A DYNAMIC MODEL FOR MOTION OF AN ROV DUE TO ON-BOARD ROBOTICS

ANDREAS EVJENTH
OTTO ANDREAS MOE
ISELIN VIOLET KJELLAND SCHØN

Bachelor's thesis in Subsea Technology
Bergen, Norway 2017

Høgskulen på Vestlandet

Western Norway University of Applied Sciences

A DYNAMIC MODEL FOR MOTION OF AN ROV DUE TO ON-BOARD ROBOTICS

Iselin Violet Kjelland Schøn
Andreas Evjenth
Otto Andreas Moe

Department of Mechanical- and Marine Engineering
Western Norway University of Applied Sciences
NO-5063 Bergen, Norway

IMM 2017-M106

Høgskulen på Vestlandet
Avdeling for ingeniør- og økonomifag
Institutt for maskin- og marinfag
Inndalsveien 28,
NO-5063 Bergen, Norge

Norsk tittel:	Dynamiske utregninger av bevegelsene til en ROV forårsaket av robotikk ombord
Authors, student number:	Iselin Violet Kjelland Schøn, h147047 Otto Andreas Moe, h143340 Andreas Evjenth, h146783
Study program:	Subsea engineering
Date:	May 2017
Report number:	IMM 2017-M106
Supervisor at HVL:	Thomas J. Impelluso
Assigned by:	HVL
Contact person:	Thomas J. Impelluso
Antall filer levert digitalt:	1

Preface and acknowledgements

This report is the result of a bachelor's thesis written at the Department of Mechanical and Marine Engineering (IMM) at Western Norway University of Applied Sciences (HVL). The bachelor's thesis is an obligatory part of the study program to obtain a degree in Mechanical Engineering, worth 20 credit points (ECTS).

We are truly grateful for the assist from our mentor Professor Thomas J. Impelluso through the whole writing process. We are thankful for his availability and his contagious interest in the field of dynamics.

This thesis is also written as a conference paper for IMECE2017. The paper will be published in Florida, USA in November 2017, with article number IMECE2017-70110. We would like to thank the developers of the Moving Frame Method (MFM), Professor Hidenori Murakami and Professor Thomas J. Impelluso, for sharing their knowledge with the world. We are honored to have learned this method which has been implement in this work.

Date and place: 23.05.2017- Western Norway University of Applied Sciences (HVL).

Signed: Iselin V.K. Schøn

Iselin Violet Kjelland Schøn

Andreas Evjenth

Andreas Evjenth

Otto A. Moe

Otto Andreas Moe

Copyright statement

Sammendrag

En ny metode i dynamikk - The Moving Frame Method (MFM) - er brukt for å analysere hvordan manipulatorarmen på en undervannsfarkost (ROV) påvirker bevegelsen til ROV-en.

En ROV utfører mange forskjellige oppgaver på sjøbunnen i oljeserviceindustrien. I de fleste tilfeller, benyttes en ROV-pilot til å overvåke og justere bevegelsene til fartøyet som oppstår på grunn av strøm, oppdrift og bevegelse fra manipulatorarmene. Å kunne regne ut forventet bevegelse til ROV-en fra manipulatorarmene, vil kunne assistere piloten, samt gjøre klart for fremtidig automatisering av prosessen.

I denne rapporten utnyttes en ny metode for å analysere de induserte bevegelsene til ROV-en. Metoden benytter Special Euclidean Group (SE(3)) og MFM. Metoden kan brukes sammen med en begrenset variasjon av vinkelhastigheten for å beregne bevegelsesligningene til ROV-en. Bevegelsesligningene løses så numerisk ved Runge-Kutta og en formel for rekonstruksjon av rotasjonsmatrisen (som bygger på Cayley-Hamilton-teoremet) for å beregne 3D-rotasjonene til fartøyet. Deler av resultatet er visualisert gjennom 2D-plot, og ligningene benyttes som grunnlag for en 3D-animasjon som vises i en nettleser.

Rapporten avslutter med et sammendrag av forenklingene som er gjort i utregningene av modellen og gir forslag for videre fremtidig arbeid.

Abstract

A new method in dynamics—the Moving Frame Method (MFM)—is used to conduct the analysis of how a robotic appendage (manipulator) on a Remotely Operated Vehicle (ROV) affects the motion of the ROV.

An ROV performs multiple tasks on the seabed in the oil service industry. In most cases, an ROV pilot monitors and adjusts the movement of the vehicle due to induced motion by currents, buoyancy and the manipulators. Simulation data would assist the pilot and improve the stability of the ROV.

This paper exploits a new method to analyze the induced movements of the ROV. The method uses the Special Euclidean Group (SE(3)) and the MFM. The method is supplemented with a restricted variation on the angular velocity to extract the equations of motion for the ROV. Then the equations of motion are solved numerically using Runge-Kutta Method and a reconstruction formula (founded upon the Cayley-Hamilton theorem) to secure the 3D rotations of the vehicle. The resulting motion is visualized with selected 2D plots. The 3D animation is displayed on a 3D web page.

This paper closes with a summary of the simplifications used in the model and suggestions for advanced work.

Table of contents

Nomenclature

$[B]$	B-matrix
C	Center of mass
$[D]$	Combined angular velocity
E	Frame connection matrix
$\dot{E}$	Time derivative of frame connection matrix
e	Frame
$\{F^*\}$	Generalized force
$\{F\}$	Force and moment list
$\mathbf{H}$	Angular Momentum
$\{H\}$	Generalized momenta
I_3	3×3 Identity matrix
$J_c^{(\alpha)}$	3×3 Mass moment of inertia matrix
J	Joint
K	Kinetic energy
$\{L\}$	Linear momentum
$\mathbf{L}$	Linear momentum
L	Lagrangian
$[M]$	Mass matrix
$[M^*]$	Reduced mass matrix
$[N^*]$	Reduced nonlinear velocity matrix
q	Generalized position
$\dot{q}$	Generalized velocity
$\ddot{q}$	Generalized acceleration
$\{\dot{q}\}$	General velocity variable list
$\{\ddot{q}\}$	Generalized acceleration variable list
R	Rotation matrix
$\dot{R}$	Time derivative of rotation matrix
r	Absolute position vector
s	Position vector
U	Potential energy
δU	Variation of potential energy
δW	Virtual work
$\{\dot{X}\}$	Velocity list
$\{\tilde{X}\}$	Virtual displacements
x	Position vector
$\dot{x}$	Linear velocity vector
α	Index counter
δ	Variation
$\delta\Pi$	Variation of frame connection matrix
θ	Angle
$\overleftrightarrow{\delta\pi}$	Virtual rotational displacement
Ω	Time rate of the frame connection matrix
ω	Angular velocity vector
$\overleftrightarrow{\omega}$	Skew symmetric angular velocity matrix

1. Introduction

A Remotely Operated Vehicle (ROV) is a subsea vehicle that is heavily used in the oil and gas service industry. It consists of a frame with thrusters, umbilical cables and usually two manipulator arms. One arm is for securing the ROV to a subsea installation, and one—the focus of this study—for operating different types of tools [1].

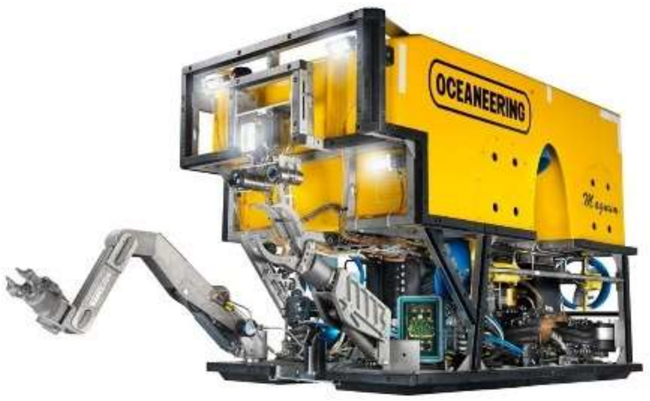


Figure 1 - Typical ROV [1]

In subsea operations, the ROV is exposed to many types of forces. Currents, buoyancy and waves cause the ROV to translate and rotate [2]. Also, motion of the manipulator can cause the ROV to rotate, when heavy tools are attached to it. These movements can cause a risk for the subsea operations.

Today, these challenges are being addressed using human pilots who continuously correct the position and rotation of the ROV. By understanding how the manipulator arms affect the rotations of the ROV, the motion can be corrected instantaneously instead of correcting the rotation after it has occurred. This paper analyzes such motion using the MFM. In the next section, the method is introduced along with a description of the model. A critical aspect of this analysis is that it was conducted by undergraduate students. This confirms that the MFM, equipped with a consistent notation across the sub-disciplines of dynamics, is a valuable approach in analysis.

2. Method

The Moving Frame Method (MFM) is founded on: 1) Lie Group Theory distilled to rotation matrices; 2) the concept of attaching a frame to all moving objects; and 3) a compact notation from the discipline of geometrical physics. A complete description of the MFM is found in reference [3].

The MFM is expanded with a notation that groups both rotation and displacement (SE(3)). Finally, a restriction on the variation of the angular velocity is obtained by ensuring the commutativity of the mixed partials (the variation and time).
Traditional methods can also readily extract the equations of motion. The point, however, is that this research was conducted entirely by undergraduate students. The MFM simplifies 3D dynamics. The goal of this work is not simply to solve the system, but to demonstrate this power of a new method. This introduction alludes to the foundation of the theory, but focuses on the application for real-world challenges. After the theory of the model has been documented, the equations of motion are extracted theoretically, and the computed symbolically and numerically using a symbolic manipulator (Matlab) and visualized in a 3D-animation using JavaScript in a HTML-script.

2.1 Model System Description

The model system analyzed in this paper is depicted in Fig. 2.

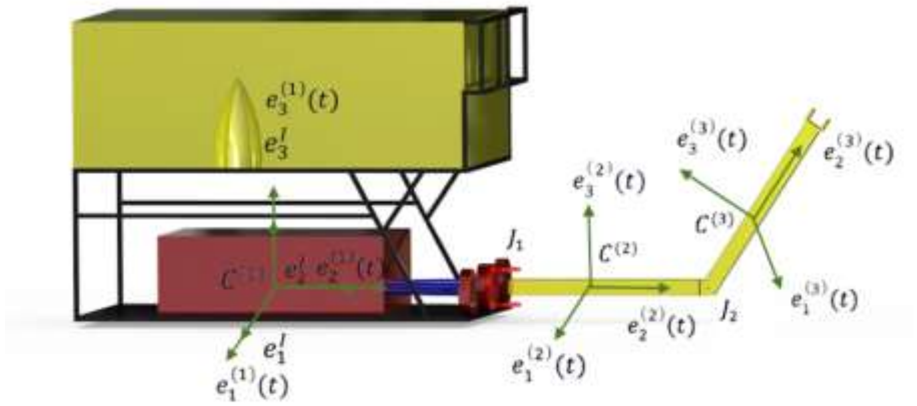


Figure 2 - Model System Description

The model system consists of an ROV—body(1)—with a two-linked robot arm (manipulator). The first arm is body(2), and the second arm is body(3). Each of these two latter bodies rotate in specified directions with a given amplitude. Each coordinate frame is defined using a vector basis consisting of three orthogonal vectors on each link or body. Furthermore, an inertial frame is deposited from the first body and this deposition defines the start time of the analysis. For drawing with dimensions of each individual part, see Appendix D.

2.2 Mathematical Model

Starting with the inner most component, the ROV itself, and progressing systematically to the first and then second arm of the manipulator. The bodies are numbered in ascending order from the ROV, to the first arm and to the second, distal, arm as the third component. This section begins, however, with a general overview of the MFM as applied to linked systems.

2.2.1 The Moving Frame Method (with SE(3))

The left side of Fig. 3 presents an inertial orthogonal coordinate system, designated by $\{x_1 \;\; x_2 \;\; x_3\}^T$ in dark arrows. The superscript "I" denotes association with the inertial system and the subscripts represent the coordinate lines. An inertial frame derives from tangent vectors to the coordinate system $\mathbf{e}^I = (\mathbf{e}_1^I \;\; \mathbf{e}_2^I \;\; \mathbf{e}_3^I)$, where $\mathbf{e}_i^I$ represents the unit tangent vector to the x_i-axis. Introducing the inertial frame defined by the coordinate basis $\mathbf{e}^I$ and the $\mathbf{0}$-position vector, this notation defines the origin as $(\mathbf{e}^I \;\; \mathbf{0})$.

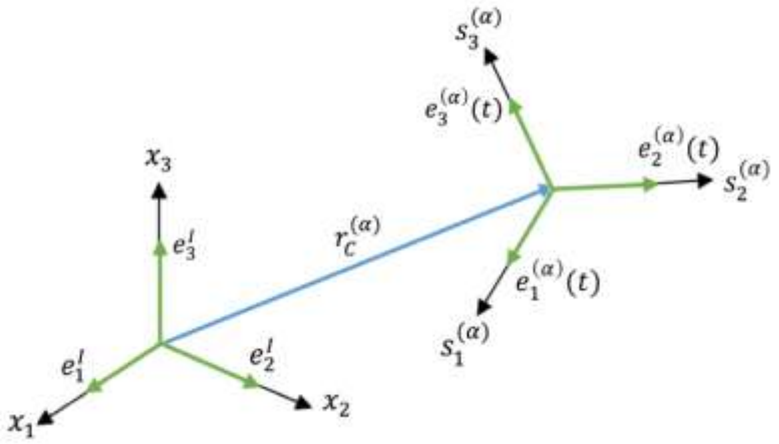


Figure 3 - Relations of the Frames

The right side of Fig. 3 presents a moving orthogonal coordinate system, designated $s_C^{(\alpha)}(t) = \left\{s_1^{(\alpha)} \; s_2^{(\alpha)} \;\; s_3^{(\alpha)}\right\}^T$, on body- α frame. It also shows the associated time dependent moving frame, expressed by $\mathbf{e}^{(\alpha)}(\mathrm{t}) = (\mathbf{e}_1^{(\alpha)}(\mathrm{t}) \;\; \mathbf{e}_2^{(\alpha)}(\mathrm{t}) \;\; \mathbf{e}_3^{(\alpha)}(\mathrm{t}))$. The superscript "$(\alpha)$" denotes variables that relates to body- α. In the same figure, it is shown that the vector $\boldsymbol{e}_i^{(\alpha)}(t)$ are the unit tangent vector to the $s_i^{(\alpha)}$-axis.

2.2.2 Kinematics of frames in general

In this section, the kinematics of the system is analyzed. A more foundational description of the method used for the kinematics, is found in reference [3]. The location of the center of mass $C^{(1)}(t)$ of the body-1 frame—the ROV—from the origin of the inertial frame, is expressed by the absolute position vector **r** with “x” as the absolute coordinate symbol:

$$\mathbf{r}_C^{(1)}(t) = \mathbf{e}^I x_C^{(1)}(\mathrm{t}) \tag{1}$$

The relative location of the center of mass $C^{(\alpha+1)}(\mathrm{t})$ of a subsequent body, from the center of mass $C^{(\alpha)}(\mathrm{t})$ of the previous body is expressed with vector $\boldsymbol{s}_C^{(\alpha+1/\alpha)}$ and coordinate symbol $s_C^{(\alpha+1/\alpha)}$and in the frame of the previous body:

$$\boldsymbol{s}_C^{(\alpha+1/\alpha)}(t) = \mathbf{e}^{(\alpha)}(\mathrm{t})\, s_C^{(\alpha+1/\alpha)}(t) \tag{2}$$

Thus, the absolute location of a frame is expressed as:

$$\mathbf{r}_C^{(\alpha+1)}(t) = \boldsymbol{r}_C^{(\alpha)}(t) + \mathbf{e}^{(\alpha)}(\mathrm{t})\, s_C^{(\alpha+1/\alpha)}(t) \tag{3}$$

Eqn. (3) reveals that to locate a frame, one first proceeds to a previous frame in a linked tree, and, from that frame, and in that frame’s basis, one proceeds to the current frame.

A 3×3 rotation matrix $R^{(\alpha)}(t)$ expresses the rotation of body-α vector-basis $\mathbf{e}^{(\alpha)}(\mathrm{t})$ from inertial vector-basis $\mathbf{e}^I$:

$$\mathbf{e}^{(\alpha)}(\mathrm{t}) = \mathbf{e}^I R^{(\alpha)}(\mathrm{t}) \tag{4}$$

Continuing, the relative rotation of a body-$(\alpha + 1)$ vector-basis $\mathbf{e}^{(\alpha+1)}(\mathrm{t})$ is given by a relative rotation matrix $R^{(\alpha+1/\alpha)}(\mathrm{t})$ as:

$$\mathbf{e}^{(\alpha+1)}(\mathrm{t}) = \mathbf{e}^{(\alpha)}(\mathrm{t})\, R^{(\alpha+1/\alpha)}(\mathrm{t}) \tag{5}$$

The orientation of the $\mathbf{e}^{(\alpha+1)}(\mathrm{t})$ body with respect to the inertial frame is, employing the algebra of SO(3):

$$\mathbf{e}^{(\alpha+1)}(\mathrm{t}) = \mathbf{e}^{I} R^{(\alpha)}(\mathrm{t}) R^{(\alpha+1/\alpha)}(\mathrm{t}) = \mathbf{e}^{I}\, R^{(\alpha+1)}(\mathrm{t}) \tag{6}$$

2.2.3 Frame Connections Matrices

The next step is to group both the rotation and displacement in one equation. The frame connection matrix, a form of a homogenous transformation matrix, combines rotations and translations. Homogenous transformation matrices were first used by Denavit and Hartenberg [4]. However they did not recognize that such transformations were members of the Special Euclidean Group denoted as SE(3). The summative development here follows that of Murakami with a consistent reference to SE(3) [5].

The body-α frame connection $(\mathbf{e}^{(\alpha)}(\mathrm{t})\ \ \boldsymbol{r}_C^{(\alpha)}(t))$ is obtained from the inertial frame by post-multiplying it with a frame connection matrix $E^{(\alpha)}(\mathrm{t})$:

$$(\mathbf{e}^{(\alpha)}(\mathrm{t})\ \ \boldsymbol{r}_C^{(\alpha)}(t) = \ (\mathbf{e}^{I}\ \ \mathbf{0})\, E^{(\alpha)}(\mathrm{t}) \tag{7}$$

The 4x4 frame connection matrix is shown as:

$$E^{(\alpha)}(\mathrm{t}) = \begin{bmatrix} R^{(\alpha)}(\mathrm{t}) & x_C^{(\alpha)}(\mathrm{t}) \\ 0_1^T & 1 \end{bmatrix} \tag{8}$$

In Eqn. (8), 0_1 is a 3×1 column zero vector, and $x_C^{(\alpha)}(\mathrm{t})$ designates coordinates from an inertial frame.

Next, the relative frame connection matrix is described for body $(\alpha + 1)$ with respect to body-α.

$$(\mathbf{e}^{(\alpha+1)}(\mathrm{t})\ \ \boldsymbol{r}_C^{(\alpha+1)}(t)) = \ (\mathbf{e}^{(\alpha)}(\mathrm{t})\, \boldsymbol{r}_C^{(\alpha)}(t)) E^{(\alpha+1/\alpha)}(\mathrm{t}) \tag{9}$$

where:

$$E^{(\alpha+1/\alpha)}(\mathrm{t}) = \begin{bmatrix} R^{(\alpha+1/\alpha)}(\mathrm{t}) & s_C^{(\alpha+1/\alpha)}(\mathrm{t}) \\ 0_1^T & 1 \end{bmatrix} \tag{10}$$

Progressing through the link-tree, the product of the absolute α-frame connection matrix and the relative $(\alpha + 1/\alpha)$-frame connection matrix, becomes an absolute connection matrix of $(\alpha + 1)$ from an inertial frame.

$$E^{(\alpha+1)}(\mathrm{t}) = E^{(\alpha)}(\mathrm{t})E^{(\alpha+1/\alpha)}(\mathrm{t}) \tag{11}$$

2.2.4 Kinematics of ROV

With this short foundation, attention is now paid to the first body in the linked tree: the ROV itself. The frame connection matrix for the ROV $E^{(1)}(t)$ is assembled as:

$$E^{(1)}(t) = \begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix} \tag{12}$$

Both the orientation and the position of the first frame (the ROV) is expressed directly with respect to the inertial frame as shown below as:

$$(\mathbf{e}^{(1)}(t) \;\; \mathbf{r}_C^{(1)}(t)) = (\mathbf{e}^I \;\; \mathbf{0})E^{(1)}(t) \tag{13}$$

This gives:

$$(\mathbf{e}^{(1)}(t) \;\; \mathbf{r}_C^{(1)}(t)) = (\mathbf{e}^I \;\; \mathbf{0})\begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix} = (\mathbf{e}^I R^{(1)}(t) \;\; \mathbf{e}^I x_C^{(1)}(t)) \tag{14}$$

The inverse, $\left(E^{(1)}(t)\right)^{-1}$ of the frame connection matrix is obtained by the matrix below where the structure of SE(3) is exploited:

$$\left(E^{(1)}(t)\right)^{-1} = \begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix}^{-1} = \begin{bmatrix} (R^{(1)}(t))^T & -(R^{(1)}(t))^T x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix} \tag{15}$$

The time derivative of the frame connection matrix:

$$\dot{E}^{(1)}(t) = \begin{bmatrix} \dot{R}^{(1)}(\mathrm{t}) & \dot{x}_C^{(1)}(\mathrm{t}) \\ 0_1^T & 0 \end{bmatrix} \tag{16}$$

The time derivative of the Eqn. (14) defines the linear velocity and the angular velocity vectors of the ROV:

$$\begin{aligned}(\dot{\mathbf{e}}^{(1)}(t) \;\; \dot{\mathbf{r}}_C^{(1)}(t)) = (\mathbf{e}^I \;\; \mathbf{0})\dot{E}^{(1)}(t) &= (\mathbf{e}^{(1)} \;\; (t) \;\; \mathbf{r}_C^{(1)}(t))\left(E^{(1)}(t)\right)^{-1}\dot{E}^{(1)}(t) \\ &= (\mathbf{e}^{(1)} \;\; (t) \;\; \mathbf{r}_C^{(1)}(t))\Omega^{(1)}(t)\end{aligned} \tag{17}$$

The upper left sub-matrix of $\Omega^{(1)}(t)$ contains the skew symmetric angular velocity matrix $\overleftrightarrow{\omega}^{(1)}(t)$. By multiplying out the terms in Eqn. (17) one finds:

$$\overleftrightarrow{\omega}^{(1)}(t) = \left(R^{(1)}(t)\right)^T \dot{R}^{(1)}(t) \tag{18}$$

The matrix $\Omega^{(1)}(t)$ in Eqn. (17), is referred to as the time rate of the frame connection matrix. The matrix includes the information describing both the linear velocity and angular velocity of the ROV coordinate frame:

$$\Omega^{(1)}(t) = \begin{bmatrix} \overleftrightarrow{\omega}^{(1)}(t) & \left(R^{(1)}(t)\right)^T \dot{x}_C^{(1)}(t) \\ 0_1^T & 0 \end{bmatrix} \tag{19}$$

The matrix above is used to express the time-rate of the ROV coordinate frame:

$$\dot{\boldsymbol{e}}^{(1)}(t) = \boldsymbol{e}^{(1)}(t)\overleftrightarrow{\omega^{(1)}(t)} = \begin{bmatrix} 0 & -\omega_3^{(1)}(t) & \omega_2^{(1)}(t) \\ \omega_3^{(1)}(t) & 0 & -\omega_1^{(1)}(t) \\ \omega_2^{(1)}(t) & \omega_1^{(1)}(t) & 0 \end{bmatrix} \tag{20}$$

Exploiting the isomorphism between the skewed form in Eqn. (20) and the column representation, one obtains the angular velocity vector for the ROV in terms of the ROV frame:

$$\boldsymbol{\omega}^{(1)}(t) = \mathbf{e}^{(1)}(t)\begin{pmatrix} \omega_1^{(1)} \\ \omega_2^{(1)} \\ \omega_3^{(1)} \end{pmatrix} \tag{21}$$

The linear velocity vector $\dot{x}_C^{(1)}(t)$ for the ROV, is also expressed with respect to the inertial frame.

$$\dot{\mathbf{r}}_C^{(1)} = \mathbf{e}^{(1)}\left(R^{(1)}(t)\right)^T \dot{x}_C^{(1)}(t) = \mathbf{e}^I \dot{x}_C^{(1)}(t) \tag{22}$$

Equation (21) and (22) are the first two required expressions needed in the analysis. Attention is now turned to the second body (the first link).

