The Dynamics Model of an Innovative Pick-and-Place Robot

TR-CIM-20160404-1 April, 2016

Maged Yassin

Jorge Angeles

A thesis submitted to McGill University in partial fulfilment of the requirements of the Undergraduate Honours Program

Centre for Intelligent Machines, Department of Mechanical Engineering
McGill University, Montréal, Canada

Abstract

Pick-and-place robots play an important role in industry, for the automation of packaging and assembly tasks. Due to the increasing demand for fast manipulators, new designs of parallel robots are being developed for their distinct speed advantages when compared to their serial counterparts. Recently, a two-limbed, four-degree-of-freedom manipulator with high rotatability of its gripper and an isostatic structure was introduced. The robot produces Schönflies motions, that is, three independent translations and one rotation about an axis of fixed direction; it is therefore classed as a Schönflies motion generator (SMG). The first prototype of this design, called the PepperMill-Carrier (PMC) was produced and comissioned at McGill University, its dynamics being the subject of this thesis. Verifying an existing mathematical model, simulation results on the inverse and forward dynamics of the robot are reported in this thesis. With a focus on investigating the potential advantages of such a design over serial robots that dominate the automation industry, torque requirements for a smoothed version of an industry-adopted test cycle are reported here, while highlighting the advantages of the novel architecture.

Résumé

Les robots à transfert rapide jouent un rôle important dans l'industrie et servent à l'emballage et à l'assemblage. Afin de répondre à la demande pour des manipulateurs de plus en plus rapides, de nouveaux concepts de robots parallèles sont actuellement en développement, bénéficiant d'avantages en terme de vitesse par rapport aux robots sériels. Récemment, un manipulateur à deux jambes, quatre degrés de liberté, une importante mobilité rotationnelle de son effecteur et une structure isostatique a été proposé, cette dernière améliorant l'assemblabilité du robot. Le robot est capable de produire des déplacements du type Schönflies, c'est-à-dire trois translations indépendantes et une rotation autour d'un axe à direction fixe ; il est ainsi désigné générateur de mouvements Schönflies (GMS), ou SMG en anglais. Le premier prototype de ce concept, nommé le "PepperMill-Carrier" (PMC), dont la dynamique est le sujet de cette thèse. Vérification du modèle mathématique de l'existence, ainsi que des solutions pour les inverses et à terme la dynamique du robot sont présentés dans cette thèse. En mettant l'accent sur l'étude des avantages potentiels d'une telle conception sur des robots en série qui dominent l'industrie de l'automatisation, les exigences de couple pour une version lissée d'un cycle d'essai adoptée par l'industrie sont rapportés ici, tout en mettant en évidence les avantages de la nouvelle architecture.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Many industrial tasks, from food packaging to automotive-part assembly to electronic-circuit manufacturing, involve manipulating and accurately positioning objects. As these operations are repetitive and well-structured, the need for automation is definite. Many of these robots see use in assemblies where the geometry of the manipulated parts are usually flat. As the demand for these devices grows, the demand for faster pick-and-place robots grows correspondingly.

Common pick-and-place operations (PPO) do not require full six-dof manipulation, which belong to a category of complex robotic architectures, and are therefore more suited for specialized robots with reduced mobility. An important category of PPO robots is known as Schönflies Motion Generators (SMG), capable of producing four-dof motions of their end effector (EE), namely, three independent translations and one rotation about an axis of fixed direction, usually vertical. The first SMG were serial robots termed SCARA (Selective Compliance Assembly Robot Arm), developed by Hiroshi Makino in 1978 [1].

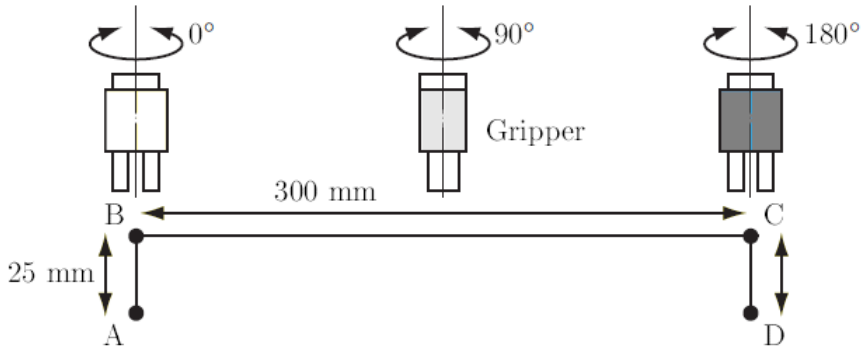

Figure 1.1: Standard Test Cycle

Currently, research into SMGs is aimed at increasing the speed with which they can perform operations. Their performance is determined by how fast they can undergo the standard trajectory illustrated in Fig.1.1, i.e., how many such operations can be conducted per second. The current record, held by Adept Technologies' Quattro, stands at five cycles per second, a cycle consisting of a 25mm vertical translation upward, a 300mm horizontal translation, a 25mm translation downward, and then following the same motions backward to arrive back at the starting position. In addition, during the first horizontal translation the moving plate turns 180°clockwise; on the second it rotates 180°counterclockwise [2].

Serial SMGs proved to be limited in their speed by the need of each motor from the base onward to drive all upstream motors and links. This motivated the development of parallel robots, composed of at least one closed kinematic chain capable of higher speeds at the cost of a higher complexity.

