RECURSIVE NEWTON-EULER DYNAMICS AND SENSITIVITY ANALYSIS FOR DYNAMIC MOTION PLANNING

By

SHUVRODEB BARMAN

Bachelor of Science in Aeronautical Engineering

Bangladesh University of Professionals

Dhaka, Bangladesh

2016

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2020

RECURSIVE NEWTON-EULER DYNAMICS AND SENSITIVITY ANALYSIS FOR DYNAMIC MOTION PLANNING

Thesis Approved:

Yujiang Xiang

Thesis Adviser

He Bai

Rushikesh Kamalapurkar

ACKNOWLEDGEMENTS

Firstly, I would like to extend my deepest gratitude to my advisor Dr. Yujiang Xiang for his continuous support and motivation through each stage of the process.

I would also like to acknowledge Dr. He Bai and Dr. Rushikesh Kamalapurkar as the committee members of this thesis, and I am thankful to them for their valuable comments on this thesis.

I must thank the National Science Foundation (NSF) for partially supporting this research.

Finally, I am extremely grateful to my parents for providing me with support and continuous encouragement.

Name: SHUVRODEB BARMAN

Date of Degree: DECEMBER, 2020

Title of Study: RECURSIVE NEWTON-EULER DYNAMICS AND SENSITIVITY ANALYSIS FOR DYNAMIC MOTION PLANNING

Major Field: MECHANICAL AND AEROSPACE ENGINEERING

Abstract: In this study, sensitivity equations are derived for recursive Newton-Euler dynamics using Denavit-Hartenberg (DH) moving coordinates. The dynamics and sensitivity equations depend on the 3×3 DH rotation matrices. Compared to recursive Lagrangian formulation, which depends on 4×4 DH transformation matrices, the recursive Newton-Euler formulation requires less computational effort. In addition, recursive Newton-Euler dynamics explicitly calculates internal joint forces that are not available in recursive Lagrangian formulation. The proposed formulation can handle both prismatic joints and revolute joints. More importantly, analytical sensitivities provide aid to dynamic motion predictions. Three numerical examples are presented to verify the efficacy of the proposed algorithms including two time-minimization trajectory-planning problems and one gait prediction problem. The example model setup, optimization formulation, and simulation results are presented. All three examples are successfully optimized using the proposed dynamics and sensitivity equations. The predicted kinematic and kinetic profiles are partially validated with the data in the literature. Finally, the recursive Newton-Euler dynamics and sensitivity equations are programmed using C++ with the latest math library (Eigen) for developing a general purpose motion prediction software.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

NOMENCLATURE

Symbol	Meaning
${}^{ls}(.)_{rs}$	left superscript *ls* indicates which reference coordinate frame the (.) is referred to, right subscript *rs* represents the local reference frame
${}^{k}\boldsymbol{R}_{j}$	coordinate rotation matrix that maps the coordinates referred to the referential coordinate frame ${}^{j}x$, into the coordinates referred to the referential system ${}^{k}x$, located at the origin of link k reference frame
${}^{j-1}\boldsymbol{R}_{j} \equiv \boldsymbol{R}_{j}$	local rotation matrix for link j, for simplicity if the coordinate rotation matrix left superscript refers just to the prior frame the right subscript corresponds to, the superscript could be dropped
$\boldsymbol{I}_{3}$	3×3 identity matrix
$\boldsymbol{Z}_{3}$	3×3 zero matrix
${}^{j}\boldsymbol{\omega}_{j-1}$	the local angular velocity of frame (j-1) in reference frame j
${}^{j}\dot{\boldsymbol{\omega}}_{j-1}$	the local angular acceleration of frame (j-1) in reference frame j
$\boldsymbol{e}_{z_{j-1}}$	$[0 \quad 0 \quad 1]^{T}$
$\Delta\boldsymbol{\omega}_{j}$	delta angular velocity in jth coordinate that gives the difference between current joint angular velocity and prior joint angular velocity

Symbol	Meaning
$\Delta\dot{\boldsymbol{\omega}}_j$	delta angular acceleration in jth coordinate that gives the difference between current joint angular acceleration and prior joint angular acceleration
$\boldsymbol{\omega}_j$	local angular velocity of link j in local jth coordinate
$\dot{\boldsymbol{\omega}}_j$	local angular acceleration for link j in local jth coordinate
${}^{j}\boldsymbol{x}_{j-1}$	the local joint position of frame (j-1) in reference frame j
${}^{j}\dot{\boldsymbol{x}}_{j-1}$	the local joint translational velocity of frame (j-1) in reference frame j
${}^{j}\ddot{\boldsymbol{x}}_{j-1}$	the local joint translational acceleration of frame (j-1) in reference frame j
$\Delta\boldsymbol{x}_j$	delta position in jth coordinate that gives the difference between current joint position and prior joint position
$\Delta\dot{\boldsymbol{x}}_j$	delta translational velocity in jth coordinate that gives the difference between current joint translational velocity and prior joint translational velocity
$\Delta\ddot{\boldsymbol{x}}_j$	delta translational acceleration in jth coordinate that gives the difference between current joint translational acceleration and prior joint translational acceleration
$\Delta\boldsymbol{x}_{Gj}$	delta position of center of mass in jth coordinate that gives the difference between current link center of mass and prior link center of mass
$\Delta\dot{\boldsymbol{x}}_{Gj}$	delta translational velocity of center of mass in jth coordinate that gives the difference between current link translational velocity of center of mass and prior link translational velocity of center of mass

Symbol	Meaning
$\Delta\ddot{\boldsymbol{x}}_{Gj}$	delta translational acceleration of center of mass in jth coordinate that gives the difference between current link translational acceleration of center of mass and prior link translational acceleration of center of mass
$\boldsymbol{x}_j$	jth joint position in jth moving coordinate
$\dot{\boldsymbol{x}}_j$	jth joint translational velocity in jth moving coordinate
$\ddot{\boldsymbol{x}}_j$	jth joint translational acceleration in jth moving coordinate
$\boldsymbol{x}_{Gj}$	position of center of mass of link j in jth moving coordinate
$\dot{\boldsymbol{x}}_{Gj}$	translational velocity of center of mass of link j in jth moving coordinate
$\ddot{\boldsymbol{x}}_{Gj}$	translational acceleration of center of mass of link j in jth moving coordinate
${}^0\boldsymbol{R}_j$	global rotation matrix for link j
$\boldsymbol{F}_{Gj}$	inertial force acting on link j
$\boldsymbol{H}_j$	angular momentum of link j
$\boldsymbol{\tau}_{Gj}$	inertial torque acting on link j
$\boldsymbol{F}_{jd}$	internal force that link (j+1) exerts on link j (distal reaction of link j)
$\boldsymbol{\tau}_{jd}$	internal torque that link (j+1) exerts on link j (distal reaction of link j)
$\boldsymbol{F}_{jp}$	internal force that link j exerts on link (j-1) (proximal reaction of link j)
$\boldsymbol{\tau}_{jp}$	internal torque that link j exerts on link (j-1) (proximal reaction of link j)
${}^0\boldsymbol{F}_k^e$	global external force applied at link k
${}^0\boldsymbol{\tau}_k^e$	global external moment applied at link k
$\boldsymbol{F}_{jd}^e$	distal reaction force of link j due to global external force

Symbol	Meaning
$\boldsymbol{\tau}_{jd}^{ef}$	distal torque of link j due to global external force
$\boldsymbol{\tau}_{jd}^{em}$	distal torque of link j due to global external moment
τ_j	actuating torque component about the local ${}^{j-1}z$ axis
f_j	actuating force component about the local ${}^{j-1}z$ axis
$(.)^T$	transpose of $(.)$
$(.)_i$	sensitivity for $(.)$ with respect to q_i
$(.)_{i'}$	sensitivity for $(.)$ with respect to $\dot{q}_i$
$(.)_{i''}$	sensitivity for $(.)$ with respect to $\ddot{q}_i$
$\boldsymbol{R}_{j,j}$	derivative of local rotation matrix for link j with respect to q_j

CHAPTER I

INTRODUCTION

Various methods available in the literature for human motion study rely heavily on experiments. In contrast, dynamic motion prediction is a popular approach which only uses limited experimental data (Sohl and Bobrow, 2001; Lo et al., 2002; De Groote et al., 2016; Bj*ö*rkenstam et al., 2018; Xiang, 2019). Optimization-based motion prediction has wide applications in both robotics and biomechanics fields. For example, it was used for robot motion planning such as welding robot, assembling robot, and rescue robot. In biomechanics field, motion prediction is used to study pathological gait, surgical planning, and rehabilitation device design. In the optimization process, equations of motion (EOM) must be considered. Furthermore, development of sensitivity expressions of EOM with respect to the state variables is needed for gradient-based optimization to solve the problem efficiently and accurately (Xiang et al., 2009a).

The recursive dynamics formulations show computational advantages over the traditional non-recursive formulations (Fu et al., 1987; Shabana, 2010). Hollerbach (1980) and Toogood (1989) presented early work for recursive Lagrangian dynamics formulation. In contrast, Luh et al. (1980) and Featherstone (1987) demonstrated recursive Newton-Euler dynamics formulation. Both these formulations require linear order of $O(n)$ calculations, and recursive Newton-Euler formulation is more efficient than the recursive Lagrangian formulation. Fu et al. (1987) gave a

good summary and illustration of different recursive dynamics formulations for robot manipulators. Recently, screw and Lie groups theory were used to build the recursive Newton-Euler dynamics equations (Müller, 2018). With the development of large-scale optimization software (Gill et al. 2002), sensitivity analysis for multibody dynamic system is becoming more important for dynamic motion planning. One efficient method for sensitivity analysis is to use symbolic automatic differentiation tool (Patterson et al., 2013; De Groote et al., 2016). The main idea of symbolic sensitivity analysis is straightforward, but the internal calculation is a black box to the user. Analytical sensitivity equations provide an alternative way for sensitivity analysis. Although the number of sensitivity equations are usually large, they can be derived systematically to obtain insight knowledge about the dynamics problem structure within the algorithm. Analytical sensitivities have been studied for recursive Lagrangian formulation, recursive Newton-Euler formulation, and recursive Kane's equations. The different sensitivity formulations have different coordinates for EOM and the corresponding math operations are also different. Björkenstam et al. (2018) developed the recursive Lagrangian sensitivity equations based on Lie groups for inverse dynamic optimization of a box lifting motion. Xiang et al. (2009a) derived sensitivity equations for recursive Lagrangian dynamics using DH method, and later these sensitivities were used to predict human walking (Xiang et al., 2009b; Xiang, 2019) and lifting (Xiang et al., 2010, 2020). Sohl and Bobrow (2001), and Lee et al. (2005) proposed sensitivity analysis for recursive Newton-Euler dynamics using Lie groups and Lie algebras, and the motion of a branched under actuated robot manipulator was optimized. Similar recursive Newton-Euler method was investigated by Lo et al. (2002) to predict human motions. Anderson and Hsu (2002) also developed recursive sensitivity equations using a velocity-based projection coordinate method, recursive Kane's equations for multibody dynamic system.

Various optimization formulations have been studied in the literature. Effect of different objective functions on maximizing the smoothness of the predicted motion pattern was investigated by Hsiang and McGorry (1997). Arjmand and Shirazi-Adl (2006) developed musculoskeletal models with the objective to calculate trunk muscle forces and joint internal loads by simultaneous consideration of recorded kinematics and equilibrium of joint loads. The multi-objective optimization (MOO) approach using a weighted sum method has been implemented for lifting motion prediction over the single objective optimization formulation with less prediction errors (Chang et al., 2001; Xiang et al., 2010). Song et al. (2016) showed about 18.9% improvement in motion prediction using MOO method. Xiang et al. (2012) studied four objective functions for lifting motion prediction including the dynamic effort, the balance criterion, the maximum shear force at spine joint and the maximum pressure force at spine joint. Minimization of muscle forces was used by Pedotti et al. (1978), later a general two-level optimization procedure for tuning a multi-joint kinematic model to a patient's experimental movement data was proposed by Reinbolt et al. (2005). An outer level optimization modifies the model's parameters (joint position and orientations) and repeated inner level optimizations modify the model's degrees of freedom given the current parameters, with the goal of minimizing errors between model and experimental marker trajectories. The predictive dynamic approach was applied to asymmetric human walking prediction (Xiang et al., 2011), and the sensitivities of the system with respect to state variables were calculated. In addition, the gait analysis results were sensitive to the locations of the external forces (Silva et al., 2004). Ardestani et al. (2015) studied sensitivity of joint moments with respect to the variations of joint kinematics, highlighting the cause and effect of joint kinematics and the resultant joint moments which can lead to optimization of current rehabilitation treatments. Fourier Series (FS) function and Genetic Algorithm (GA) based optimization formulations are not limited by kinematics, so they can be used with various motion predictions (Yang et al., 2006).

Although different algorithms for sensitivity analysis of dynamic equations have been studied (Sohl and Bobrow, 2001; Lee et al., 2005; Anderson and Hsu, 2002; Xiang et al., 2009a; Björkenstam et al., 2018), limited work is found for inverse recursive Newton-Euler formulation with sensitivity analysis using DH moving coordinates. By using 3×3 DH rotation matrices and moving coordinates (Fu et al., 1987), recursive Newton-Euler formulation can not only efficiently and conveniently calculate joint actuating torques and forces but also the internal forces at the joints, a feature not presenting in recursive Lagrangian formulation (Xiang et al, 2009a). In addition, the recursive Newton-Euler formulation is based mostly on the calculation of vector products, which alleviates the computational complexity of multiplying numerous 4×4 DH transformation matrices for the recursive Lagrangian formulation (Fu et al., 1987).

In this study, the general sensitivity algorithms for recursive Newton-Euler dynamic equations are presented. In the recursive formulation, the system kinematics propagates forward along the branch and the kinetics transfers back from end-effector to the origin. The developed sensitivities are used to solve the dynamic motion optimization problems for multibody dynamic system. Three numerical examples are presented to show the validity of the proposed algorithms. The first example is to minimize the total travel time for a two-link manipulator with revolute joints by given initial and final configurations subject to torque limits. The second example is a multi-objective optimization (MOO) for a two-link manipulator with one prismatic and one revolute joint. The total travel time, actuating force, and actuating torque are minimized simultaneously. Both prismatic and revolute joints are tested for this simple two-link manipulator. After that, the proposed sensitivity equations are applied to a complicated gait problem using a 55 degree-of-freedom (DOF) skeletal model. For these three examples, the acquired optimal travel time,

kinematics, and kinetics are compared with those in the literature. Finally, internal forces at different joints are also retrieved from the simulation, a benefit for Newton-Euler formulation. The general algorithm of moving coordinates (Fu et al., 1987) performs recursive forward kinematics and backward dynamics for a serial link system by using the Newton-Euler dynamics model. According to this model, the dynamics equilibrium is performed for each link at a time as a free body where appropriate forces and torques are introduced at the joints. Therefore, the interaction forces between adjacent links are accounted for, which are however not obtained by using the Lagrange's model.

CHAPTER II

GENERAL ALGORITHM WITH MOVING COORDINATES

This chapter discusses the general algorithm used for recursive Newton-Euler dynamics. One of the steps in EOM formulation for a multibody system is the choice of kinematics formalism. This task is twofold: selection of the reference frame and type of mathematical description. The coordinate reference frame may be selected between a global or base reference frame and local or moving reference frames. The mathematical description may be by 4×4 transformation matrices, or by position vectors and 3×3 rotation matrices, or by position vectors and angular velocity vectors. Another step is the choice of the dynamics formulation: Lagrangian dynamics or Newton-Euler dynamics. The general algorithm of moving coordinates (Fu et al., 1987) performs recursive forward kinematics and backward dynamics for a serial link system by using the Newton-Euler dynamics model. According to this model, the dynamics equilibrium is performed for each link at a time as a free body where appropriate forces and torques are introduced at the joints. Therefore, the interaction forces between adjacent links are accounted for, which are however not obtained by using the Lagrange's model.

