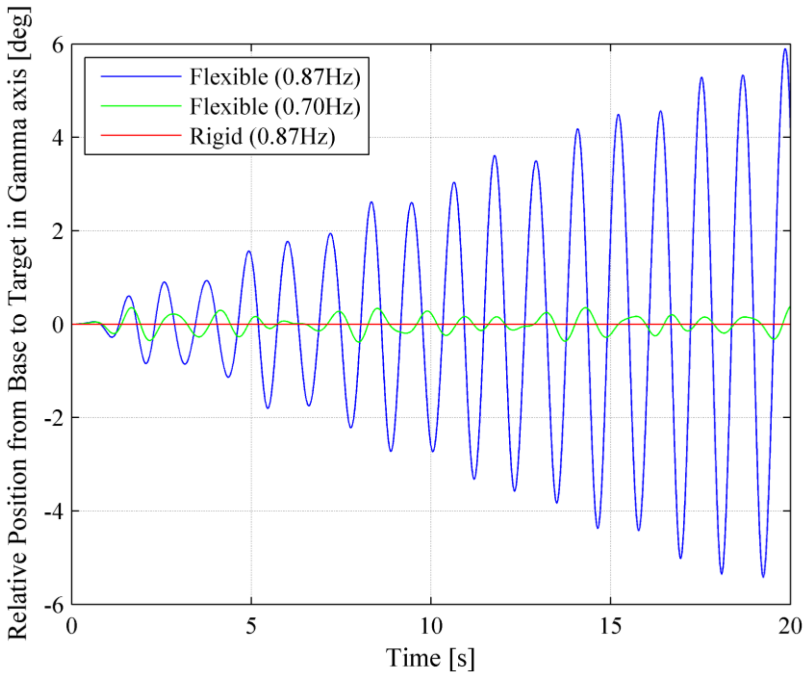


Fig. 3.5.8 Relative attitude from base to target in Gamma axis.

The relative position and attitude of the target from the base were oscillated by joint motion on Joint 2 in each case. In particular, the vibrations' amplitudes in the case of the flexible links with the eigenfrequency were dramatically increased due to the resonance. In the case of the flexible links with the non-eigenfrequency, the vibration amplitudes were not constant, but they were not diverged like the resonance case. In the case of the rigid links, the vibrations were constant even though the same frequency with the resonance case was applied on Joint 2 as an input.

< Target Position with respect to Inertial Frame >

The following figures compares the simulation results of target position with respect to the inertial frame during first 10 [sec] in each axis.

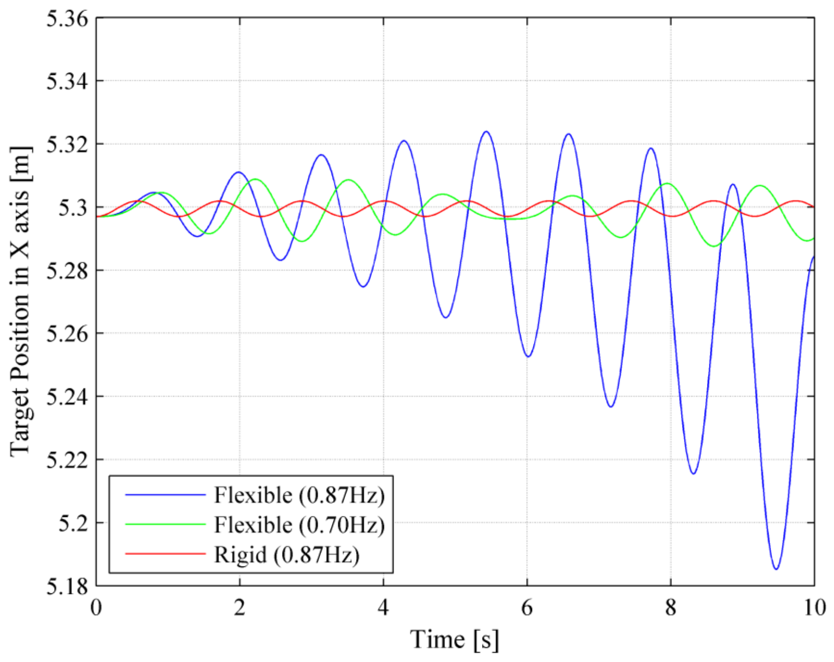


Fig. 3.5.9 Target position in X axis.