2.2.5 Kinematics for the first arm

For the first robotic arm (the second body), a frame is placed at the center of mass $C^{(2)}$ of the first arm. The system is designed with revolute joints. The MFM is also capable of handle other types of joints like ball and socket joints, but in this analysis planar rotation is assumed.

A Cartesian coordinate system for the first arm is set to be $\left\{ \mathrm{s}_1^{(2)}\ \mathrm{s}_2^{(2)}\ \mathrm{s}_3^{(2)} \right\}^T$, and the coordinate frame for the first arm is $\mathbf{e}^{(2)}(\mathrm{t}) = (\mathbf{e}_1^{(2)}(\mathrm{t})\ \ \mathbf{e}_2^{(2)}(\mathrm{t})\ \ \mathbf{e}_3^{(2)}(\mathrm{t}))$.

The relative position vector from the center of mass for the ROV $C^{(1)}$ to the center of mass for the first arm $C^{(2)}$ is found by translating, rotating, and translating again. Thus, to reach the first joint J_1at the end of the ROV where the first arm connects:

$$\boldsymbol{s}_{\mathrm{J}_1} = \mathbf{e}^{(1)}(t) s_{J_1} = \mathbf{e}^{(1)}(t) \begin{pmatrix} 0 \\ s_{J_1} \\ 0 \end{pmatrix} \tag{23}$$

The first arm rotates about one single axis, specifically the $\mathbf{e}_3^{(2)}(t)$ axis, so the relation between the ROV and the first arm is expressed as follows:

$$\mathbf{e}^{(2)}(t) = \mathbf{e}^{(1)}(t) R^{(2/1)}(t) \tag{24}$$

where:

$$R^{(2/1)}(t) = R_3^{(2/1)}(\theta^{(2)}) = \begin{bmatrix} cos(\theta^{(2)}) & -sin(\theta^{(2)}) & 0 \\ sin(\theta^{(2)}) & cos(\theta^{(2)}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{25}$$

To reach the center of mass for the first arm $C^{(2)}$ from the first joint J_1 at the end of the ROV, second frame is used as (where the second axis points directly along the first arm):

$$\boldsymbol{s}_{C_2} = \mathbf{e}^{(2)}(t) s_{C_2} = \mathbf{e}^{(2)}(t) \begin{pmatrix} 0 \\ s_{C_2} \\ 0 \end{pmatrix} \tag{26}$$

The relative position vector from the first frame $\mathbf{e}^{(1)}(t)$ at the center of mass of the ROV $C^{(1)}$ to the center of mass of the first arm $C^{(2)}$ is found as:

$$\boldsymbol{s}_C^{(2/1)} = \mathbf{e}^{(1)}(t) s_C^{(2/1)} \tag{27}$$

The connection matrix for the first arm is be obtained as:

$$\left(\mathbf{e}^{(2)}(t) \;\; \mathbf{r}_c^{(2)}(t)\right) = \left(\mathbf{e}^{(1)} \;\; (t)\, \mathbf{r}_c^{(1)}(t)\right) E^{(2/1)}(t) \tag{28}$$

Thus, finally, $E^{(2/1)}(t)$ is the compact notation for the frame connection matrix:

$$\begin{aligned} E^{(2/1)}(\mathrm{t}) &= \begin{bmatrix} R^{(2/1)}(\mathrm{t}) & s_C^{(2/1)} \\ 0_1^T & 1 \end{bmatrix} = \begin{bmatrix} I_3 & s_{J_1} \\ 0_1^T & 1 \end{bmatrix} \begin{bmatrix} R^{(2/1)}(t) & 0_1 \\ 0_1^T & 1 \end{bmatrix} \begin{bmatrix} I_3 & s_{C_2} \\ 0_1^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} R^{(2/1)}(t) & R^{(2/1)}(t) s_{C_2} + s_{J_1} \\ 0_1^T & 1 \end{bmatrix} \end{aligned} \tag{29}$$

The relative connection matrix for the first arm written with respect to inertial frame:

$$\left(\mathbf{e}^{(2)}(t) \;\; \mathbf{r}_c^{(2)}(t)\right) = (\mathbf{e}^I \;\; \mathbf{0}) E^{(2)}(t) = \left(\boldsymbol{e}^{\mathrm{I}} \;\; \mathbf{0}\right) E^{(1)} E^{(2/1)} \tag{30}$$

where the $E^{(2)}(t)$ frame connection matrix is related to the inertial frame.

In Eqn. (30) $E^{(2)}(t)$ consists of $R^{(2)}(t)$, (in the upper left quadrant) and $x_C^{(2)}(t)$ (in the upper right quadrant):

$$R^{(2)}(t) = R^{(1)}(t) R^{(2/1)}(t) \tag{31}$$

$$x_C^{(2)}(t) = R^{(2)}(t) s_{C_2} + R^{(1)}(t) s_{J_1} + x_c^{(1)}(t) \tag{32}$$

With this information, a rate analysis is conducted like that leading up to Eqn. (19), and specific information is extracted as follows:

The angular velocity vector for the first arm:

$$\omega^{(2)}(t) = (R^{(2/1)}(t))^T\omega^{(1)}(t) + \omega^{(2/1)}(t) = (R^{(2/1)}(t))^T\omega^{(1)}(t) + \dot{\theta}^{(2)}e_3 \tag{33}$$

where $\omega^{(2/1)}(t)$ is the rotation for the first arm and only about one axis, therefore it can be represented with the notation $\dot{\theta}^{(2)}e_3$. This notation is also used for the second arm.

The linear velocity vector for the first arm with respect to inertial frame:

$$\dot{x}_C^{(2)}(t) = R^{(1)}(t)\overleftrightarrow{\omega}^{(1)}(t)s_{J_1} + \dot{x}_C^{(1)}(t) = R^{(1)}(t)\left(\overleftrightarrow{s_{J_1}}\right)^T\omega^{(1)}(t) + \dot{x}_C^{(1)}(t) \tag{34}$$

Equation (33) and (34) are the second two required expressions needed in the analysis. Attention is now turned to the third body (the second link).

2.2.6 Kinematics for the second arm

Similarly, to the first frame, a Cartesian coordinate system for the second arm is set to be $\left\{\mathrm{s}_1^{(3)}\ \mathrm{s}_2^{(3)}\ \mathrm{s}_3^{(3)}\right\}^T$, and the frame for the second arm is: $\mathbf{e}^{(3)}(\mathrm{t}) = (\mathbf{e}_1^{(3)}(\mathrm{t})\ \ \mathbf{e}_2^{(3)}(\mathrm{t})\ \ \mathbf{e}_3^{(3)}(\mathrm{t}))$.

For the second robotic arm, a frame is placed at the center of mass $C^{(3)}$. The relative position vector from the center of mass for the first arm $C^{(2)}$ to the center of mass for the second arm $C^{(3)}$, is found by an analysis like that leading up to Eqn. (27).

The distance from the center of mass for the first arm $C^{(2)}$ to the second joint J_2, and the distance from the second joint J_2 to the center of mass for the second arm $C^{(3)}$ is respectively defined as:

$$\boldsymbol{s}_{J_2} = \mathbf{e}^{(2)}(t)s_{J_2} = \mathbf{e}^{(2)}(t)\begin{pmatrix}0\\ s_{J_2}\\ 0\end{pmatrix} \tag{35}$$

$$\boldsymbol{s}_{C_3} = \mathbf{e}^{(3)}(t)s_{C_3} = \mathbf{e}^{(3)}(t)\begin{pmatrix}0\\ s_{C_3}\\ 0\end{pmatrix} \tag{36}$$

As with the first arm, the second arm also rotates about one single axis, the $\mathbf{e}_1^{(3)}(t)$ axis. This relation between the first arm and the second arm is expressed as follows:

$$\mathbf{e}^{(3)}(t) = \mathbf{e}^{(2)}(t)R^{(3/2)}(t) \tag{37}$$

$$R^{(3/2)}(t) = R_1^{(3/2)}(\theta^3) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta^{(3)}) & -sin(\theta^{(3)}) \\ 0 & sin(\theta^{(3)}) & sin(\theta^{(3)}) \end{bmatrix} \tag{38}$$

This is also a planar rotation, but now both arms are rotating in different planes and this is now a full 3D problem.

The relative position vector from the center of mass of the first arm $C^{(2)}$ to the center of mass of the second arm $C^{(3)}$ is found, using the second frame as:

$$\boldsymbol{s}_C^{(3/2)} = \mathbf{e}^{(2)}(t)s_C^{(3/2)} \tag{39}$$

The second arm connection matrices are expressed with respect to the inertial frame as:

$$\left(\mathbf{e}^{(3)}(t)\ \ \mathbf{r}_C^{(3)}\right) = \left(\mathbf{e}^{(2)}(t)\ \mathbf{r}_c^{(2)}\right)E^{(3/2)}(t)$$

$$\left(\mathbf{e}^{(3)}(t)\ \ \mathbf{r}_C^{(3)}\right) = (\mathbf{e}^{I}\quad \mathbf{0})E^{(1)}(t)E^{(2/1)}(t)E^{(3/2)}(t) \tag{40}$$

$$\left(\mathbf{e}^{(3)}(t)\ \ \mathbf{r}_C^{(3)}\right) = (\mathbf{e}^{I}\quad \mathbf{0})E^{(3)}(t)$$

Where the frame connection matrix between the first arm and the second arm are now obtained as:

$$E^{(3/2)}(t) = \begin{bmatrix} I_3 & s_{J_2} \\ 0_1^T & 1 \end{bmatrix}\begin{bmatrix} R^{(3/2)}(t) & 0_1 \\ 0_1^T & 1 \end{bmatrix}\begin{bmatrix} I_3 & s_{C_3} \\ 0_1^T & 1 \end{bmatrix} = \begin{bmatrix} R^{(3/2)}(t) & R^{(3/2)}(t)s_{C_3} + s_{J_2} \\ 0_1^T & 1 \end{bmatrix} \tag{41}$$

In Eqn. (40) $E^{(3)}(t)$ consists of $R^{(3)}(t)$, and $x_C^{(3)}(t)$:

$$R^{(3)}(t) = R^{(1)}(t)R^{(2/1)}(t)R^{(3/2)}(t) \tag{42}$$

$$x_C^{(3)}(t) = R^{(3)}(t)s_{C_3} + R^{(2)}(t)(s_{C_2} + s_{J_2}) + R^{(1)}(t)s_{J_1} + x_C^{(1)}(t) \tag{43}$$

The angular velocity vector for the second arm is obtained as:

$$\omega^{(3)}(t) = (R^{(3/1)}(t))^T\omega^{(1)}(t) + (R^{(3/2)}(t))^T\dot{\theta}^{(2)}\mathbf{e}_3 + \dot{\theta}^{(3)}\mathbf{e}_1 \tag{44}$$

The linear velocity vector for the second arm with respect to inertial frame:

$$\begin{gathered}\dot{x}_C^{(3)}(t) = \; R^{(1)}(t)R^{(2/1)}(t)R^{(3/2)}(t)\left(\overleftrightarrow{s_{J_2}}\right)^T\omega^{(3)}(t) + R^{(1)}(t)R^{(2/1)}\left(s_{J1}\right)^T\omega^{(2)}(t) \\ +R^{(1)}(t)\left(\overleftrightarrow{s_{J_1}}\right)^T\omega^{(1)}(t) + \dot{x}_C^{(1)}(t)\end{gathered} \tag{45}$$

Equation (44) and (45) are the last two required expressions needed in the analysis. This closes the study of kinematics. Attention is now turned to kinetics.

2.3 Kinetics

Now turning to the Kinetics, and commencing with the generalized coordinate, Hamilton's Principle is applied. The details regarding the method are found in the references [6, 7].

2.3.1 Generalized Coordinates

The generalized velocity $\{\dot{\mathrm{X}}(\mathrm{t})\}$ is defined as a 6n×1 matrix that consists of both linear velocity $\dot{\mathrm{x}}_\mathrm{C}^{(\alpha)}$and angular velocities $\omega^{(\alpha)}$ at the center of mass, in alternating order, for each body.

For a three-linked system, as analyzed in this paper, the vector $\{\dot{X}(t)\}$ has 18 rows. The coordinates $\dot{x}_C^{(1)}(t), \dot{x}_C^{(2)}(t)$ and $\dot{x}_C^{(3)}(t)$ are all presented from the inertial frame.

$$\{\dot{X}(t)\} = \begin{pmatrix}\dot{x}_C^{(1)}(t)\\ \omega^{(1)}(t)\\ \dot{x}_C^{(2)}(t)\\ \omega^{(2)}(t)\\ \dot{x}_C^{(3)}(t)\\ \omega^{(3)}(t)\end{pmatrix} \tag{46}$$

The generalized velocity $\{\dot{X}(t)\}$ as in Eqn. (46) is related linearly to essential generalized velocity $\{\dot{q}(t)\}$[6]. This is an $n^{*}\times 1$ matrix for a system with n^{*}degrees of freedom. For the ROV, there is 8 degrees of freedom, and $\{\dot{q}(t)\}$ is defined as:

$$\{\dot{q}(t)\} = \begin{pmatrix} \dot{x}_C^{(1)}(t) \\ \omega^{(1)}(t) \\ \dot{\theta}^{(2)}(t) \\ \dot{\theta}^{(3)}(t) \end{pmatrix} \tag{47}$$

These terms are, respectively: the translational velocity of the ROV (3 components), the angular velocity of the ROV (3 components), the rotational velocity of the first arm and the rotational velocity of the second arm—a total of eight.

The linear relationship between $\{\dot{X}(t)\}$ and $\{\dot{q}(t)\}$ is expressed as:

$$\{\dot{X}(t)\} = [B(t)]\{\dot{q}(t)\} \tag{48}$$

The B matrix $[B(t)]$ in Eqn. (48) is obtained by considering Eqn. (21), (22), (33), (34), (44), and (45).

2.3.2 Hamilton's Principle

For Kinetics, the Newton and Euler's equations is derived (the equations of motion). Hamilton's Principle is [6]:

$$\delta \int_{t_0}^{t_1} L^{(\alpha)}(t)+ W^{(\alpha)}(t))dt = 0 \tag{49}$$

The Lagrangian $L^{(\alpha)}(t)$ is defined as the difference between kinetic energy $K^{(\alpha)}(t)$ and potential energy $U^{(\alpha)}(t)$.

$$L^{(\alpha)}(t) \equiv K^{(\alpha)}(t) - U^{(\alpha)}(t) \tag{50}$$

Principle of Virtual work is redefined from Hamilton's principle as:

$$\delta \int_{t_0}^{t_1}(K^{(\alpha)}(t) - U^{(\alpha)}(t)+ W^{(\alpha)}(t))dt = 0 \tag{51}$$

Here, "U" continues to designate potential energy from conservative forces such as gravity, while "W" represents the work done by the non-conservative forces.

The Kinetic energy $K^{(\alpha)}(t)$ and the potential energy $U^{(\alpha)}(t)$ is respectively defined as:

$$K^{(\alpha)}(t) = \frac{1}{2}\left\{\dot{\mathbf{r}}_C^{(\alpha)}\mathbf{L}_C^{(\alpha)} + \boldsymbol{\omega}^{(\alpha)}\mathbf{H}_C^{(\alpha)}\right\} \tag{52}$$

$$U^{(\alpha)}(t) = m^{(\alpha)}gx_{3C}^{(\alpha)}(t) \tag{53}$$

The Kinetic energy $K^{(\alpha)}(t)$ consists of translational and rotational components with respect to the center of mass of each link in the system.

The linear momentum $\mathbf{L}_C^{(\alpha)}$ and the angular momentum $\mathbf{H}_C^{(\alpha)}$,are respectively defined as:

$$\mathbf{L}_C^{(\alpha)}(t) \equiv \mathbf{e}^I m^{(\alpha)}\dot{x}_C^{(\alpha)}(t) \tag{54}$$

$$\mathbf{H}_C^{(\alpha)}(t) \equiv \mathbf{e}^{(\alpha)}(t)^{(\alpha)}J_C^{(\alpha)}\omega^{(\alpha)}(t) \tag{55}$$

Where the mass moment of inertia matrix $J_C^{(\alpha)}$ is a 3×3 matrix. If the coordinate systems are placed at the center of mass in each link, and oriented such that the axes coincide with the principal axes of each link, then the $J_C^{(\alpha)}$ matrices are diagonal.

$$J_C^{(\alpha)} = \begin{bmatrix} J_{11}^{(\alpha)} & 0 & 0 \\ 0 & J_{22}^{(\alpha)} & 0 \\ 0 & 0 & J_{33}^{(\alpha)} \end{bmatrix} \tag{56}$$

When substituting Eqn. (54) and (55) into Eqn. (52), the component form of kinetic energy becomes:

$$K^{(\alpha)}(t) = \frac{1}{2}\left\{\dot{x}_C^{(\alpha)^T} m^{(\alpha)}\dot{x}_C^{(\alpha)} + \omega^{(\alpha)}J_C^{(\alpha)}\omega^{(\alpha)}\right\} \tag{57}$$

Eqn. (57) is expressed more compactly as:

$$K(t) = \tfrac{1}{2}\{\dot{X}(t)\}^T\{H(t)\} = \tfrac{1}{2}\{\dot{X}(t)\}^T[M]\{\dot{X}(t)\} \tag{58}$$

The generalized momenta $\{H(t)\}$ is a 6n×1 matrix that contains both the linear momentum $\mathbf{L}_{\mathrm{C}}^{(\alpha)}$and the angular momentum $\mathbf{H}_{C}^{(\alpha)}$, respectively defined as in Eqn. (54) and (55):

$$\{H(t)\} = \begin{pmatrix} L_C^{(1)}(t) \\ H_C^{(1)}(t) \\ L_C^{(2)}(t) \\ H_C^{(2)}(t) \\ L_C^{(3)}(t) \\ H_C^{(3)}(t) \end{pmatrix} = [M]\{\dot{X}(t)\} \tag{59}$$

To build the generalized momenta $\{H(t)\}$ as in Eqn. (59) the Generalized mass matrix $[M]$ is defined. It is defined as a $6n$×$6n$ matrix, that contains the mass $m^{(\alpha)}$ and moment of inertia $\mathrm{J}_C^{(\alpha)}$ for each link in the system:

$$[M] = \begin{bmatrix} m^{(1)}I_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & J_C^{(1)} & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & m^{(2)}I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & J_C^{(2)} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & m^{(3)}I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & J_C^{(3)} \end{bmatrix} \tag{60}$$

where I_3 is a 3×3 identity matrix, and 0_3 is a 3×3 zero matrix.

When inserting Eqn. (46), (59), (60) into Eqn. (58), the equation for kinetic energy for the ROV turns out:

$$K(t) = \tfrac{1}{2}\begin{pmatrix} \dot{x}_C^{(1)}(t) \\ \omega^{(1)}(t) \\ \dot{x}_C^{(2)}(t) \\ \omega^{(2)}(t) \\ \dot{x}_C^{(3)}(t) \\ \omega^{(3)}(t) \end{pmatrix}^T \begin{pmatrix} L_C^{(1)}(t) \\ H_C^{(1)}(t) \\ L_C^{(2)}(t) \\ H_C^{(2)}(t) \\ L_C^{(3)}(t) \\ H_C^{(3)}(t) \end{pmatrix} \tag{61}$$

To derive the principal of virtual work, the variation of the Lagrangian $L^{(\alpha)}(t)$ as in Eqn. (51) is required, and hence the variation of the generalized velocities must be obtained.

To take the variation of the Lagrangian, the variation of the generalized velocities must be obtained. More specifically, the variation of the angular velocity is needed. It will be seen that there exists a restriction on the angular velocity. To this end, some definitions are presented and their utility is shown, shortly.