Figure 2.1 shows free body diagram of a link, where the inertial force and toque at the center of mass are represented by $\boldsymbol{F}_{Gj}$ and $\boldsymbol{\tau}_{Gj}$ respectively, and the reaction forces and torques at the proximal and distal joints are represented by $\boldsymbol{F}_p$ and $\boldsymbol{\tau}_p$, and $\boldsymbol{F}_d$ and $\boldsymbol{\tau}_d$, respectively.

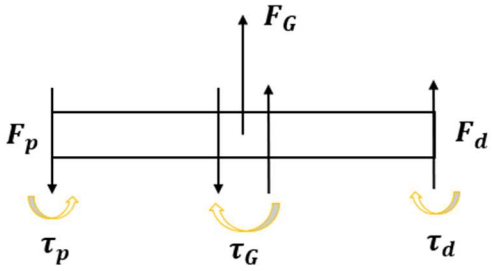


Figure 2.1: Free body diagram of a link

The equilibrium forces can be expressed as:

$$\boldsymbol{F}_p = \boldsymbol{F}_G + \boldsymbol{F}_d \tag{2.1}$$

Taking moments relatively to the left (proximal) end, we have the torque equilibrium as

$$\boldsymbol{\tau}_p = \boldsymbol{\tau}_G + \boldsymbol{\tau}_d - \boldsymbol{F}_G \times (\boldsymbol{x}_G - \boldsymbol{x}_p) - \boldsymbol{F}_d \times (\boldsymbol{x}_d - \boldsymbol{x}_p) \tag{2.2}$$

where $(\boldsymbol{x}_G - \boldsymbol{x}_p)$ is the vector from the proximal end to the center of mass and $(\boldsymbol{x}_d - \boldsymbol{x}_p)$ is the vector from the proximal end to the distal end.

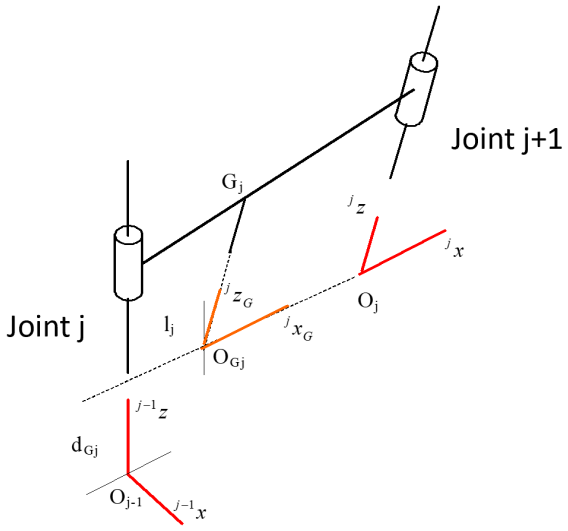


Figure 2.2: Joint coordinate systems and center of mass referential frame

The local coordinates are set to the end of each link such that the ^{j-1}z axis lies along the axis of motion of joint $j+1$, the ^{j}x axis is normal to the ^{j-1}z axis pointing away from it. The Denavit-

Hartenberg (D-H) representation results in 4×4 homogeneous transformation matrices that describe the translational and rotational relationships between joint j and joint $j+1$ in Figure 2.2, which can be defined as:

$$^{j-1}\boldsymbol{T}_j = \boldsymbol{T}_{z,\theta}\,\boldsymbol{T}_{z,d}\,\boldsymbol{T}_{x,a}\,\boldsymbol{T}_{x,\alpha} \tag{2.3}$$

where $\boldsymbol{T}_{z,\theta}$ represents the transformation matrix due to the rotation angle θ_i from ^{j-1}x axis to the ^{j}x axis about the ^{j-1}z axis; $\boldsymbol{T}_{z,d}$ represents the transformation matrix due to the distance d_i from the origin of ^{j-1}x axis to the intersection of the ^{j-1}z axis and ^{j}x axis along the ^{j-1}z axis; $\boldsymbol{T}_{x,a}$ represents the transformation matrix due to the offset distance a_i from the intersection of the ^{j-1}z axis and ^{j}x axis to the origin of ^{j}x axis to the along the ^{j}x axis and $\boldsymbol{T}_{x,\alpha}$ represents the transformation matrix due to the offset angle α_i from ^{j-1}z axis to the ^{j}z axis about the ^{j}x axis. These transformation matrices can be mathematically given by:

$$\boldsymbol{T}_{z,\theta} = \begin{bmatrix} \cos\theta_j & -\sin\theta_j & 0 & 0 \\ \sin\theta_j & \cos\theta_j & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$\boldsymbol{T}_{z,d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$\boldsymbol{T}_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a_j \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

$$\boldsymbol{T}_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_j & -\sin\alpha_j & 0 \\ 0 & \sin\alpha_j & \cos\alpha_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

They can then be multiplied together to get the homogeneous transformation matrix expression.

$$ {}^{j-1}\boldsymbol{T}_j = \begin{bmatrix} \cos\theta_j & -\cos\alpha_j \sin\theta_j & \sin\alpha_j \sin\theta_j & a_j \cos\theta_j \\ \sin\theta_j & \cos\alpha_j \cos\theta_j & -\sin\alpha_j \cos\theta_j & a_j \sin\theta_j \\ 0 & \sin\alpha_j & \cos\alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 2.8 $$

Here, $\theta_j, d_j, a_j, \alpha_j$ are also known as DH parameters, the first two determine relative positions between adjacent links and the second two determine the structure of the link. The upper left 3×3 submatrix represents the rotation matrix and the upper right 3×1 submatrix represents the position vector of the origin of the rotated coordinate system with respect to the reference system;. Here, the 3×3 rotation matrix is defined as a transformation matrix that operates on a position vector in a three dimensional Euclidean space and maps its coordinate in a rotated coordinates system to a reference coordinates system.

The DH coordinates of the center of mass are represented as $\theta_j, d_{Gj}, l_j, \alpha_j$ in Figure 2.2. The corresponding DH transformation matrix for the center of mass for the j-th link is expressed similarly as follows:

$$ {}^{j-1}\boldsymbol{T}_{Gj} = \begin{bmatrix} \cos\theta_j & -\cos\alpha_j \sin\theta_j & \sin\alpha_j \sin\theta_j & l_j \cos\theta_j \\ \sin\theta_j & \cos\alpha_j \cos\theta_j & -\sin\alpha_j \cos\theta_j & l_j \sin\theta_j \\ 0 & \sin\alpha_j & \cos\alpha_j & d_{Gj} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 2.9 $$

where the parameter l_j is the distance, along the ${}^{j}x$ direction, from the intersection of ${}^{j}x_G$ and ${}^{j-1}z$ to the origin O_{Gj} (perpendicular distance between ${}^{j}z_G$ and ${}^{j-1}z$) as shown in Figure 2.2. The parameter d_{Gj} is the distance from O_{j-1} to the intersection of ${}^{j-1}z_G$ and ${}^{j}x$ (perpendicular distance between ${}^{j}x_G$ and ${}^{j-1}x$) as shown in Figure 2.2.

Then the split is done for the coordinate transformation matrix of link j into local rotation matrix and position of origin of reference frame j relatively to the reference frame $(j-1)$. After that, the EOM are described in terms of local moving coordinate frames in order to have the inertia moments of the links independent of the links position. Also, the basic calculations are mostly

3×1 vector products, computationally more efficient than 4×4 matrix products in recursive Lagrangian formulation.

$$\boldsymbol{R}_j = \begin{bmatrix} \cos\theta_j & -\cos\alpha_j \sin\theta_j & \sin\alpha_j \sin\theta_j \\ \sin\theta_j & \cos\alpha_j \cos\theta_j & -\sin\alpha_j \cos\theta_j \\ 0 & \sin\alpha_j & \cos\alpha_j \end{bmatrix} \tag{2.10}$$

From Eqs. 2.8 and 2.10 we get,

$$^{j-1}\boldsymbol{x}_j = [a_j \cos\theta_j \quad a_j \sin\theta_j \quad d_j]^T \tag{2.11}$$

$$^{j-1}\boldsymbol{x}_{Gj} = [l_j \cos\theta_j \quad l_j \sin\theta_j \quad d_{Gj}]^T \tag{2.12}$$

In the formulation, for quantities as position, velocity and acceleration, as well as forces and torques, if the left superscript is same as the right subscript, then the superscript may be dropped.

This chapter discussed the basics of setting up DH coordinates for robot manipulators. The general expressions for setting up the corresponding transformation and rotation matrices, position vectors are explained. The following chapter gives the detailed Newton-Euler formulation along with sensitivity analysis.

CHAPTER III

SENSITIVITY ANALYSIS FOR RECURSIVE NEWTON-EULER EQUATIONS

In this chapter, the detailed formulation for recursive Newton-Euler equations is presented. First the state variables, DH parameters, external forces and moments, gravity, position, translational and angular velocity and acceleration, and unit vectors are initialized. Then, there is the recursive formulation. During the forward kinematics calculation, the kinematics terms are computed with respect to the next reference frame, hence propagating forward. The sequence for the forward recursive kinematics formulation along with the sensitivities for both revolute and prismatic joint is given through steps (1-8). The backward dynamics calculations are given in steps (9, 10).

Step 1

a. The starting state variables $q_j, \dot{q}_j, \ddot{q}_j$ $(j = 1, 2, 3, \dots, n)$ are initialized for the links;
b. The system DH parameters, $\theta_j, d_j, a_j, \alpha_j$, including the location of the centers of mass, l_j, d_{Gj} of the links $(j = 1, 2, 3, \dots, n)$ are set up;
c. External forces and moments ${}^0\boldsymbol{F}_k^e, {}^0\boldsymbol{\tau}_k^e$ acting on the kth link referred to the global coordinate frame are set up;
d. Gravity ${}^0\boldsymbol{g}_j = \begin{bmatrix} {}^0g_x & {}^0g_y & {}^0g_z \end{bmatrix}^T$ acting on the links $(j = 1, 2, 3, \dots, n)$ referred to the global coordinate frame are set up;

e. The position, translational and angular velocity, acceleration for the base link are set up as $\boldsymbol{x}_0 = \mathbf{0},\ \dot{\boldsymbol{x}}_0 = \mathbf{0},\ \ddot{\boldsymbol{x}}_0 = \mathbf{0},\ \boldsymbol{\omega}_0 = \mathbf{0},\ \dot{\boldsymbol{\omega}}_0 = \mathbf{0}$;

f. Local rotation matrix and its derivatives for the base link are set up as $\boldsymbol{R}_0 = \boldsymbol{I}_3, \dot{\boldsymbol{R}}_0 = \boldsymbol{Z}_3, \ddot{\boldsymbol{R}}_0 = \boldsymbol{Z}_3$;

g. Define local unit vector $\boldsymbol{e}_{z_{j-1}}$ in the jth coordinate $\boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} = [0 \quad \sin\alpha_j \quad \cos\alpha_j]^T$;

h. Prismatic joint specific parameters were set up as:

$$\boldsymbol{R}_{j,i}^T = \mathbf{0}$$
$$d_{Gj,i} = 1$$
$$\dot{d}_{Gj,i'} = 1$$
$$\ddot{d}_{Gj,i''} = 1 \qquad (3.1)$$
$$\dot{d}_j \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} = \dot{d}_j [0 \quad \sin\alpha_j \quad \cos\alpha_j]^T$$
$$\dot{d}_{Gj} \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} = \dot{d}_{Gj} [0 \quad \sin\alpha_j \quad \cos\alpha_j]^T$$

Step 2

a. Calculate and store local rotation matrices and their derivatives for all the links $(j = 1, 2, 3, \ldots, n)$;

b. Now, calculate and store the delta position for all the links $(j = 1, 2, 3, \ldots, n)$ along with their sensitivities;

$$\Delta \boldsymbol{x}_{Gj} = \boldsymbol{R}_j^{T\,j-1} \boldsymbol{x}_{Gj} = [l_j \quad d_{Gj} \sin\alpha_j \quad d_{Gj} \cos\alpha_j]^T \qquad (3.2)$$

Revolute Joint

$$\Delta \boldsymbol{x}_{Gj,i} = \mathbf{0}$$
$$\Delta \boldsymbol{x}_{Gj,i'} = \mathbf{0} \qquad (3.3)$$

$$\Delta \boldsymbol{x}_{Gj,i''} = \mathbf{0}$$

Prismatic Joint

$$\Delta \boldsymbol{x}_{Gj,i} = \begin{cases} \mathbf{0}\,; when\ i < j \\ d_{Gj,i} \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}}\,; when\ i = j \end{cases}$$

$$\Delta \boldsymbol{x}_{Gj,i'} = \mathbf{0} \qquad 3.4$$

$$\Delta \boldsymbol{x}_{Gj,i''} = \mathbf{0}$$

c. Calculate and store the delta angular velocity for all the links $(j = 1, 2, 3, \dots, n)$.

$$\Delta \boldsymbol{\omega}_j = \begin{cases} \dot{\theta}_j {\boldsymbol{R}_j}^T \boldsymbol{e}_{z_{j-1}}\,; for\ revolute\ joint \\ \mathbf{0}\,; for\ prismatic\ joint \end{cases} \qquad 3.5$$

Revolute Joint

$$\Delta \boldsymbol{\omega}_{j,i} = \mathbf{0}$$

$$\Delta \boldsymbol{\omega}_{j,i'} = \begin{cases} \mathbf{0}; & when\ i < j \\ {\boldsymbol{R}_j}^T \boldsymbol{e}_{z_{j-1}}; & when\ i = j \end{cases} \qquad 3.6$$

$$\Delta \boldsymbol{\omega}_{j,i''} = \mathbf{0}$$

Prismatic Joint

$$\Delta \boldsymbol{\omega}_{j,i} = \mathbf{0}$$

$$\Delta \boldsymbol{\omega}_{j,i'} = \mathbf{0} \qquad 3.7$$

$$\Delta \boldsymbol{\omega}_{j,i''} = \mathbf{0}$$

Step 3

a. Calculate and store the angular velocity of local frame $j-1$ with respect to the next reference frame, j for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$^{j}\boldsymbol{\omega}_{j-1} = \boldsymbol{R}_j^T \boldsymbol{\omega}_{j-1} \qquad 3.8$$

Revolute Joint

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{\omega}_{(j-1),i}; & when\ i < j \\ \boldsymbol{R}_{j,i}^T \boldsymbol{\omega}_{j-1}; & when\ i = j \end{cases}
$$

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{\omega}_{(j-1),i'}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases} \qquad 3.9
$$

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i''} = \mathbf{0}
$$

Prismatic Joint

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{\omega}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T \boldsymbol{\omega}_{j-1}\ ; when\ i = j \end{cases}
$$

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{\omega}_{(j-1),i'}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases} \qquad 3.10
$$

$$
{}^{j}\boldsymbol{\omega}_{(j-1),i''} = \mathbf{0}
$$

b. Calculate and store the angular velocity and its sensitivities for all the links $(j = 1, 2, 3, \dots, n)$.

$$
\boldsymbol{\omega}_j = {}^{j}\boldsymbol{\omega}_{j-1} + \Delta\boldsymbol{\omega}_j \qquad 3.11
$$