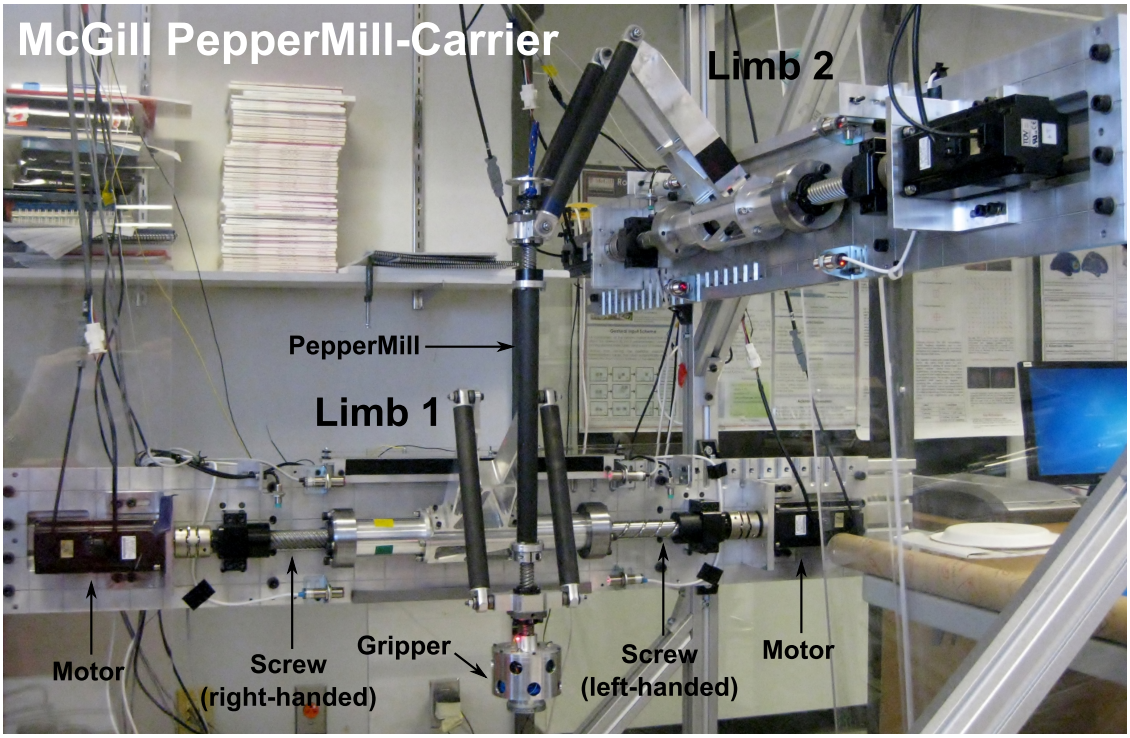

Figure 1.2: McGill's PMC Prototype

Recently, an alternative architecture for a SMG was proposed by Lee and Lee [3]. This architecture is aimed at overcoming the low rotability of the moving plate by using two helical joints to achieve the desired rotation. The architecture has a moving link, referred to as "The Peppermill", which carries the gripper. The name comes from the manner in which the link is manipulated, which is similar to the motions of a waiter dispensing pepper with a long peppermill. The whole robot can therefore be referred to as the Peppermill-carrier, or PMC. A prototype of the PMC, shown in Fig.1.2, was built at *McGill's Robotic Mechanical Systems Laboratory* to facilitate research into the implementation, kinematics, dynamics, and control of the robot with this architecture.

This thesis reports on the inverse dynamics of the McGill PMC prototype, allowing us to determine the torques to be delivered by the drive motors for any speed and path. The

forward dynamics allows predicting the trajectory of the PMC, in joint-coordinate space, given the torque profiles delivered by the motors.

Chapter 2 builds upon previous research on PMC kinematics and the derivation of its mathematical model. Chapter 3 offers an overview of the standard test cycle that the PMC is currently operating on. Chapter 4 includes the solution to the inverse dynamics of the PMC, as well as a recap over the method of the *natural orthogonal complement*. Simulation of the robot response to the results reported in the inverse dynamics solution is highlighted in chapter 5. In Chapter 6, conclusions on the work done are included, along with recommendations for the continuation of this work.

Chapter 2

Model Formulation

This SMG is a symmetric, single-loop kinematic chain of the CRRHHRRC type, where R, C and H denote revolute, cylindrical [1], and helical (screw) joints, respectively.

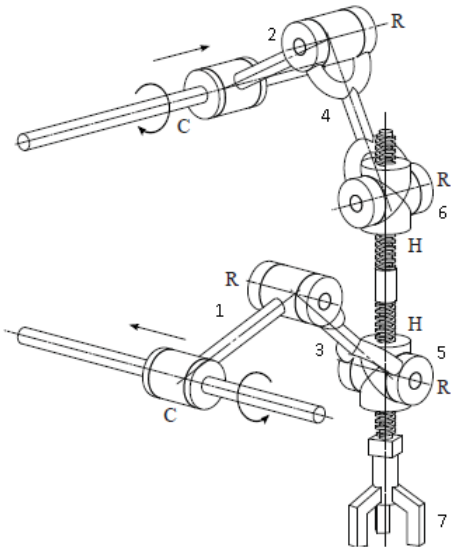

Figure 2.1: PMC Kinematic Chain

The common link to both limbs is composed of two rigidly connected coaxial screws of different pitches, and termed the PepperMill (PM) as its operation resembles that of a long pepper mill. The robot is therefore known as the PepperMill-Carrier (PMC).