First, define $\delta\Pi^{(\alpha)}$ as:

$$\delta\Pi^{(\alpha)} = \begin{bmatrix} \overleftrightarrow{\delta\pi^{(\alpha)}(t)} & \left(R^{(\alpha)}(t)\right)^T \delta x_c^{(\alpha)}(t) \\ 0_1^T & 0 \end{bmatrix} \tag{62}$$

Where the virtual rotational displacement $\overleftrightarrow{\delta\pi^{(\alpha)}}$ is defined as:

$$\overleftrightarrow{\delta\pi^{(\alpha)}(t)} = \left(R^{(\alpha)}(t)\right)^T \delta R^{(\alpha)}(t) \tag{63}$$

From the definition in Eqn. (62) the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ is defined as a $6n\times1$ column matrix:

$$\{\delta\tilde{X}(t)\} = \begin{pmatrix} \delta x_C^{(1)}(t) \\ \delta\pi^{(1)}(t) \\ \delta x_C^{(2)}(t) \\ \delta\pi^{(2)}(t) \\ \delta x_C^{(3)}(t) \\ \delta\pi^{(3)}(t) \end{pmatrix} \tag{64}$$

As in Eqn. (48), the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ has its relationship also through the $[B(t)]$ matrix as per Murakami [3]. The linear relation gives the virtual essential generalized displacement $\{\delta q(t)\}$.

$$\{\delta\tilde{X}(t)\} = [B(t)]\{\delta q(t)\} \tag{65}$$

For a three-linked system, the relationship between $\{\delta\tilde{X}(t)\}$ and $\{\delta q(t)\}$ is expressed as:

$$\begin{pmatrix} \delta x_C^{(1)}(t) \\ \delta\pi^{(1)}(t) \\ \delta x_C^{(2)}(t) \\ \delta\pi^{(2)}(t) \\ \delta x_C^{(3)}(t) \\ \delta\pi^{(3)}(t) \end{pmatrix} = [B(t)] \begin{pmatrix} \delta x_C^{(1)}(t) \\ \delta\pi^{(1)}(t) \\ \delta\theta^{(2)}(t) \\ \delta\theta^{(3)}(t) \end{pmatrix} \tag{66}$$

Introducing the commutativity of time differentiation and variation of a frame.

$$\delta\omega^{(\alpha)}(t) = \frac{d}{dt}\delta\pi^{(\alpha)}(t) + \overleftrightarrow{\omega^{(\alpha)}(t)}\delta\pi^{(\alpha)}(t) \tag{67}$$

$$\frac{d}{dt}\delta x_C^{(\alpha)}(t) = \delta\dot{x}_C^{(\alpha)}(t) \tag{68}$$

Obtaining Eqn. (67) is a bit of work and shown reference [5]. Eqn. (67) and (68) are more compactly expressed in the matrix form of virtual generalized velocity $\{\delta\dot{X}(t)\}$, which is defined as:

$$\{\delta\dot{X}(t)\} = \{\delta\dot{\tilde{X}}(t)\} + [D(t)]\{\delta\tilde{X}(t)\} \tag{69}$$

Where $[D(t)]$ is a $6n\times6n$ skew symmetric matrix, defined for a three-linked system as:

$$[D(t)] = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & \overleftrightarrow{\omega^{(1)}(t)} & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(2)}(t)} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(3)}(t)} \end{bmatrix} \tag{70}$$

Next the variation of the kinetic energy $\delta K^{(\alpha)}(t)$, the potential energy $\delta U^{(\alpha)}(t)$, and the the virtual work $\delta W^{(\alpha)}(t)$ is derived.

The variation of the kinetic energy $\delta K^{(\alpha)}(t)$, is derived by taking the variation of Eqn. (58):

$$\delta K(t) = \left\{\delta \dot{X}(t)\right\}^T \{H(t)\} = \left\{\delta \dot{X}(t)\right\}^T [M]\left\{\dot{X}(t)\right\} \tag{71}$$

The variation of work is $W^{(\alpha)}(t)$. This is defined as the work done by resultant external force $F_C^{(\alpha)^I}(t)$ and external torque $M_C^{(\alpha)}(t)$, defined with respect to the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ as in Eqn. (64):

$$\delta W(t) = \{\delta\tilde{X}(t)\}^T\{F(t)\} \tag{72}$$

Where the external force $\{F(t)\}$ is a $6n\times 1$matrix, defined for a three-linked system as:

$$\{F(t)\} = \begin{pmatrix} F_C^{(1)^I}(t) \\ M_C^{(1)^I}(t) \\ F_C^{(2)^I}(t) \\ M_C^{(2)^I}(t) \\ F_C^{(3)^I}(t) \\ M_C^{(3)^I}(t) \end{pmatrix} \tag{73}$$

By substitution Eqn. (65) into (72), the virtual work $\delta W(t)$ expressed with respect to essential generalized displacement becomes:

$$\delta W(t) = \{\delta q(t)\}^T\{F^*(t)\} \tag{74}$$

Where the essential force $\{F^*(t)\}$ is defined as a $n^* \times 1$ matrix, where n^* defines the number of essential generalized coordinates:

$$\{F^*(t)\} = [B(t)]^T\{F(t)\} \tag{75}$$

The virtual work done by conservative external forces is taken to account by variation of potential energy $\delta U^{(\alpha)}(t)$:

$$\delta W^{(\alpha)}(t) = -\delta U^{(\alpha)}(t) \tag{76}$$

Therefore, the virtual work done by conservative forces is included in the virtual work $\delta W^{(\alpha)}(t)$ as in Eqn. (74).

Substitution of Eqn. (71) and (74) into Eqn. (51), with the use of Eqn. (76), Hamilton's principle turns to:

$$\int_{t_0}^{t_1}\left\{\{\delta\dot{X}(t)\}^T[M]\{\dot{X}(t)\}+\{\delta q(t)\}^T\{F^*(t)\}\right\}dt=0 \tag{77}$$

Where $\{\delta\dot{X}(t)\}$ from Eqn. (69) is expressed for a three-linked system as:

$$\{\delta\dot{X}(t)\}=\begin{pmatrix}\delta\dot{x}_C^{(1)}(t)\\ \delta\omega^{(1)}(t)\\ \delta\dot{x}_C^{(2)}(t)\\ \delta\omega^{(2)}(t)\\ \delta\dot{x}_C^{(3)}(t)\\ \delta\omega^{(3)}(t)\end{pmatrix}=\frac{d}{dt}\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta x_C^{(2)}(t)\\ \delta\pi^{(2)}(t)\\ \delta x_C^{(3)}(t)\\ \delta\pi^{(3)}(t)\end{pmatrix}$$

$$+\begin{bmatrix}0_3&0_3&0_3&0_3&0_3&0_3\\ 0_3&\overleftrightarrow{\omega^{(1)}(t)}&0_3&0_3&0_3&0_3\\ 0_3&0_3&0_3&0_3&0_3&0_3\\ 0_3&0_3&0_3&\overleftrightarrow{\omega^{(2)}(t)}&0_3&0_3\\ 0_3&0_3&0_3&0_3&0_3&0_3\\ 0_3&0_3&0_3&0_3&0_3&\overleftrightarrow{\omega^{(3)}(t)}\end{bmatrix}\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta x_C^{(2)}(t)\\ \delta\pi^{(2)}(t)\\ \delta x_C^{(3)}(t)\\ \delta\pi^{(3)}(t)\end{pmatrix} \tag{78}$$

In Eqn. (77) the variation of omega $\delta\omega^{(\alpha)}$(t) is eliminated by inserting Eqn. (69) into Eqn. (77):

$$\int_{t_0}^{t_1}\{\delta\tilde{X}\}^T\frac{d}{dt}\left([M]\{\dot{X}(t)\}\right)+\{\delta\tilde{X}(t)\}^T[D(t)][M]\{\dot{X}(t)\}-\{\delta q(t)\}^T\{F^*(t)\}dt=0 \tag{79}$$

Performing the derivation in Eqn. (79), and substitute $\{\delta\tilde{X}(t)\}$ and $\{\dot{X}(t)\}$ with the definitions in Eqn. (65) and (39) to obtain the equations of motion:

$$\begin{gathered}\int_{t_0}^{t_1}\{\delta q(t)\}^T([B(t)]^T[M][B(t)]\{\ddot{q}(t)\}+[B(t)]^T[M][\dot{B}(t)]\{\dot{q}(t)\}+\\ [B(t)]^T[D(t)][M][B(t)]\{\dot{q}(t)\}-\{F^*(\mathrm{t})\})\}dt=0\end{gathered} \tag{80}$$

Hence the equations of motion become:

$$[M^*(t)]\{\ddot{q}(t)\} + [N^*(t)]\{\dot{q}(t)\} - \{F^*(t)\} = 0 \tag{81}$$

Looking at Eqn. (81), the two matrices $[M^*(t)]$ and $[N^*(t)]$ are respectively defined as:

$$[M^*(t)] = [B(t)]^T[M][B(\mathrm{t})] \tag{82}$$

$$[N^*(t)] = [B(t)]^T\left([M]\left[\dot{B}(t)\right] + [D(t)][M][B(t)]\right) \tag{83}$$

Eqn. (81) is the equation of motion with the essential generalized velocities. The mathematical model for deriving the equations of motion of an ROV with a three-linked robotic arm has been set.

2.4 Simplified Model

Equation (81) must be solved numerically. However, this paper will attempt to use the Matlab symbolic manipulator to obtain more direct equations. Thus, the simplifications that follow really amount to an attempt by student authors, to obtain a qualitative solution under certain simplifying cases; and, also, simpler equation systems. The main issue up to this point, and the emphasis to be made, is that the MFM and the restriction on the angular velocity can enable undergraduate students to conduct this work.

To see the impact of the system parameters, a numerical experiment is conducted with a 3D simulation. The mathematical model built above is simplified with several assumptions.

Each term in the generalized velocity $\{\dot{X}(t)\}$ as in Eqn. (46) contributes to the total momentum. For the sake of the experiment translation is neglected, and only rotations are analyzed.

Therefore, the first assumption made is for the translation of the ROV to be prescribed zero in velocity and acceleration, and no translation of the body applies. Hence the ROV only rotates around its center of mass.

$$\delta x_C^{(1)}(t) = 0, \qquad \dot{x}_C^{(1)}(t) = 0, \qquad \ddot{x}_C^{(1)}(t) = 0 \tag{84}$$

The second assumption made, is prescribed angular velocities and accelerations for the first and second arm.

$$\theta^{(2)}(t) = \frac{\pi}{2}\sin\left(t\frac{2*\pi}{10}\right), \qquad \theta^{(3)}(t) = \frac{\pi}{2}\sin\left(t\frac{2*\pi}{3}\right) \tag{85}$$

$$\text{Hence: } \delta\theta^{(2)}(t) = \delta\theta^{(3)}(t) = 0$$

The third assumption made, regards the dimensions to the body of the ROV. It is assumed to have the shape of a square prism, with the width (w) equal to the height (h) and the length (l)

$$J_1^1 = J_2^1 = J_3^1 = \frac{2}{3}m_1 \tag{86}$$

$$\text{where } m_1 = 2000kg$$

The fourth assumption made, regards the first arm. The first arm is assumed to be a slender rod with no mass

$$m^{(2)} = 0\,, \qquad J_1^2 = j_3^2, \qquad J_2^2 = 0 \tag{87}$$

The fifth assumption made, is for the second arm. The second arm is assumed to be slender rod, symmetric about its first and third axis.

$$J_1^3 = J_3^3 = \frac{1}{3}\mathrm{m}_3, \qquad J_2^3 = 0 \tag{88}$$

$$\text{where } m_3 = 100kg$$

With these assumptions made, the system simplifies and with the first and second assumption as in Eqn. (84) and (85). Because of the prescribed rates, the virtual essential generalized displacement $\{\delta q(t)\}$ as in Eqn. (66) turns out:

$$\{\delta q(t)\} = \begin{pmatrix} 0 \\ \delta\pi^{(1)}(t) \\ 0 \\ 0 \end{pmatrix} \tag{89}$$

Eqn. (89) is inserted into Eqn. (81) with the definitions in Eqn. (82) and (83).

Eqn. (81) is then simplified. The first assumption as in Eqn. (84) yields that the translation of the ROV is prescribed to be zero in velocity as with acceleration, hence the top 3 rows of the essential generalized velocity $\{\dot{q}(t)\}$ and acceleration $\{\ddot{q}(t)\}$ is zero.

$$[M^*(t)]_{8\times8}\begin{pmatrix}0\\ \dot{\omega}^{(1)}(t)\\ \ddot{\theta}^{(2)}(t)\\ \ddot{\theta}^{(3)}(t)\end{pmatrix}_{8\times1}+[N^*(t)]_{8\times8}\begin{pmatrix}0\\ \omega^{(1)}(t)\\ \dot{\theta}^{(2)}(t)\\ \dot{\theta}^{(3)}(t)\end{pmatrix}_{8\times1}-\{F^*(t)\}_{8\times1}=0 \qquad (90)$$

Eqn. (90) is simplified in two steps. First due to the three top rows of $\{\dot{q}(t)\}$ and $\{\ddot{q}(t)\}$ are zero, then [M*(t)], and [N*(t)] reduces from 8×8 to 8×5 matrices by removing the first three columns. In this process $\{\dot{q}(t)\}$, and $\{\ddot{q}(t)\}$ reduce from 8×1 to 5×1.

A second reduction comes from $\{\delta q(t)\}$. The three top rows and two bottom rows of $\{\delta q(t)\}$ are zero, hence the three top rows and two bottom rows of the total product in Eqn. (90) is zero. Then [M*(t)] and [N*(t)] reduces from 8×5 to 3×5 matrices by removing the three top rows and two bottom rows.

Any force or moment that impacts a prescribed motion and angel is zero reduces {F*(t)} from 8×1 to a 3×1 vector, by removing the three top rows and two bottom rows.

Eqn. (90) results in the reduced form:

$$\left([M^*(t)]_{Reduced_{3\times5}}\begin{pmatrix}\dot{\omega}^{(1)}(t)\\ \ddot{\theta}^{(2)}(t)\\ \ddot{\theta}^{(3)}(t)\end{pmatrix}_{5\times1}\right)_{3\times1}=\{F^*(t)\}_{3\times1}-\left([N^*(t)]_{Reduced_{3\times5}}\begin{pmatrix}\omega^{(1)}(t)\\ \dot{\theta}^{(2)}(t)\\ \dot{\theta}^{(3)}(t)\end{pmatrix}_{5\times1}\right)_{3\times1} \qquad (91)$$

Eqn. (91) gives three equations:

$$\dot{\omega}_1{}^{(1)}(t) = \frac{-[N^*(t)]_{1,j}\{\dot{q}(t)\}_j - [M^*(t)]_{1,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_1}{[M^*(t)]_{11}}$$

$$j\epsilon\{1,\dots,5\}, \qquad i\epsilon\{2,3,4,5\} \tag{92}$$

$$\dot{\omega}_2{}^{(1)}(t) = \frac{-[N^*(t)]_{2,j}\{\dot{q}(t)\}_j - [M^*(t)]_{2,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_2}{[M^*(t)]_{22}}$$

$$j\epsilon\{1,\dots,5\}, \qquad i\epsilon\{1,3,4,5\} \tag{93}$$

$$\dot{\omega}_3{}^{(1)}(t) = \frac{-[N^*(t)]_{3,j}\{\dot{q}(t)\}_j - [M^*(t)]_{3,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_3}{[M^*(t)]_{33}}$$

$$j\epsilon\{1,\dots,5\}, \qquad i\epsilon\{1,2,4,5\} \tag{94}$$

2.4.1 Reconstruction of the rotation matrix

In $[M^*(t)]$, the rotation matrix for the ROV appears, and thus the rotation matrix for the ROV must be calculated at each time step. If $R(t)$ and $\omega(t)$ is known, then $\omega(t+\Delta t)$ is calculated using 4th order Runge-Kutta Method [8]. $R(t+\Delta t)$ needs to be obtained before moving on to the next time step of Runge-Kutta Method.

The rate of change of the rotation matrix is defined from Eqn. (18):

$$\dot{R}^{(1)}(t) = R^{(1)}(t)\,\overleftrightarrow{\omega^{(1)}(t)} \tag{95}$$

For a constant angular velocity problem, this could be solved analytically by integrating Eqn. (95), but ω is not constant. However, by taking the average of $\omega(t)$ and $\omega(t+\Delta t)$ and assuming a constant omega between time steps, the rotation matrix is obtained with the following formula.

$$R(t+\Delta t) = R(t)\exp(\overleftrightarrow{\Delta t\omega(t+\Delta t/2)}) \tag{96}$$

Where the exponential of the angular velocity is found using the Cayley-Hamilton theorem, which leads to the following equation:

$$\exp(\Delta t \overleftrightarrow{\omega}) = I_3 + \frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}\sin(\Delta t\|\omega(t+\Delta t/2)\|) + \left(\frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}\right)^2 (1-\cos(\Delta t\|\omega(t+\Delta t/2)\|)) \tag{97}$$

If a unit vector $u = \frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}$ is defined in the direction of the angular velocity vector, then Eqn. (97) agrees with the Rodrigues formula. Eqn. (97) is used in the numerical integration of Eqn. (95) to obtain the rotation matrix [9, 10].

2.5 Numerical solution

A symbolic manipulator is used to obtain the equation of motion. Due
to lack of computer processing power and limitations of the program used, some simplifications were necessary. By collapsing $\{\dot{X}(t)\}$ to only hold $\omega^{(\alpha)}(t)$, hence collapsing $[B(t)]$, $[M]$ and $[D(t)]$, information about the linear momentum is lost, but the angular momentum still stands. For future iterations of this problem, linear momentum can be considered. The three coupled first order differential equations Eqn. (92), (93) and (94), are decoupled using a symbolic manipulator through Gauss-Jordan elimination. Then, the three decoupled equations are solved numerically using the Runge-Kutta Method, to obtain $\omega_1^{(1)}(t)$, $\omega_2^{(1)}(t)$ and $\omega_3^{(1)}(t)$.

3. Results

Using Eqn. (92), (93) and (94) and the simplifications above, $\dot{\omega}^{(1,2,3)}(t)$ are solved numerically. Runge-Kutta is used to solve these equations with time steps of 0.02 seconds for 1000 steps. The plots are presented below in Fig. 4, 5 and 6:

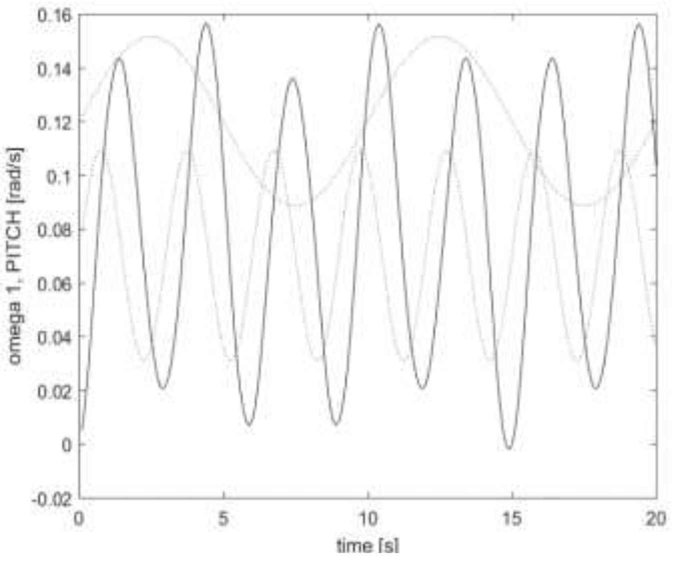


Figure 4 - Angular velocity about first axis (pitch) of ROV and prescribed angular motion of first and second arm (dashed)

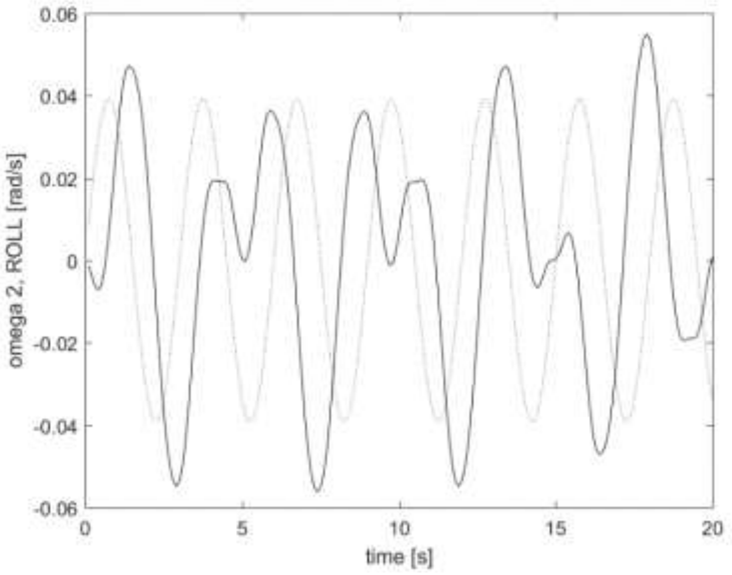


Figure 5 - Angular velocity about second axis (roll) of the ROV and prescribed angular motion of just the second arm (dashed)

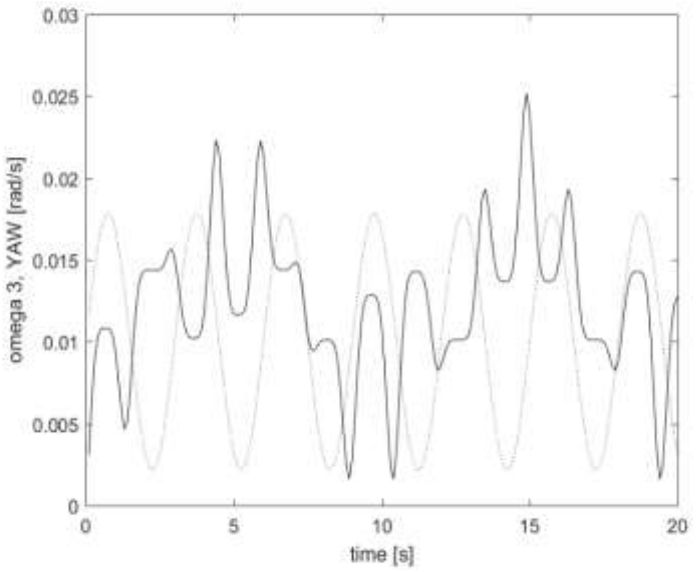


Figure 6 - Plot of the angular velocity about the third axis (yaw axis) of the ROV and prescribed motion of the second arm (dashed)

In these plots, one observes the induced ROV angular velocity due to the input angular velocity of the first and second arms. The response of the ROV closely matches the input angular velocities of the two links. However, the reader is reminded that many simplifications were undertaken, especially, ignoring wave forces, added mass and the input torque generator on the two arms. However, this does confirm the qualitative response.

More informative, perhaps, is observing the induced rocking on the ROV, itself.

Finally, the authors point the reader to a 3D web page [11]:
http://home.hib.no/prosjekter/dynamics/2017/robot/

On this web page, one can alter the input parameters and observe the rocking motion of the craft. Left, middle and right mouse buttons translate as: zoom, pan and rotate.

The scripts used in the symbolic manipulator to extract the symbolic equations are added in the appendix B and C.

4. Conclusion

The results show that it is possible to analyze the induced rotations due to movement of the manipulators. By utilizing the approach used in this paper, combined with coding the solution of Eqn. (81) directly, a broader in-depth analysis can be performed to include both rotations and displacements, as well as reducing the number of simplification necessary to carry out the calculations.