Revolute Joint

$$
\boldsymbol{\omega}_{j,i} = {}^{j}\boldsymbol{\omega}_{(j-1),i}
$$

$$
\boldsymbol{\omega}_{j,i'} = \begin{cases} {}^{j}\boldsymbol{\omega}_{(j-1),i'}\ ; & when\ i < j \\ \Delta\boldsymbol{\omega}_{j,i'}\ ; & when\ i = j \end{cases} \qquad 3.12
$$

$$
\boldsymbol{\omega}_{j,i''} = \mathbf{0}
$$

Prismatic Joint

$$
\boldsymbol{\omega}_{j,i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{\omega}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T \boldsymbol{\omega}_{j-1}\ ; when\ i = j \end{cases}
$$

$$
\boldsymbol{\omega}_{j,i'} = \mathbf{0} \qquad 3.13
$$

$$
\boldsymbol{\omega}_{j,i''} = \mathbf{0}
$$

Step 4

a. Calculate and store the angular acceleration of local frame $j-1$ with respect to the next - reference frame, j for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$ {}^{j}\dot{\boldsymbol{\omega}}_{j-1} = \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{j-1} \tag{3.14} $$

Revolute Joint

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i}; & when\ i < j \\ \boldsymbol{R}_{j,i}^T \dot{\boldsymbol{\omega}}_{(j-1)}; & when\ i = j \end{cases} $$

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i'}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases} \tag{3.15} $$

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i''} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i''}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases} $$

Prismatic Joint

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T \dot{\boldsymbol{\omega}}_{j-1}; when\ i = j \end{cases} $$

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i'}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases} \tag{3.16} $$

$$ {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i''} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{\omega}}_{(j-1),i''}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases} $$

b. Calculate and store the delta angular acceleration for all the links $(j = 1, 2, 3, \dots, n)$.

$$ \boldsymbol{\Delta\dot{\omega}}_j = \begin{cases} \ddot{\theta}_j \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} + {}^{j}\boldsymbol{\omega}_{j-1} \times \boldsymbol{\Delta\omega}_j\ ; for\ revolute\ joint \\ \mathbf{0}\ ; for\ prismatic\ joint \end{cases} \tag{3.17} $$

Revolute Joint

$$ \Delta\dot{\boldsymbol{\omega}}_{j,i} = {}^{j}\boldsymbol{\omega}_{(j-1),i} \times \Delta\boldsymbol{\omega}_j $$

$$ \Delta\dot{\boldsymbol{\omega}}_{j,i'} = \begin{cases} {}^{j}\boldsymbol{\omega}_{(j-1),i'} \times \Delta\boldsymbol{\omega}_j; & when\ i < j \\ {}^{j}\boldsymbol{\omega}_{(j-1)} \times \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}}; & when\ i = j \end{cases} \tag{3.18} $$

$$ \Delta\dot{\boldsymbol{\omega}}_{j,i''} = \begin{cases} \mathbf{0}; & when\ i < j \\ \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}}; & when\ i = j \end{cases} $$

Prismatic Joint

$$\Delta\dot{\boldsymbol{\omega}}_{j,i} = \mathbf{0}$$

$$\Delta\dot{\boldsymbol{\omega}}_{j,i'} = \mathbf{0} \qquad 3.19$$

$$\Delta\dot{\boldsymbol{\omega}}_{j,i''} = \mathbf{0}$$

c. Calculate and store the angular acceleration and its sensitivities for all the links $(j = 1, 2, 3, \dots, n)$.

$$\dot{\boldsymbol{\omega}}_j = {}^{j}\dot{\boldsymbol{\omega}}_{j-1} + \boldsymbol{\Delta\dot{\omega}}_j \qquad 3.20$$

Revolute Joint

$$\dot{\boldsymbol{\omega}}_{j,i} = {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i} + \Delta\dot{\boldsymbol{\omega}}_{j,i}$$

$$\dot{\boldsymbol{\omega}}_{j,i'} = \begin{cases} {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i'} + \Delta\dot{\boldsymbol{\omega}}_{j,i'}; & when\ i < j \\ \Delta\dot{\boldsymbol{\omega}}_{j,i'}; & when\ i = j \end{cases} \qquad 3.21$$

$$\dot{\boldsymbol{\omega}}_{j,i''} = \begin{cases} {}^{j}\dot{\boldsymbol{\omega}}_{(j-1),i''}; & when\ i < j \\ \Delta\dot{\boldsymbol{\omega}}_{j,i''}; & when\ i = j \end{cases}$$

Prismatic Joint

$$\dot{\boldsymbol{\omega}}_{j,i} = \begin{cases} \boldsymbol{R}_j^T\dot{\boldsymbol{\omega}}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T\dot{\boldsymbol{\omega}}_{j-1}\ ; when\ i = j \end{cases}$$

$$\dot{\boldsymbol{\omega}}_{j,i'} = \begin{cases} \boldsymbol{R}_j^T\dot{\boldsymbol{\omega}}_{(j-1),i'}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases} \qquad 3.22$$

$$\dot{\boldsymbol{\omega}}_{j,i''} = \begin{cases} \boldsymbol{R}_j^T\dot{\boldsymbol{\omega}}_{(j-1),i''}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases}$$

Step 5

a. Calculate and store the position of local frame $j-1$ with respect to the next reference frame, j for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$${}^{j}\boldsymbol{x}_{j-1} = \boldsymbol{R}_j^T\,\boldsymbol{x}_{j-1} \qquad 3.23$$

Revolute Joint

$$
{}^{j}\boldsymbol{x}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{x}_{(j-1),i}; & when\ \ i < j \\ \boldsymbol{R}_{j,i}^T\, \boldsymbol{x}_{j-1}; & when\ i = j \end{cases}
$$

$$
{}^{j}\boldsymbol{x}_{(j-1),i'} = \mathbf{0} \tag{3.24}
$$

$$
{}^{j}\boldsymbol{x}_{(j-1),i''} = \mathbf{0}
$$

Prismatic Joint

$$
{}^{j}\boldsymbol{x}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{x}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T \boldsymbol{x}_{j-1}\ ; when\ i = j \end{cases}
$$

$$
{}^{j}\boldsymbol{x}_{(j-1),i'} = \mathbf{0} \tag{3.25}
$$

$$
{}^{j}\boldsymbol{x}_{(j-1),i''} = \mathbf{0}
$$

b. Calculate the local position and their sensitivities for all the links $(j = 1, 2, 3, \dots, n)$;

$$
\boldsymbol{x}_j = {}^{j}\boldsymbol{x}_{j-1} + \Delta \boldsymbol{x}_j \tag{3.26}
$$

Revolute Joint

$$
\boldsymbol{x}_{j,i} = {}^{j}\boldsymbol{x}_{(j-1),i}
$$

$$
\boldsymbol{x}_{j,i'} = \mathbf{0} \tag{3.27}
$$

$$
\boldsymbol{x}_{j,i''} = \mathbf{0}
$$

Prismatic Joint

$$
\boldsymbol{x}_{j,i} = \begin{cases} \boldsymbol{R}_j^T \boldsymbol{x}_{(j-1),i}\ ; when\ i < j \\ \boldsymbol{R}_{j,i}^T \boldsymbol{x}_{j-1} + \Delta \boldsymbol{x}_{j,i}; when\ i = j \end{cases}
$$

$$
\boldsymbol{x}_{j,i'} = \mathbf{0} \tag{3.28}
$$

$$
\boldsymbol{x}_{j,i''} = \mathbf{0}
$$

c. Calculate the local position of the center of mass and their sensitivities for all the links $(j = 1, 2, 3, \dots, n)$;

$$
\boldsymbol{x}_{Gj} = {}^{j}\boldsymbol{x}_{j-1} + \Delta \boldsymbol{x}_{Gj} \tag{3.29}
$$

Revolute Joint

$$\boldsymbol{x}_{Gj,i} = {}^{j}\boldsymbol{x}_{(j-1),i}$$

$$\boldsymbol{x}_{Gj,i'} = \mathbf{0} \tag{3.30}$$

$$\boldsymbol{x}_{Gj,i''} = \mathbf{0}$$

Prismatic Joint

$$\boldsymbol{x}_{Gj,i} = {}^{j}\boldsymbol{x}_{(j-1),i} + \Delta\boldsymbol{x}_{Gj,i}$$

$$\boldsymbol{x}_{Gj,i'} = \mathbf{0} \tag{3.31}$$

$$\boldsymbol{x}_{Gj,i''} = \mathbf{0}$$

Step 6

a. Calculate and store the translational velocity of local frame $j-1$ with respect to the next reference frame, j for all the links $(j = 1, 2, 3, \ldots, n)$ along with their sensitivities;

$${}^{j}\dot{\boldsymbol{x}}_{j-1} = \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{j-1} \tag{3.32}$$

Revolute Joint

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{(j-1),i}; & when\ i < j \\ \boldsymbol{R}_{j,i}^T \dot{\boldsymbol{x}}_{j-1}; & when\ i = j \end{cases}$$

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{(j-1),i'}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases} \tag{3.33}$$

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i''} = \mathbf{0}$$

Prismatic Joint

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{(j-1),i}\ ;\ when\ i < j \\ \boldsymbol{R}_{j,i}^T \dot{\boldsymbol{x}}_{j-1}\ ; when\ i = j \end{cases}$$

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{(j-1),i'}\ ; when\ i < j \\ \mathbf{0}\ ; when\ i = j \end{cases} \tag{3.34}$$

$${}^{j}\dot{\boldsymbol{x}}_{(j-1),i''} = \mathbf{0}$$

b. Calculate and store the delta translational velocity for all the links ($j = 1, 2, 3, \dots, n$) along with their sensitivities;

$$\Delta\dot{\boldsymbol{x}}_j = \begin{cases} \boldsymbol{\omega}_j \times \Delta\boldsymbol{x}_j \,; for\ revolute\ joint \\ \boldsymbol{\omega}_j \times \Delta\boldsymbol{x}_j + \dot{d}_j \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} \,; for\ prismatic\ joint \end{cases} \tag{3.35}$$

Revolute Joint

$$\begin{gathered} \Delta\dot{\boldsymbol{x}}_{j,i} = \boldsymbol{\omega}_{j,i} \times \Delta\boldsymbol{x}_j \\ \Delta\dot{\boldsymbol{x}}_{j,i'} = \boldsymbol{\omega}_{j,i'} \times \Delta\boldsymbol{x}_j \\ \Delta\dot{\boldsymbol{x}}_{j,i''} = \mathbf{0} \end{gathered} \tag{3.36}$$

Prismatic Joint

$$\begin{gathered} \Delta\dot{\boldsymbol{x}}_{j,i} = \begin{cases} \boldsymbol{\omega}_{j,i} \times \Delta\boldsymbol{x}_j \,; when\ i < j \\ \boldsymbol{\omega}_j \times \Delta\boldsymbol{x}_{j,i} \,; when\ i = j \end{cases} \\ \Delta\dot{\boldsymbol{x}}_{j,i'} = \begin{cases} \boldsymbol{\omega}_{j,i'} \times \Delta\boldsymbol{x}_j \,; when\ i < j \\ \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} ; when\ i = j \end{cases} \\ \Delta\dot{\boldsymbol{x}}_{j,i''} = \mathbf{0} \end{gathered} \tag{3.37}$$

c. Calculate and store the translational velocity for all the links ($j = 1, 2, 3, \dots, n$) along with their sensitivities;

$$\dot{\boldsymbol{x}}_j = {}^j\dot{\boldsymbol{x}}_{j-1} + \Delta\dot{\boldsymbol{x}}_j \tag{3.38}$$

Revolute Joint

$$\begin{gathered} \dot{\boldsymbol{x}}_{j,i} = {}^j\dot{\boldsymbol{x}}_{(j-1),i} + \Delta\dot{\boldsymbol{x}}_{j,i} \\ \dot{\boldsymbol{x}}_{j,i'} = \begin{cases} {}^j\dot{\boldsymbol{x}}_{(j-1),i'} + \Delta\dot{\boldsymbol{x}}_{j,i'}; & when\ i < j \\ \Delta\dot{\boldsymbol{x}}_{j,i'}; & when\ i = j \end{cases} \\ \dot{\boldsymbol{x}}_{j,i''} = \mathbf{0} \end{gathered} \tag{3.39}$$

Prismatic Joint

$$\dot{\boldsymbol{x}}_{j,i} = \begin{cases} {}^j\dot{\boldsymbol{x}}_{(j-1),i} + \Delta\dot{\boldsymbol{x}}_{j,i} \,; when\ i < j \\ \Delta\dot{\boldsymbol{x}}_{j,i} \,; when\ i = j \end{cases} \tag{3.40}$$

$$\dot{\boldsymbol{x}}_{j,i'} = \begin{cases} \boldsymbol{R}_j^T \dot{\boldsymbol{x}}_{(j-1),i'} \; ; when \; i < j \\ \Delta \dot{\boldsymbol{x}}_{j,i'} \; ; when \; i = j \end{cases}$$

$$\dot{\boldsymbol{x}}_{j,i''} = \boldsymbol{0}$$

d. Calculate and store the delta velocity of the center of the mass for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\Delta \dot{\boldsymbol{x}}_{Gj} = \begin{cases} \boldsymbol{\omega}_j \times \Delta \boldsymbol{x}_{Gj} \; ; for \; revolute \; joint \\ \boldsymbol{\omega}_j \times \Delta \boldsymbol{x}_{Gj} + \dot{d}_{Gj} \boldsymbol{R}_j^T e_{z_{j-1}} \; ; for \; prismatic \; joint \end{cases} \quad 3.41$$

Revolute Joint

$$\Delta \dot{\boldsymbol{x}}_{Gj,i} = \boldsymbol{\omega}_{j,i} \times \Delta \boldsymbol{x}_{Gj}$$

$$\Delta \dot{\boldsymbol{x}}_{Gj,i'} = \boldsymbol{\omega}_{j,i'} \times \Delta \boldsymbol{x}_{Gj} \quad 3.42$$

$$\Delta \dot{\boldsymbol{x}}_{Gj,i''} = \boldsymbol{0}$$

Prismatic Joint

$$\Delta \dot{\boldsymbol{x}}_{Gj,i} = \begin{cases} \boldsymbol{\omega}_{j,i} \times \Delta \boldsymbol{x}_{Gj} \; ; when \; i < j \\ \boldsymbol{\omega}_j \times \Delta \boldsymbol{x}_{Gj,i} \; ; when \; i = j \end{cases}$$

$$\Delta \dot{\boldsymbol{x}}_{Gj,i'} = \begin{cases} \boldsymbol{\omega}_{j,i'} \times \Delta \boldsymbol{x}_{Gj} \; ; when \; i < j \\ \dot{d}_{Gj,i'} \boldsymbol{R}_j^T e_{z_{j-1}} \; ; when \; i = j \end{cases} \quad 3.43$$

$$\Delta \dot{\boldsymbol{x}}_{Gj,i''} = \boldsymbol{0}$$

e. Calculate and store the translational velocity of the center of the mass for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\dot{\boldsymbol{x}}_{Gj} = {}^{j}\dot{\boldsymbol{x}}_{j-1} + \Delta \dot{\boldsymbol{x}}_{Gj} \quad 3.44$$