[1] A cylindrical joint is capable of independent rotations about an axis and translations in the direction of the axis.

2.1 Inverse Displacement Analysis

The Inverse Displacement Analysis (IDA) [4, 5, 6] is briefly recalled here for completeness. It determines the displacements of the actuated C-joints from the pose—position (x, y, z) and orientation ϕ—of the PM.

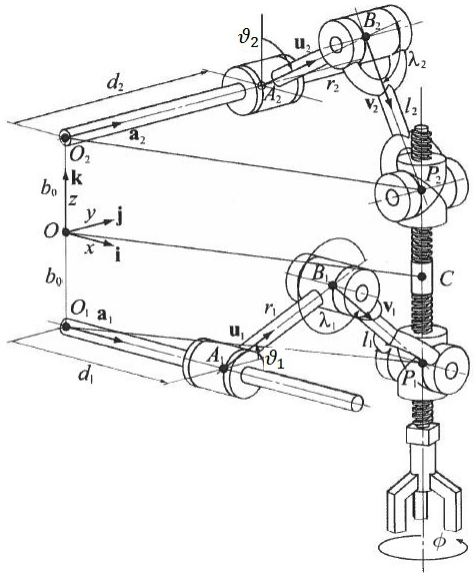

Figure 2.2: Model of the SMG showing the driving and passive joint variables

The IDA is required when the desired path of the PM is known, and the corresponding actuator displacements are needed. The x and y coordinates of the PM lead directly to the C joint translation variables d_1 and d_2:

$$d_1 = x, \quad d_2 = y \tag{2.1}$$

It is worth mentioning that symmetry relations are imposed on the link dimensions of the PMC in order to obtain a modular, simple design. The two limbs are identical, as are the lengths of the proximal and distal links. Furthermore, it is recalled [7] that the PM screw

pitches are identical, but of opposite hands. Hence, the symmetry relations are:

$$p \equiv p_1 = -p_2, \quad r \equiv r_i = l_i, \quad i = 1, 2 \tag{2.2}$$

where r_i and l_i are the lengths of the proximal and distal links, respectively, while p_1 and p_2 are the pitches of the screws belonging to the first and second limbs.

From previous results [7], the C-joint angular displacements θ_1 and θ_2 are the solutions of the relation

$$(\mathbf{q}_i^T \mathbf{x}_i) \cos\theta_i + (\mathbf{q}_i^T \mathbf{y}_i) \sin\theta_i = \frac{\mathbf{q}_i^T \mathbf{q}_i + r_i^2 - l_i^2}{2r_i} \tag{2.3}$$

with

$$\mathbf{x}_1 \equiv \mathbf{j}, \quad \mathbf{x}_2 \equiv \mathbf{k}, \quad \mathbf{y}_1 \equiv \mathbf{k}, \quad \mathbf{y}_2 \equiv \mathbf{i} \tag{2.4}$$

$$\mathbf{q}_1 \equiv \begin{bmatrix} x - d_1 \\ y \\ z - b_0 + p\phi \end{bmatrix}, \quad \mathbf{q}_2 \equiv \begin{bmatrix} x \\ y - d_2 \\ z + b_0 - p\phi \end{bmatrix} \tag{2.5}$$

while the coordinate frame unit vectors $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ are defined as in Fig. 2.2. Relation (2.3) is solved by resorting to the *tan-half identities*:

$$\cos\theta_i \equiv \frac{1 - T_i^2}{1 + T_i^2}, \quad \cos\theta_i \equiv \frac{2T_i}{1 + T_i^2}, \quad T_i \equiv \tan\frac{\theta_i}{2} \tag{2.6}$$

which allows relation (2.3) to be recast as

$$A_i \frac{1 - T_i^2}{1 + T_i^2} + B_i \frac{2T_i}{1 + T_i^2} + C_i = 0 \tag{2.7}$$

where

$$A_i \equiv \mathbf{q}_i^T \mathbf{x}_i, \quad B_i \equiv \mathbf{q}_i^T \mathbf{y}_i, \quad C_i \equiv -\frac{\mathbf{q}_i^T \mathbf{q}_i + r_i^2 - l_i^2}{2r_i} \tag{2.8}$$

Equation (2.7) is readily rearranged as a quadratic polynomial in T_i, namely,

$$(C_i - A_i)T_i^2 + 2B_i T_i + C_i + A_i = 0 \tag{2.9}$$

whose roots are

$$T_{ij} = \frac{-B_i \mp \sqrt{A_i^2 + B_i^2 - C_i^2}}{C_i - A_i}, \quad j = 1, 2 \tag{2.10}$$

The solutions for θ_i are computed from the roots T_{ij}, $j = 1, 2$, and denoted θ_{ij}:

$$\theta_{ij} = 2 \arctan \left(-B_i + (-1)^j \sqrt{A_i^2 + B_i^2 - C_i^2}, \quad C_i - A_i \right) \tag{2.11}$$