Also, by prescribing the motion of the first arm to be zero, the results reduce to a sine wave about the first axis with a period equal to the second arm. However, by studying such a 2D-simplification, it would not have been possible to construct a set of instructions that could be of any use to control an ROV. With the MFM, the authors have been able to construct and utilize 3D-dynamics. Together with PID-settings it is possible construct a set of instructions that update the motors of an ROV to prevent any rotation induced by motion of the arms.

References

[1] I. Oceaneering. (2016, 03.04.2017). ROV Systems. Available: http://www.oceaneering.com/rovs/rov-systems/

[2] F. Driscoll, R. Lueck, and M. Nahon, "The motion of a deep-sea remotely operated vehicle system Part 1: Motion observations," (in English), Ocean Engineering, Article vol. 27, no. 1, pp. 29-56, JAN 2000.

[3] H. Murakami and T. J. Impelluso, "Chapter 12," 2014. Pending, by Pearson.

[4] J. Denavit and R. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," vol. 22, pp. 215-221

[5] H. Murakami, "A moving frame method for multi-body dynamics using SE(3)," presented at the IMECE2015, Houston, Texas, USA, 2015.

[6] H. Murakami and T. J. Impelluso, "Chapter 13 Multi-Body Dynamics." Pending, by Pearson.

[7] O. Rios and A. Amini, "Model of a gyroscopic roll stabilizer with preliminary experiments," presented at the IMECE2016, Phoenix, Arizona, USA, 2016.

[8] C. A. Barker. (2017, 12.02.2017). Numerical Methods for Solving Differential Equations The Runge-Kutta Method, Theoretical Introduction. Available: http://calculuslab.deltacollege.edu/ODE/7-C-3/7-C-3-h.html

[9] T. J. Impelluso, "Reconstruction," February 2017.

[10] H. Murakami, O. Rios, and T. J. Impelluso, "A theoretical and numerical study of the dzhanibekov and tennis racket
phenomena," presented at the IMECE2015, Houston, Texas, USA, 2015.

[11] A. Evjenth, I.V.K. Schøn, O.A. Moe (2017). *ROV MOTION.*
http://home.hib.no/prosjekter/dynamics/2017/robot/

Appendix A

The following equations gives angular velocity about the first, second and third axis, respectively. Admittedly, this was an overwrought attempt—to obtain expressions for the three angular velocities in this form. In future work, it would be best to simply conduct a numerical integration directly with Eqn. (81-83) or just with (80) alone.

$$\dot{\omega}^{(1)}(t) = (2 \ast J31 \ast w3 \ast sin(th1t(t)) \ast sin(th2t(t))\text{\textasciicircum}2 \ast dth2t(t))/(J11 + J31) - (J31 \ast w1 \ast sin(2 \ast th2t(t)) \ast dth2t(t))/(J11 + J31) - (J31 \ast w2 \ast w3 \ast (cos(th2t(t))\text{\textasciicircum}2 - 1))/(J11 + J31) - (J31 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)) \ast dth1t(t)\text{\textasciicircum}2)/(J11 + J31) - (J31 \ast w2 \ast w3 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t))\text{\textasciicircum}2)/(J11 + J31) - (J31 \ast cos(th1t(t)) \ast ddth2t(t))/(J11 + J31) - (2 \ast J31 \ast w2 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t))\text{\textasciicircum}2 \ast dth1t(t))/(J11 + J31) + (2 \ast J31 \ast sin(th1t(t)) \ast sin(th2t(t))\text{\textasciicircum}2 \ast dth1t(t) \ast dth2t(t))/(J11 + J31) + (J31 \ast w2\text{\textasciicircum}2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)))/(J11 + J31) - (J31 \ast w3\text{\textasciicircum}2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)))/(J11 + J31) - (J31 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)) \ast ddth1t(t))/(J11 + J31) + (J31 \ast w1 \ast w3 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast sin(th1t(t)))/(J11 + J31) - (2 \ast J31 \ast w3 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)) \ast dth1t(t))/(J11 + J31) + (2 \ast J31 \ast w1 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast sin(th1t(t)) \ast dth1t(t))/(J11 + J31) + (2 \ast J31 \ast w1 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t)) \ast sin(th2t(t)) \ast dth2t(t))/(J11 + J31) - (J31 \ast w1 \ast w2 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)))/(J11 + J31) + (2 \ast J31 \ast w2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)) \ast dth2t(t))/(J11 + J31)$$

$$\dot{\omega}^{(2)}(t) = (2 \ast J31 \ast w1 \ast w3 \ast cos(th2t(t))\text{\textasciicircum}2)/(J11 + J31) - (J31 \ast w1 \ast w3)/(J11 + J31) - (J31 \ast sin(th1t(t)) \ast ddth2t(t))/(J11 + J31) - (2 \ast J31 \ast w3 \ast cos(th1t(t)) \ast dth2t(t))/(J11 + J31) + (2 \ast J31 \ast w1 \ast cos(th2t(t))\text{\textasciicircum}2 \ast dth1t(t))/(J11 + J31) - (2 \ast J31 \ast cos(th1t(t)) \ast dth1t(t) \ast dth2t(t))/(J11 + J31) - (J31 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)) \ast dth1t(t)\text{\textasciicircum}2)/(J11 + J31) - (J31 \ast w1 \ast w3 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t))\text{\textasciicircum}2)/(J11 + J31) + (2 \ast J31 \ast w3 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast dth2t(t))/(J11 + J31) - (2 \ast J31 \ast w1 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t))\text{\textasciicircum}2 \ast dth1t(t))/(J11 + J31) + (2 \ast J31 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast dth1t(t) \ast dth2t(t))/(J11 + J31) + (J31 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)) \ast ddth1t(t))/(J11 + J31) + (J31 \ast w1\text{\textasciicircum}2 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)))/(J11 + J31) - (J31 \ast w3\text{\textasciicircum}2 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)))/(J11 + J31) - (J31 \ast w2 \ast w3 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast sin(th1t(t)))/(J11 + J31) - (2 \ast J31 \ast w3 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)) \ast dth1t(t))/(J11 + J31) - (2 \ast J31 \ast w2 \ast cos(th1t(t)) \ast cos(th2t(t))\text{\textasciicircum}2 \ast sin(th1t(t)) \ast dth1t(t))/(J11 + J31) - (2 \ast J31 \ast w2 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t)) \ast sin(th2t(t)) \ast dth2t(t))/(J11 + J31) - (J31 \ast w1 \ast w2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th2t(t)))/(J11 + J31) + (2 \ast J31 \ast w1 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th1t(t)) \ast sin(th2t(t)) \ast dth2t(t))/(J11 + J31)$$

$$\dot{\omega}^{(3)}(t) = (J31 \ast cos(\theta^{(2)}(t)) \ast (2 \ast w3 \ast sin(th2t(t)) \ast dth2t(t) - cos(th2t(t)) \ast ddth1t(t) + 2 \ast sin(th2t(t)) \ast dth1t(t) \ast dth2t(t) - w1 \ast w2 \ast cos(th2t(t)) + w2 \ast w3 \ast sin(th1t(t)) \ast sin(th2t(t)) - w1\text{\textasciicircum}2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th1t(t)) + w2\text{\textasciicircum}2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast sin(th1t(t)) + 2 \ast w1 \ast w2 \ast cos(th1t(t))\text{\textasciicircum}2 \ast cos(th2t(t)) + 2 \ast w2 \ast cos(th1t(t)) \ast cos(th2t(t)) \ast dth2t(t) - 2 \ast w1 \ast cos(th2t(t)) \ast sin(th1t(t)) \ast dth2t(t) + w1 \ast w3 \ast cos(th1t(t)) \ast sin(th2t(t))))/(J11 + J31)$$

Appendix B

Matlab script

```
function reduced
% Author: Andreas Evjenth
syms t
syms th1t(t) th2t(t)
syms dth1t(t) dth2t(t)
syms ddth1t(t) ddth2t(t)
syms th1 th2
syms dth1 dth2
syms ddth1 ddth2

syms dx1 dx2 dx3 w1 w2 w3 dw1 dw2 dw3
syms ddx1 ddx2 ddx3

syms L
syms m1 m2 m3
syms J11 J12 J13 J21 J22 J23 J31 J32 J33

e1 = sym([1;0;0]);
e3 = sym([0;0;1]);

R2_1 = sym(eye(3,3));
R2_1(1,1) = cos(th1t);
R2_1(2,2) = cos(th1t);
R2_1(1,2) = -sin(th1t);
R2_1(2,1) = sin(th1t);

R3_2 = sym(eye(3,3));
R3_2(2,2) = cos(th2t);
R3_2(3,3) = cos(th2t);
R3_2(2,3) = -sin(th2t);
R3_2(3,2) = sin(th2t);

I3 = sym(eye(3,3));

B_matrix = sym(zeros(9,5));
B_matrix(1:3,1:3) = I3;
B_matrix(4:6,1:3) = R2_1.';
B_matrix(4:6,4) = e3;
B_matrix(7:9,1:3) = R3_2.' * R2_1.';
B_matrix(7:9,4) = R3_2.' * e3;
B_matrix(7:9,5) = e1;

Bt_matrix = B_matrix.';
dB_matrix = diff(B_matrix,t);
```

```
J13 = J11;
J12 = J11;

J21 = 0;
J22 = 0;
J23 = J21;

J32 = 0;
J33 = J31;

M_matrix = sym(diag([J11,J12,J13,J21,J22,J23,J31,J32,J33]));

%Define the omegas
w = [w1;w2;w3];
O2 = sym(zeros(3,1));
O2(1:3,1) = (R2_1.' * w) + (dth1t * e3);
O3 = sym(zeros(3,1));
O3(1:3,1) = R3_2.' * R2_1.' * w + R3_2.' * dth1t * e3 + dth2t * e1;
Omega1 = symSkew(w);
Omega2 = symSkew(O2);
Omega3 = symSkew(O3);

D_matrix = sym(zeros(9,9));
D_matrix(1:3,1:3) = Omega1(1:3,1:3);
D_matrix(4:6,4:6) = Omega2(1:3,1:3);
D_matrix(7:9,7:9) = Omega3(1:3,1:3);

Mstar = simplify(Bt_matrix * M_matrix * B_matrix);
Nstar = simplify(Bt_matrix * (M_matrix * dB_matrix + D_matrix * M_matrix * B_matrix));


dqr = sym(zeros(5,1));
dqr(1,1) = dw1;
dqr(2,1) = dw2;
dqr(3,1) = dw3;
dqr(4,1) = ddth1t;
dqr(5,1) = ddth2t;

qr = sym(zeros(5,1));
qr(1,1) = w1;
qr(2,1) = w2;
qr(3,1) = w3;
qr(4,1) = dth1t;
qr(5,1) = dth2t;

ess = sym(zeros(3,1));
for a= 1:3
   for j=1:5
      ess(a) = ess(a) - Nstar(a,j)*qr(j);
      if(j~= a)
```

```
         ess(a) = ess(a) - Mstar(a,j)*dqr(j);
      end
   end
   ess(a) = ess(a) / Mstar(a,a);
end


%SETTING UP FOR GAUSS ELIMINATION

syms ga gb gc gd gf gg gh gi

ga = sym(ess(1));
gb = sym(ess(1));
gc = sym(ess(1));
gd = sym(ess(2));
ge = sym(ess(2));
gf = sym(ess(2));
gg = sym(ess(3));
gh = sym(ess(3));
gi = sym(ess(3));

ga = subs(ga,dw2,0);
ga = subs(ga,dw3,0);

gb = subs(gb,dw2,1);
gb = subs(gb,dw3,0);
gb = gb - ga;

gc = subs(gc,dw2,0);
gc = subs(gc,dw3,1);
gc = gc - ga;

gd = subs(gd,dw1,0);
gd = subs(gd,dw3,0);

ge = subs(ge,dw1,1);
ge = subs(ge,dw3,0);
ge = ge - gd;

gf = subs(gf,dw1,0);
gf = subs(gf,dw3,1);
gf = gf - gd;

gg = subs(gg,dw1,0);
gg = subs(gg,dw2,0);

gh = subs(gh,dw1,1);
gh = subs(gh,dw2,0);
gh = gh - gg;

gi = subs(gi,dw1,0);
gi = subs(gi,dw2,1);
gi = gi - gg;

gauss_matrix = [-1,gb,gc,-ga;ge,-1,gf,-gd;gh,gi,-1,-gg];
gauss_matrix = rref(gauss_matrix);
q_unc = sym(zeros(3,1));
q_unc(1:3) = gauss_matrix(1:3,4);

q_unc = subs(q_unc,diff(th1t(t), t),dth1t);
q_unc = subs(q_unc,diff(th2t(t), t),dth2t);

time = 0;
step = 1;
step_max = 2000;
while step < step_max
   step = step * 2;
   parfor a=1:3
      q_unc(a) = simplify(q_unc(a),'Steps',step);
   end
   clc;
   disp(step);
end

disp(q_unc);
end

function [Mat] = symSkew(vec)
   Mat = sym(zeros(3));
   Mat(1,2) = -vec(3);
   Mat(1,3) =  vec(2);
   Mat(2,3) = -vec(1);
   Mat(2,1) = -Mat(1,2);
   Mat(3,1) = -Mat(1,3);
   Mat(3,2) = -Mat(2,3);
end
```

Appendix C

Main.js, Starter.js, Dynamics.js

```
$(document).ready(function () {
  //Author: Andreas Evjenth
  var camera, renderer, controls;
  var scene = new THREE.Scene();
  var frames = [];
  var dq = [0, 0, 0];
  var q = [0, 0, 0];
  var guiParams = gui();
  var t = 0;
  const dt = 0.016;


  function init() {
    setRenderer();
    frames = loadObjects(scene).slice();
    setCamera();
    setLights();
    seafloor();
    setControls();
    window.addEventListener('resize', onWindowResize, false);
    renderer.render(scene, camera);
    animate();
  }

  function seafloor() {
    var mesh, texture;
    var worldWidth = 1024, worldDepth = 1024,
    worldHalfWidth = worldWidth / 2, worldHalfDepth = worldDepth / 2;
    var data = generateHeight(worldWidth, worldDepth);
    var geometry = new THREE.PlaneBufferGeometry(3000, 3000, worldWidth - 1, worldDepth - 1);
    geometry.rotateX(-Math.PI / 2);
    var vertices = geometry.attributes.position.array;
    for (var i = 0, j = 0, l = vertices.length; i < l; i++, j += 3) {
      vertices[j + 1] = data[i] * 5;
    }
    texture = new THREE.CanvasTexture(generateTexture(data, worldWidth, worldDepth));
    texture.wrapS = THREE.ClampToEdgeWrapping;
    texture.wrapT = THREE.ClampToEdgeWrapping;
    mesh = new THREE.Mesh(geometry, new THREE.MeshBasicMaterial({ map: texture }));
    mesh.rotation.x = Math.PI / 2;
    mesh.position.z = -1500;
    mesh.scale.x = 20;
    mesh.scale.z = 20;
    scene.add(mesh);
  }

  function animate() {
    requestAnimationFrame(animate);
    var ROV_data = updateROV(dq, dt, t, q);
    for (var i = 0 ; i < 3 ; dq[i] = ROV_data.ddq[i] * 0.9, i++);
    for (var i = 0 ; i < 3 ; q[i] = ROV_data.angles[i] * 0.9, i++);
    //dq *= 0.5;
    rotate(q, getTheta(t));
    controls.update();
    renderer.render(scene, camera);
    t += dt;
}
function rotate(q, theta) {
    frames[0].rotation.x = q[0];
    frames[0].rotation.y = q[1];
    frames[0].rotation.z = q[2];

    frames[2].rotation.z = theta.th1t;
    frames[3].rotation.x = theta.th2t;
}

function setCamera() {
    camera = new THREE.PerspectiveCamera(60, window.innerWidth / window.innerHeight, 100, 50000);
    camera.position.set(5000, 5000, 0);
    camera.lookAt(0, 0, 0);
    camera.up.set(0, 0, 1);
}

function setRenderer() {
    renderer = new THREE.WebGLRenderer({ antialias: true });
    renderer.setSize(window.innerWidth, window.innerHeight);
    renderer.setClearColor(0x000055, 1.0);
    var container = document.getElementById("myCanvas");
    container.appendChild(renderer.domElement);
}

function setLights() {
    var light = new THREE.HemisphereLight(0xffffbb, 0x080820, 1);
    light.castShadow = true;
    scene.add(light);

    var amblight = new THREE.AmbientLight(0x555555);
    scene.add(amblight);

}
function setControls() {
    controls = new THREE.OrbitControls(camera, renderer.domElement);
    controls.rotateSpeed = 0.5;
    controls.zoomSpeed = 2.0;
    controls.panSpeed = 5.;
    controls.minDistance = 5.0;
```

```
        controls.maxDistance = 100000;
    }

    function onWindowResize() {
        camera.aspect = window.innerWidth / window.innerHeight;
        camera.updateProjectionMatrix();
        renderer.setSize(window.innerWidth, window.innerHeight);
    }
    init();
});
```

```
function loadObjects(scene) {
  var loader = new THREE.OBJMTLLoader();
  var objects = loadObject(loader);
  var origo = new THREE.Vector3(0, 0, 0);

  var linelength = 4000;
  var material = new THREE.LineBasicMaterial({
    color: 0xff0000
  });
  var linegeo = new THREE.Geometry();
  linegeo.vertices.push(
    new THREE.Vector3(linelength, 0, 0),
    origo
  );
  var line = new THREE.Line(linegeo, material);

  scene.add(line);

  var material1 = new THREE.LineBasicMaterial({
    color: 0x00ff00
  });
  var linegeo1 = new THREE.Geometry();
  linegeo1.vertices.push(
    new THREE.Vector3(0, linelength, 0),
    origo
  );
  var line1 = new THREE.Line(linegeo1, material1);

  scene.add(line1);

  var material2 = new THREE.LineBasicMaterial({
    color: 0x0000ff
  });
  var linegeo2 = new THREE.Geometry();
  linegeo2.vertices.push(
    new THREE.Vector3(0, 0, linelength),
    origo
  );
  var line2 = new THREE.Line(linegeo2, material2);

  scene.add(line2);

  var frames = [];
  frames[0] = new THREE.Object3D(0, 0, 0);
  frames[1] = new THREE.Object3D(0, 0, 0);
  frames[1].position.y = 1350;
  frames[1].position.z = -170;
  frames[2] = new THREE.Object3D(0, 0, 0);
```

```
    frames[2].position.y = 50;
    frames[3] = new THREE.Object3D(0, 0, 0);
    frames[3].position.y = 1300;

    scene.add(frames[0]);
    frames[0].add(objects.o1);
    frames[0].add(frames[1]);
    frames[1].add(objects.o2);
    frames[1].add(frames[2]);
    frames[2].add(objects.o3);
    frames[2].add(frames[3]);
    frames[3].add(objects.o4);
    return frames;
}

function loadObject(loader, num) {
    var o1 = new THREE.Object3D();
    loader.load('models/realistic_rov.obj', 'models/realistic_rov.mtl', function (object) { o1.add(object); });
    o1.position.y = -1300;
    o1.position.z = 600;;
    o1.position.x = 800;
    o1.rotation.y = Math.PI / 2;
    o1.rotation.x = Math.PI / 2;

    var o2 = new THREE.Object3D();
    loader.load('models/base.obj', 'models/base.mtl', function (object) { o2.add(object); });
    o2.rotation.z = Math.PI;
    var o3 = new THREE.Object3D();
    loader.load('models/arm1.obj', 'models/arm1.mtl', function (object) { o3.add(object); });

    o3.rotation.z = Math.PI / 2;
    o3.rotation.y = Math.PI / 2;
    o3.scale.y = 3;
    o3.scale.z = 2;

    var o4 = new THREE.Object3D();
    loader.load('models/arm1.obj', 'models/arm1.mtl', function (object) { o4.add(object); });
    o4.rotation.z = Math.PI / 2;
    o4.scale.y = 3;
    o4.scale.z = 2;

    return {
        o1: o1,
        o2: o2,
        o3: o3,
        o4: o4
    }
```

```
}

function generateHeight(width, height) {
  var size = width * height, data = new Uint8Array(size),
  perlin = new ImprovedNoise(), quality = 1, z = Math.random() * 5;
  for (var j = 0; j < 4; j++) {
    for (var i = 0; i < size; i++) {
      var x = i % width, y = ~~(i / width);
      data[i] += Math.abs(perlin.noise(x / quality, y / quality, z) * quality * 1.75);
    }
    quality *= 1.2;
  }
  return data;
}

function generateTexture(data, width, height) {
  var canvas, canvasScaled, context, image, imageData,
  level, diff, vector3, sun, shade;
  vector3 = new THREE.Vector3(0, 0, 0);
  sun = new THREE.Vector3(1, 1, 1);
  sun.normalize();
  canvas = document.createElement('canvas');
  canvas.width = width;
  canvas.height = height;
  context = canvas.getContext('2d');
  context.fillStyle = '#000';
  context.fillRect(0, 0, width, height);
  image = context.getImageData(0, 0, canvas.width, canvas.height);
  imageData = image.data;
  for (var i = 0, j = 0, l = imageData.length; i < l; i += 4, j++) {
    vector3.x = data[j - 2] - data[j + 2];
    vector3.y = 2;
    vector3.z = data[j - width * 2] - data[j + width * 2];
    vector3.normalize();
    shade = vector3.dot(sun);
    imageData[i] = (shade * 128) * (0.5 + data[j] * 0.007);
    imageData[i + 1] = (32 + shade * 96) * (0.5 + data[j] * 0.007);
    imageData[i + 2] = (shade * 96) * (0.5 + data[j] * 0.007);
  }
  context.putImageData(image, 0, 0);
  // Scaled 4x
  canvasScaled = document.createElement('canvas');
  canvasScaled.width = width * 4;
  canvasScaled.height = height * 4;
  context = canvasScaled.getContext('2d');
  context.scale(4, 4);
  context.drawImage(canvas, 0, 0);
  image = context.getImageData(0, 0, canvasScaled.width, canvasScaled.height);
  imageData = image.data;
```

```
    for (var i = 0, l = imageData.length; i < l; i += 4) {

        var v = ~~(Math.random() * 5);
        imageData[i] += v;
        imageData[i + 1] += v;
        imageData[i + 2] += v;
    }

    context.putImageData(image, 0, 0);
    return canvasScaled;
}



function gui() {
    var gui = new dat.GUI({
        height: 5 * 32 - 1
    });
    guiParams = {
        PER1: 10,
        PER2: 3,
        AMP1: 0.7,
        AMP2: 1.0

    };
    gui.add(guiParams, 'PER1', 1, 20).name('Period arm 1');
    gui.add(guiParams, 'PER2', 1, 20).name('Period arm 2');
    gui.add(guiParams, 'AMP1', 0.1, Math.PI / 4).name('Amplitude arm 1');
    gui.add(guiParams, 'AMP2', 0.1, Math.PI / 2).name('Amplitude arm 2');
    return guiParams;
}

function period(seconds) {
    return (Math.PI * 2) / seconds;
}
```