Revolute Joint

$$\dot{\boldsymbol{x}}_{Gj,i} = {}^{j}\dot{\boldsymbol{x}}_{(j-1),i} + \Delta \dot{\boldsymbol{x}}_{Gj,i}$$

$$\dot{\boldsymbol{x}}_{Gj,i'} = \begin{cases} {}^{j}\dot{\boldsymbol{x}}_{(j-1),i'} + \Delta \dot{\boldsymbol{x}}_{Gj,i'}; & when \; i < j \\ \Delta \dot{\boldsymbol{x}}_{Gj,i'}; & when \; i = j \end{cases} \quad 3.45$$

$$\dot{\boldsymbol{x}}_{Gj,i''} = \boldsymbol{0}$$

Prismatic Joint

$$\dot{\boldsymbol{x}}_{Gj,i} = {}^{j}\dot{\boldsymbol{x}}_{(j-1),i} + \Delta\dot{\boldsymbol{x}}_{Gj,i}$$

$$\dot{\boldsymbol{x}}_{Gj,i'} = \begin{cases} {}^{j}\dot{\boldsymbol{x}}_{(j-1),i'} + \Delta\dot{\boldsymbol{x}}_{Gj,i'}\ ;\ when\ i < j \\ \Delta\dot{\boldsymbol{x}}_{Gj,i'}\ ;when\ i = j \end{cases} \qquad 3.46$$

$$\dot{\boldsymbol{x}}_{Gj,i''} = \mathbf{0}$$

Step 7

a. Calculate and store the translational acceleration of local frame $j-1$ with respect to the next reference frame, j for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$${}^{j}\ddot{\boldsymbol{x}}_{j-1} = \ \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{j-1} \qquad 3.47$$

Revolute Joint

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i}; & when\ i < j \\ \boldsymbol{R}_{j,i}^T\ddot{\boldsymbol{x}}_{j-1}; & when\ i = j \end{cases}$$

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i'}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases} \qquad 3.48$$

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i''} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i''}; & when\ i < j \\ \mathbf{0}; & when\ i = j \end{cases}$$

Prismatic Joint

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i}\ ;when\ i < j \\ \boldsymbol{R}_{j,i}^T\ddot{\boldsymbol{x}}_{j-1}\ ;when\ i = j \end{cases}$$

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i'} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i'}\ ;when\ i < j \\ \mathbf{0}\ ;when\ i = j \end{cases} \qquad 3.49$$

$${}^{j}\ddot{\boldsymbol{x}}_{(j-1),i''} = \begin{cases} \boldsymbol{R}_j^T\ddot{\boldsymbol{x}}_{(j-1),i''}\ ;when\ i < j \\ \mathbf{0}\ ;when\ i = j \end{cases}$$

b. Calculate and store the delta translational acceleration for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\Delta\ddot{\boldsymbol{x}}_j = \begin{cases} \dot{\boldsymbol{\omega}}_j \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_j \; ; for\ revolute\ joint \\ \dot{\boldsymbol{\omega}}_j \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_j + \boldsymbol{\omega}_j \times \dot{d}_j \boldsymbol{R}_j^{\boldsymbol{T}} \boldsymbol{e}_{z_{j-1}} + \ddot{d}_j \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} \; ; for\ prismatic\ joint \end{cases} \quad 3.50$$

Revolute Joint

$$\Delta\ddot{\boldsymbol{x}}_{j,i} = \dot{\boldsymbol{\omega}}_{j,i} \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_{j,i} \times \Delta\dot{\boldsymbol{x}}_j + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{j,i}$$

$$\Delta\ddot{\boldsymbol{x}}_{j,i'} = \dot{\boldsymbol{\omega}}_{j,i'} \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_{j,i'} \times \Delta\dot{\boldsymbol{x}}_j + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{j,i'} \quad 3.51$$

$$\Delta\ddot{\boldsymbol{x}}_{\boldsymbol{j,i''}} = \boldsymbol{\omega}_{\boldsymbol{j,i''}} \times \Delta\boldsymbol{x}_{\boldsymbol{j}}$$

Prismatic Joint

$$\Delta\ddot{\boldsymbol{x}}_{j,i} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i} \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_{j,i} \times \left(\Delta\dot{\boldsymbol{x}}_j + \dot{d}_j \boldsymbol{R}_j^T \boldsymbol{e}_{\boldsymbol{z_{j-1}}}\right) + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{j,i} \; ; when\ i < j \\ \dot{\boldsymbol{\omega}}_j \times \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{j,i} \; ; when\ i = j \end{cases}$$

$$\Delta\ddot{\boldsymbol{x}}_{j,i'} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i'} \times \Delta\boldsymbol{x}_j + \boldsymbol{\omega}_{j,i'} \times \left(\Delta\dot{\boldsymbol{x}}_j + \dot{d}_j \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}}\right) + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{j,i'} \; ; when\ i < j \\ 2\boldsymbol{\omega}_j \times \boldsymbol{R}_j^T \boldsymbol{e}_{\boldsymbol{z_{j-1}}} \; ; when\ i = j \end{cases} \quad 3.52$$

$$\Delta\ddot{\boldsymbol{x}}_{j,i''} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i''} \times \Delta\boldsymbol{x}_j \; ; when\ i < j \\ \boldsymbol{R}_j^T \boldsymbol{e}_{z_{j-1}} \; ; when\ i = j \end{cases}$$

c. Calculate and store the translational acceleration for all the links ($j = 1, 2, 3, \ldots, n$) along with their sensitivities;

$$\ddot{\boldsymbol{x}}_j = {}^j\ddot{\boldsymbol{x}}_{j-1} + \Delta\ddot{\boldsymbol{x}}_j \quad 3.53$$

Revolute Joint

$$\ddot{\boldsymbol{x}}_{j,i} = {}^j\ddot{\boldsymbol{x}}_{(j-1),i} + \Delta\ddot{\boldsymbol{x}}_{j,i}$$

$$\ddot{\boldsymbol{x}}_{j,i'} = \begin{cases} {}^j\ddot{\boldsymbol{x}}_{(j-1),i'} + \Delta\ddot{\boldsymbol{x}}_{j,i'}; & when\ i < j \\ \Delta\ddot{\boldsymbol{x}}_{j,i'}; & when\ i = j \end{cases} \quad 3.54$$

$$\ddot{\boldsymbol{x}}_{j,i''} = \begin{cases} {}^j\ddot{\boldsymbol{x}}_{(j-1),i''} + \Delta\ddot{\boldsymbol{x}}_{j,i''}; & when\ i < j \\ \Delta\ddot{\boldsymbol{x}}_{j,i''}; & when\ i = j \end{cases}$$

Prismatic Joint

$$\ddot{\boldsymbol{x}}_{j,i} = {}^j\ddot{\boldsymbol{x}}_{(j-1),i} + \Delta\ddot{\boldsymbol{x}}_{j,i} \quad 3.55$$

$$\ddot{\boldsymbol{x}}_{j,i'} = \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i'} + \Delta\ddot{\boldsymbol{x}}_{j,i'} \; ; when \; i < j \\ \Delta\ddot{\boldsymbol{x}}_{j,i'} \; ; when \; i = j \end{cases}$$

$$\ddot{\boldsymbol{x}}_{j,i''} = \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i''} + \Delta\ddot{\boldsymbol{x}}_{j,i''} \; ; when \; i < j \\ \Delta\ddot{\boldsymbol{x}}_{j,i''} \; ; when \; i = j \end{cases}$$

d. Calculate and store the delta translational acceleration of the center of mass for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\Delta\ddot{\boldsymbol{x}}_{Gj} = \begin{cases} \dot{\boldsymbol{\omega}}_j \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj} \; ; for \; revolute \; joint \\ \dot{\boldsymbol{\omega}}_j \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj} + \boldsymbol{\omega}_j \times \dot{d}_{Gj}\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}} + \ddot{d}_{Gj}\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}} \; ; for \; prismatic \; joint \end{cases} \quad 3.56$$

Revolute Joint

$$\begin{aligned} \Delta\ddot{\boldsymbol{x}}_{Gj,i} &= \dot{\boldsymbol{\omega}}_{j,i} \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_{j,i} \times \Delta\dot{\boldsymbol{x}}_{Gj} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj,i} \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i'} &= \dot{\boldsymbol{\omega}}_{j,i'} \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_{j,i'} \times \Delta\dot{\boldsymbol{x}}_{Gj} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj,i'} \quad 3.57 \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i''} &= \dot{\boldsymbol{\omega}}_{j,i''} \times \Delta\boldsymbol{x}_{Gj} \end{aligned}$$

Prismatic Joint

$$\Delta\ddot{\boldsymbol{x}}_{Gj,i} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i} \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_{j,i} \times \left(\Delta\dot{\boldsymbol{x}}_{Gj} + \dot{d}_j\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}}\right) + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj,i} \; ; when \; i < j \\ \dot{\boldsymbol{\omega}}_j \times \Delta\boldsymbol{x}_{Gj,i} + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj,i} \; ; when \; i = j \end{cases}$$

$$\Delta\ddot{\boldsymbol{x}}_{Gj,i'} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i'} \times \Delta\boldsymbol{x}_{Gj} + \boldsymbol{\omega}_{j,i'} \times \left(\Delta\dot{\boldsymbol{x}}_{Gj} + \dot{d}_j\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}}\right) + \boldsymbol{\omega}_j \times \Delta\dot{\boldsymbol{x}}_{Gj,i'} \; ; when \; i < j \\ 2\boldsymbol{\omega}_j \times \dot{d}_{Gj,i'}\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}} \; ; when \; i = j \end{cases} \quad 3.58$$

$$\Delta\ddot{\boldsymbol{x}}_{Gj,i''} = \begin{cases} \dot{\boldsymbol{\omega}}_{j,i''} \times \Delta\boldsymbol{x}_{Gj} \; ; when \; i < j \\ \ddot{d}_{Gj,i''}\boldsymbol{R}_j^T\boldsymbol{e}_{z_{j-1}} \; ; when \; i = j \end{cases}$$

e. Calculate and store the translational acceleration of the center of mass for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\ddot{\boldsymbol{x}}_{Gj} = {}^{j}\ddot{\boldsymbol{x}}_{j-1} + \Delta\ddot{\boldsymbol{x}}_{Gj} \quad 3.59$$

Revolute Joint

$$\begin{aligned} \ddot{\boldsymbol{x}}_{Gj,i} &= {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i} + \Delta\ddot{\boldsymbol{x}}_{Gj,i} \\ \ddot{\boldsymbol{x}}_{Gj,i'} &= \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i'} + \Delta\ddot{\boldsymbol{x}}_{Gj,i'}; & when \; i < j \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i'}; & when \; i = j \end{cases} \end{aligned} \quad 3.60$$

$$\ddot{\boldsymbol{x}}_{Gj,i''} = \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i''} + \Delta\ddot{\boldsymbol{x}}_{Gj,i''}; & when\ i < j \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i''}; & when\ i = j \end{cases}$$

Prismatic Joint

$$\ddot{\boldsymbol{x}}_{Gj,i} = {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i} + \Delta\ddot{\boldsymbol{x}}_{Gj,i}$$

$$\ddot{\boldsymbol{x}}_{Gj,i'} = \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i'} + \Delta\ddot{\boldsymbol{x}}_{Gj,i'}\ ; when\ i < j \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i'}\ ; when\ i = j \end{cases} \qquad 3.61$$

$$\ddot{\boldsymbol{x}}_{Gj,i''} = \begin{cases} {}^{j}\ddot{\boldsymbol{x}}_{(j-1),i''} + \Delta\ddot{\boldsymbol{x}}_{Gj,i''}\ ; when\ i < j \\ \Delta\ddot{\boldsymbol{x}}_{Gj,i''}\ ; when\ i = j \end{cases}$$

Step 8

a. Calculate and store the rotation matrices referred to the global reference frame for all the links ($j = 1, 2, 3, \dots, n$) along with their sensitivities;

$${}^{0}\boldsymbol{R}_{j} = {}^{0}\boldsymbol{R}_{j-1}\boldsymbol{R}_{j} \qquad 3.62$$

$${}^{0}\boldsymbol{R}_{j,i} = \begin{cases} {}^{0}\boldsymbol{R}_{(j-1),i}\boldsymbol{R}_{j}; when\ i < j \\ {}^{0}\boldsymbol{R}_{j-1}\boldsymbol{R}_{j,i}; when\ i = j \end{cases}$$

$${}^{0}\boldsymbol{R}_{j,i'} = \boldsymbol{0} \qquad 3.63$$

$${}^{0}\boldsymbol{R}_{j,i''} = \boldsymbol{0}$$

b. Calculate and store the angular momentum for all the links ($j = 1, 2, 3, \dots, n$) along with their sensitivities;

$$\boldsymbol{H}_{j} = \boldsymbol{I}_{j}\boldsymbol{\omega}_{j} \qquad 3.64$$

Revolute Joint

$$\boldsymbol{H}_{j,i} = \boldsymbol{I}_{j}\boldsymbol{\omega}_{j,i}$$

$$\boldsymbol{H}_{j,i'} = \boldsymbol{I}_{j}\boldsymbol{\omega}_{j,i'} \qquad 3.65$$

$$\boldsymbol{H}_{j,i''} = \mathbf{0}$$

Prismatic Joint

$$\boldsymbol{H}_{j,i} = \begin{cases} \boldsymbol{I}_j \boldsymbol{\omega}_{j,i} \,; when\ i < j \\ \mathbf{0} \,; when\ i = j \end{cases}$$

$$\boldsymbol{H}_{j,i'} = \begin{cases} \boldsymbol{I}_j \boldsymbol{\omega}_{j,i'} \,; when\ i < j \\ \mathbf{0} \,; when\ i = j \end{cases} \quad 3.66$$

$$\boldsymbol{H}_{j,i''} = \mathbf{0}$$

c. Calculate and store the inertia force for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\boldsymbol{F}_{Gj} = m_j \left(\ddot{\boldsymbol{x}}_{Gj} + {}^0\boldsymbol{R}_j^T {}^0\boldsymbol{g}_j \right) \quad 3.67$$

$$\boldsymbol{F}_{Gj,i} = m_j \left(\ddot{\boldsymbol{x}}_{Gj,i} + {}^0\boldsymbol{R}_{j,i}^T {}^0\boldsymbol{g}_j \right)$$

$$\boldsymbol{F}_{Gj,i'} = m_j \ddot{\boldsymbol{x}}_{Gj,i'} \quad 3.68$$

$$\boldsymbol{F}_{Gj,i''} = m_j \ddot{\boldsymbol{x}}_{Gj,i''}$$

d. Calculate and store the torque for all the links $(j = 1, 2, 3, \dots, n)$ along with their sensitivities;

$$\boldsymbol{\tau}_{Gj} = \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_j + \boldsymbol{\omega}_j \times \boldsymbol{H}_j \quad 3.69$$

Revolute Joint

$$\boldsymbol{\tau}_{Gj,i} = \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i} + \boldsymbol{\omega}_{j,i} \times \boldsymbol{H}_j + \boldsymbol{\omega}_j \times \boldsymbol{H}_{j,i}$$

$$\boldsymbol{\tau}_{Gj,i'} = \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i'} + \boldsymbol{\omega}_{j,i'} \times \boldsymbol{H}_j + \boldsymbol{\omega}_j \times \boldsymbol{H}_{j,i'} \quad 3.70$$

$$\boldsymbol{\tau}_{Gj,i''} = \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i''}$$

Prismatic Joint

$$\boldsymbol{\tau}_{Gj,i} = \begin{cases} \boldsymbol{\omega}_{j,i} \times \boldsymbol{H}_j + \boldsymbol{\omega}_j \times \boldsymbol{H}_{j,i} + \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i} \,; when\ i < j \\ \mathbf{0} \,; when\ i = j \end{cases} \quad 3.71$$

$$\boldsymbol{\tau}_{Gj,i'} = \begin{cases} \boldsymbol{\omega}_{j,i'} \times \boldsymbol{H}_j + \boldsymbol{\omega}_j \times \boldsymbol{H}_{j,i'} + \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i'} \; ; when \; i < j \\ \mathbf{0} \; ; when \; i = j \end{cases}$$

$$\boldsymbol{\tau}_{Gj,i''} = \begin{cases} \boldsymbol{I}_j \dot{\boldsymbol{\omega}}_{j,i''} \; ; when \; i < j \\ \mathbf{0} \; ; when \; i = j \end{cases}$$

Only in this step, the rotational matrix referred to the global reference frame is needed.

If $j < n$ go to step 2, if $j = n$ continue.