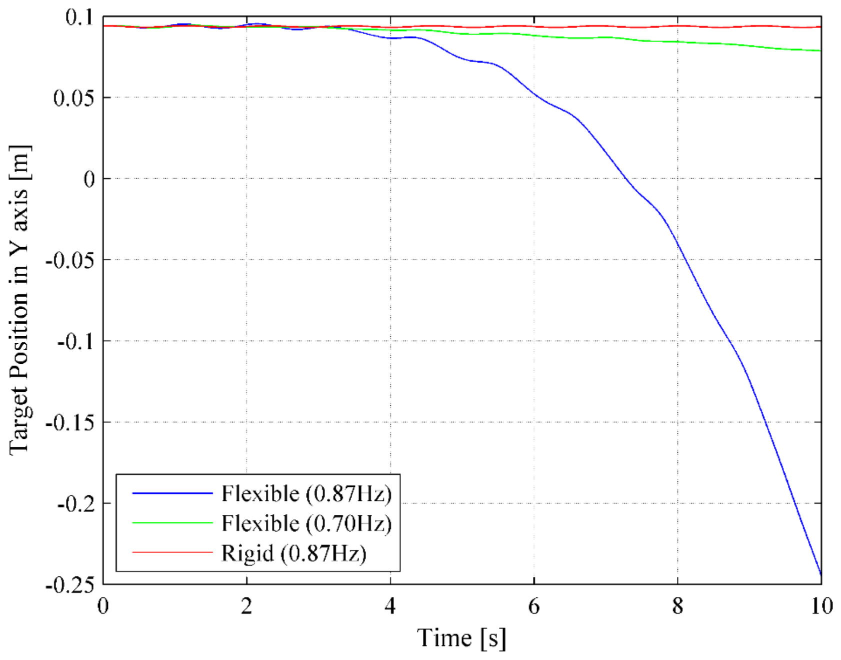


Fig. 3.5.10 Target position in Y axis.

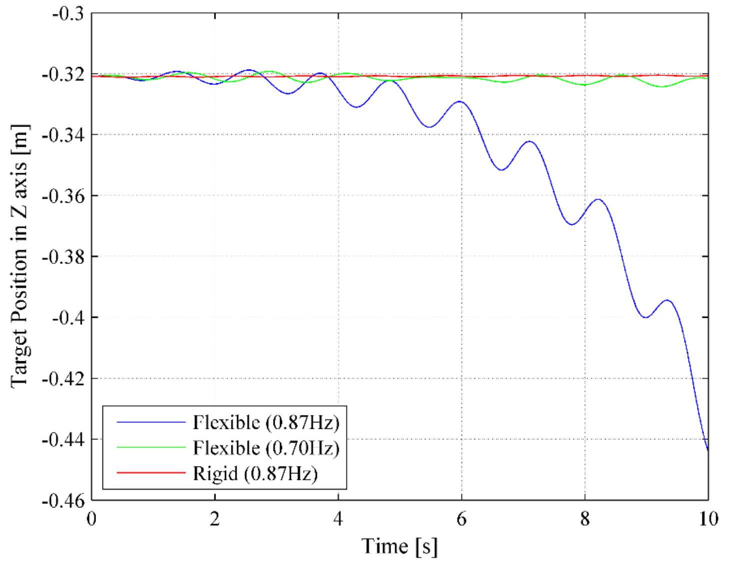


Fig. 3.5.11 Target position in Z axis.

The target position was oscillated and decreased from the initial position in each axis, while the position in the case of the rigid links was not changed. In particular, the large translational motion change was observed in the case of the flexible links with the first eigenfrequency. This behavior is likely to be an integration error due to the high speed motion by resonance effect. In general, the position of the center of mass for whole system should be stationery, even though a non-holonomic motion is observed. For more detailed analysis, it is recommended to verify the position of the center of mass for whole system.

< Base Position with respect to Inertial Frame >

The following graphs show the translational motion of base position with respect to the inertial frame during the first 10 [sec] in each axis.

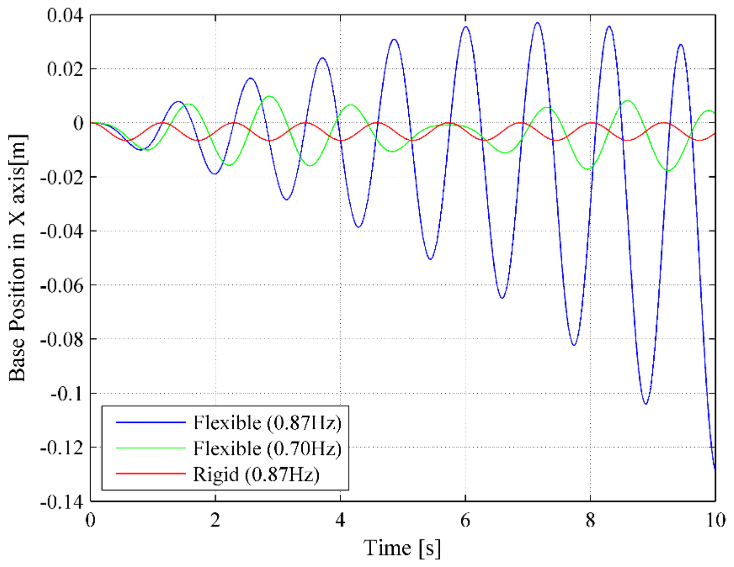


Fig. 3.5.12 Base position in X axis.