Limb Index	Elbow-up	Elbow-down
$i = 1$	$j = 2$	$j = 1$
$i = 2$	$j = 1$	$j = 2$

Table 2.1: Correspondence between j and inverse/forward conjugate postures

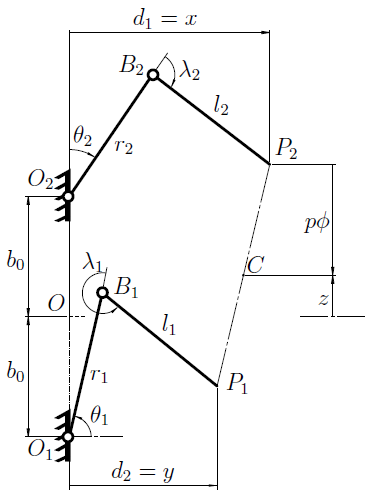

Figure 2.3: planar diagram of the PMC at an arbitrary posture

where $\arctan(\cdot, \cdot)$ is the four-quadrant arctangent function. For a given limb i, the two angles θ_{ij} correspond to the solutions for the two possible limb layouts[2]; each limb i can be oriented with the distal link pointing either up or down with respect to the ground, termed the *up* and *down* postures, respectively. The correspondence between j and the limb posture is given in Table 2.1. Figures 2.2 and 2.3 show an up-up inverse kinematic posture.

For the prototype, θ_1 and θ_2 take the values of θ_{11} and θ_{22}, respectively. Plugging in the design specification from eq. (2.2) yields the closed-form solutions

$$\theta_1 = 2\arctan\left(-P_{1z} - \frac{1}{2}\sqrt{4y^2 + 4P_{1z}^2 - \frac{\left(y^2 + P_{1z}^2\right)^2}{r^2}}, \quad -\frac{y^2 + P_{1z}^2}{2r} - y\right) \tag{2.12}$$

$$\theta_2 = 2\arctan\left(-x + \frac{1}{2}\sqrt{4x^2 + 4P_{2z}^2 - \frac{\left(x^2 + P_{2z}^2\right)^2}{r^2}}, \quad -\frac{x^2 + P_{2z}^2}{2r} - P_{2z}\right) \tag{2.13}$$

[2] θ_{i1} and θ_{i2} are equal when the proximal and distal link of the i^{th} limb are parallel.

where P_{1z} and P_{2z} are the z-coordinates of points P_1 and P_2 from Fig. 2.2, namely

$$P_{1z} \equiv z + b_0 + p\phi, \quad P_{2z} \equiv z - b_0 - p\phi \tag{2.14}$$

A change of variables is required here to maintain consistency throughout this document as attention will be brought to the motor angular displacements, velocities, and torques. The PMC is driven by four identical motors: motors 1 and 3 actuate the first C drive while 2 and 4 the second C drive, as numbered in Fig. 2.2.

A new vector $\boldsymbol{\theta_a}$ is defined to carry angular displacements of the four motors; thus, eqs. (2.12) and (2.13) can be finally recast as[3]

$$\boldsymbol{\theta_a} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = 2 \arctan \begin{bmatrix} -P_{1z} - \frac{1}{2}\sqrt{4y^2 + 4P_{1z}^2 - \frac{(y^2 + P_{1z}^2)^2}{r^2}} \\ -x + \frac{1}{2}\sqrt{4x^2 + 4P_{2z}^2 - \frac{(x^2 + P_{2z}^2)^2}{r^2}} \\ -\frac{y^2 + P_{1z}^2}{2r} - y \\ -\frac{x^2 + P_{2z}^2}{2r} - P_{2z} \end{bmatrix} \tag{2.15}$$

2.2 Mathematical Model

Starting from defining the twists of the seven moving rigid bodies to the robot equation of motion, Klingenberg derived the robot mathematical model [8] using the methodology of the *natural orthogonal complement*, as recalled below,

The twist of a rigid body is defined as a representation of the body velocity field, in terms of the body angular velocity and the velocity of one specific body point. In dynamics, this

[3]Notice the use of an abusive shorthand notation in eq. (2.15), as the arctan of a vector doesn't actually exist.

point is chosen to be the centre of mass (c.o.m.) of the link. The twist is thus expressed as:

$$\mathbf{t} = \begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{bmatrix} \tag{2.16}$$

where $\boldsymbol{\omega}$ represents the body angular velocity and $\mathbf{v}$ the velocity of the body c.o.m.

From there, a *twist-shaping matrix* can be defined as mapping the actuated-joint rates into the link twists. The relationship between the twist of the ith link and the vector $\dot{\boldsymbol{\theta}}_a$ of joint rates is, thus,

$$\mathbf{t}_i = \mathbf{T}_i \dot{\boldsymbol{\theta}}_a \tag{2.17}$$

Since we have both $\mathbf{t}_i$ and $\dot{\boldsymbol{\theta}}_a$, we can find the 6×4 twist-shaping matrix $\mathbf{T}_i$ that performs the mapping in the form

$$\mathbf{T}_i = \frac{\partial \mathbf{t}_i}{\partial \dot{\boldsymbol{\theta}}_a} \tag{2.18}$$