The sequence for the backward recursive dynamics formulation along with the sensitivities for both revolute and prismatic joint is given through steps (9-10):

Step 9

a. Calculate internal forces for all the links $(j = n, n-1, \dots, 3, 2, 1)$ along with their sensitivities, consider for $j = n+1$, $\boldsymbol{F}_{jp} = \mathbf{0}$;

$$\boldsymbol{F}_{jd} = \boldsymbol{R}_{j+1} \boldsymbol{F}_{(j+1)p} \tag{3.72}$$

$$\boldsymbol{F}_{jd,i} = \begin{cases} \boldsymbol{R}_{j+1} \boldsymbol{F}_{(j+1)p,i}; & when \; i \neq j+1 \\ \boldsymbol{R}_{j+1} \boldsymbol{F}_{(j+1)p,i} + \boldsymbol{R}_{(j+1),i} \boldsymbol{F}_{(j+1)p}; & when \; i = j+1 \end{cases}$$

$$\boldsymbol{F}_{jd,i'} = \boldsymbol{R}_{j+1} \boldsymbol{F}_{(j+1)p,i'} \tag{3.73}$$

$$\boldsymbol{F}_{jd,i''} = \boldsymbol{R}_{j+1} \boldsymbol{F}_{(j+1)p,i''}$$

$$\boldsymbol{F}_{jp} = \boldsymbol{F}_{jd} + \boldsymbol{F}_{Gj} \tag{3.74}$$

$$\boldsymbol{F}_{jp,i} = \begin{cases} \boldsymbol{F}_{jd,i} + \boldsymbol{F}_{Gj,i}; & when \; i \leq j \\ \boldsymbol{F}_{jd,i}; & when \; i > j \end{cases}$$

$$\boldsymbol{F}_{jp,i'} = \begin{cases} \boldsymbol{F}_{jd,i'} + \boldsymbol{F}_{Gj,i'}; & when \; i \leq j \\ \boldsymbol{F}_{jd,i'}; & when \; i > j \end{cases} \tag{3.75}$$

$$\boldsymbol{F}_{jp,i''} = \begin{cases} \boldsymbol{F}_{jd,i''} + \boldsymbol{F}_{Gj,i''}; & when\ i \le j \\ \boldsymbol{F}_{jd,i''}; & when\ i > j \end{cases}$$

b. Calculate internal forces due to global external force ${}^0\boldsymbol{F}_k^e$ applied at link k which transfers internal joint force $\boldsymbol{F}_{jd}^e$ backward $(j = k, k-1, \dots, 3, 2, 1)$, where $k \le n$.

$$\boldsymbol{F}_{jd}^e = \begin{cases} {}^0\boldsymbol{R}_k^T\, {}^0\boldsymbol{F}_k^e; & when\ j = k \\ \boldsymbol{R}_{j+1}\boldsymbol{F}_{(j+1)d}^e; & when\ j < k \end{cases} \tag{3.76}$$

$$\boldsymbol{F}_{jd,i}^e = \begin{cases} {}^0\boldsymbol{R}_{k,i}^T\, {}^0\boldsymbol{F}_k^e; & when\ j = k\ and\ i \le j \\ \boldsymbol{R}_{(j+1),i}\boldsymbol{F}_{(j+1)d}^e + \boldsymbol{R}_{j+1}\boldsymbol{F}_{(j+1)d,i}^e; & when\ j < k\ and\ i = j+1 \\ \boldsymbol{R}_{j+1}\boldsymbol{F}_{(j+1)d,i}^e; & when\ j < k\ and\ i \ne j+1 \end{cases}$$

$$\boldsymbol{F}_{jd,i'}^e = \boldsymbol{0} \tag{3.77}$$

$$\boldsymbol{F}_{jd,i''}^e = \boldsymbol{0}$$

$$\boldsymbol{F}_{jp} = \boldsymbol{F}_{jp} + \boldsymbol{F}_{jd}^e \tag{3.78}$$

$$\boldsymbol{F}_{jp,i} = \boldsymbol{F}_{jp,i} + \boldsymbol{F}_{jd,i}^e;\ \ when\ \ i \le k$$

$$\boldsymbol{F}_{jp,i'} = \boldsymbol{0} \tag{3.79}$$

$$\boldsymbol{F}_{jp,i''} = \boldsymbol{0}$$

c. Calculate internal torques for all the links $(j = n, n-1, \dots, 3, 2, 1)$ along with their sensitivities, consider for $j = n+1$, $\boldsymbol{\tau}_{jp} = \boldsymbol{0}$;

$$\boldsymbol{\tau}_{jd} = \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)p} \tag{3.80}$$

$$\boldsymbol{\tau}_{jd,i} = \begin{cases} \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)p,i}; & when\ i \ne j+1 \\ \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)p,i} + \boldsymbol{R}_{(j+1),i}\boldsymbol{\tau}_{(j+1)p}; & when\ i = j+1 \end{cases} \tag{3.81}$$

$$\boldsymbol{\tau}_{jd,i'} = \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)p,i'}$$

$$\boldsymbol{\tau}_{jd,i''} = \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)p,i''}$$

$$\boldsymbol{\tau}_{jp} = \boldsymbol{\tau}_{jd} + \boldsymbol{\tau}_{Gj} - \boldsymbol{F}_{jd} \times \Delta\boldsymbol{x}_j - \boldsymbol{F}_{Gj} \times \Delta\boldsymbol{x}_{Gj} \tag{3.82}$$

$$\boldsymbol{\tau}_{jp,i} = \begin{cases} \boldsymbol{\tau}_{jd,i} + \boldsymbol{\tau}_{Gj,i} - \boldsymbol{F}_{jd,i} \times \Delta\boldsymbol{x}_j - \boldsymbol{F}_{Gj,i} \times \Delta\boldsymbol{x}_{Gj}; & when\ i \le j \\ \boldsymbol{\tau}_{jd,i} - \boldsymbol{F}_{jd,i} \times \Delta\boldsymbol{x}_j; & when\ i > j \end{cases}$$

$$\boldsymbol{\tau}_{jp,i'} = \begin{cases} \boldsymbol{\tau}_{jd,i'} + \boldsymbol{\tau}_{Gj,i'} - \boldsymbol{F}_{jd,i'} \times \Delta\boldsymbol{x}_j - \boldsymbol{F}_{Gj,i'} \times \Delta\boldsymbol{x}_{Gj}; & when\ \ i \le j \\ \boldsymbol{\tau}_{jd,i'} - \boldsymbol{F}_{jd,i'} \times \Delta\boldsymbol{x}_j; & when\ i > j \end{cases} \tag{3.83}$$

$$\boldsymbol{\tau}_{jp,i''} = \begin{cases} \boldsymbol{\tau}_{jd,i''} + \boldsymbol{\tau}_{Gj,i''} - \boldsymbol{F}_{jd,i''} \times \Delta\boldsymbol{x}_j - \boldsymbol{F}_{Gj,i''} \times \Delta\boldsymbol{x}_{Gj}; & when\ i \le j \\ \boldsymbol{\tau}_{jd,i''} - \boldsymbol{F}_{jd,i''} \times \Delta\boldsymbol{x}_j; & when\ i > \ j \end{cases}$$

d. Calculate joint torque due to global external force ${}^0\boldsymbol{F}_k^e$ applied at local coordinate $\boldsymbol{r}_k$ of the kth link which transfers internal joint torque $\boldsymbol{\tau}_{jd}^{ef}$ backward $(j = k, k-1, \dots, 3, 2, 1)$, where $k \le n$.

$$\boldsymbol{\tau}_{jd}^{ef} = \begin{cases} (\boldsymbol{r}_k + \Delta\boldsymbol{x}_j) \times \boldsymbol{F}_{jd}^e; & when\ j = k \\ \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d}^e + \Delta\boldsymbol{x}_j \times \boldsymbol{F}_{jd}^e; & when\ j < k \end{cases} \tag{3.84}$$

$$\boldsymbol{\tau}_{jd,i}^{ef} = \begin{cases} \Delta\boldsymbol{x}_{j,i} \times \boldsymbol{F}_{jd}^e + \Delta\boldsymbol{x}_j \times \boldsymbol{F}_{jd,i}^e; & when\ j = k\ and\ i \le j \\ \boldsymbol{R}_{(j+1),i}\boldsymbol{\tau}_{(j+1)d}^e + \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d,i}^e + \Delta\boldsymbol{x}_j \times \boldsymbol{F}_{jd,i}^e; & when\ j < k\ and\ i = j+1 \\ \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d,i}^e + \Delta\boldsymbol{x}_{j,i} \times \boldsymbol{F}_{jd}^e + \Delta\boldsymbol{x}_j \times \boldsymbol{F}_{jd,i}^e; & when\ j < k\ and\ i \ne j+1 \end{cases}$$

$$\boldsymbol{\tau}_{jd,i'}^{ef} = \boldsymbol{0} \tag{3.85}$$

$$\boldsymbol{\tau}_{jd,i''}^{ef} = \boldsymbol{0}$$

$$\boldsymbol{\tau}_{jp} = \boldsymbol{\tau}_{jp} + \boldsymbol{\tau}_{jd}^{ef} \tag{3.86}$$

$$\boldsymbol{\tau}_{jp,i} = \boldsymbol{\tau}_{jp,i} + \boldsymbol{\tau}_{jd,i}^{ef}; \quad when \;\; i \leq k$$

$$\boldsymbol{\tau}_{jp,i'} = \mathbf{0} \tag{3.87}$$

$$\boldsymbol{\tau}_{jp,i''} = \mathbf{0}$$

e. Calculate joint torque due to global external moment ${}^{0}\boldsymbol{\tau}_{k}^{e}$ applied at kth link which transfers internal joint torque $\boldsymbol{\tau}_{jd}^{em}$ backward $(j = k, k-1, \dots, 3, 2, 1)$, where $k \leq n$.

$$\boldsymbol{\tau}_{jd}^{em} = \begin{cases} {}^{0}\boldsymbol{R}_{k}^{T}\,{}^{0}\boldsymbol{\tau}_{k}^{e}; & when\; j = k \\ \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d}^{e}; & when\; j < k \end{cases} \tag{3.88}$$

$$\boldsymbol{\tau}_{jd,i}^{em} = \begin{cases} {}^{0}\boldsymbol{R}_{k,i}^{T}\,{}^{0}\boldsymbol{\tau}_{k}^{e}; & when\; j = k \;and\; i \leq j \\ \boldsymbol{R}_{(j+1),i}\boldsymbol{\tau}_{(j+1)d}^{e} + \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d,i}^{e}; & when\; j < k \;and\; i = j+1 \\ \boldsymbol{R}_{j+1}\boldsymbol{\tau}_{(j+1)d,i}^{e}; & when\; j < k \;and\; i \neq j+1 \end{cases} \tag{3.89}$$

$$\boldsymbol{\tau}_{jd,i'}^{em} = \mathbf{0}$$

$$\boldsymbol{\tau}_{jd,i''}^{em} = \mathbf{0}$$

$$\boldsymbol{\tau}_{jp} = \boldsymbol{\tau}_{jp} + \boldsymbol{\tau}_{jd}^{em} \tag{3.90}$$

$$\boldsymbol{\tau}_{jp,i} = \boldsymbol{\tau}_{jp,i} + \boldsymbol{\tau}_{jd,i}^{em}; \quad when \;\; i \leq k$$

$$\boldsymbol{\tau}_{jp,i'} = \mathbf{0} \tag{3.91}$$

$$\boldsymbol{\tau}_{jp,i''} = \mathbf{0}$$

Step 10

a. Calculate the torque component about the local ${}^{j-1}z$ axis for all the joint $(j = n, n-1, \dots, 3, 2, 1)$ along with their sensitivities;

$$\tau_{j} = \boldsymbol{\tau}_{jp}^{T}\boldsymbol{R}_{j}^{T}\boldsymbol{e}_{z_{j-1}} \tag{3.92}$$

$$\tau_{j,i} = \begin{cases} \boldsymbol{\tau}_{jp,i}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}}; & when\ i \neq j \\ \boldsymbol{\tau}_{jp,i}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}} + \boldsymbol{\tau}_{jp}^{T} \boldsymbol{R}_{j,i}^{T}\, \boldsymbol{e}_{z_{j-1}}; & when\ i = j \end{cases}$$

$$\tau_{j,i'} = \boldsymbol{\tau}_{jp,i'}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}} \qquad 3.93$$

$$\tau_{j,i''} = \boldsymbol{\tau}_{jp,i''}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}}$$

b. Calculate the force component along the local ${}^{j-1}z$ axis for all the joint $(j = n, n-1, \ldots, 3, 2, 1)$ along with their sensitivities;

$$f_j = \boldsymbol{F}_{jp}^{T} \boldsymbol{R}_{j}^{T\,j} \boldsymbol{e}_{z_{j-1}} \qquad 3.94$$

$$f_{j,i} = \begin{cases} \boldsymbol{F}_{jp,i}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}}; & when\ i \neq j \\ \boldsymbol{F}_{jp,i}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}} + \boldsymbol{F}_{jp}^{T} \boldsymbol{R}_{j,i}^{T}\, \boldsymbol{e}_{z_{j-1}}; & when\ i = j \end{cases}$$

$$f_{j,i'} = \boldsymbol{F}_{jp,i'}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}} \qquad 3.95$$

$$f_{j,i''} = \boldsymbol{F}_{jp,i''}^{T} \boldsymbol{R}_{j}^{T}\, \boldsymbol{e}_{z_{j-1}}$$

In step 10 only the left-hand side subscript j stands for the joints, all the other subscripts and superscripts indicate the link reference as before.

In summary, the last two steps show how the kinetics terms are transferred back from end-effector to the origin. Formula for internal forces due to global external force, and internal torques due to global external moment are all given. Also, the torque and force components can be computed from proximal and distal forces and torques for each link. The next two chapters discuss the applications of the proposed recursive dynamics and sensitivity analysis to the dynamic motion planning problems.

CHAPTER IV

DYNAMIC MOTION PREDICTION WITH TWO-LINK MANIPULATOR

Two simple robot manipulator examples are first setup to test the proposed recursive dynamics formulation and its sensitivity analysis. The first example is a time-minimization motion-planning problem for a simple two-link robot manipulator with two revolute joints. The second example is also a time-minimization motion-planning problem for a two-link manipulator but with one prismatic joint and one revolute joint. The detailed model setup, optimization formulations, and numerical results are described in this chapter.

4.1 Two-link manipulator with two revolute joints

Two-link robot manipulator is a widely studied model in dynamics simulation (Dissanayake et al., 1991; Wang et al., 2005; Xiang et al., 2009a). The same example is examined here to verify the proposed algorithms for revolute joints. This manipulator model is designed by simply connecting two links with two revolute joints that are moving on the horizontal plane. The local coordinates for each link are set up at the tip of that link. The relative joint angles q_1 and q_2 represent the independent generalized coordinates for each link. A visual representation of the example setup is given in Figure 4.1 showing all physical parameters for both links. The manipulator can be moved from a preset initial position $[q_1(0), q_2(0)]$ to a desired position $[q_1(T), q_2(T)]$ in time interval T. The DH parameters are set up in Table 4.1, and the physical properties of the manipulator are given in Table 4.2.