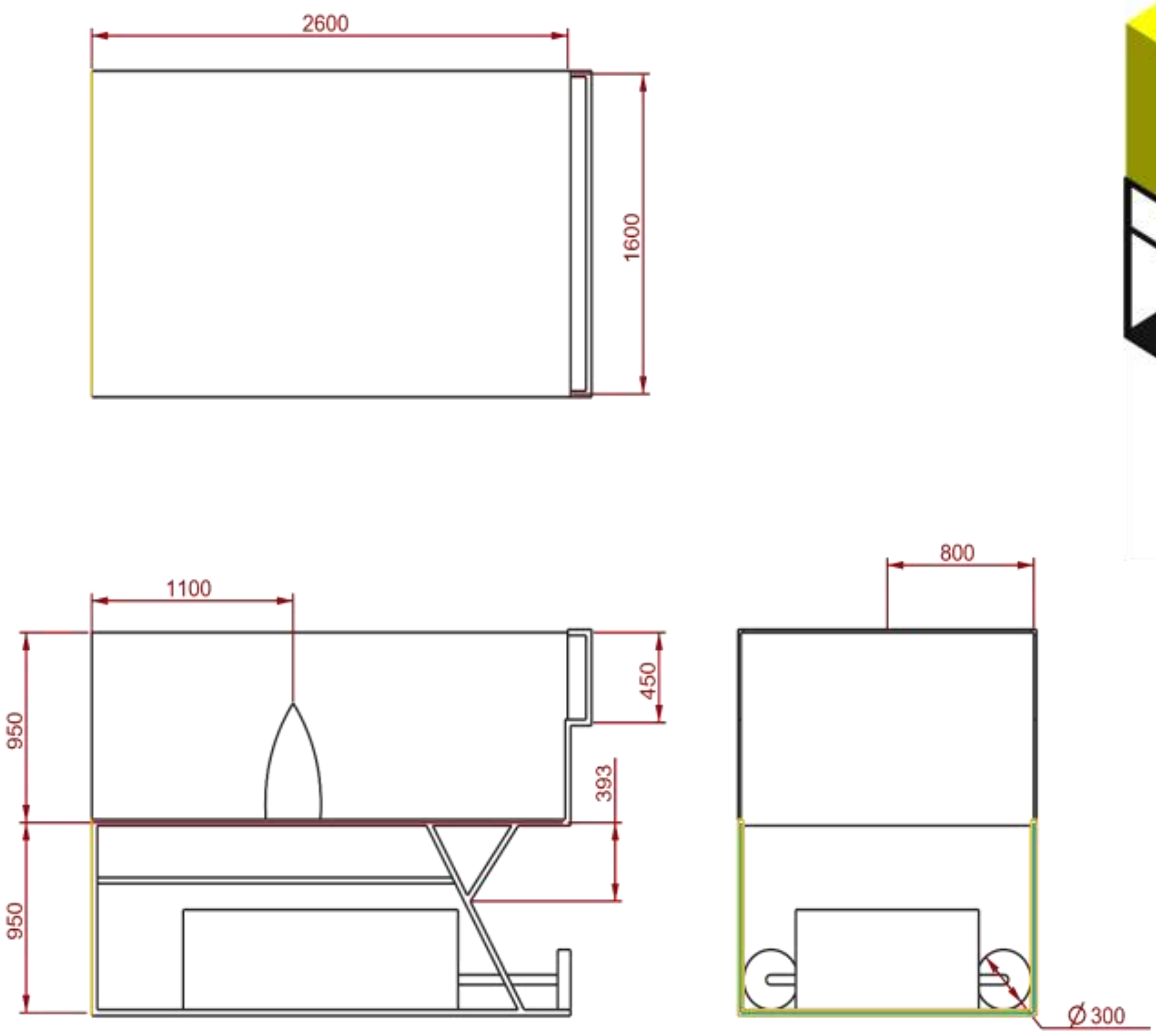
```
var AMP1 = 0;
var AMP2 = 0;
var PER1 = period(10);
var PER2 = period(3);
var x = 0;
var y = 0;
const phase = Math.PI/2;

const ROV_length = 1.0;
const V_length = .5;
const arm_length = .5;
const m1 = 2000;
const m3 = 300;
const J11 = ROV_length * ROV_length * m1 / 6;
const J31 = m3 * arm_length * arm_length / 3;
const geo =  0.5;

var rotationMatrix = eyeMat(3, 3);
var identityMat = eyeMat(3, 3);
function getTheta(t) {

   return {
      th1t: AMP1 * Math.sin(t * PER1 + phase + x),
      dth1t: AMP1 * PER1 * Math.cos(t * PER1 + phase),
      ddth1t: -AMP1 * PER1 * PER1 * Math.sin(t * PER1 +phase),
      th2t: AMP2 * Math.sin(t * PER2 + phase+y),
      dth2t: AMP2 * PER2 * Math.cos(t * PER2 + phase),
      ddth2t: -AMP2 * PER2 * PER2 * Math.sin(t * PER2 +phase)
   }
}

function matlabFunction(om_flag, prev_dq, t,q) {
```

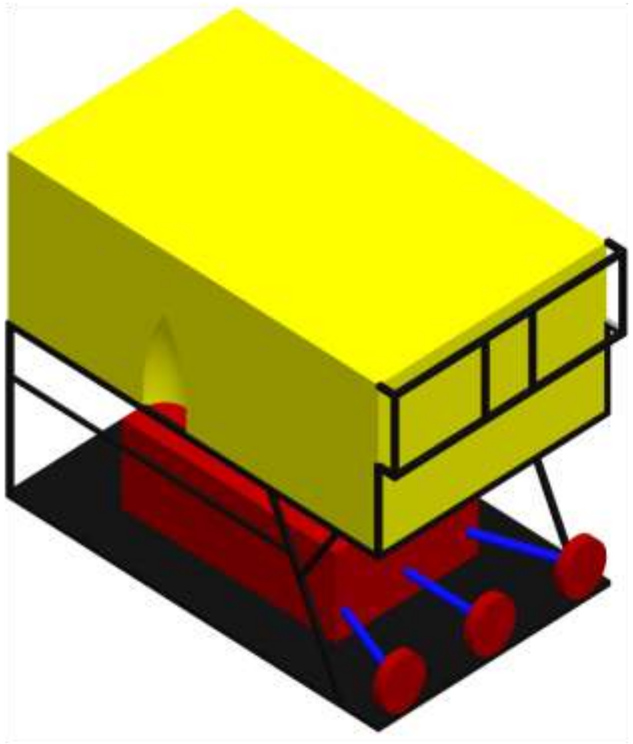
```
var w1 = prev_dq[0];
   var w2 = prev_dq[1];
   var w3 = prev_dq[2];
   var theta = getTheta(t);


   var th1 = theta.th1t;
   var dth1 = theta.dth1t;
   var ddth1 = theta.ddth1t;
   var th2 = theta.th2t;
   var dth2 = theta.dth2t;
   var ddth2 = theta.ddth2t;
   var M = getMoments(q,m1,prev_dq);
   const M1x = M.M1x;
   const M1y = M.M1y;

   switch (om_flag) {
      case 0:
         return (J11*J31*ddth2*Math.cos(th1) - J31*M1x*Math.cos(th2)**2 - J11*M1x +
J31*M1x*Math.cos(th1)**2*Math.cos(th2)**2 - J11*J31*w2*w3 + J11*J31*dth2*w1*Math.sin(2*th2) +
J11*J31*w2*w3*Math.cos(th2)**2 - 2*J11*J31*dth1*dth2*Math.sin(th1) -
2*J11*J31*dth2*w3*Math.sin(th1) + J31*M1y*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) +
2*J11*J31*dth1*w2*Math.cos(th1)**2*Math.cos(th2)**2 +
J11*J31*dth1**2*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
J11*J31*w2*w3*Math.cos(th1)**2*Math.cos(th2)**2 -
J11*J31*w2**2*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
J11*J31*w3**2*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
2*J11*J31*dth1*dth2*Math.cos(th2)**2*Math.sin(th1) +
2*J11*J31*dth2*w3*Math.cos(th2)**2*Math.sin(th1) +
J11*J31*ddth1*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) +
2*J11*J31*dth1*w3*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
J11*J31*w1*w2*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) -
2*J11*J31*dth1*w1*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) -
2*J11*J31*dth2*w1*Math.cos(th1)**2*Math.cos(th2)*Math.sin(th2) -
J11*J31*w1*w3*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) -
2*J11*J31*dth2*w2*Math.cos(th1)*Math.cos(th2)*Math.sin(th1)*Math.sin(th2))/(J11*(J11 + J31));
      case 1:
         return  (J11*J31*ddth2*Math.sin(th1) - J11*M1y - J31*M1y*Math.cos(th1)**2*Math.cos(th2)**2
- 2*J11*J31*dth1*w1*Math.cos(th2)**2 + 2*J11*J31*dth1*dth2*Math.cos(th1) +
2*J11*J31*dth2*w3*Math.cos(th1) - J11*J31*w1*w3*(2*Math.cos(th2)**2 - 1) +
J31*M1x*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) +
2*J11*J31*dth1*w1*Math.cos(th1)**2*Math.cos(th2)**2 +
J11*J31*w1*w3*Math.cos(th1)**2*Math.cos(th2)**2 +
J11*J31*dth1**2*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) -
J11*J31*w1**2*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) +
J11*J31*w3**2*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) -
2*J11*J31*dth1*dth2*Math.cos(th1)*Math.cos(th2)**2 -
2*J11*J31*dth2*w3*Math.cos(th1)*Math.cos(th2)**2 -
```

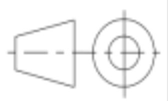
```
J11*J31*ddth1*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
2*J11*J31*dth1*w3*Math.cos(th2)*Math.sin(th1)*Math.sin(th2) +
J11*J31*w1*w2*Math.cos(th1)*Math.cos(th2)*Math.sin(th2) +
2*J11*J31*dth1*w2*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) +
2*J11*J31*dth2*w2*Math.cos(th1)**2*Math.cos(th2)*Math.sin(th2) +
J11*J31*w2*w3*Math.cos(th1)*Math.cos(th2)**2*Math.sin(th1) -
2*J11*J31*dth2*w1*Math.cos(th1)*Math.cos(th2)*Math.sin(th1)*Math.sin(th2))/(J11*(J11 + J31));
        case 2:
            return - (J31*Math.cos(th2)*(2*dth1*dth2*Math.sin(th2) - ddth1*Math.cos(th2) +
2*dth2*w3*Math.sin(th2) + w1*w3*Math.cos(th1)*Math.sin(th2) +
w1*w2*Math.cos(th2)*(2*Math.cos(th1)**2 - 1) + w2*w3*Math.sin(th1)*Math.sin(th2) -
w1**2*Math.cos(th1)*Math.cos(th2)*Math.sin(th1) +
w2**2*Math.cos(th1)*Math.cos(th2)*Math.sin(th1) + 2*dth2*w2*Math.cos(th1)*Math.cos(th2) -
2*dth2*w1*Math.cos(th2)*Math.sin(th1)))/(J11 + J31) -
(J31*Math.cos(th2)*(M1y*Math.cos(th1)*Math.sin(th2) - M1x*Math.sin(th1)*Math.sin(th2)))/(J11*(J11
+ J31));
    }
    console.log("ERROR – matlab switch statemet. No results returned");
    return 0;
}

function getMoments(q,m1,dq){

    const length = 0.5;
    const gravity = 9.81;
    const mass = m1;
    return{
        M1x :   - m1 * length * gravity * Math.sin(q[0]),
        M1y :    m1 * length * gravity * Math.sin(q[1])
    }
}
function updateROV(dq, dt, t,q) {
    if(period(guiParams.PER1)!=PER1){
        x = Math.asin(getTheta(t).th1t/AMP1)-t*period(guiParams.PER1)-phase;
        while(x > 2 * Math.PI){
            x -= (2 * Math.PI);
        }
        while(x < 0){
            x += 2 * Math.PI;
        }
        PER1 = period(guiParams.PER1);
    }
    if(period(guiParams.PER2)!=PER2){
        y = Math.asin(getTheta(t).th2t/AMP2)-t*period(guiParams.PER2)-phase;
        while(y > 2 * Math.PI){
            y -= (2 * Math.PI);
        }
        while(y < 0){
            y += 2 * Math.PI;
        }
        PER2 = period(guiParams.PER2);
    }
    AMP1 = AMP1 > guiParams.AMP1 + 0.05 ? AMP1 - 0.005 : AMP1 < guiParams.AMP1 - 0.05 ? AMP1 +
0.005 : AMP1;
    AMP2 = AMP2 > guiParams.AMP2 + 0.05 ? AMP2 - 0.005 : AMP2 < guiParams.AMP2 - 0.05 ? AMP2 +
0.005 : AMP2;
    var ddq = RK4(dq, dt, t,q);
    rotationMatrix = getRotMat(rotationMatrix, ddq, dt);
    var angles = rotMatToAngle(rotationMatrix);
    return {
        angles: angles,
        ddq: ddq
    }

}



function RK4(last_dq, dt, t,q) {
    var k1 = [], k2 = [], k3 = [], k4 = [];
    var arr = [];
    var dq = last_dq.slice();
    for (var a = 0 ; a < dq.length ; a++) {
        k1[a] = dt * matlabFunction(a, dq, t,q);
    }
    for (var a = 0 ; a < dq.length ; a++) {
        for (var i = 0 ; i < k1.length ; arr[i] = dq[i] + k1[i] / 2, i++);
        k2[a] = dt * matlabFunction(a, arr, t + dt / 2,q)
    }
    for (var a = 0 ; a < dq.length ; a++) {
        for (var i = 0 ; i < k2.length ; arr[i] = dq[i] + k2[i] / 2, i++);
        k3[a] = dt * matlabFunction(a, arr, t + dt / 2,q);
    }
    for (var a = 0 ; a < dq.length ; a++) {
        for (var i = 0 ; i < k3.length ; arr[i] = dq[i] + k3[i], i++);
        k4[a] = dt * matlabFunction(a, arr, t + dt,q);
    }
    for (var i = 0 ; i < dq.length ; dq[i] = dq[i] + (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6, i++);

    return dq;
}
```

```
function rotMatToAngle(rotMat) {
    var r00 = rotMat[0][0];
    var r01 = rotMat[0][1];
    var r10 = rotMat[1][0];
    var r11 = rotMat[1][1];
    var r02 = rotMat[0][2];
    var r20 = rotMat[2][0];
    var r12 = rotMat[1][2];
    var r21 = rotMat[2][1];
    var r22 = rotMat[2][2];

    var rot1 = Math.atan2(r12, r22);
    var s1 = Math.sin(rot1);
    var c1 = Math.cos(rot1);
    var c2 = Math.sqrt(r00 * r00 + r01 * r01);
    var rot2 = Math.atan2(r02, c2);
    var rot3 = Math.atan2(s1 * r20 - c1 * r10, c1 * r11 - s1 * r21);
    return [rot1, rot2, rot3];
}

function eyeMat(rows, cols) {
    var Mat = zerosMat([rows, cols]);
    for (var i = 0; i < rows; i++) {
        Mat[i][i] = 1;
    }
    return Mat;
}

function getRotMat(rotMat, w, dt) {
    var w1 = w[0];
    var w2 = w[1];
    var w3 = w[2];
    var wNorm = Math.sqrt(w1 * w1 + w2 * w2 + w3 * w3);
    if (w1 * w2 * w3 < 0) wNorm = -wNorm;

    var wSkew = [[0, -w3, w2], [w3, 0, -w1], [-w2, w1, 0]];

    var c2 = Math.sin(dt * wNorm) / wNorm;
    var c3 = (1 - Math.cos(dt * wNorm)) / (wNorm * wNorm);

    var term1 = eyeMat(3, 3);
    var term2 = constMatMult(c2, wSkew);
    var term3 = constMatMult(c3, MatMatMult(wSkew, wSkew));

    var expwt = MatMatAdd(term1, MatMatAdd(term2, term3));
    rotMat = MatMatMult(rotMat, expwt);

    return rotMat;
}

function constMatMult(c, Mat) {
    var rows = Mat.length;
    var cols = Mat[0].length;
    for (var i = 0; i < rows; i++) {
        for (var j = 0; j < cols; j++) {
            Mat[i][j] = c * Mat[i][j];
        }
    }
    return Mat;
}

function MatMatAdd(Mat1, Mat2) {
    var rows = Mat1.length;
    var cols = Mat1[0].length;
    for (var i = 0; i < rows; i++) {
        for (var j = 0; j < cols; j++) {
            Mat1[i][j] += Mat2[i][j];
        }
    }
    return Mat1;
}

function MatMatMult(Mat1, Mat2) {
    var rows1 = Mat1.length;
    var cols1 = Mat1[0].length;
    var rows2 = Mat2.length;
    var cols2 = Mat2[0].length;
    var Mat = zerosMat([rows1, cols2]);

    for (var i = 0; i < rows1; i++) {
        for (var j = 0; j < cols2; j++) {
            for (var k = 0; k < cols1; k++) {
                Mat[i][j] += Mat1[i][k] * Mat2[k][j];
            }
        }
    }
    return Mat;
}

function zerosMat(dimensions) {
    var mat = [];
    for (var i = 0; i < dimensions[0]; ++i) {
        mat.push(dimensions.length == 1 ? 0 : zerosMat(dimensions.slice(1)));
    }
    return mat;
}

function period(seconds) {
    return (Math.PI * 2) / seconds;
}
```

MODEL NAME	TITLE
REALISTIC_ROV	ROV



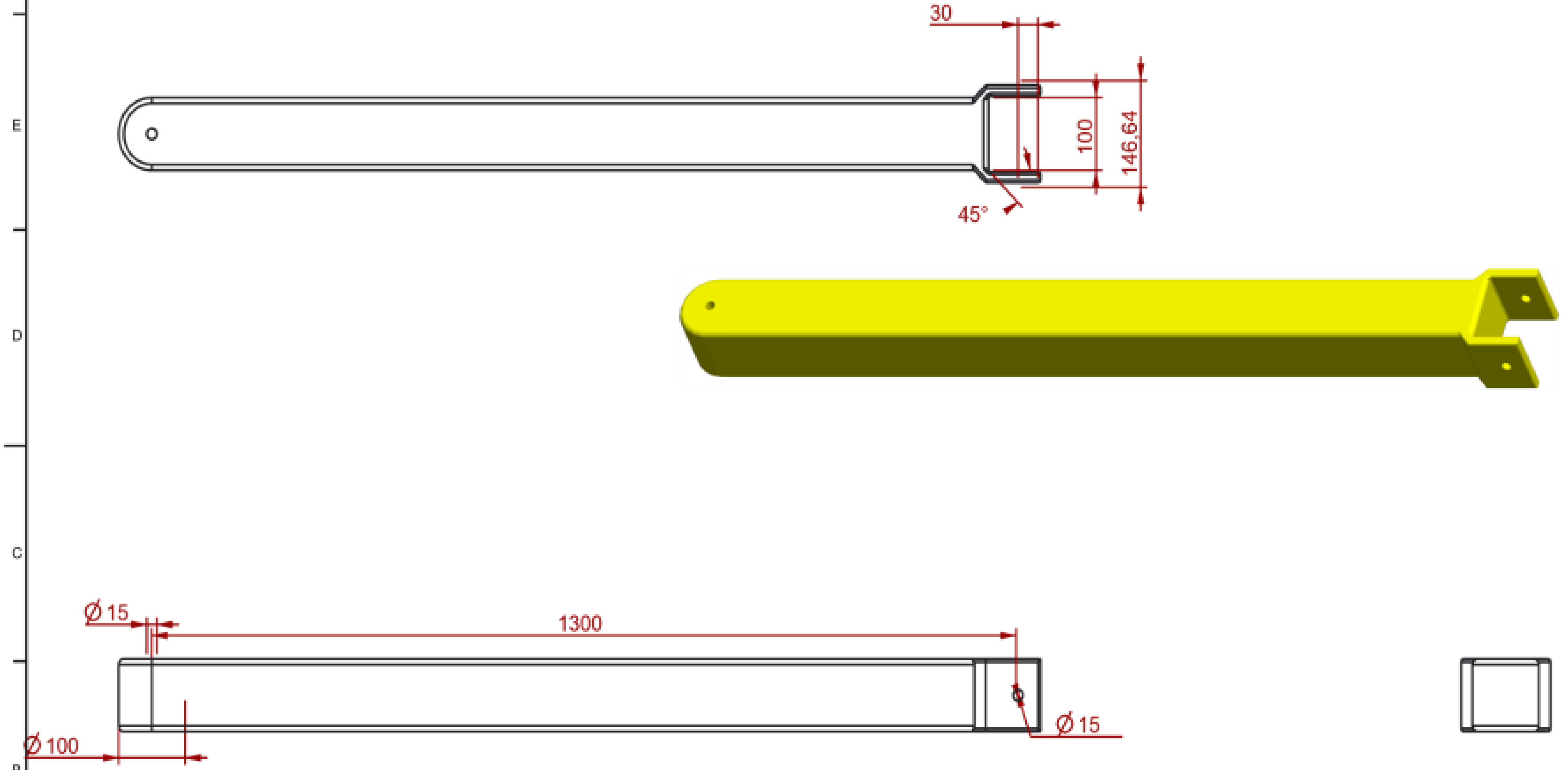




		DRAWN BY A. Evjenth	DATE
		CHECKED BY	DATE
		APPROVED BY	DATE
	[illegible]	SCALE 0,040	SHEET 1 OF 1
UNLESS OTHERWISE SPECIFIED ALL DIMENSIONS ARE IN MILLIMETERS TOLERANCES ARE +/- 0.1MM +/- 1°	TITLE ROV		
DO NOT SCALE	PROJECT	MODEL NAME REALISTIC_ROV	ISSUE

ISO STANDARD,MM,1ST ANGLE

MODEL NAME	TITLE
ARM1	ROV MANIPULATOR



UNLESS OTHERWISE SPECIFIED ALL DIMENSIONS ARE IN MILLIMETERS. TOLERANCES ARE +/- 0.1MM +/- 1°	DRAWN BY A. Evjenth	DATE 23.05.2017
DO NOT SCALE	CHECKED BY	DATE
	APPROVED BY	DATE
	SCALE 0,150	SHEET 1 OF 1
	TITLE ROV MANIPULATOR	
	PROJECT	MODEL NAME ARM1	ISSUE

ISO STANDARD,MM,1ST ANGLE

Appendix E

In parallel with the bachelor project, a draft for an IMECE article has been produced. It is written with the help of Professor Thomas J. Impelluso, and it is pending acceptance as a conference article in the fall of 2017. In short, it is a more compactly written version of the bachelor project. The article is presented on the next pages.