The kinetic energy of the the ith rigid body can be expressed as

$$T_i = \frac{1}{2} \mathbf{t}_i^T \mathbf{M}_i \mathbf{t}_i \tag{2.19}$$

where $\mathbf{M}_i$ is the constant 6×6 *inertia dyad* of the ith rigid body. This matrix is defined in terms of the 3×3 moment-of-inertia matrix $\mathbf{I}_i$ computed at the body centre of mass, and

the mass m_i of the ith rigid body:

$$\mathbf{M}_i \equiv \begin{bmatrix} \mathbf{I}_i & \mathbf{O}_{3x3} \\ \mathbf{O}_{3x3} & m_i \mathbf{1}_{3x3} \end{bmatrix} \tag{2.20}$$

where $\mathbf{I}_i$ is expressed in a body-fixed frame. Thus the 4×4 generalized inertia matrix for the ith rigid body becomes

$$\mathbf{I}_i = \mathbf{T}_i^T \mathbf{M}_i \mathbf{T}_i \tag{2.21}$$

The matrix of the Coriolis and centrifugal forces associated with the ith body is defined as:

$$\mathbf{C}_i = \mathbf{T}_i^T \mathbf{M}_i \dot{\mathbf{T}}_i + \mathbf{T}_i^T \mathbf{W}_i \mathbf{M}_i \mathbf{T}_i \tag{2.22}$$

where $\dot{\mathbf{T}}_i$ denotes the time-derivative of the twist-shaping matrix of the ith rigid body, and $\mathbf{W}_i$ is the 6×6 *angular velocity dyad* of the ith rigid body, which is known [9] to be:

$$\mathbf{W}_i \equiv \begin{bmatrix} \mathbf{\Omega}_i & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \mathbf{0}_{3x3} \end{bmatrix} \tag{2.23}$$

where $\mathbf{\Omega}_i$ is the 3×3 cross-product matrix of $\boldsymbol{\omega_i}$, defined as:

$$\boldsymbol{\omega}_i \times \mathbf{r} = \mathbf{\Omega_i r} \quad \forall \quad \mathbf{r} \in \mathbb{R}^3 \tag{2.24}$$

Since the system kinetic energy is additive, the 4×4 generalized inertia matrix $\mathbf{I}$ and its counterpart $\mathbf{C}$ of Coriolis and centrifugal forces of the whole system are also additive, we have,

$$\mathbf{I} = \sum_{i=1}^{7} \mathbf{T}_i^T \mathbf{M}_i \mathbf{T}_i, \quad \mathbf{C} = \sum_{i=1}^{7} (\mathbf{T}_i^T \mathbf{M}_i \dot{\mathbf{T}}_i + \mathbf{T}_i^T \mathbf{W}_i \mathbf{M}_i \mathbf{T}_i) \tag{2.25}$$

where $\mathbf{I}$ is a function of the vector of actuated-joint variables $\boldsymbol{\theta}_a$, while $\mathbf{C}$ is a function of both $\boldsymbol{\theta}_a$ and the vector of actuated-joint rates $\dot{\boldsymbol{\theta}}_a$.

This brings us to the mathematical model:

$$\mathbf{I}(\boldsymbol{\theta}_a)\ddot{\boldsymbol{\theta}}_a + \mathbf{C}(\boldsymbol{\theta}_a, \dot{\boldsymbol{\theta}}_a)\dot{\boldsymbol{\theta}}_a = \boldsymbol{\tau} + \boldsymbol{\delta} + \boldsymbol{\gamma} \tag{2.26}$$

where $\boldsymbol{\tau}$ is the four-dimensional array of motor torques, $\boldsymbol{\delta}$ represents the dissipative and disturbance generalized forces, and $\boldsymbol{\gamma}$ the gravitational generalized forces.

The PMC consists of two proximal links, two distal links, two peppermill nuts and the peppermill itself, described below with their twist shaping matrices listed as functions of the actuated joint variables and rates solely.

2.2.1 Proximal Links

The twist-shaping matrices $\boldsymbol{T}_i$ carry the subscript assigned to each link, as per the numbering in Fig. 2.1. The proximal link of each limb is driven by a cylindrical joint, a combination of two joints, a prismatic and a revolute. The twist-shaping matrix of the proximal

link of each of the two arms, denoted $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$, are shown below.

$$\mathbf{T}_1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{p}{4\pi} & \frac{-p}{4\pi} & 0 & 0 \\ \frac{-\|\boldsymbol{\rho}_{I1}\|}{2}\sin\left(\frac{\theta_1+\theta_2}{2}\right) & \frac{-\|\boldsymbol{\rho}_{I1}\|}{2}\sin\left(\frac{\theta_1+\theta_2}{2}\right) & 0 & 0 \\ \frac{\|\boldsymbol{\rho}_{I1}\|}{2}\cos\left(\frac{\theta_1+\theta_2}{2}\right) & \frac{\|\boldsymbol{\rho}_{I1}\|}{2}\cos\left(\frac{\theta_1+\theta_2}{2}\right) & 0 & 0 \end{bmatrix} \tag{2.27}$$

$$\mathbf{T}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-\|\boldsymbol{\rho}_{II1}\|}{2}\cos\left(\frac{\theta_3+\theta_4}{2}\right) & \frac{-\|\boldsymbol{\rho}_{II1}\|}{2}\cos\left(\frac{\theta_3+\theta_4}{2}\right) \\ 0 & 0 & \frac{-p}{4\pi} & \frac{p}{4\pi} \\ 0 & 0 & \frac{\|\boldsymbol{\rho}_{II1}\|}{2}\sin\left(\frac{\theta_3+\theta_4}{2}\right) & \frac{\|\boldsymbol{\rho}_{II1}\|}{2}\sin\left(\frac{\theta_3+\theta_4}{2}\right) \end{bmatrix} \tag{2.28}$$