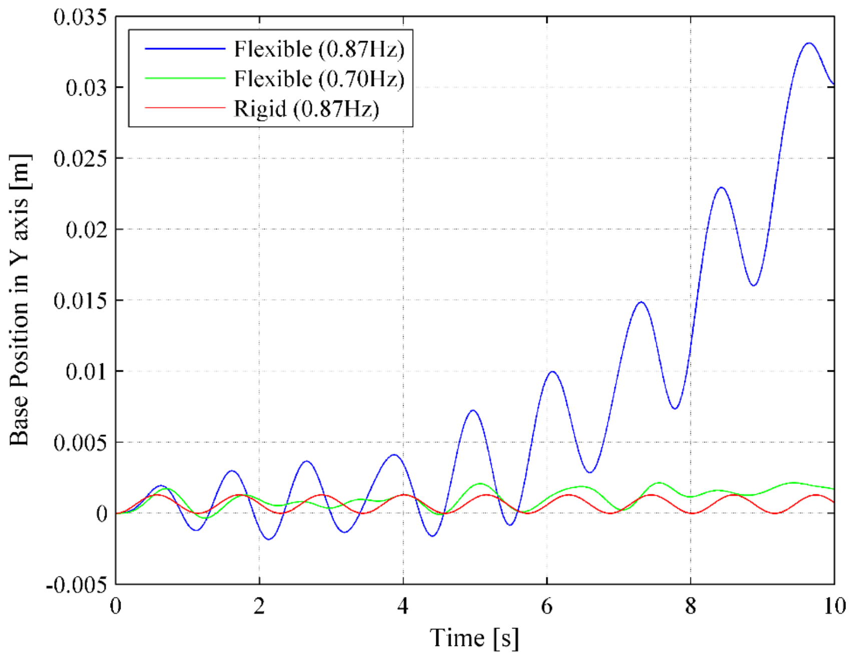


Fig. 3.5.13 Base position in Y axis.

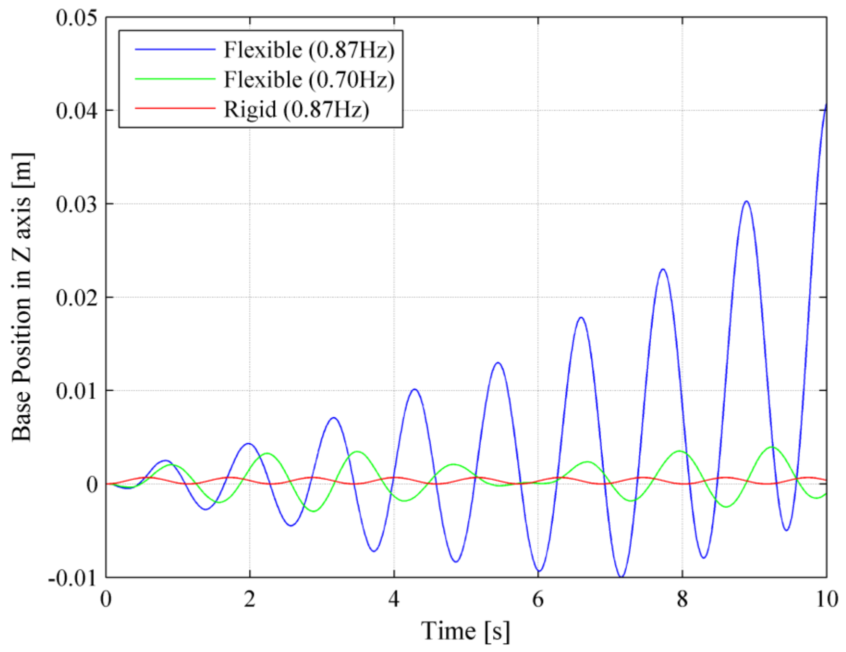


Fig. 3.5.14 Base position in Z axis.

The translational motions of the base position were observed as well as the target position. In the case of the rigid links, the base position was not dramatically changed, while the vibrational motion was observed. In contrast, the translational sliding motions were observed in the cases of the flexible links. The integration errors are considered a major cause of these sliding motions.

4. Conclusion

This report described the technical manual and the simulation results using SIMPACK. The first chapter introduced the overview of SIMPACK. The second chapter described the details on how to create a dynamic model in SIMPACK and how to perform dynamic simulation, including the setup for co-simulation with MATLAB/Simulink. In addition, the third chapter verified the developed model and analyzed the docking motion for DEOS. The result of model verification indicated that the develop model can emulate a similar behavior with a finite element model that was made by ASTRIUM. The analysis of docking motion simulation using the developed model showed that the maximum deformation of the relative target position from the base is less than 2 [mm], and therefore the effect of link flexibility is not critical for the actual operation. Finally, the resonance simulation was performed, and the simulation results showed the resonance behavior due to the link flexibility.