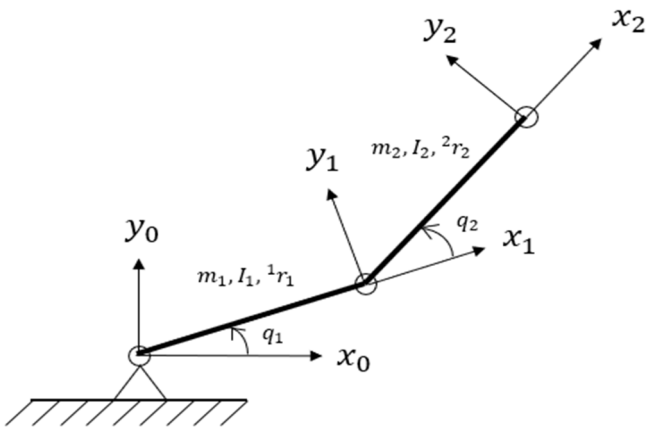


Figure 4.1: Two-link manipulator with two revolute joints

Table 4.1: DH table for a two-link manipulator with revolute joints

Link	θ_i	d_i	a_i	α_i
1	0	0	L_1	0
2	0	0	L_2	0

Table 4.2: Physical parameters for a two-link manipulator with revolute joints

Length:	$L_1 = L_2 = 0.4\ m$
Mass:	$m_1 = m_2 = 0.5\ kg$
Inertia at center of mass:	$I_1 = I_2 = 0.1\ kg \cdot m^2$
Center of mass of link 1:	${}^1r_1 = (-L_1/2, 0)$
Center of mass of link 2:	${}^2r_2 = (-L_2/2, 0)$
Torque upper bound:	$\boldsymbol{\tau}^U = 10\ Nm$
Torque lower bound:	$\boldsymbol{\tau}^L = -10\ Nm$

4.1.1 Joint Profile Discretization

The proposed recursive Newton-Euler formulation is used to calculate the joint actuating torques, internal joint reaction forces, and their sensitivities with respect to state variables $(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$. A joint profile $q(t, \boldsymbol{s}, \boldsymbol{P})$ is discretized using cubic B-splines:

$$q(t, \boldsymbol{s}, \boldsymbol{P}) = \sum_{i=1}^{m} B_i(t, \boldsymbol{s}) p_i \qquad 0 \le t \le T \tag{4.13}$$

where $B_i(t, \boldsymbol{s})$ are the basis functions, $\boldsymbol{s} = \{s_0, \dots, s_l\}$ is the knot vector, and $\boldsymbol{P} = \{p_1, \dots, p_m\}$ is the control point vector.

Knot vector $\boldsymbol{s}$, which uniquely determines the B-spline basis functions, is evenly spaced on the time interval $[0 \quad T]$ with the time step Δs,

$$s_{i+1} = s_i + \Delta s, \qquad \Delta s = \frac{T}{l}, \qquad i = 0, \dots, l-1 \tag{4.14}$$

Then, the state variables calculated from Eq. 4.1 are functions of both the control point vector and knot vector. Therefore, the torque and force calculated from EOM are also an explicit function of control point vector and knot vector. Finally, the sensitivities are calculated using the chain rule as follows,

$$\begin{aligned} \frac{\partial \boldsymbol{\tau}}{\partial p_i} &= \frac{\partial \boldsymbol{\tau}}{\partial q}\frac{\partial q}{\partial p_i} + \frac{\partial \boldsymbol{\tau}}{\partial \dot{q}}\frac{\partial \dot{q}}{\partial p_i} + \frac{\partial \boldsymbol{\tau}}{\partial \ddot{q}}\frac{\partial \ddot{q}}{\partial p_i} \\ \frac{\partial \boldsymbol{F}}{\partial p_i} &= \frac{\partial \boldsymbol{F}}{\partial q}\frac{\partial q}{\partial p_i} + \frac{\partial \boldsymbol{F}}{\partial \dot{q}}\frac{\partial \dot{q}}{\partial p_i} + \frac{\partial \boldsymbol{F}}{\partial \ddot{q}}\frac{\partial \ddot{q}}{\partial p_i} \end{aligned} \tag{4.15}$$

4.1.2 Optimization Formulation

The two-link trajectory optimization problem is formulated as a parameter optimization problem as below, where the design variables are computed as $\boldsymbol{x} = [\boldsymbol{P}^{\mathrm{T}} \quad \boldsymbol{s}^{\mathrm{T}}]^{\mathrm{T}}$, and the total time T is minimized subject to boundary conditions and torque limits constraint.

Minimize

$$T(\boldsymbol{x}) \tag{4.16}$$

Subject to

$$q_1(0) = 0.0, q_2(0) = -2.0$$
$$q_1(T) = 1.0, q_2(T) = -1.0$$
$$\dot{q}_1(0) = \dot{q}_2(0) = \dot{q}_1(T) = \dot{q}_2(T) = 0.0 \quad (4.17)$$
$$-10 \leq \boldsymbol{\tau}(\boldsymbol{x}) \leq 10$$

The optimization problem is solved using a sequential quadratic programming (SQP) algorithm in SNOPT and the *snoptB* routine (Gill et al., 2002) is used for this purpose.

4.1.3 Results and Discussion

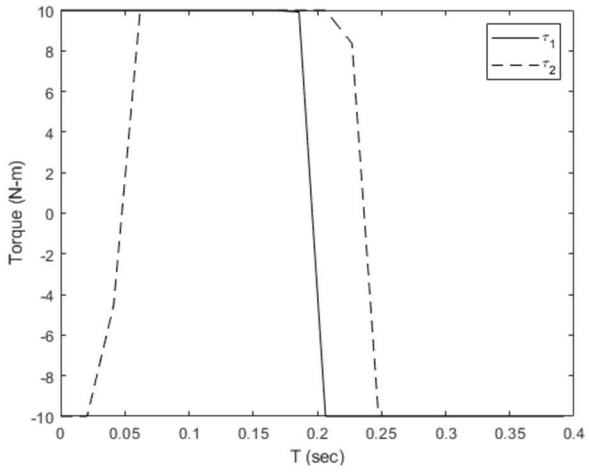


Figure 4.2: Joint torque profiles with two revolute joints

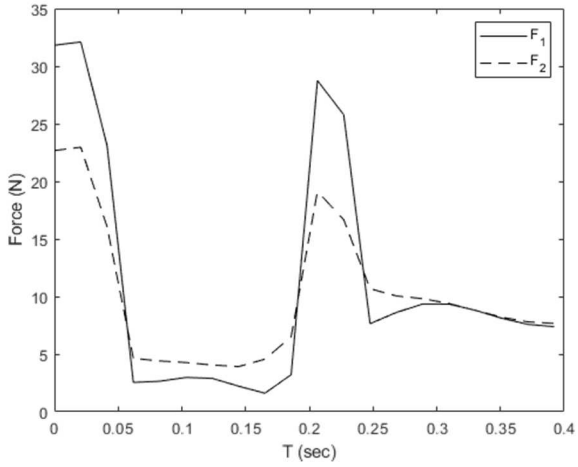


Figure 4.3: Joint internal force vector norm profiles with two revolute joints

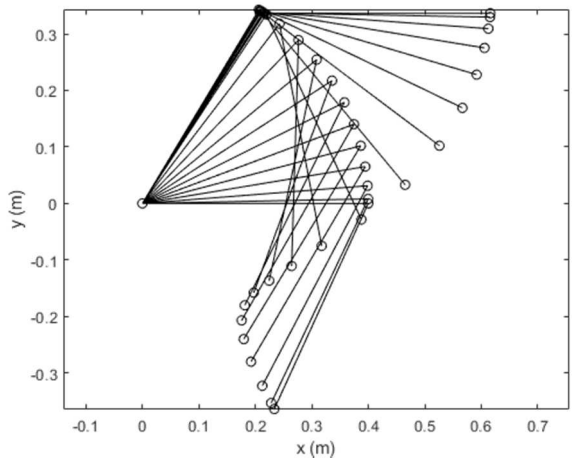


Figure 4.4: Motion trajectory with two revolute joints

This two-link minimum-time motion-planning problem has been well studied in the literature with different dynamics formulations (Dissanayake et al., 1991; Wang et al., 2005; Xiang et al., 2009a). In this example, internal joint force can be directly calculated from the proposed recursive Newton-Euler equations as shown in Figure 4.3. This feature is unavailable in recursive Lagrangian formulation (Xiang et al., 2009a). The minimum travel time is obtained as $T = 0.392$ seconds, which is comparable to that reported in the literature (0.394 seconds by Wang et al.

(2005) and 0.393 seconds by Xiang et al. (2009a)). Also, the predicted joint torque and trajectory (Figs. 4.2 and 4.4) have similar shapes to those reported in the literature (Xiang et al., 2009a).

The joint torque value for the first link stays on the upper bound until just over 0.2 seconds, then it shifts to the lower bound and continues until optimal time is achieved. For the second link, its torque value stays on the lower bound for a short time period, then shifts to upper bound, finally returns back to lower bound after 0.25 seconds. This is a bound-bound type of control to achieve the minimal travel time as shown in Figure 4.2. The magnitude of internal joint reaction force, which is calculated as the second norm of the internal joint force vector at revolute joint, shows a double-peak pattern in Figure 4.3. This is due to the bound-bound type of control for the two-link manipulator. However, the first link has higher internal joint reaction force value than the second link before they merge around the optimal time.

4.2 Two-link manipulator with one prismatic and one revolute joint

To examine the proposed algorithms for prismatic joint, a similar two-link manipulator in Section 4.1 is discussed here. For this example, the first joint is replaced by a prismatic joint. The second joint is still a revolute joint, giving this manipulator an opportunity to both translate and rotate about their DOF on a horizontal plane. The relative joint translation q_1 and joint rotation q_2 represent the independent generalized coordinates for the links of L_1 and L_2. The mass of the links is given by m_1 and m_2, while the inertia is given by I_1 and I_2 at the center of mass of each link, respectively. The manipulator is moving from an initial position $[q_1(0), q_2(0)]$ to a desired position $[q_1(T), q_2(T)]$ in time interval T. The DH parameters are set up in Table 4.3, and the physical properties of the manipulator are given in Table 4.4. Cubic B-splines are used to discretize the joint profiles as discussed in previous example in Section 4.1.

Table 4.3: DH table for a two-link manipulator with one prismatic joint and one revolute joint

Link	θ_i	d_i	a_i	α_i
1	$\pi/2$	L_1	0	$\pi/2$
2	$\pi/2$	0	L_2	0

Table 4.4: Physical parameters for a two-link manipulator with one prismatic joint and one revolute joint

Length:	$L_1 = L_2 = 0.4\ m$
Mass:	$m_1 = m_2 = 0.5\ kg$
Inertia at center of mass:	$I_1 = I_2 = 0.1\ kg \cdot m^2$
Center of Mass of Link 1:	${}^1r_1 = (0, -L_1/2)$
Center of Mass of Link 2:	${}^2r_2 = (-L_2/2, 0)$
Torque upper bound:	$\tau^U = 10\ Nm$
Torque lower bound:	$\tau^L = -10\ Nm$
Force upper bound:	$f^U = 10\ N$
Force lower bound:	$f^L = -10\ N$

4.2.1 Optimization formulation

A MOO scheme has been set up for this example. The optimization formulation is given as follows:

Minimize

$$\int_{t=0}^{T} [(1-u) + u(f_1^2 + \tau_2^2)]dt\,, \qquad u \in [0,1] \tag{4.18}$$

Subject to

$$\begin{gathered} q_1(0) = 0.1, q_2(0) = 0.6 \\ q_1(T) = \frac{\pi}{4}, q_2(T) = -\frac{\pi}{4} \\ \dot{q}_1(0) = \dot{q}_2(0) = \dot{q}_1(T) = \dot{q}_2(T) = 0.0 \\ -10 \leq \tau(\boldsymbol{x}) \leq 10 \\ -10 \leq f(\boldsymbol{x}) \leq 10 \end{gathered} \tag{4.19}$$

where $u = 0.01$ is a preset parameter that prevents sharp functioning of the actuating force (f_1) and torque (τ_2). As the value of u decreases, the MOO problem turns out to be a time-minimization trajectory-planning problem. The actuation square is related to energy consumption in the optimization formulation.

4.2.2 Results and Discussion

The optimal actuating force and torque profiles are depicted in Figure 4.5. The optimal internal joint reaction force (magnitude) profiles are presented in Figure 4.6, and the snapshots of the optimal trajectory is shown in Figure 4.7. This example problem is new that has not been solved in the literature before. In this study, the minimum travel time is obtained as $T = 0.5585$ seconds and the predicted motion trajectory has a desired shape with a translation and a rotation.

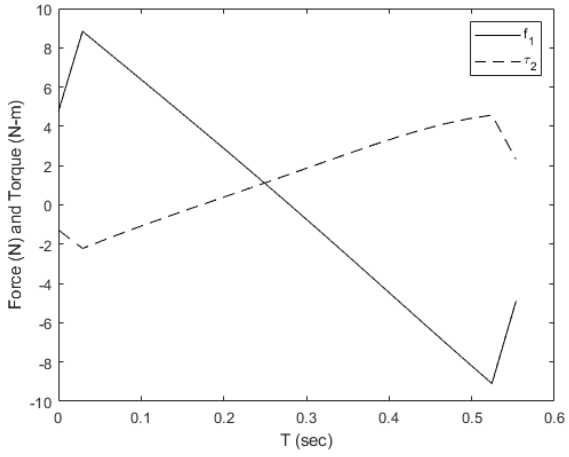


Figure 4.5: Joint actuating force and torque profiles with one prismatic and one revolute joint

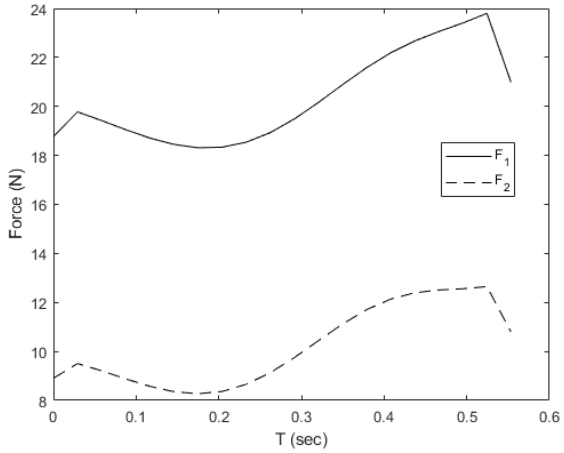


Figure 4.6: Joint internal force vector norm profiles with one prismatic joint and one revolute joint

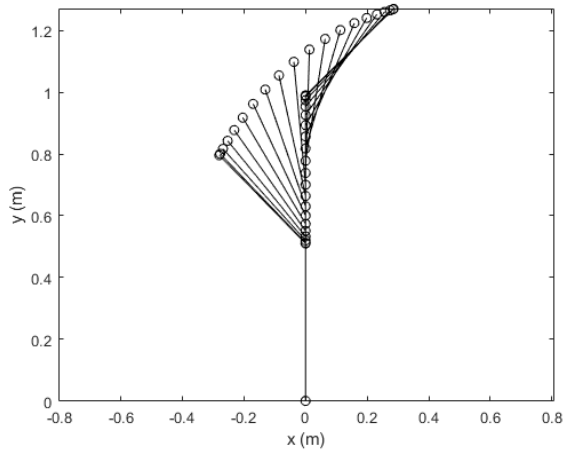


Figure 4.7: Motion trajectory with one prismatic joint and one revolute joint

In Figure 4.5, both the actuating force for the first prismatic joint and the actuating torque for the second revolute joint have an accelerating and a decelerating process since their initial and final velocities are set to be zero. This is because the parameter u in MOO is a small value (0.01), which gives some time-minimization features. The actuating force changes from a positive peak value to a negative peak value. In contrast, the actuating torque changes from a negative peak

value to a positive peak value in its moving direction. Figure 4.6 shows the internal joint reaction force vector norm profiles. Both internal force curves have similar trends and double-peak pattern. In addition, the internal force vector norm profile of the revolute joint is lower than that of the prismatic joint.