2.2.2 Distal Links

The distal links act as passive connection between the proximal links and the peppermill nuts; their twist-shaping matrices, $\boldsymbol{T}_3$ and $\boldsymbol{T}_4$ are

$$\mathbf{T}_3 = \begin{bmatrix} \mathbf{a}_{I2} & \mathbf{b}_{I2} & \mathbf{c}_{I2} & \mathbf{d}_{I2} \end{bmatrix} \tag{2.29}$$

whose blocks are defined below:

$$\mathbf{a}_{I2} = \begin{bmatrix} \dfrac{-r_1 \sin\left(\dfrac{\theta_1+\theta_2}{2}\right)}{2l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \\ 0 \\ 0 \\ \dfrac{p}{4\pi} \\ \dfrac{r_1}{2}\sin\left(\dfrac{\theta_1+\theta_2}{2}\right)\left(\dfrac{\|\boldsymbol{\rho}_{I2}\|}{l_1}-1\right) \\ \dfrac{-\|\boldsymbol{\rho}_{I2}\|}{16\pi^2}\dfrac{\sin\left(\dfrac{\theta_1+\theta_2}{2}\right)\left(2p\pi r_1(\theta_4-\theta_3)-8r_1^2\pi^2\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)}{l_1^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}}+\dfrac{r_1}{2}\cos\left(\dfrac{\theta_1+\theta_2}{2}\right) \end{bmatrix} \tag{2.30}$$

$$\mathbf{b}_{I2} = \begin{bmatrix} \dfrac{-r_1 \sin\left(\dfrac{\theta_1+\theta_2}{2}\right)}{2l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \\ 0 \\ 0 \\ \dfrac{-p}{4\pi} \\ \dfrac{r_1}{2}\sin\left(\dfrac{\theta_1+\theta_2}{2}\right)\left(\dfrac{\|\boldsymbol{\rho}_{I2}\|}{l_1}-1\right) \\ \dfrac{-\|\boldsymbol{\rho}_{I2}\|}{16\pi^2}\dfrac{\sin\left(\dfrac{\theta_1+\theta_2}{2}\right)\left(2p\pi r_1(\theta_4-\theta_3)-8r_1^2\pi^2\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)}{l_1^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}}+\dfrac{r_1}{2}\cos\left(\dfrac{\theta_1+\theta_2}{2}\right) \end{bmatrix} \tag{2.31}$$

$$\mathbf{c}_{I2} = \begin{bmatrix} \dfrac{p}{4\pi l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \\ 0 \\ 0 \\ 0 \\ \dfrac{-p\,\|\boldsymbol{\rho}_{I2}\|}{4\pi l_1} \\ \dfrac{-\|\boldsymbol{\rho}_{I2}\|}{16\pi^2}\dfrac{\left(p^2(\theta_3-\theta_4)+4p\pi r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)}{l_1^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \end{bmatrix} \tag{2.32}$$

$$\mathbf{d}_{I2} = \begin{bmatrix} \dfrac{-p}{4\pi l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \\ 0 \\ 0 \\ 0 \\ \dfrac{p\,\|\boldsymbol{\rho}_{I2}\|}{4\pi l_1} \\ \dfrac{-\|\boldsymbol{\rho}_{I2}\|}{16\pi^2}\dfrac{\left(p^2(\theta_4-\theta_3)-4p\pi r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)}{l_1^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} \end{bmatrix} \tag{2.33}$$

The twist-shaping matrix of the second arm distal link is

$$\mathbf{T}_4 = \begin{bmatrix} \mathbf{a}_{II2} & \mathbf{b}_{II2} & \mathbf{c}_{II2} & \mathbf{d}_{II2} \end{bmatrix} \tag{2.34}$$

its blocks being

$$\mathbf{a}_{II2} = \begin{bmatrix} 0 \\ \dfrac{p}{4\pi l_2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_1-\theta_2)-r_2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)^2}{l_2^2}}} \\ 0 \\ \dfrac{p\,\|\boldsymbol{\rho}_{II2}\|}{4\pi l_2} \\ 0 \\ \dfrac{\|\boldsymbol{\rho}_{II2}\|}{16\pi^2}\dfrac{p^2(\theta_2-\theta_1)+4p\pi r_2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)}{l_2^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_1-\theta_2)-r_2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)^2}{l_2^2}}} \end{bmatrix} \tag{2.35}$$

$$\mathbf{b}_{II2} = \begin{bmatrix} 0 \\ \dfrac{-p}{4\pi l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} \\ 0 \\ \dfrac{-p \left\| \boldsymbol{\rho}_{II2} \right\|}{4\pi l_2} \\ 0 \\ \dfrac{\left\| \boldsymbol{\rho}_{II2} \right\|}{16\pi^2} \dfrac{p^2(\theta_1 - \theta_2) - 4p\pi r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)}{l_2^2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} \end{bmatrix} \tag{2.36}$$

$$\mathbf{c}_{II2} = \begin{bmatrix} 0 \\ \dfrac{-r_2 \cos\left(\dfrac{\theta_3 + \theta_4}{2}\right)}{2l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} \\ 0 \\ \dfrac{r_2}{2} \cos\left(\dfrac{\theta_3 + \theta_4}{2}\right) \left(\dfrac{\left\| \boldsymbol{\rho}_2 \right\|}{l_2} + 1\right) \\ \dfrac{-p}{4\pi} \\ \dfrac{\left\| \boldsymbol{\rho}_{II2} \right\|}{16\pi^2} \cos\left(\dfrac{\theta_3 + \theta_4}{2}\right) \dfrac{2p\pi r_2(\theta_1 - \theta_2) - 8\pi^2 r_2^2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)}{l_2^2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} - \dfrac{r_2}{2} \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right) \end{bmatrix} \tag{2.37}$$