Proceedings of the ASME 2017 International Mechanical Engineering & Exposition
IMECE2017
November 03-09, 2017, Tampa, Florida, USA

IMECE2017-70110

A DYNAMIC MODEL FOR MOTION OF AN ROV DUE TO ON-BOARD ROBOTICS

Andreas Evjenth
Mechanical and Marine Engineering
Western Norway University of Applied Sciences
Bergen, Norway
anstroen@gmail.com

Otto Andreas Moe
Mechanical and Marine Engineering
Western Norway University of Applied Sciences
Bergen, Norway
ottoandreasmoe@gmail.com

Iselin Violet Kjelland Schøn
Mechanical and Marine Engineering
Western Norway University of Applied Sciences
Bergen, Norway
iselinvks@gmail.com

Thomas J. Impelluso
Mechanical and Marine Engineering
Western Norway University of Applied Sciences
Bergen, Norway
tjm@hvl.no

ABSTRACT

A new method in dynamics—the Moving Frame Method (MFM)—is used to conduct the analysis of how a robotic appendage (manipulator) on a Remotely Operated Vehicle (ROV) affects the motion of the ROV. An ROV performs multiple tasks on the seabed in the oil service industry. In most cases, an ROV pilot monitors and adjusts the movement of the vehicle due to induced motion by currents, buoyancy and the manipulators. Simulation data would assist the pilot and improve the stability of the ROV. This paper exploits a new method to analyze the induced movements of the ROV. The method uses the Special Euclidean Group (SE(3)) and the MFM. The method is supplemented with a restricted variation on the angular velocity to extract the equations of motion for the ROV. Then the equations of motion are solved numerically using Runge-Kutta Method and a reconstruction formula (founded upon the Cayley-Hamilton theorem) to secure the 3D rotations of the vehicle. The resulting motion is visualized with selected 2D plots. The 3D animation is displayed on a 3D web page. This paper closes with a summary of the simplifications used in the model and suggestions for advanced work.

FIGURE 1: Typical ROV

INTRODUCTION

A Remotely Operated Vehicle (ROV) is a subsea vehicle that is heavily used in the oil and gas service industry. It consists of a frame with thrusters, umbilical cables and usually two manipulator arms. One arm is for securing the ROV to a subsea installation, and one—the focus of this study—for operating different types of tools [1].

In subsea operations, the ROV is exposed to many types of forces. Currents, buoyancy and waves cause the ROV to translate and rotate [2]. Also, motion of the manipulator can cause the ROV to rotate, when heavy tools are attached to it. These movements can cause a risk for the subsea operations.

Today, these challenges are being addressed using human pilots who continuously correct the position and rotation of the ROV. By understanding how the manipulator arms affect the rotations of the ROV, the motion can be corrected instantaneously instead of correcting the rotation after it has occurred. This paper analyzes such motion using the MFM. In the next section, the method is introduced along with a description of the model. A critical aspect of this analysis is that it was conducted by undergraduate students. This confirms that the MFM, equipped with a consistent notation across the sub-disciplines of dynamics, is a valuable approach in analysis.

MODEL SYSTEM DESCRIPTION

The Moving Frame Method (MFM) is founded on: 1) Lie Group Theory distilled to rotation matrices; 2) the concept of attaching a frame to all moving objects; and 3) a compact notation from the discipline of geometrical physics. A complete description of the MFM is found in reference [3].

The MFM is expanded with a notation that groups both rotation and displacement (SE(3)). Finally, a restriction on the variation of the angular velocity is obtained by ensuring the commutativity of the mixed partials (the variation and time).

Traditional methods can also readily extract the equations of motion. The point, however, is that this research was conducted entirely by undergraduate students. The MFM simplifies 3D dynamics. The goal of this work is not simply to solve the system, but to demonstrate this power of a new method. This introduction alludes to the foundation of the theory, but focuses on the application for real-world challenges. The key with the MFM is that from 2D dynamics to 3D, the notation remains unchanged; from contact, to linkages, from self-contracting robotics to beam deformation and gyroscopic phenomena.

The model system analyzed in this paper is depicted in Fig. 2.

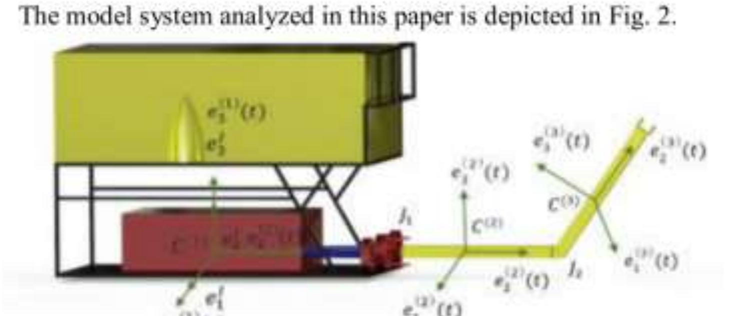


FIGURE 2: Model System Description

The model system consists of an ROV—body(1)—with a two-linked robot arm (manipulator). The first arm is body(2), and the second arm is body(3). Each of these two latter bodies rotate in specified directions with a given amplitude. Each coordinate frame is defined using a vector basis consisting of three orthogonal vectors on each link or body. Furthermore, an inertial frame is deposited from the first body and this deposition defines the start time of the analysis.

MATHEMATICAL MODEL

Starting with the inner most component, the ROV itself, and progressing systematically to the first and then second arm of the manipulator. The bodies are numbered in ascending order from the ROV, to the first arm and to the second, distal, arm as the third component. This section begins, however, with a general overview of the MFM as applied to linked systems.

The Moving Frame Method (with SE(3))

The left side of Fig. 3 presents an inertial orthogonal coordinate system, designated by $\{x_1 \ x_2 \ x_3\}^T$ in dark arrows. The superscript "I" denotes association with the inertial system and the subscripts represent the coordinate lines. An inertial frame derives from tangent vectors to the coordinate system $\mathbf{e}^I = (\mathbf{e}_1^I \ \mathbf{e}_2^I \ \mathbf{e}_3^I)$, where $\mathbf{e}_i^I$ represent the unit tangent vector to the x_i-axis. Introducing the inertial frame defined by the coordinate basis $\mathbf{e}^I$ and the $\mathbf{0}$-position vector, this notation defines the origin as $(\mathbf{e}^I \ \ \mathbf{0})$.

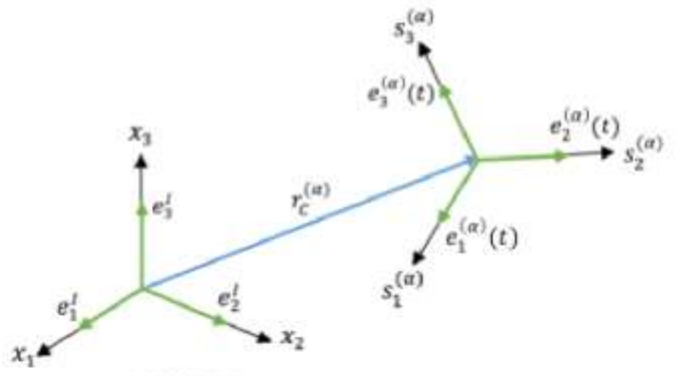


FIGURE 3: Relations of the Frames

The right side of Fig. 3 presents a moving orthogonal coordinate system, designated $s_C^{(\alpha)}(t) = \left\{s_1^{(\alpha)} \ s_2^{(\alpha)} \ s_3^{(\alpha)}\right\}^T$, on body- α frame. It also shows the associated time dependent moving frame, expressed by $\mathbf{e}^{(\alpha)}(\mathrm{t}) = (\mathbf{e}_1^{(\alpha)}(\mathrm{t}) \ \mathbf{e}_2^{(\alpha)}(\mathrm{t}) \ \mathbf{e}_3^{(\alpha)}(\mathrm{t}))$. The superscript "$(\alpha)$" denotes variables that relates to body- α. In the same figure, it is shown that the vector $\boldsymbol{e}_i^{(\alpha)}(t)$ are the unit tangent vector to the $s_i^{(\alpha)}$-axis.

Kinematics of frames in general

In this section, the kinematics of the system is analyzed. A more foundational description of the method used for the kinematics, is found in reference [3]. The location of the center of mass $C^{(1)}(t)$ of the body-1 frame—the ROV—from the origin of the inertial frame, is expressed by the absolute position vector **r** with "x" as the absolute coordinate symbol:

$$\mathbf{r}_C^{(1)}(t) = \ \mathbf{e}^I x_C^{(1)}(\mathrm{t}). \tag{1}$$

By placing the frame, as a row vector, before the components [3], one is able to maintain the traditional sense of components as a column vector and still carry out the computation. The ensuing notation enables one to view rotation matrices as matrix operators on columns of components. Furthermore, the notation of placing the basis to the left of the components is linguistically equivalent to "go in this direction, a certain amount". The traditional notation, with the basis on the right of the components, implies "go a certain amount, in this direction". Thus, the new notation is also more intuitive as language. Furthermore, it leads to definitive analyses of problems [4].

The relative location of the center of mass $C^{(\alpha+1)}(\mathrm{t})$ of a subsequent body, from the center of mass $C^{(\alpha)}(\mathrm{t})$ of the previous body is expressed with vector $\boldsymbol{s}_C^{(\alpha+1/\alpha)}$ and coordinate symbol $s_C^{(\alpha+1/\alpha)}$and in the frame of the previous body:

$$\mathbf{s}_C^{(\alpha+1/\alpha)}(t) = \ \mathbf{e}^{(\alpha)}(\mathrm{t}) s_C^{(\alpha+1/\alpha)}(t). \tag{2}$$

Thus, the absolute location of a frame is expressed as:

$$\mathbf{r}_C^{(\alpha+1)}(t) = \boldsymbol{r}_C^{(\alpha)}(t) + \mathbf{e}^{(\alpha)}(\mathrm{t}) s_C^{(\alpha+1/\alpha)}(t). \quad (3)$$

Equation (3) reveals that to locate a frame, one first proceeds to a previous frame in a linked tree, and, from that frame, and in that frame's basis, one proceeds to the current frame.

A 3×3 rotation matrix $R^{(\alpha)}(t)$ expresses the rotation of body-α vector-basis $\mathbf{e}^{(\alpha)}(\mathrm{t})$ from inertial vector-basis $\mathbf{e}^I$:

$$\mathbf{e}^{(\alpha)}(\mathrm{t}) = \mathbf{e}^I R^{(\alpha)}(\mathrm{t}). \quad (4)$$

Continuing, the relative rotation of a body-$(\alpha+1)$ vector-basis $\mathbf{e}^{(\alpha+1)}(\mathrm{t})$ is given by a relative rotation matrix $R^{(\alpha+1/\alpha)}(\mathrm{t})$ as:

$$\mathbf{e}^{(\alpha+1)}(\mathrm{t}) = \mathbf{e}^{(\alpha)}(\mathrm{t}) R^{(\alpha+1/\alpha)}(\mathrm{t}). \quad (5)$$

The orientation of the $\mathbf{e}^{(\alpha+1)}(\mathrm{t})$ body with respect to the inertial frame is, employing the algebra of SO(3):

$$\mathbf{e}^{(\alpha+1)}(\mathrm{t}) = \mathbf{e}^I R^{(\alpha)}(\mathrm{t}) R^{(\alpha+1/\alpha)}(\mathrm{t}) = \mathbf{e}^I\, R^{(\alpha+1)}(\mathrm{t}). \quad (6)$$

Frame Connections Matrices

The next step is to group both the rotation and displacement in one equation. The frame connection matrix, a form of a homogenous transformation matrix, combines rotations and translations. Homogenous transformation matrices were first used by Denavit and Hartenberg [5]. However, they did not recognize that such transformations were members of the Special Euclidean Group denoted as SE(3). The summative development here follows that of Murakami with a consistent reference to SE(3) [6].

The body-α frame connection $(\mathbf{e}^{(\alpha)}(\mathrm{t})\ \boldsymbol{r}_C^{(\alpha)}(t))$ is obtained from the inertial frame by post-multiplying it with a frame connection matrix $E^{(\alpha)}(\mathrm{t})$:

$$(\mathbf{e}^{(\alpha)}(\mathrm{t})\ \boldsymbol{r}_C^{(\alpha)}(t) = (\mathbf{e}^I\ \mathbf{0}) E^{(\alpha)}(\mathrm{t}). \quad (7)$$

where the 4x4 frame connection matrix is shown as:

$$E^{(\alpha)}(\mathrm{t}) = \begin{bmatrix} R^{(\alpha)}(\mathrm{t}) & x_C^{(\alpha)}(\mathrm{t}) \\ 0_1^T & 1 \end{bmatrix}. \quad (8)$$

where 0_1 is a 3×1 column zero vector. Here, $x_C^{(\alpha)}(\mathrm{t})$ designates coordinates from an inertial frame.

Next, the *relative* frame connection matrix is described for body $(\alpha+1)$ with respect to body-α.

$$(\mathbf{e}^{(\alpha+1)}(\mathrm{t})\ \boldsymbol{r}_C^{(\alpha+1)}(t)) = \left(\mathbf{e}^{(\alpha)}(\mathrm{t})\, \boldsymbol{r}_C^{(\alpha)}(t)\right) E^{(\alpha+1/\alpha)}(\mathrm{t}). \quad (9)$$

$$\text{where } E^{(\alpha+1/\alpha)}(\mathrm{t}) = \begin{bmatrix} R^{(\alpha+1/\alpha)}(\mathrm{t}) & s_C^{(\alpha+1/\alpha)}(\mathrm{t}) \\ 0_1^T & 1 \end{bmatrix}. \quad (10)$$

Progressing through the link-tree, the product of the absolute α-frame connection matrix and the relative $(\alpha+1/\alpha)$-frame connection matrix, becomes an absolute connection matrix of $(\alpha+1)$ from an inertial frame.

$$E^{(\alpha+1)}(\mathrm{t}) = E^{(\alpha)}(\mathrm{t}) E^{(\alpha+1/\alpha)}(\mathrm{t}). \quad (11)$$

Kinematics of ROV

With this short foundation, attention is now on the first body in the linked tree: the ROV itself. The frame connection matrix for the ROV $E^{(1)}(t)$ is assembled as:

$$E^{(1)}(t) = \begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix}. \quad (12)$$

Both the orientation and the position of the first frame (the ROV) is expressed directly with respect to the inertial frame as shown below as:

$$(\mathbf{e}^{(1)}(t)\ \ \mathbf{r}_C^{(1)}(t)) = (\mathbf{e}^I\ \ \mathbf{0}) E^{(1)}(t). \quad (13)$$

This gives:

$$(\mathbf{e}^{(1)}(t)\ \ \mathbf{r}_C^{(1)}(t)) = (\mathbf{e}^I\ \ \mathbf{0}) \begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix} = (\mathbf{e}^I R^{(1)}(t)\ \ \ \mathbf{e}^I x_C^{(1)}(t)). \quad (14)$$

The inverse, $\left(E^{(1)}(t)\right)^{-1}$ of the frame connection matrix is obtained by the matrix below, where the structure of SE(3) is exploited:

$$\left(E^{(1)}(t)\right)^{-1} = \begin{bmatrix} R^{(1)}(t) & x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix}^{-1} = \begin{bmatrix} (R^{(1)}(t))^T & -(R^{(1)}(t))^T x_C^{(1)}(t) \\ 0_1^T & 1 \end{bmatrix}. \quad (15)$$

The time derivative of the frame connection matrix:

$$\dot{E}^{(1)}(t) = \begin{bmatrix} \dot{R}^{(1)}(\mathrm{t}) & \dot{x}_C^{(1)}(\mathrm{t}) \\ 0_1^T & 0 \end{bmatrix}. \quad (16)$$

The time derivative of the Eq. (14) defines the linear velocity and the angular velocity vectors of the ROV:

$$(\dot{\mathbf{e}}^{(1)}(t)\ \ \dot{\mathbf{r}}_C^{(1)}(t)) = (\mathbf{e}^I\ \ \mathbf{0}) \dot{E}^{(1)}(t) = (\mathbf{e}^{(1)}\ (t)\ \ \mathbf{r}_C^{(1)}(t)) \left(E^{(1)}(t)\right)^{-1} \dot{E}^{(1)}(t) = (\mathbf{e}^{(1)}\ (t)\ \ \mathbf{r}_C^{(1)}(t)) \Omega^{(1)}(t). \quad (17)$$

The upper left sub-matrix of $\Omega^{(1)}(t)$ contains the skew symmetric angular velocity matrix $\overleftrightarrow{\omega}^{(1)}(t)$. By multiplying out the terms in Eq. (17) one finds:

$$\overleftrightarrow{\omega}^{(1)}(t) = \left(R^{(1)}(t)\right)^T \dot{R}^{(1)}(t). \tag{18}$$

The matrix $\Omega^{(1)}(t)$ in Eq. (17) is referred to as the time rate of the frame connection matrix. The matrix includes the information describing both the linear velocity and angular velocity of the ROV coordinate frame:

$$\Omega^{(1)}(t) = \begin{bmatrix} \overleftrightarrow{\omega}^{(1)}(t) & \left(R^{(1)}(t)\right)^T \dot{x}_C^{(1)}(t) \\ 0_1^T & 0 \end{bmatrix}. \tag{19}$$

The matrix above is used to express the time-rate of the ROV coordinate frame:

$$\dot{\boldsymbol{e}}^{(1)}(t) = \boldsymbol{e}^{(1)}(t)\overleftrightarrow{\omega^{(1)}(t)} = \\ \boldsymbol{e}^{(1)}(t)\begin{bmatrix} 0 & -\omega_3^{(1)}(t) & \omega_2^{(1)}(t) \\ \omega_3^{(1)}(t) & 0 & -\omega_1^{(1)}(t) \\ \omega_2^{(1)}(t) & \omega_1^{(1)}(t) & 0 \end{bmatrix}. \tag{20}$$

Exploiting the isomorphism between the skewed form in Eq. (20) and the column representation, one obtains the angular velocity vector for the ROV in terms of the ROV frame:

$$\boldsymbol{\omega}^{(1)}(t) = \mathbf{e}^{(1)}(t)\begin{pmatrix} \omega_1^{(1)} \\ \omega_2^{(1)} \\ \omega_3^{(1)} \end{pmatrix}. \tag{21}$$

The linear velocity vector $\dot{x}_C^{(1)}(t)$ for the ROV, is also expressed with respect to the inertial frame.

$$\dot{\mathbf{r}}_C^{(1)} = \mathbf{e}^{(1)}\left(R^{(1)}(t)\right)^T \dot{x}_C^{(1)}(t) = \mathbf{e}^I \dot{x}_C^{(1)}(t). \tag{22}$$

Equation (21) and (22) are the first two required expressions needed in the analysis. Attention is now turned to the second body (the first link).

Kinematics for the first arm

For the first robotic arm (the second body), a frame is placed at the center of mass $C^{(2)}$ of the first arm. The system is designed with revolute joints. The MFM is also capable of handle other types of joints like ball and socket joints, but in this analysis planar rotation is assumed.

A Cartesian coordinate system for the first arm is set to be $\left\{ \mathrm{s}_1^{(2)}\ \mathrm{s}_2^{(2)}\ \mathrm{s}_3^{(2)} \right\}^T$, and the coordinate frame for the first arm is $\mathbf{e}^{(2)}(\mathrm{t}) = (\mathbf{e}_1^{(2)}(\mathrm{t})\ \mathbf{e}_2^{(2)}(\mathrm{t})\ \mathbf{e}_3^{(2)}(\mathrm{t}))$.

The *relative position vector* from the center of mass for the ROV $C^{(1)}$ to the center of mass for the first arm $C^{(2)}$ is found by translating, rotating, and translating again. Thus, to reach the first joint J_1 at the end of the ROV where the first arm connects:

$$\boldsymbol{s}_{J_1} = \mathbf{e}^{(1)}(t)s_{J_1} = \mathbf{e}^{(1)}(t)\begin{pmatrix} 0 \\ s_{J_1} \\ 0 \end{pmatrix}. \tag{23}$$

The first arm rotates about one single axis, specifically the $\mathbf{e}_3^{(2)}(t)$ axis, so the relation between the ROV and the first arm is expressed as follows:

$$\mathbf{e}^{(2)}(t) = \mathbf{e}^{(1)}(t)R^{(2/1)}(t), \tag{24}$$

$$R^{(2/1)}(t) = R_3^{(2/1)}\left(\theta^{(2)}\right) = \begin{bmatrix} \cos\left(\theta^{(2)}\right) & -\sin\left(\theta^{(2)}\right) & 0 \\ \sin\left(\theta^{(2)}\right) & \cos\left(\theta^{(2)}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{25}$$

To reach the center of mass for the first arm $C^{(2)}$ from the first joint J_1 at the end of the ROV, second frame is used as (where the second axis points directly along the first arm):

$$\boldsymbol{s}_{C_2} = \mathbf{e}^{(2)}(t)s_{C_2} = \mathbf{e}^{(2)}(t)\begin{pmatrix} 0 \\ s_{C_2} \\ 0 \end{pmatrix}. \tag{26}$$

The relative position vector from the first frame $\mathbf{e}^{(1)}(t)$ at the center of mass of the ROV $C^{(1)}$ to the center of mass of the first arm $C^{(2)}$ is found as:

$$\boldsymbol{s}_C^{(2/1)} = \mathbf{e}^{(1)}(t)s_C^{(2/1)}. \tag{27}$$

The connection matrix for the first arm is be obtained as:

$$\left(\mathbf{e}^{(2)}(t)\ \ \mathbf{r}_c^{(2)}(t)\right) = \left(\mathbf{e}^{(1)}\ (t)\ \mathbf{r}_c^{(1)}(t)\right)E^{\left(\frac{2}{1}\right)}(t). \tag{28}$$

Thus, finally, $E^{(2/1)}(t)$ is the compact notation for the frame connection matrix:

$$\begin{aligned} E^{(2/1)}(\mathrm{t}) &= \begin{bmatrix} R^{(2/1)}(\mathrm{t}) & s_C^{(2/1)} \\ 0_1^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} I_3 & s_{J_1} \\ 0_1^T & 1 \end{bmatrix}\begin{bmatrix} R^{(2/1)}(t) & 0_1 \\ 0_1^T & 1 \end{bmatrix}\begin{bmatrix} I_3 & s_{C_2} \\ 0_1^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} R^{(2/1)}(t) & R^{(2/1)}(t)s_{C_2} + s_{J_1} \\ 0_1^T & 1 \end{bmatrix}. \end{aligned} \tag{29}$$

The relative connection matrix for the first arm written with respect to inertial frame:

$$\left(\mathbf{e}^{(2)}(t)\ \ \mathbf{r}_c^{(2)}(t)\right) = (\mathbf{e}^I\ \ \mathbf{0})E^{(2)}(t) = (\boldsymbol{e}^I\ \ \mathbf{0})E^{(1)}E^{(2/1)}. \tag{30}$$

where the $E^{(2)}(t)$ frame connection matrix is related to the inertial frame.