In summary, this chapter gave detailed discussions on two simple two-link manipulator trajectory planning examples for the proposed recursive Newton-Euler formulation. The simulation results are compared to available data in literature and similar results have been found. This paves the way to further investigate the efficacy of the proposed algorithms with a much more complex gait prediction problem, which will be discussed in the next chapter.

CHAPTER V

DYNAMIC MOTION PREDICTION WITH GAIT

The one-step gait prediction in Xiang et al. (2009b) is revisited in this chapter. The mechanical model has been set up, then the proposed algorithm is used to calculate the joint torque and forces. Internal joint reaction forces and their sensitivities will also be calculated. A two-step passive-active algorithm is used to calculate the ground reaction forces (GRF). Various time dependent and independent constraints are applied for optimization formulation.

5.1 Mechanical Model

The 3D skeletal model shown in Figure 5.1 has 55 DOF and 6 of them are global DOF including three translations and three rotations. The five physical branches are spine, neck, right arm, left arm, right leg, and left leg. The global DOF start from the origin of the Cartesian coordinates and end at the current pelvis location. To simplify the simulation, the upper spine, neck, clavicle, and wrist joints are frozen to their neutral angles. Therefore, the total number of active joints are 38. One-step formulation is used to simulate continuous walking motion. Each step consists of a single support and a double support phase. Single support phase occurs when one foot contacts the ground while the other leg is swinging. After one step, the roles of the arms and legs are swapped and the motion is repeated.

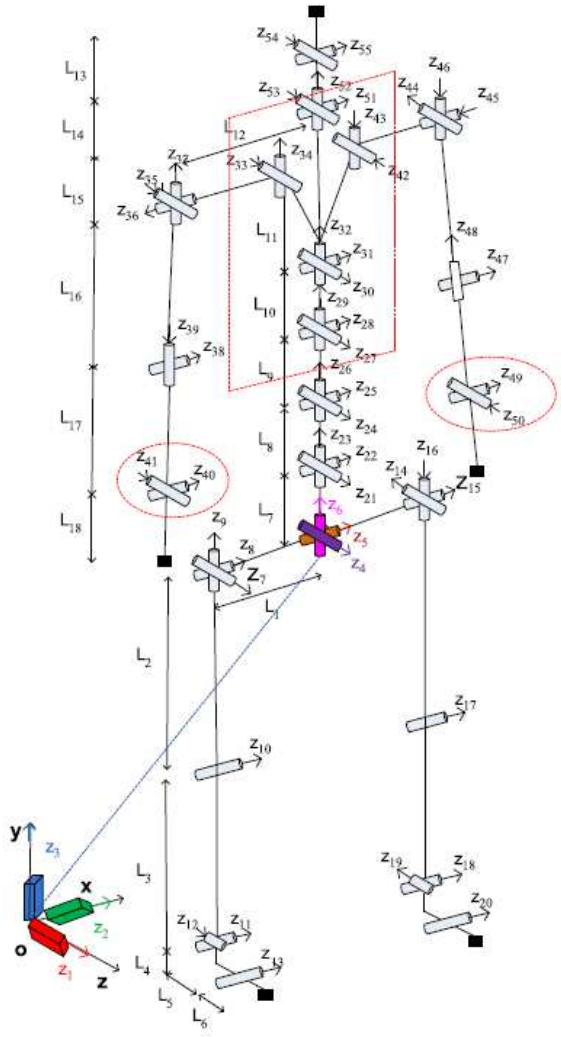


Figure 5.1: The 3D Skeletal Digital Human Model, the frozen DOFs for the walking task are shown in the dashed enclosures

The proposed recursive Newton-Euler formulation is used to calculate the revolute joint actuating torque, prismatic joint actuating force, internal joint reaction force, and their sensitivities with respect to state variables ($\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$). The ground reaction forces (GRF) are calculated using a two-step active-passive algorithm (Xiang et al., 2009b). The main idea of the algorithm is to first calculate the resultant global active forces and zero-moment point (ZMP) location from EOM based on the given state variables. ZMP location also coincides with center of pressure (COP). Then GRF

$$J(\boldsymbol{q},\boldsymbol{\tau},\boldsymbol{t}) = \int_{t=0}^{T} \left(\frac{\boldsymbol{\tau}(\boldsymbol{q},\boldsymbol{t})}{|\tau|_{max}}\right)^{\mathrm{T}} \left(\frac{\boldsymbol{\tau}(\boldsymbol{q},\boldsymbol{t})}{|\tau|_{max}}\right) dt \qquad 5.1$$

where $|\tau|_{max}$ is the maximum absolute value of joint torque limit.

5.2 Constraints

Two types of constraints are considered for the walking optimization problem: one is the time-dependent constraints which include (1) joint angle limits, (2) joint torque limits, (3) ground penetration, (4) dynamic balance, (5) arm-leg coupling, and (6) self-avoidance. These constraints are imposed throughout the time interval. The second type is time-independent constraints which comprise the (7) symmetry conditions, (8) ground clearance and (9) initial and final foot positions; these constraints are considered only at a specific time point during the step.

5.2.1 Time-dependent constraints

(1) The joint angle limits are taken into account in the formulation.

$$\boldsymbol{q}^L \leq \boldsymbol{q}(\boldsymbol{t}) \leq \boldsymbol{q}^U \qquad 5.2$$

where $\mathbf{q}^L$ and $\mathbf{q}^U$ are lower and upper bounds of joint angle, respectively.

(2) Joint torque limits are imposed as:

$$\boldsymbol{\tau}^L \leq \boldsymbol{\tau}(\boldsymbol{q},\boldsymbol{t}) \leq \boldsymbol{\tau}^U \qquad 5.3$$

where $\boldsymbol{\tau}^L$ and $\boldsymbol{\tau}^U$ are lower and upper bounds of joint torque, respectively.

(3) Ground penetration is defined as while the foot contacts the ground, the height and velocity of contacting points are zero. In contrast, the heights of non-contacting points are greater than zero.

$$y_i = 0, \quad \dot{x}_i = 0, \quad \dot{y}_i = 0, \quad \dot{z}_i = 0, \quad i \in \Omega$$
$$y_i > 0, \quad i \notin \Omega \tag{5.4}$$

where Ω is the set of foot contacting points.

(4) The dynamic balance is considered during walking process by enforcing the ZMP to remain within the foot support polygon (FSP).

$$^{o}\boldsymbol{r}_{ZMP}(\boldsymbol{q}, \boldsymbol{t}) \in FSP \tag{5.5}$$

where $^{o}\boldsymbol{r}_{ZMP}$ is the calculated ZMP location.

(5) The arm-leg coupling constraint is defined as the arm swing on one side counteracting the leg swing on the other side. We introduced one pendulum to represent left arm and another pendulum to represent right leg. The coupling constraint states that these two pendulums should swing in the same direction along the sagittal plane during walking motion.

$$(\boldsymbol{\rho}_1 \cdot \boldsymbol{n}_z)(\boldsymbol{\rho}_2 \cdot \boldsymbol{n}_z) \geq 0 \tag{5.6}$$

where $\boldsymbol{\rho}_1$ is left arm pendulum vector and $\boldsymbol{\rho}_2$ is right leg pendulum vector, $\boldsymbol{n}_z$ is the unit vector along vertical direction.

(6) Self-avoidance is considered in the current formulation to prevent penetration of the arm into the body.

$$d(t) \geq r_1 + r_2 \tag{5.7}$$

where d is calculated distance between wrist and hip during walking, r_1 is the radius of the sphere filled at wrist, and r_2 is the radius of the sphere filled at hip.

5.2.2 Time-independent constraints

(7) The initial and final postures and velocities should satisfy the symmetry conditions to generate continuous walking motion.

$$\boldsymbol{q}(0) = \boldsymbol{q}_S(T)$$
$$\dot{\boldsymbol{q}}(0) = \dot{\boldsymbol{q}}_S(T) \qquad 5.8$$

where $\boldsymbol{q}_S$ are symmetric joint angles, $\dot{\boldsymbol{q}}_S$ are symmetric joint angular velocities.

(8) Ground clearance constraint is imposed during the walking motion to avoid foot drag motion. The value of knee flexion at mid-swing is set around $60°$ to formulate ground clearance constraint.

$$|q_{knee} - 60°| \leq \varepsilon \qquad 5.9$$

where $\varepsilon = 5°$.

(9) The initial and final foot contacting positions are specified at the initial and final times to satisfy the step length constraint.

$${}^{o}\boldsymbol{r}_{foot}(\boldsymbol{q}, t) = {}^{o}\boldsymbol{r}^{L}_{foot}(t); \qquad t = 0, T \qquad 5.10$$

where ${}^{o}\boldsymbol{r}_{foot}$ is the calculated global foot position, and ${}^{o}\boldsymbol{r}^{E}_{foot}$ is the given foot position based on the given step length L.

For a normal gait, the given velocity is $V = 1.2\ m/s$ and the step length is $L = 0.6m$. There are 330 nonlinear design variables (55 DOF, each with six control points) and 1726 nonlinear constraints. First the optimization problem is solved to obtain a feasible solution for the walking problem, where $\boldsymbol{q}(\boldsymbol{t}) = \boldsymbol{0}$ used as the starting point with $J(\boldsymbol{q}, \boldsymbol{\tau}, \boldsymbol{t}) = constant$ as the objective function and all constraints imposed. Once a feasible solution has been obtained, it is used as the

starting point for the optimization problem with dynamic effort as the objective function. The optimality and feasibility tolerances are both set $\varepsilon = 10^{-3}$ for SNOPT and it takes 34.10 seconds of CPU time for a desktop computer with an Intel (R) Xeon (R) E-2186G CPU @ 3.80GHz to solve the nonlinear optimization problem. There are 244 active constraints at the optimal solution.

5.3 Results and Discussion

Figure 5.2 shows the snapshots of a 3D human walking on level ground. The predicted walking motion is quite smooth, and the joint angle and velocity symmetry conditions are satisfied. ZMP stays in the FSP to satisfy the balance condition. In addition, knee flexion at mid-swing, arm-leg coupling, and a straight spine are all predicted as expected.

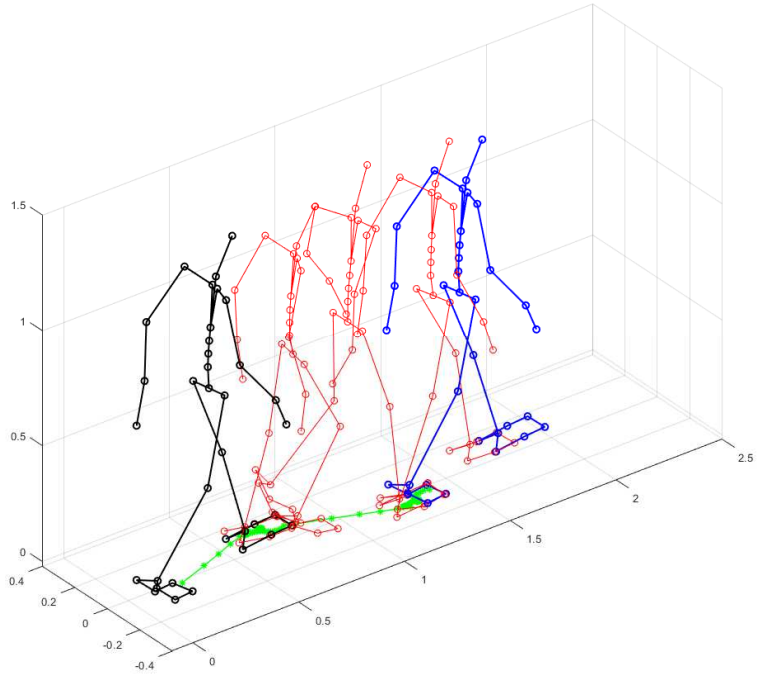
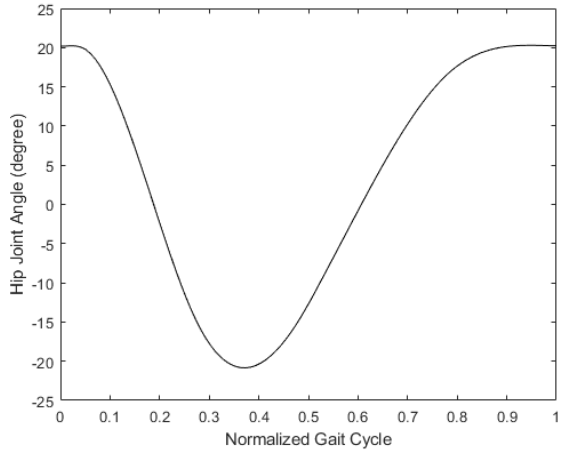
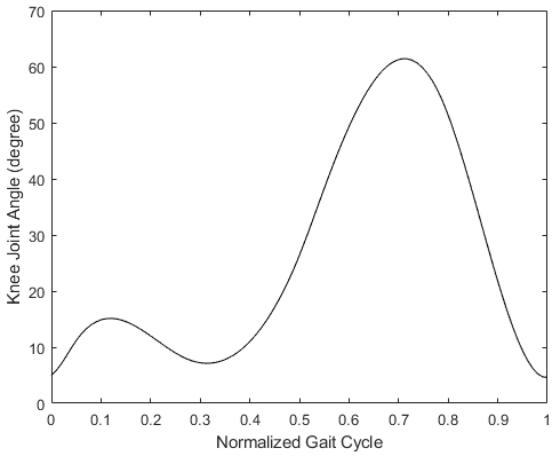


Figure 5.2: The snapshots of the optimized cyclic walking motion

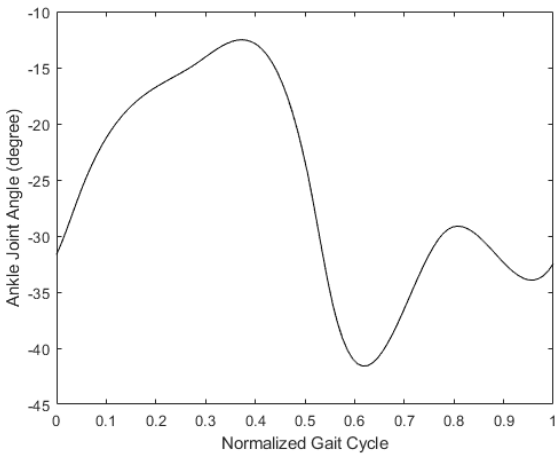
The predicted hip, knee, and ankle joint angles of normal walking for a stride starting from heel strike and ending with the same heel strike are plotted in Figure 5.3. The joint angles' magnitudes and trends are consistent with the data in the literature (Xiang et al., 2009b).