$$\mathbf{d}_{II2} = \begin{bmatrix} 0 \\ \dfrac{-r_2 \cos\left(\dfrac{\theta_3+\theta_4}{2}\right)}{2l_2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_1-\theta_2)-r_2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)^2}{l_2^2}}} \\ \dfrac{r_2}{2}\cos\left(\dfrac{\theta_3+\theta_4}{2}\right)\left(\dfrac{\|\boldsymbol{\rho}_{II2}\|}{l_2}+1\right) \\ \dfrac{p}{4\pi} \\ \dfrac{\|\boldsymbol{\rho}_{II2}\|}{16\pi^2}\cos\left(\dfrac{\theta_3+\theta_4}{2}\right)\dfrac{2p\pi r_2(\theta_1-\theta_2)-8\pi^2 r_2^2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)}{l_2^2\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_1-\theta_2)-r_2\sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)^2}{l_2}}}-\dfrac{r_2}{2}\sin\left(\dfrac{\theta_3+\theta_4}{2}\right) \end{bmatrix} \tag{2.38}$$

2.2.3 Peppermill Nuts

The peppermill nuts are the links driving the Peppermill. From Fig. 2.2, it can be seen that these nuts P_1 and P_2, are constrained to move under pure translation, The twist-shaping matrix of the nut of limb I is:

$$\mathbf{T}_5 = \begin{bmatrix} \mathbf{a}_{I3} & \mathbf{b}_{I3} & \mathbf{c}_{I3} & \mathbf{d}_{I3} \end{bmatrix} \tag{2.39}$$

with

$$\mathbf{a}_{I3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{p}{4\pi} \\ 0 \\ \frac{-\sin\left(\frac{\theta_1+\theta_2}{2}\right)\left(2p\pi r_1(\theta_4-\theta_3) - 8r_1^2\pi^2\cos\left(\frac{\theta_1+\theta_2}{2}\right)\right)}{16\pi^2 l_1\sqrt{1-\frac{\left(\frac{p}{4\pi}(\theta_4-\theta_3) - r_1\cos\left(\frac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} + \frac{r_1}{2}\cos\left(\frac{\theta_1+\theta_2}{2}\right) \end{bmatrix} \tag{2.40}$$

$$\mathbf{b}_{I3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{-p}{4\pi} \\ 0 \\ \frac{-\sin\left(\frac{\theta_1+\theta_2}{2}\right)\left(2p\pi r_1(\theta_4-\theta_3) - 8r_1^2\pi^2\cos\left(\frac{\theta_1+\theta_2}{2}\right)\right)}{16\pi^2 l_1\sqrt{1-\frac{\left(\frac{p}{4\pi}(\theta_4-\theta_3) - r_1\cos\left(\frac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}}} + \frac{r_1}{2}\cos\left(\frac{\theta_1+\theta_2}{2}\right) \end{bmatrix} \tag{2.41}$$

$$\mathbf{c}_{I3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dfrac{-p}{4\pi} \\ -\left(p^2(\theta_3-\theta_4)+4p\pi r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right) \\ \hline 16\pi^2 l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)^2}{l_1^2}} \end{bmatrix} \tag{2.42}$$

$$\mathbf{d}_{I3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dfrac{p}{4\pi} \\ -\left(p^2(\theta_4-\theta_3)-4p\pi r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right) \\ \hline 16\pi^2 l_1\sqrt{1-\dfrac{\left(\dfrac{p}{4\pi}(\theta_4-\theta_3)-r_1\cos\left(\dfrac{\theta_1+\theta_2}{2}\right)\right)}{l_1^2}} \end{bmatrix} \tag{2.43}$$

The twist-shaping matrix of the nut on the second chain is

$$\mathbf{T}_6 = \begin{bmatrix} \mathbf{a}_{II3} & \mathbf{b}_{II3} & \mathbf{c}_{II3} & \mathbf{d}_{II3} \end{bmatrix} \tag{2.44}$$

with

$$\mathbf{a}_{II3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dfrac{p}{4\pi} \\ 0 \\ \dfrac{p^2(\theta_2 - \theta_1) + 4p\pi r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)}{16\pi^2 l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} \end{bmatrix} \tag{2.45}$$

$$\mathbf{b}_{II3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dfrac{-p}{4\pi} \\ 0 \\ \dfrac{p^2(\theta_1 - \theta_2) - 4p\pi r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)}{16\pi^2 l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} \end{bmatrix} \tag{2.46}$$

$$\mathbf{c}_{II3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dfrac{-p}{4\pi} \\ \dfrac{\cos\left(\dfrac{\theta_3 + \theta_4}{2}\right)\left(2p\pi r_2(\theta_1 - \theta_2) - 8\pi^2 r_2^2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)}{16\pi^2 l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1 - \theta_2) - r_2 \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right)\right)^2}{l_2^2}}} - \dfrac{r_2}{2} \sin\left(\dfrac{\theta_3 + \theta_4}{2}\right) \end{bmatrix} \tag{2.47}$$

$$\mathbf{d}_{II3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dfrac{p}{4\pi} \\ \dfrac{\cos\left(\dfrac{\theta_3+\theta_4}{2}\right)\left(2p\pi r_2(\theta_1-\theta_2) - 8\pi^2 r_2^2 \sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)}{16\pi^2 l_2 \sqrt{1 - \dfrac{\left(\dfrac{p}{4\pi}(\theta_1-\theta_2) - r_2 \sin\left(\dfrac{\theta_3+\theta_4}{2}\right)\right)^2}{l_2^2}}} - \dfrac{r_2}{2}\sin\left(\dfrac{\theta_3+\theta_4}{2}\right) \end{bmatrix} \tag{2.48}$$