In Eq. (30) $E^{(2)}(t)$ consists of $R^{(2)}(t)$ (in the upper left quadrant) and $x_C^{(2)}(t)$ (in the upper right quadrant):

$$R^{(2)}(t) = R^{(1)}(t)R^{(2/1)}(t), \tag{31}$$

$$x_C^{(2)}(t) = R^{(2)}(t)s_{C_2} + R^{(1)}(t)s_{J_1} + x_c^{(1)}(t). \tag{32}$$

With this information, a rate analysis is conducted similar to that leading up to Eq. (19), and specific information is extracted as follows:

The angular velocity vector for the first arm:

$$\begin{aligned}\omega^{(2)}(t) &= (R^{(2/1)}(t))^T\omega^{(1)}(t) + \omega^{(2/1)}(t) = \\ &(R^{(2/1)}(t))^T\omega^{(1)}(t) + \dot{\theta}^{(2)}e_3.\end{aligned} \tag{33}$$

where $\omega^{(2/1)}(t)$ is the rotation for the first arm and only about one axis, therefore it can be represented with the notation $\dot{\theta}^{(2)}e_3$. This notation is also used for the second arm.

The linear velocity vector for the first arm with respect to inertial frame:

$$\begin{aligned}\dot{x}_C^{(2)}(t) &= R^{(1)}(t)\vec{\omega}^{(1)}(t)s_{J_1} + \dot{x}_C^{(1)}(t) = \\ &R^{(1)}(t)\left(\overrightarrow{s_{J_1}}\right)^T\omega^{(1)}(t) + \dot{x}_C^{(1)}(t).\end{aligned} \tag{34}$$

Equation (33) and (34) are the second two required expressions needed in the analysis. Attention is now turned to the third body (the second link).

Kinematics for the second arm

Similarly, to the first frame, a Cartesian coordinate system for the second arm is set to be $\left\{s_1^{(3)}\ s_2^{(3)}\ s_3^{(3)}\right\}^T$, and the frame for the second arm is $\mathbf{e}^{(3)}(\mathrm{t}) = (\mathbf{e}_1^{(3)}(\mathrm{t})\ \mathbf{e}_2^{(3)}(\mathrm{t})\ \mathbf{e}_3^{(3)}(\mathrm{t}))$.

For the second robotic arm, a frame is placed at the center of mass $C^{(3)}$. The relative position vector from the center of mass for the first arm $C^{(2)}$ to the center of mass for the second arm $C^{(3)}$, is found by an analysis similar to that leading up to Eq. (27). The distance from the center of mass for the first arm $C^{(2)}$ to the second joint J_2, and the distance from the second joint J_2 to the center of mass for the second arm $C^{(3)}$ is respectively defined as:

$$\boldsymbol{s}_{J_2} = \mathbf{e}^{(2)}(t)s_{J_2} = \mathbf{e}^{(2)}(t)\begin{pmatrix}0\\ s_{J_2}\\ 0\end{pmatrix}. \tag{35}$$

$$\boldsymbol{s}_{C_3} = \mathbf{e}^{(3)}(t)s_{C_3} = \mathbf{e}^{(3)}(t)\begin{pmatrix}0\\ s_{C_3}\\ 0\end{pmatrix}. \tag{36}$$

As with the first arm, the second arm also rotates about one single axis, the $\mathbf{e}_1^{(3)}(t)$ axis. This relation between the first arm and the second arm is expressed as follows:

$$\mathbf{e}^{(3)}(t) = \mathbf{e}^{(2)}(t)R^{(3/2)}(t). \tag{37}$$

$$R^{(3/2)}(t) = R_1^{(3/2)}(\theta^3) = \begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\theta^{(3)}) & -\sin(\theta^{(3)})\\ 0 & \sin(\theta^{(3)}) & \sin(\theta^{(3)})\end{bmatrix}. \tag{38}$$

This is also a planar rotation, but now both arms are rotating in different planes and this is now a full 3D problem.

The relative position vector from the center of mass of the first arm $C^{(2)}$ to the center of mass of the second arm $C^{(3)}$ is found, using the second frame as:

$$\boldsymbol{s}_C^{(3/2)} = \mathbf{e}^{(2)}(t)s_C^{(3/2)}. \tag{39}$$

The second arm connection matrices are expressed with respect to the inertial frame as:

$$\begin{aligned}\left(\mathbf{e}^{(3)}(t)\ \mathbf{r}_C^{(3)}\right) &= \left(\mathbf{e}^{(2)}(t)\ \mathbf{r}_c^{(2)}\right)E^{(3/2)}(t),\\ \left(\mathbf{e}^{(3)}(t)\ \mathbf{r}_C^{(3)}\right) &= (\mathbf{e}^I\ \ \mathbf{0})E^{(1)}(t)E^{(2/1)}(t)E^{(3/2)}(t),\\ \left(\mathbf{e}^{(3)}(t)\ \mathbf{r}_C^{(3)}\right) &= (\mathbf{e}^I\ \ \mathbf{0})E^{(3)}(t).\end{aligned} \tag{40}$$

Where the frame connection matrix between the first arm and the second arm are now obtained as:

$$\begin{aligned}E^{(3/2)}(t) &= \begin{bmatrix}I_3 & s_{J_2}\\ 0_1^T & 1\end{bmatrix}\begin{bmatrix}R^{(3/2)}(t) & 0_1\\ 0_1^T & 1\end{bmatrix}\begin{bmatrix}I_3 & s_{C_3}\\ 0_1^T & 1\end{bmatrix}\\ &= \begin{bmatrix}R^{(3/2)}(t) & R^{(3/2)}(t)s_{C_3} + s_{J_2}\\ 0_1^T & 1\end{bmatrix}.\end{aligned} \tag{41}$$

In Eq. (40) $E^{(3)}(t)$ consists of $R^{(3)}(t)$, and $x_C^{(3)}(t)$:

$$R^{(3)}(t) = R^{(1)}(t)R^{(2/1)}(t)R^{(3/2)}(t), \tag{42}$$

$$\begin{aligned}x_C^{(3)}(t) &= R^{(3)}(t)s_{C_3} + R^{(2)}(t)(s_{C_2} + s_{J_2}) + \\ &R^{(1)}(t)s_{J_1} + x_C^{(1)}(t).\end{aligned} \tag{43}$$

The angular velocity vector for the second arm is obtained as:

$$\begin{aligned}\omega^{(3)}(t) &= (R^{(3/1)}(t))^T\omega^{(1)}(t)\\ &+(R^{(3/2)}(t))^T\dot{\theta}^{(2)}\mathbf{e}_3 + \dot{\theta}^{(3)}\mathbf{e}_1.\end{aligned} \tag{44}$$

The linear velocity vector for the second arm with respect to inertial frame:

$$\dot{x}_C^{(3)}(t) = R^{(1)}(t)R^{(2/1)}(t)R^{(3/2)}(t)\left(\overleftarrow{s_{J_2}}\right)^T \omega^{(3)}(t) + R^{(1)}(t)R^{(2/1)}\left(s_{J1}\right)^T \omega^{(2)}(t) + R^{(1)}(t)\left(\overleftarrow{s_{J_1}}\right)^T \omega^{(1)}(t) + \dot{x}_C^{(1)}(t). \quad (45)$$

Equation (44) and (45) are the last two required expressions needed in the analysis. This closes the study of kinematics. Attention is now turned to kinetics.

Kinetics

Now turning to the Kinetics, and commencing with the generalized coordinate, Hamilton's Principle is applied. The details regarding the method are found in the references [6, 7].

Generalized Coordinates

The generalized velocity $\{\dot{\mathrm{X}}(\mathrm{t})\}$ is defined as a $6\mathrm{n} \times 1$ matrix that consists of both linear velocity $\dot{\mathrm{x}}_{\mathrm{C}}^{(\alpha)}$ and angular velocities $\omega^{(\alpha)}$ at the center of mass, in alternating order, for each body.

For a three-linked system, as analyzed in this paper, the vector $\{\dot{X}(t)\}$ has 18 rows. The coordinates $\dot{x}_C^{(1)}(t), \dot{x}_C^{(2)}(t)$ and $\dot{x}_C^{(3)}(t)$ are all presented from the inertial frame.

$$\{\dot{X}(t)\} = \begin{pmatrix} \dot{x}_C^{(1)}(t) \\ \omega^{(1)}(t) \\ \dot{x}_C^{(2)}(t) \\ \omega^{(2)}(t) \\ \dot{x}_C^{(3)}(t) \\ \omega^{(3)}(t) \end{pmatrix}. \quad (46)$$

The generalized velocity $\{\dot{X}(t)\}$ as in Eq. (46) is related linearly to *essential* generalized velocity $\{\dot{q}(t)\}$. This is an $n^* \times 1$ matrix for a system with n^*degrees of freedom. For the ROV, there is eight degrees of freedom, and $\{\dot{q}(t)\}$ is defined as:

$$\{\dot{q}(t)\} = \begin{pmatrix} \dot{x}_C^{(1)}(t) \\ \omega^{(1)}(t) \\ \dot{\theta}^{(2)}(t) \\ \dot{\theta}^{(3)}(t) \end{pmatrix}. \quad (47)$$

These terms are, respectively: the translational velocity of the ROV (3 components), the angular velocity of the ROV (3 components), the rotational velocity of the first arm and the rotational velocity of the second arm—a total of eight.

The linear relationship between $\{\dot{X}(t)\}$ and $\{\dot{q}(t)\}$ is expressed as:

$$\{\dot{X}(t)\} = [B(t)]\{\dot{q}(t)\}. \quad (48)$$

The B matrix $[B(t)]$ in Eq. (48) is obtained by considering Eq. (21), (22), (33), (34), (44), and (45).

Hamilton's Principle

For Kinetics, the Newton and Euler's equations is derived (the equations of motion). Hamilton's Principle is:

$$\delta \int_{t_0}^{t_1} L^{(\alpha)}(t) + W^{(\alpha)}(t)\,)dt = 0. \quad (49)$$

The Lagrangian $\mathrm{L}^{(\alpha)}(\mathrm{t})$ is defined as the difference between kinetic energy $\mathrm{K}^{(\alpha)}(\mathrm{t})$ and potential energy $\mathrm{U}^{(\alpha)}(\mathrm{t})$.

$$L^{(\alpha)}(t) \equiv K^{(\alpha)}(t) - U^{(\alpha)}(t). \quad (50)$$

Principle of Virtual work is redefined from Hamilton's principle as:

$$\delta \int_{t_0}^{t_1} (K^{(\alpha)}(t) - U^{(\alpha)}(t) + W^{(\alpha)}(t)\,)dt = 0. \quad (51)$$

Here, "U" continues to designate potential energy from conservative forces such as gravity, while "W" represents the work done by the non-conservative forces.

The Kinetic energy $K^{(\alpha)}(t)$ and the potential energy $U^{(\alpha)}(t)$ is respectively defined as:

$$K^{(\alpha)}(t) = \tfrac{1}{2}\left\{\dot{\mathbf{r}}_C^{(\alpha)} \mathbf{L}_C^{(\alpha)} + \boldsymbol{\omega}^{(\alpha)} \mathbf{H}_C^{(\alpha)}\right\}. \quad (52)$$

$$U^{(\alpha)}(t) = m^{(\alpha)} g x_{3C}^{(\alpha)}(t). \quad (53)$$

The Kinetic energy $K^{(\alpha)}(t)$ consists of translational and rotational components with respect to the center of mass of each link in the system.

The linear momentum $\mathbf{L}_C^{(\alpha)}$ and the angular momentum $\mathbf{H}_C^{(\alpha)}$ are respectively defined as:

$$\mathbf{L}_C^{(\alpha)}(t) \equiv \mathbf{e}^I m^{(\alpha)} \dot{x}_C^{(\alpha)}(t), \quad (54)$$

$$\mathbf{H}_C^{(\alpha)}(t) \equiv \mathbf{e}^{(\alpha)}(t)^{(\alpha)} J_C^{(\alpha)} \omega^{(\alpha)}(t). \quad (55)$$

Where the mass moment of inertia matrix $J_C^{(\alpha)}$ is a 3×3 matrix.

If the coordinate systems are placed at the center of mass in each link, and oriented such that the axes coincide with the principal axes of each link, then the $J_C^{(\alpha)}$ matrices are diagonal.

$$J_C^{(\alpha)} = \begin{bmatrix} J_{11}^{(\alpha)} & 0 & 0 \\ 0 & J_{22}^{(\alpha)} & 0 \\ 0 & 0 & J_{33}^{(\alpha)} \end{bmatrix}. \tag{56}$$

When substituting Eq. (54) and (55) into Eq. (52), the component form of kinetic energy becomes:

$$K^{(\alpha)}(t) = \tfrac{1}{2}\left\{\dot{x}_C^{(\alpha)^T} m^{(\alpha)} \dot{x}_C^{(\alpha)} + \omega^{(\alpha)} J_C^{(\alpha)} \omega^{(\alpha)}\right\}. \tag{57}$$

Equation (57) is expressed more compactly as:

$$K(t) = \tfrac{1}{2}\{\dot{X}(t)\}^T\{H(t)\} = \tfrac{1}{2}\{\dot{X}(t)\}^T[M]\{\dot{X}(t)\}. \tag{58}$$

The generalized momenta $\{H(t)\}$ is a $6\text{n} \times 1$ matrix that contains both the linear momentum $\mathbf{L}_C^{(\alpha)}$and the angular momentum $\mathbf{H}_C^{(\alpha)}$, respectively defined as in Eq. (54) and (55):

$$\{H(t)\} = \begin{pmatrix} L_C^{(1)}(t) \\ H_C^{(1)}(t) \\ L_C^{(2)}(t) \\ H_C^{(2)}(t) \\ L_C^{(3)}(t) \\ H_C^{(3)}(t) \end{pmatrix} = [M]\{\dot{X}(t)\}. \tag{59}$$

To build the generalized momenta $\{H(t)\}$ as in Eq. (59) the generalized mass matrix $[M]$ is defined. It is defined as a $6n \times 6n$ matrix, that contains the mass $m^{(\alpha)}$ and moment of inertia $\mathrm{J}_C^{(\alpha)}$ for each link in the system:

$$[M] = \begin{bmatrix} m^{(1)}I_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & J_C^{(1)} & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & m^{(2)}I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & J_C^{(2)} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & m^{(3)}I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & J_C^{(3)} \end{bmatrix}. \tag{60}$$

where I_3 is a 3×3 identity matrix, and 0_3 is a 3×3 zero matrix.

When inserting Eq. (46), (59), (60) into Eq. (58), the equation for kinetic energy for the ROV turns out:

$$K(t) = \tfrac{1}{2}\begin{pmatrix} \dot{x}_C^{(1)}(t) \\ \omega^{(1)}(t) \\ \dot{x}_C^{(2)}(t) \\ \omega^{(2)}(t) \\ \dot{x}_C^{(3)}(t) \\ \omega^{(3)}(t) \end{pmatrix}^T \begin{pmatrix} L_C^{(1)}(t) \\ H_C^{(1)}(t) \\ L_C^{(2)}(t) \\ H_C^{(2)}(t) \\ L_C^{(3)}(t) \\ H_C^{(3)}(t) \end{pmatrix}. \tag{61}$$

To derive the principal of virtual work, the variation of the Lagrangian $L^{(\alpha)}(t)$ as in Eq. (51) is required, and hence the variation of the generalized velocities must be obtained.

To take the variation of the Lagrangian, the variation of the generalized velocities must be obtained. More specifically, the variation of the angular velocity is needed. It will be seen that there exists a restriction on the angular velocity. To this end, some definitions are presented and their utility is shown, shortly.

First, define $\delta\Pi^{(\alpha)}$ as:

$$\delta\Pi^{(\alpha)} = \begin{bmatrix} \overleftrightarrow{\delta\pi^{(\alpha)}(t)} & \left(R^{(\alpha)}(t)\right)^T \delta x_c^{(\alpha)}(t) \\ 0_3^T & 0 \end{bmatrix}. \tag{62}$$

Where the virtual rotational displacement $\overleftrightarrow{\delta\pi^{(\alpha)}}$ is defined as:

$$\overleftrightarrow{\delta\pi^{(\alpha)}(t)} = \left(R^{(\alpha)}(t)\right)^T \delta R^{(\alpha)}(t). \tag{63}$$

As an aside, π does not exist. It exists only in its representation of the variation, $\delta\boldsymbol{\pi}^{(\alpha)} = \mathbf{e}^{(\alpha)}(\delta\pi_1^{(\alpha)}\ \delta\pi_2^{(\alpha)}\ \delta\pi_3^{(\alpha)})^T$.

From the definition in Eq. (62) the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ is defined as a $6n \times 1$ column matrix:

$$\{\delta\tilde{X}(t)\} = \begin{pmatrix} \delta x_C^{(1)}(t) \\ \delta\pi^{(1)}(t) \\ \delta x_C^{(2)}(t) \\ \delta\pi^{(2)}(t) \\ \delta x_C^{(3)}(t) \\ \delta\pi^{(3)}(t) \end{pmatrix}. \tag{64}$$

As in Eq. (48), the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ has its relationship *also* through the $[B(t)]$ matrix as per Murakami [3]. The linear relation gives the virtual essential generalized displacement $\{\delta q(t)\}$.

$$\{\delta\tilde{X}(t)\} = [B(t)]\{\delta q(t)\}. \tag{65}$$

For a three-linked system, the relationship between $\{\delta\tilde{X}(t)\}$ and $\{\delta q(t)\}$ is expressed as:

$$\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta x_C^{(2)}(t)\\ \delta\pi^{(2)}(t)\\ \delta x_C^{(3)}(t)\\ \delta\pi^{(3)}(t)\end{pmatrix} = [B(t)]\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta\theta^{(2)}(t)\\ \delta\theta^{(3)}(t)\end{pmatrix}. \qquad (66)$$

Introducing the commutativity of time differentiation and variation of a frame.

$$\delta\omega^{(\alpha)}(t) = \tfrac{d}{dt}\delta\pi^{(\alpha)}(t) + \overleftrightarrow{\omega^{(\alpha)}(t)}\delta\pi^{(\alpha)}(t), \qquad (67)$$

$$\tfrac{d}{dt}\delta x_C^{(\alpha)}(t) = \delta\dot{x}_C^{(\alpha)}(t). \qquad (68)$$

Obtaining Eq. (67) is a bit of work and shown reference [5]. Equations (67) and (68) are more compactly expressed in the matrix form of virtual generalized velocity $\{\delta\dot{X}(t)\}$, which is defined as:

$$\{\delta\dot{X}(t)\} = \left\{\delta\dot{\tilde{X}}(t)\right\} + [D(t)]\{\delta\tilde{X}(t)\}. \qquad (69)$$

Where $[D(t)]$ is a $6n \times 6n$ skew symmetric matrix, defined for a three-linked system as:

$$[D(t)] = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & \overleftrightarrow{\omega^{(1)}(t)} & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(2)}(t)} & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(3)}(t)}\end{bmatrix}. \qquad (70)$$

Next the variation of the kinetic energy $\delta K^{(\alpha)}(t)$, the potential energy $\delta U^{(\alpha)}(t)$, and the the virtual work $\delta W^{(\alpha)}(t)$ is derived.

The variation of the kinetic energy $\delta K^{(\alpha)}(t)$ is derived by taking the variation of Eq. (58):

$$\delta K(t) = \{\delta\dot{X}(t)\}^T\{H(t)\} = \{\delta\dot{X}(t)\}^T[M]\{\dot{X}(t)\}. \qquad (71)$$

The variation of work is $W^{(\alpha)}(t)$. This is defined as the work done by resultant external force $F_C^{(\alpha)^I}(t)$ and external torque $M_C^{(\alpha)}(t)$, defined with respect to the virtual generalized displacement $\{\delta\tilde{X}(t)\}$ as in Eq. (64):

$$\delta W(t) = \{\delta\tilde{X}(t)\}^T\{F(t)\}. \qquad (72)$$

Where the external force $\{F(t)\}$ is a $6n \times 1$matrix, defined for a three-linked system as:

$$\{F(t)\} = \begin{pmatrix} F_C^{(1)^I}(t)\\ M_C^{(1)^I}(t)\\ F_C^{(2)^I}(t)\\ M_C^{(2)^I}(t)\\ F_C^{(3)^I}(t)\\ M_C^{(3)^I}(t)\end{pmatrix}. \qquad (73)$$

By substitution Eq. (65) into (72), the virtual work $\delta W(t)$ expressed with respect to essential generalized displacement becomes:

$$\delta W(t) = \{\delta q(t)\}^T\{F^*(t)\}. \qquad (74)$$

Where the essential force $\{F^*(t)\}$ is defined as a $n^* \times 1$ matrix, where n^* defines the number of essential generalized coordinates:

$$\{F^*(t)\} = [B(t)]^T\{F(t)\}. \qquad (75)$$

The virtual work done by conservative external forces is taken to account by variation of potential energy $\delta U^{(\alpha)}(t)$:

$$\delta W^{(\alpha)}(t) = -\delta U^{(\alpha)}(t). \qquad (76)$$

Therefore, the virtual work done by conservative forces is included in the virtual work $\delta W^{(\alpha)}(t)$ as in Eq. (74).