(a)

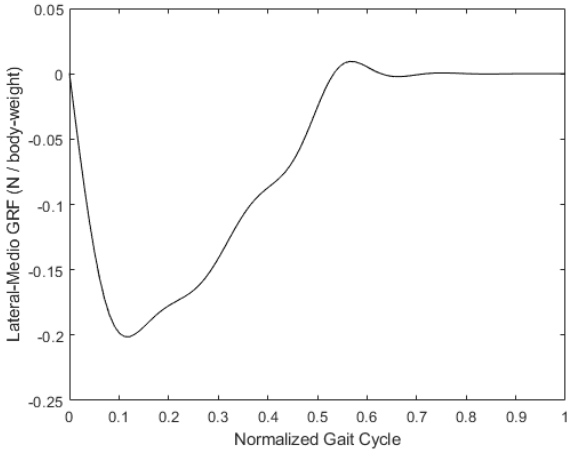


(b)

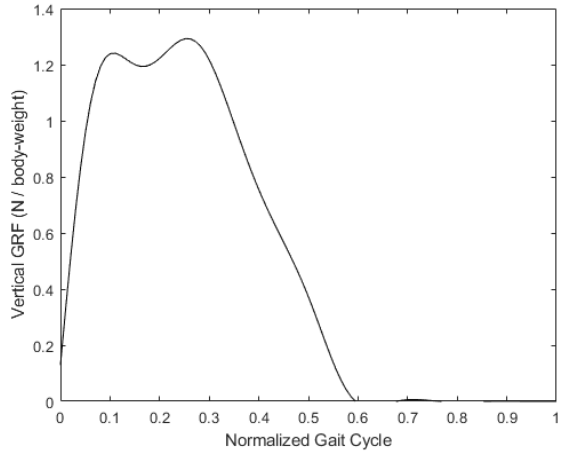


(c)

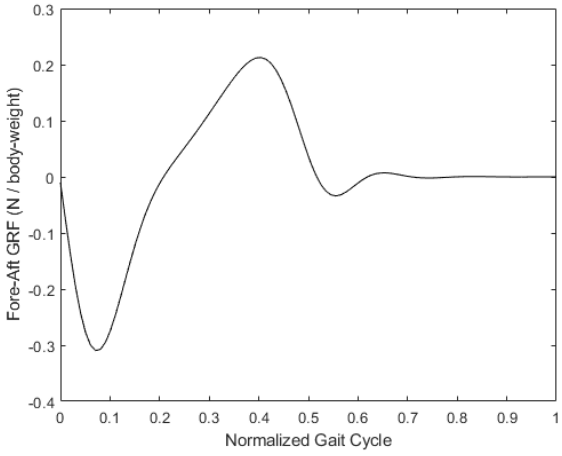
Figure 5.3: Joint angle profiles for gait: (a) Hip, (b) knee, and (c) ankle



(a)

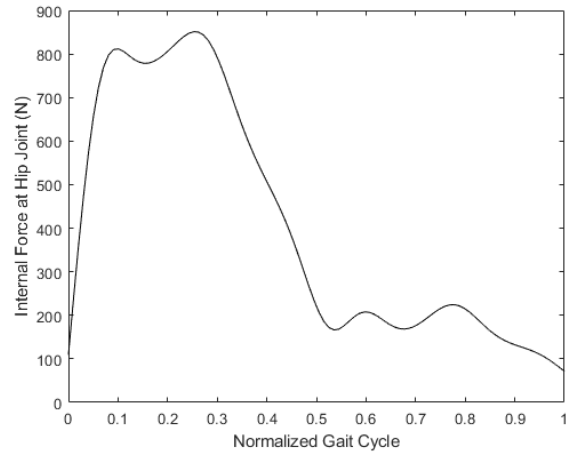


(b)

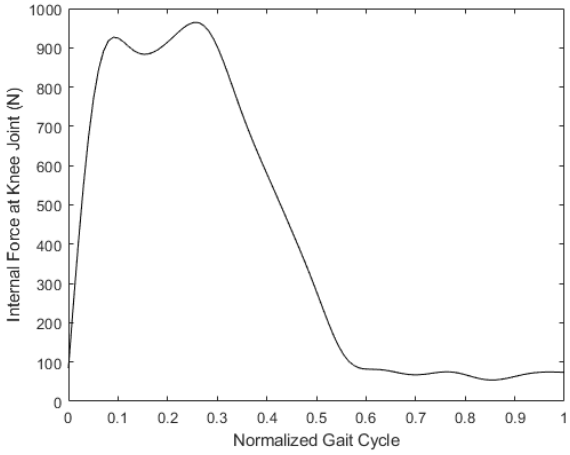


(c)

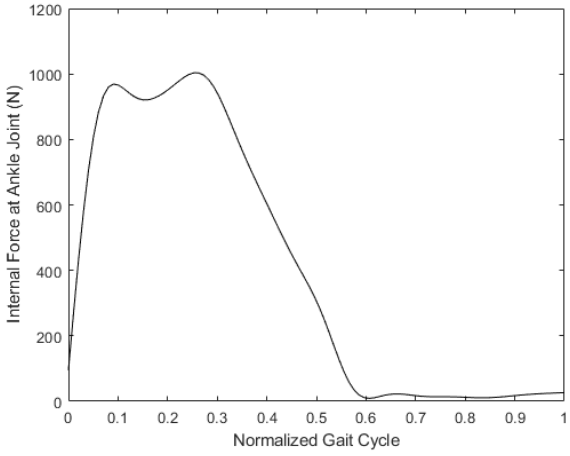
Figure 5.4: GRF profiles for gait: (a) Lateral, (b) vertical, and (c) fore-aft



(a)



(b)



(c)

Figure 5.5: Joint internal force vector norm profiles for gait: (a) Hip, (b) knee, and (c) ankle

Figure 5.4 shows the predicted GRF for normal gait. The vertical GRF has a familiar double-peak pattern. There are decelerating and acceleration processes for fore-aft GRF. Meanwhile, the foot also pushes laterally during the entire stance phase. In Figure 5.5, the resultant internal forces (vector norm) for hip, knee, and ankle have similar patterns to the vertical GRF during stance phase of gait cycle. This is because the vertical GRF is the largest reaction force applied at COP during stance phase, and its magnitude is much larger than those of lateral and fore-aft GRF. In contrast, during swinging phase, the resultant internal forces are difference at hip, knee, and ankle joints because these joints have different motions.

This chapter discusses the proposed algorithms being applied to a complex optimization problem of human gait prediction. There were 6 types of time-dependent and 3 types of time-independent constraints, which results in a total of 1726 nonlinear constraints. A total of 330 nonlinear design variables were considered for the optimization problem. The optimized cyclic walking motion is presented along with the joint angle profiles, GRF and internal joint force vector norm for hip, knee and ankle joints are shown and discussed.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In this study, the recursive Newton-Euler formulation with analytical sensitivities were derived for dynamic motion prediction. The derived formulation was based on 3×3 DH rotation matrices and DH moving coordinates. Therefore, it is computationally efficient than the recursive Lagrangian formulation which is based on 4×4 DH transformation matrices. The proposed formulation can handle both prismatic joints and revolute joints, external force and moment, single branch and articulated branch. Three numerical examples were presented to demonstrate the efficacy and efficiency of the proposed algorithm. First, the trajectory planning problem using a simple two-link manipulator with two revolute joints was studied, where the total time was minimized subject to initial and final boundary conditions and joint torque limits. The minimized total time, predicted joint torque, and motion trajectory were quite similar to those reported in the literature. Furthermore, the internal joint reaction forces are readily obtained in the process of calculation for recursive Newton-Euler dynamics. Secondly, a similar example for a two-link manipulator with one prismatic joint and one revolute joint was examined. A MOO problem was formulated and solved for the optimal actuating force and torque. Third, a more complex example of normal walking was predicted by using a 55-DOF skeletal human model. The normal walking was considered as a cyclic and symmetric motion, where the initial and final

postures were repeatable. The recursive Newton-Euler formulation was used to build human kinematics dynamics and sensitivity equations. A two-step algorithm was used to calculate the passive GRF at ZMP based on global equilibrium. The energy-related objective function, integral of squares of all joint torques, was minimized for the dynamic walking motion prediction. Cubic B-splines were used to discretize the joint profiles, and the corresponding control points and knot vectors were treated as design variables in all three cases. An SQP algorithm in SNOPT (*snoptB* routine) was used to solve the motion planning problems. The predicted kinematic and kinetic graphs were comparable to those found in the literature, validating the robustness of the optimization formulations which used the proposed sensitivity algorithms. Finally, the recursive Newton-Euler dynamics and sensitivity equations are programmed in C++ with the latest math library (Eigen).

6.2 Future Work

Besides the foregoing work, the following topics will be studied in the future:

(1) building musculoskeletal models; (2) studying the muscle fatigue problem for repetitive motions; (3) developing a general purpose motion prediction software using the proposed algorithms.

6.2.1 Building musculoskeletal model

OpenSim muscle models will be used to develop musculoskeletal model. This, when coupled with the internal joint forces and torques provided by the new formulation, can get the motion prediction more accurately and efficiently.

6.2.2 Effect of muscle fatigue

Muscle fatigue is a common phenomenon while performing any repetitive motion that can reduce worker's efficiency. Tasks such as lifting can be subjected to fatigue for an extended period of time. The muscle fatigue model will be added to the current recursive Newton-Euler dynamics formulation to predict the repetitive lifting to reduce injury. This can be useful for both upper and lower body motion simulation for different tasks.

6.2.3 The general purpose motion prediction software

A general purpose motion prediction software will be developed for motion planning studies. This software has applications in both robotic planning field and human health and ergonomic field, such as industrial robot manipulator planning, human lifting prediction, and human-robot interaction. Finally, this real-time simulation software will be delivered as a general motion simulation and analysis tool for common motion planning tasks.

REFERENCES

[1] Anderson, K. and Hsu, Y., 2002, Analytical fully-recursive sensitivity analysis for multibody dynamic chain systems. Multibody System Dynamics, 8(1), 1-27.

[2] Ardestani, M., Moazen, M., and Jin, Z., 2015, Sensitivity analysis of human lower extremity joint moments due to changes in joint kinematics. Medical Engineering and Physics, 37, 165–174.

[3] Björkenstam, S., Leyendecker, S., Linn, J., Carlson, J. S., and Lennartson, B., 2018, Inverse dynamics for discrete geometric mechanics of multibody systems with application to direct optimal control. ASME Journal of Computational and Nonlinear Dynamics, 13(10), 101001.

[4] Chang, C., Brown, D., Bloswick, D., and Hsiang, S., 2001, Biomechanical simulation of manual lifting using spacetime optimization. Journal of Biomechanics, 34(4), 527–532.

[5] De Groote, F., Kinney, A.L., Rao, A.V., and Fregly, B.J., 2016, Evaluation of direct collocation optimal control problem formulations for solving the muscle redundancy problem. Annals of Biomedical Engineering, 44(10), 2922-2236.

[6] Denavit, J., and Hartenberg, R.S., 1955, A kinematic notation for lower-pair mechanisms based on matrices. Journal of Applied Mechanics, 22, 215-221.

[7] Dissanayake, M.W.M.G., Goh, C.J., Phan-Thien, N., 1991, Time optimal trajectories for robot manipulators. Robotica, 9, 131-138.

[8] Featherstone, R., 1987, Robot Dynamics Algorithms. Kluwer, Boston.

[9] Fu, K.S., Gonzalez, R.C., and Lee, C.S.G., 1987, Robotics: Control, Sensing, Vision, and Intelligence. McGraw-Hill, New York.

[10] Gill, P.E., Murray, W., Saunders, M.A., 2002, SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM Journal of Optimization, 12(4), 979-1006.

[11] Hollerbach, J.M., 1980, A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. IEEE Transactions on Systems, Man, and Cybernetics, 11(10), 730-736.

[12] Hsiang, S., and McGorry, R., 1997, Three different lifting strategies for controlling the motion patterns of the external load. Ergonomics, 40(9), 928–939.

[13] Inman, V.T., Ralston, R.J., Todd, F., 1981, Human Walking. Wilkins & Wilkins, Baltimore, MD.

[14] Lee, S.H., Kim, J., Park, F.C., Kim, M., and Bobrow, J.E., 2005, Newton-type algorithm for dynamics-based robot movement optimization. IEEE Transactions on Robotics, 21(4), 657-667.

[15] Lo, J., Huang, G., and Metaxas, D., 2002, Human motion planning based on recursive dynamics and optimal control techniques. Multibody System Dynamics, 8(4), 433-458.

[16] Müller, A., 2018, Screw and Lie group theory in multibody dynamics. Multibody System Dynamics, 42, 219-248.

[17] Patterson, M.A., Weinstein, M., and Rao, A.V., 2013, An efficient overloaded method for computing derivatives of mathematical functions in MATLAB. ACM Transactions on Mathematical Software, 39(3): Article 17, 1-36.

[18] Pedotti, A., Krishnan, V., and Stark, L., 1978, Optimization of Muscle-Force Sequencing in Human locomotion. Mathematical Biosciences, 38(1-2), 57–76.

[19] Reinbolt, J., Schutte, J., Fregly, B., Koh, B., Haftka, R., George, A., and Mitchell, K., 2005, Determination of patient-specific multi-joint kinematic models through two-level optimization. Journal of Biomechanics, 38, 621–626.

[20] Shabana, A.A., 2010, Computational Dynamics. John Wiley & Sons.

[21] Silva, M. and Ambrósio, J., 2004, Sensitivity of the results produced by the inverse dynamic analysis of a human stride to perturbed input data. Gait and Posture, 19, 35–49.

[22] Sohl, G.A., Bobrow, J.E., 2001, A recursive multibody dynamics and sensitivity algorithm for branched kinematic chains. ASME Journal of Dynamic Systems, Measurement, and Control, 123, 391-399.

[23] Song, J., Qu, X., and Chen, C., 2016, Simulation of lifting motions using a novel multi-objective optimization approach. International Journal of Industrial Ergonomics, 53, 37–47.

[24] Toogood, R.W., 1989, Efficient robot inverse and direct dynamics algorithms using micro-computer based symbolic generation. Proceedings of IEEE International Conference on Robotics and Automation, 3, 1827-1832.

[25] Wang, Q., Xiang, Y.J., Kim, H.J., Arora, J.S., Abdel-Malek, K., 2005, Alternative formulations for optimization-based digital human motion prediction. 2005 Digital Human Modeling for Design and Engineering Symposium, Iowa City, IA, June 14-16.

[26] Xiang, Y., Arora, J.S., and Abdel-Malek, K., 2009a, Optimization-based motion prediction of mechanical systems: sensitivity analysis. Structural and Multidisciplinary Optimization, 37, 595-608.

[27] Xiang, Y., Arora, J.S., Rahmatalla, S., and Abdel-Malek, K., 2009b, Optimization-based dynamic human walking prediction: one step formulation. International Journal for Numerical Methods in Engineering, 79(6), 667-695.

[28] Xiang, Y., Arora, J.S., Rahmatalla, S., Marler, T., Bhatt, R., and Abdel-Malek, K., 2010, Human lifting simulation using a multi-objective optimization approach. Multibody System Dynamics, 23(4), pp. 431-451.

[29] Xiang, Y., Arora, J., and Abdel-Malek, K., 2011, Optimization-based prediction of asymmetric human gait. Journal of Biomechanics, 44, 683–693.

[30] Xiang, Y., Arora, J., and Abdel-Malek, K., 2012, 3D Human Lifting Motion Prediction with Different Performance Measures. International Journal of Humanoid Robotics, 9(2), 1250012.

[31] Xiang, Y., 2019, An inverse dynamics optimization formulation with recursive B-spline derivatives and partition of unity contacts: demonstration using two-dimensional musculoskeletal arm and gait. ASME Journal of Biomechanical Engineering, 141(3), 034503.

[32] Xiang Y., Cruz, J., Zaman, R., and Yang, J., 2020, Multi-objective optimization for two-dimensional maximum weight lifting prediction considering dynamic strength. Engineering Optimization, (in press).

[33] Yang, L., Chew, C., and Poo, A., 2006, Adjustable Bipedal Gait Generation using Genetic Algorithm Optimized Fourier Series Formulation. International Conference

on Intelligent Robots and Systems, 4435–4440.

VITA

Shuvrodeb Barman

Candidate for the Degree of

Master of Science

Thesis: RECURSIVE NEWTON-EULER DYNAMICS AND SENSITIVITY ANALYSIS FOR DYNAMIC MOTION PLANNING

Major Field: Mechanical and Aerospace Engineering

Biographical:

Education:

Completed the requirements for the Master of Science in Mechanical Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2020.

Completed the requirements for the Bachelor of Science in Aeronautical Engineering at Bangladesh University of Professionals, Dhaka, Bangladesh in 2016.

Experience:
Graduate Research Assistant, Biodynamics Optimization Laboratory, Oklahoma State University (January 2019 - December 2020).

Professional Memberships: ASME