Substitution of Eq. (71) and (74) into Eq. (51) with the use of Eq. (76), Hamilton's principle turns to:

$$\int_{t_0}^{t_1}\left\{\{\delta\dot{X}(t)\}^T[M]\{\dot{X}(t)\} + \{\delta q(t)\}^T\{F^*(t)\}\right\}dt = 0. \qquad (77)$$

Where $\{\delta\dot{X}(t)\}$ from Eq. (69) is expressed for a three-linked system as:

$$\{\delta\dot{X}(t)\} = \begin{pmatrix}\delta\dot{x}_C^{(1)}(t)\\ \delta\omega^{(1)}(t)\\ \delta\dot{x}_C^{(2)}(t)\\ \delta\omega^{(2)}(t)\\ \delta\dot{x}_C^{(3)}(t)\\ \delta\omega^{(3)}(t)\end{pmatrix} = \frac{d}{dt}\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta x_C^{(2)}(t)\\ \delta\pi^{(2)}(t)\\ \delta x_C^{(3)}(t)\\ \delta\pi^{(3)}(t)\end{pmatrix} + \qquad (78)$$

$$\begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & \overleftrightarrow{\omega^{(1)}(t)} & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(2)}(t)} & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & 0_3\\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 & \overleftrightarrow{\omega^{(3)}(t)}\end{bmatrix}\begin{pmatrix}\delta x_C^{(1)}(t)\\ \delta\pi^{(1)}(t)\\ \delta x_C^{(2)}(t)\\ \delta\pi^{(2)}(t)\\ \delta x_C^{(3)}(t)\\ \delta\pi^{(3)}(t)\end{pmatrix}.$$

In Eq. (77) the variation of omega $\delta\omega^{(\alpha)}$(t) is eliminated by inserting Eq. (69) into Eq. (77):

$$\int_{t_0}^{t_1}\{\delta\tilde{X}\}^T\frac{d}{dt}\left([M]\{\dot{X}(t)\}\right)+\{\delta\tilde{X}(t)\}^T[D(t)][M]\{\dot{X}(t)\}-\{\delta q(t)\}^T\{F^*(t)\}dt=0. \tag{79}$$

Performing the derivation in Eq. (79), and substitute $\{\delta\tilde{X}(t)\}$ and $\{\dot{X}(t)\}$ with the definitions in Eq. (65) and (39) to obtain the equations of motion:

$$\int_{t_0}^{t_1}\{\delta q(t)\}^T([B(t)]^T[M][B(t)]\{\ddot{q}(t)\}+[B(t)]^T[M]\left[\dot{B}(t)\right]\{\dot{q}(t)\}+[B(t)]^T[D(t)][M][B(t)]\{\dot{q}(t)\}-\{F^*(t)\})\}dt=0. \tag{80}$$

Hence the equations of motion become:

$$[M^*(t)]\{\ddot{q}(t)\}+[N^*(t)]\{\dot{q}(t)\}-\{F^*(t)\}=0. \tag{81}$$

Looking at Eq. (81), the two matrices $[M^*(t)]$ and $[N^*(t)]$ are respectively defined as:

$$[M^*(t)]=[B(t)]^T[M][B(t)], \tag{82}$$

$$[N^*(t)]=[B(t)]^T\left([M]\left[\dot{B}(t)\right]+[D(t)][M][B(t)]\right). \tag{83}$$

Eq. (81) is the equation of motion with the essential generalized velocities. The mathematical model for deriving the equations of motion of an ROV with a three-linked robotic arm has been set.

SIMPLIFIED MODEL

Equation (81) must be solved numerically. However, this paper will attempt to use the Matlab symbolic manipulator to obtain more direct equations. Thus, the simplifications that follow really amount to an attempt by student authors, to obtain a qualitative solution under certain simplifying cases, also to simpler equation systems. The main issue up to this point, and the emphasis to be made, is that the MFM and the restriction on the angular velocity can enable undergraduate students to conduct this work.

To see the impact of the system parameters, a numerical experiment is conducted with a 3D simulation. The mathematical model built above is simplified with several assumptions.

Each term in the generalized velocity $\{\dot{X}(t)\}$ as in Eq. (46) contributes to the total momentum. For the sake of the experiment translation is neglected, and only rotations are analyzed.

Therefore, the first assumption made is for the translation of the ROV to be prescribed zero in velocity and acceleration, and no translation of the body applies. Hence the ROV only rotates around its center of mass.

$$\delta x_C^{(1)}(t)=0, \quad \dot{x}_C^{(1)}(t)=0, \quad \ddot{x}_C^{(1)}(t)=0. \tag{84}$$

The second assumption made, is prescribed angular velocities and accelerations for the first and second arm.

$$\theta^{(2)}(t)=\frac{\pi}{2}\sin\left(t\frac{2*\pi}{10}\right), \quad \theta^{(3)}(t)=\frac{\pi}{2}\sin\left(t\frac{2*\pi}{3}\right). \quad \text{Hence: } \delta\theta^{(2)}(t)=\delta\theta^{(3)}(t)=0. \tag{85}$$

The third assumption made, regards the dimensions to the body of the ROV. It is assumed to have the shape of a square prism, with the width (w) equal to the height (h) and the length (l)

$$J_1^1=J_2^1=J_3^1=\frac{2}{3}m_1. \quad \text{where } m_1=2000kg \tag{86}$$

The fourth assumption made, regards the first arm. The first arm is assumed to be a slender rod with no mass

$$m^{(2)}=0\ , J_1^2=j_3^2, \qquad J_2^2=0. \tag{87}$$

The fifth assumption made, is for the second arm. The second arm is assumed to be slender rod, symmetric about its first and third axis.

$$J_1^3=J_3^3=\frac{1}{3}m_3, \qquad J_2^3=0, \quad \text{where } m_3=100kg \tag{88}$$

With these assumptions made, the system simplifies and with the first and second assumption as in Eq. (84) and (85). Because of the prescribed rates, the virtual essential generalized displacement $\{\delta q(t)\}$ as in Eq. (66) turns out:

$$\{\delta q(t)\}=\begin{pmatrix}0\\ \delta\pi^{(1)}(t)\\ 0\\ 0\end{pmatrix}. \tag{89}$$

Equation (89) is inserted into Eq. (81) with the definitions in Eq. (82) and (83).
Equation (81) is then simplified. The first assumption as in Eq. (84) yields that the translation of the ROV is prescribed to be zero in velocity as with acceleration, hence the top three rows of the essential generalized velocity $\{\dot{q}(t)\}$ and acceleration $\{\ddot{q}(t)\}$ is zero.

$$[M^*(t)]_{8\times8}\begin{pmatrix}0\\ \dot{\omega}^{(1)}(t)\\ \ddot{\theta}^{(2)}(t)\\ \ddot{\theta}^{(3)}(t)\end{pmatrix}_{8\times1}+[N^*(t)]_{8\times8}\begin{pmatrix}0\\ \omega^{(1)}(t)\\ \dot{\theta}^{(2)}(t)\\ \dot{\theta}^{(3)}(t)\end{pmatrix}_{8\times1}-\{F^*(t)\}_{8\times1}=0. \tag{90}$$

Equation (90) is simplified in two steps. First due to the three top rows of $\{\dot{q}(t)\}$ and $\{\ddot{q}(t)\}$ are zero, then $[M^*(t)]$, and $[N^*(t)]$ reduces from 8×8 to 8×5 matrices by removing the first three columns. In this process $\{\dot{q}(t)\}$, and $\{\ddot{q}(t)\}$ reduce from 8×1 to 5×1.

A second reduction comes from $\{\delta q(t)\}$. The three top rows and two bottom rows of $\{\delta q(t)\}$ are zero, hence the three top rows and two bottom rows of the total product in Eq. (90) is zero. Then $[M^*(t)]$ and $[N^*(t)]$ reduces from 8×5 to 3×5 matrices by removing the three top rows and two bottom rows.

Any force or moment that impacts a prescribed motion and angel is zero reduces $\{F^*(t)\}$ from 8×1 to a 3×1 vector, by removing the three top rows and two bottom rows.

Equation (90) results in the reduced form:

$$\left([M^*(t)]_{Reduced_{3\times5}} \begin{pmatrix} \dot{\omega}^{(1)}(t) \\ \ddot{\theta}^{(2)}(t) \\ \ddot{\theta}^{(3)}(t) \end{pmatrix}_{5\times1} \right)_{3\times1} = \{F^*(t)\}_{3\times1} - \left([N^*(t)]_{Reduced_{3\times5}} \begin{pmatrix} \omega^{(1)}(t) \\ \dot{\theta}^{(2)}(t) \\ \dot{\theta}^{(3)}(t) \end{pmatrix}_{5\times1} \right)_{3\times1}. \tag{91}$$

Equation (91) gives three equations:

$$\dot{\omega}_1{}^{(1)}(t) = \frac{-[N^*(t)]_{1,j}\{\dot{q}(t)\}_j - [M^*(t)]_{1,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_1}{[M^*(t)]_{11}} \qquad j\epsilon\{1,\dots,5\}, \quad i\epsilon\{2,3,4,5\}, \tag{92}$$

$$\dot{\omega}_2{}^{(1)}(t) = \frac{-[N^*(t)]_{2,j}\{\dot{q}(t)\}_j - [M^*(t)]_{2,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_2}{[M^*(t)]_{22}} \qquad j\epsilon\{1,\dots,5\}, \quad i\epsilon\{1,3,4,5\}, \tag{93}$$

$$\dot{\omega}_3{}^{(1)}(t) = \frac{-[N^*(t)]_{3,j}\{\dot{q}(t)\}_j - [M^*(t)]_{3,i}\{\ddot{q}(t)\}_i + \{F^*(t)\}_3}{[M^*(t)]_{33}} \qquad j\epsilon\{1,\dots,5\}, \quad i\epsilon\{1,2,4,5\}. \tag{94}$$

Reconstruction of the rotation matrix
In $[M^*(t)]$, the rotation matrix for the ROV appears, and thus the rotation matrix for the ROV must be calculated at each time step. If $R(t)$ and $\omega(t)$ is known, then $\omega(t + \Delta t)$ is calculated using 4th order Runge-Kutta Method [8]. $R(t + \Delta t)$ needs to be obtained before moving on to the next time step of Runge-Kutta Method.

The rate of change of the rotation matrix is defined from Eq. (18):

$$\dot{R}^{(1)}(t) = R^{(1)}(t)\,\overleftrightarrow{\omega^{(1)}(t)}. \tag{95}$$

For a constant angular velocity problem, this could be solved analytically by integrating Eq. (95), but ω is not constant. However, by taking the average of $\omega(t)$ and $\omega(t + \Delta t)$ and assuming a constant omega between time steps, the rotation matrix is obtained with the following formula.

$$R(t + \Delta t) = R(t)\exp\left(\Delta t\overleftrightarrow{\omega(t + \Delta t/2)}\right). \tag{96}$$

Where the exponential of the angular velocity is found using the *Cayley-Hamilton theorem,* which leads to the following equation:

$$\exp(\Delta t\overleftrightarrow{\omega}) = I_3 + \frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}\sin(\Delta t\|\omega(t + \Delta t/2)\|) + \left(\frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}\right)^2 (1 - \cos(\Delta t\|\omega(t + \Delta t/2)\|)). \tag{97}$$

If a unit vector $u = \frac{\overleftrightarrow{\omega(t+\Delta t/2)}}{\|\omega(t+\Delta t/2)\|}$ is defined in the direction of the angular velocity vector, then Eq. (97) agrees with the *Rodrigues formula*. Equation (97) is used in the numerical integration of Eq. (95) to obtain the rotation matrix [6, 8].

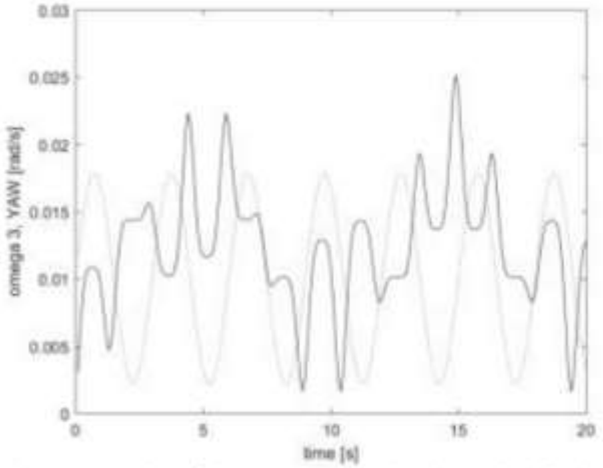


FIGURE 4: Plot of the angular velocity about the third axis (yaw axis) of the ROV and prescribed motion of the second arm (dashed)

Numerical solution
A symbolic manipulator is used to obtain the equation of motion. Due to lack of computer processing power and limitations of the program used, some simplifications were necessary. By collapsing $\{\dot{X}(t)\}$ to only hold $\omega^{(a)}(t)$, hence collapsing $[B(t)]$, $[M]$ and $[D(t)]$, information about the linear momentum is lost,

but the angular momentum still stands. For future iterations of this problem, linear momentum can be considered. The three coupled first order differential equations Eq. (92), (93) and (94), are decoupled using a symbolic manipulator through Gauss-Jordan elimination. Then, the three decoupled equations are solved numerically using the Runge-Kutta Method, to obtain $\omega_1^{(1)}(t)$, $\omega_2^{(1)}(t)$ and $\omega_3^{(1)}(t)$.

RESULTS

Using Eq. (92), (93) and (94) and the simplifications above, $\dot{\omega}^{(1,2,3)}(t)$ are solved numerically. Runge-Kutta is used to solve these equations with time steps of 0.02 seconds for 1000 steps. The plots are presented below in Fig. 4, 5 and 6:

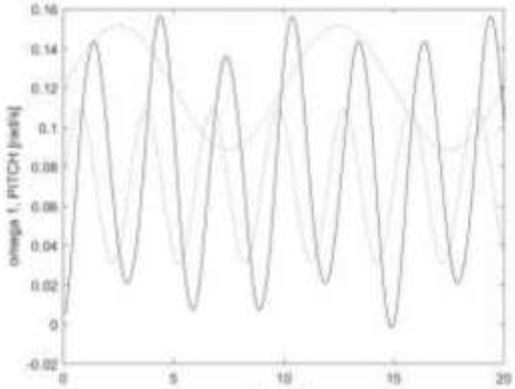


FIGURE 5: Angular velocity about first axis (pitch) of ROV and prescribed angular motion of first and second arm (dashed)

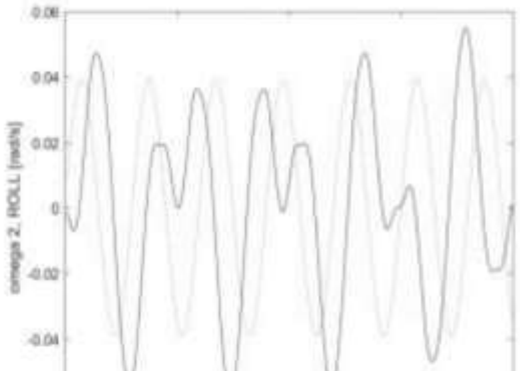


FIGURE 6: Angular velocity about second axis (roll) of the ROV and prescribed angular motion of just the second arm (dashed).

In these plots, one observes the induced ROV angular velocity due to the input angular velocity of the first and second arms. The response of the ROV closely matches the input angular velocities of the two links. However, the reader is reminded that many simplifications were undertaken, especially, ignoring wave forces, added mass and the input torque generator on the two arms. However, this does confirm the qualitative response.

More informative, perhaps, is observing the induced rocking on the ROV, itself.

Finally, the authors point the reader to a 3D web page (viewable on most mobile devices) (control box in upper right corner of web page):
http://home.hib.no/prosjekter/dynamics/2017/robot/

On this web page, one can alter the input parameters and observe the rocking motion of the craft. Left, middle and right mouse buttons translate as: zoom, pan and rotate.

CONCLUSION

Primarily, this work needs experimental confirmation. This work is next and will utilize a wave tank at HVL.

The results show that it is possible to analyze the induced rotations due to movement of the manipulators. By utilizing the approach used in this paper, combined with coding the solution of Eq. (81) directly, a broader in-depth analysis can be performed to include both rotations and displacements, as well as reducing the number of simplification necessary to carry out the calculations.

Also, by prescribing the motion of the first arm to be zero, the results reduce to a sine wave about the first axis with a period equal to the second arm. However, by studying such a 2D-simplification, it would not have been possible to construct a set of instructions that could be of any use to control an ROV. With the MFM, the authors have been able to construct and utilize 3D-dynamics. Together with PID-settings it is possible construct a set of instructions that update the motors of an ROV to prevent any rotation induced by motion of the arms.

REFERENCES

[1] I. Oceaneering. (2016, 03.04.2017). *ROV Systems*. Available: http://www.oceaneering.com/rovs/rov-systems/

[2] F. Driscoll, R. Lueck, and M. Nahon, "The motion of a deep-sea remotely operated vehicle system Part 1: Motion observations", (in English), *Ocean Engineering,* Article vol. 27, no. 1, pp. 29-56, JAN 2000 2000.

[3] Frankel, T., 2012, The Geometry of Physics, an Introduction, third edition, Cambridge University Press, New York; First edition published in 1997.

[4] A Theoretical and Numerical Study of the Dzhanibekov and Tennis Racket Phenomena. *Journal of applied mechanics* 2016; Volume 83, No. 11.

[5] J. Denavit and R. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," vol. 22, pp. 215-221

[6] H. Murakami, "A moving frame method for multi-body dynamics using SE(3)", presented at the IMECE2015, Houston, Texas, USA, 2015.

[7] C. A. Barker. (2017, 12.02.2017). *Numerical Methods for*

Solving Differential Equations, The Runge-Kutta Method, Theoretical Introduction. Available: http://calculuslab.deltacollege.edu/ODE/7-C-3/7-C-3-h.html

ANNEX A

The following equations gives angular velocity about the first, second and third axis, respectively. Admittedly, this was an overwrought attempt—to obtain expressions for the three angular velocities in this form. In future work, it would be best to simply conduct a numerical integration directly with Eq. (81-83) or just with Eq. (80) alone.

$\boxed{\dot{\omega}^{(1)}(t)}$ = (2 * J31 * w3 * sin(th1t(t)) * sin(th2t(t))^2 * dth2t(t))/(J11 + J31) − (J31 * w1 * sin(2 * th2t(t)) * dth2t(t))/(J11 + J31) − (J31 * w2 * w3 * (cos(th2t(t))^2 1))/(J11 + J31) − (J31 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)) * dth1t(t)^2)/(J11 + J31) − (J31 * w2 * w3 * cos(th1t(t))^2 * cos(th2t(t))^2)/(J11 + J31) − (J31 * cos(th1t(t)) * ddth2t(t))/(J11 + J31) − (2 * J31 * w2 * cos(th1t(t))^2 * cos(th2t(t))^2 * dth1t(t))/(J11 + J31) + (2 * J31 * sin(th1t(t)) * sin(th2t(t))^2 * dth1t(t) * dth2t(t))/(J11 + J31) + (J31 * w2^2 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)))/(J11 + J31) − (J31 * w3^2 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)))/(J11 + J31) − (J31 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)) * ddth1t(t))/(J11 + J31) + (J31 * w1 * w3 * cos(th1t(t)) * cos(th2t(t))^2 * sin(th1t(t)))/(J11 + J31) − (2 * J31 * w3 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)) * dth1t(t))/(J11 + J31) + (2 * J31 * w1 * cos(th1t(t)) * cos(th2t(t))^2 * sin(th1t(t)) * dth1t(t))/(J11 + J31) + (2 * J31 * w1 * cos(th1t(t))^2 * cos(th2t(t)) * sin(th2t(t)) * dth2t(t))/(J11 + J31) − (J31 * w1 * w2 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)))/(J11 + J31) + (2 * J31 * w2 * cos(th1t(t)) * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)) * dth2t(t))/(J11 + J31)

$\boxed{\dot{\omega}^{(2)}(t)}$ = (2 * J31 * w1 * w3 * cos(th2t(t))^2)/(J11 + J31) − (J31 * w1 * w3)/(J11 + J31) − (J31 * sin(th1t(t)) * ddth2t(t))/(J11 + J31) − (2 * J31 * w3 * cos(th1t(t)) * dth2t(t))/(J11 + J31) + (2 * J31 * w1 * cos(th2t(t))^2 * dth1t(t))/(J11 + J31) − (2 * J31 * cos(th1t(t)) * dth1t(t) * dth2t(t))/(J11 + J31) − (J31 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)) * dth1t(t)^2)/(J11 + J31) − (J31 * w1 * w3 * cos(th1t(t))^2 * cos(th2t(t))^2)/(J11 + J31) + (2 * J31 * w3 * cos(th1t(t)) * cos(th2t(t))^2 * dth2t(t))/(J11 + J31) − (2 * J31 * w1 * cos(th1t(t))^2 * cos(th2t(t))^2 * dth1t(t))/(J11 + J31) + (2 * J31 * cos(th1t(t)) * cos(th2t(t))^2 * dth1t(t) * dth2t(t))/(J11 + J31) + (J31 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)) * ddth1t(t))/(J11 + J31) + (J31 * w1^2 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)))/(J11 + J31) − (J31 * w3^2 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)))/(J11 + J31) − (J31 * w2 * w3 * cos(th1t(t)) * cos(th2t(t))^2 * sin(th1t(t)))/(J11 + J31) − (2 * J31 * w3 * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)) * dth1t(t))/(J11 + J31) − (2 * J31 * w2 * cos(th1t(t)) * cos(th2t(t))^2 * sin(th1t(t)) * dth1t(t))/(J11 + J31) − (2 * J31 * w2 * cos(th1t(t))^2 * cos(th2t(t)) * sin(th2t(t)) * dth2t(t))/(J11 + J31) − (J31 * w1 * w2 * cos(th1t(t)) * cos(th2t(t)) * sin(th2t(t)))/(J11 + J31) + (2 * J31 * w1 * cos(th1t(t)) * cos(th2t(t)) * sin(th1t(t)) * sin(th2t(t)) * dth2t(t))/(J11 + J31)

$\boxed{\dot{\omega}^{(3)}(t)}$ = (J31 * cos(th2t(t)) * (2 * w3 * sin(th2t(t)) * dth2t(t) − cos(th2t(t)) * ddth1t(t) + 2 * sin(th2t(t)) * dth1t(t) * dth2t(t) − w1 * w2 * cos(th2t(t)) + w2 * w3 * sin(th1t(t)) * sin(th2t(t)) − w1^2 * cos(th1t(t)) * cos(th2t(t)) * sin(th1t(t)) + w2^2 * cos(th1t(t)) * cos(th2t(t)) * sin(th1t(t)) + 2 * w1 * w2 * cos(th1t(t))^2 * cos(th2t(t)) + 2 * w2 * cos(th1t(t)) * cos(th2t(t)) * dth2t(t) − 2 * w1 * cos(th2t(t)) * sin(th1t(t)) * dth2t(t) + w1 * w3 * cos(th1t(t)) * sin(th2t(t))))/(J11 + J31)