2.2.4 The Peppermill

The Peppermill is capable of translation and rotation about one axis of fixed direction, the latter linearly related with the translation in the same direction by the common pitch p_3 of the threads on the Peppermill, with $p_4 = -p_3 = -p$. Therefore, the twist-shaping matrix of the Peppermill is

$$\mathbf{T}_7 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\dfrac{1}{2p_3}(E_1 - F_1) & -\dfrac{1}{2p_3}(E_2 - F_2) & -\dfrac{1}{2p_3}(E_3 - F_3) & -\dfrac{1}{2p_3}(E_4 - F_4) \\ \dfrac{p}{4\pi} & -\dfrac{p}{4\pi} & 0 & 0 \\ 0 & 0 & -\dfrac{p}{4\pi} & \dfrac{p}{4\pi} \\ E_1 + F_1 & E_2 + F_2 & E_3 + F_3 & E_4 + F_4 \end{bmatrix} \tag{2.49}$$

where E_i and F_i are the entries in the 6th row and ith column of the twist-shaping matrices $\mathbf{T}_5$ and $\mathbf{T}_6$ respectively.

Chapter 3

Standard Test Cycle

The PMC prototype is intended to go through the standard test-cycle trajectory described in Fig. 1.1, which is commonly adopted by the industry to represent the standard pick-and place task to be performed by a Schönflies motion generator. It is highly desirable that the PPO be a smooth trajectory, which is accomplished by ensuring the continuity of the curve, its tangent and its curvature. This cannot be accomplished by implementing a test cycle with corners between horizontal and vertical segments, like the one shown in Fig. 1.1; these corners are obvious sources of acceleration discontinuities.

3.1 Trajectory Displacement Profile

The above-mentioned trajectory was smoothed for a SCARA robot [10] using a combination of Lamé curves and a 4-5-6-7 polynomial displacement program; a similar technique was carried out to optimize the PMC trajectory. The trajectory is optimized by minimizing the variation of the kinetic energy of the payload alone with the goal of maintaining a G^2-continuity[1]. In order to achieve a G^2-continuity, a compromise between accelerations and

[1] G^2-continuity means position, tangent and curvature continuity of a given geometric curve.

maximum velocities is made.

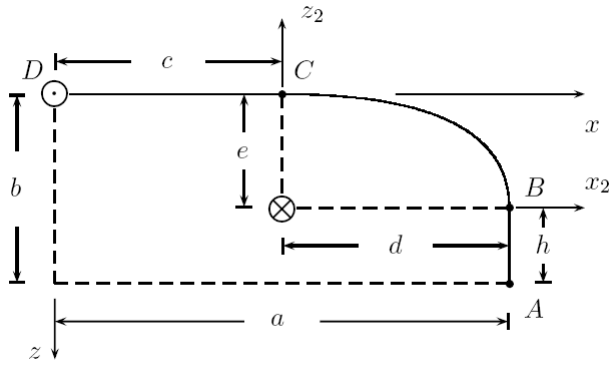

Figure 3.1: Half of the Optimized Trajectory [10]

The first half of the optimum trajectory is shown in Fig. 3.1, where t_{AD} is one quarter of the cycle time, a and b are set at 150 mm and 25 mm, respectively. The curve BC is used to blend the vertical and horizontal portions of the trajectory, thus eliminating the discontinuities. For the synthesis of the curve BC, Lamé curves were used because they allow a parametrization of the curve in Cartesian coordinates and are analytic throughout the required path.

Lamé curves are generally defined by the following equation:

$$|u|^m + |v|^m = 1 \quad m = 1, 2, ... \tag{3.1}$$

It was shown in [10] that two lines cannot be blent smoothly by a quadratic Lamé curve, i.e., a circle; thus, a cubic Lamé curve was chosen since it is the next curve of this family that allows for such blending. Therefore, looking at Fig. 3.1, for given parameters d and e

in the coordinate frame x_2-z_2, the equation of the cubic Lamé curve becomes

$$\left|\frac{x_2}{d}\right|^3 + \left|\frac{z_2}{e}\right|^3 = 1 \tag{3.2}$$

The above equation can be parametrized by defining θ to be the angle measured with respect to the x_2-coordinate axis, defined positive counterclockwise.

$$x_2(\theta) = d/(1+\tan^3\theta)^{1/3}, \quad z_2(\theta) = e\ \tan\theta/(1+\tan^3\theta)^{1/3} \tag{3.3}$$

the length of the Lamé curve being given by

$$l_c = \int dl = \int \sqrt{{dx_2}^2 + {dz_2}^2} = \int_0^{\pi/4} \frac{1+\tan^2\theta}{(1+\tan^3\theta)^{4/3}} \sqrt{d^2\tan^4\theta + e^2} d\theta + \int_0^{\pi/4} \frac{1+\tan^2\theta}{(1+\tan^3\theta)^{4/3}} \sqrt{e^2\tan^4\theta + d^2} d\theta \tag{3.4}$$

This leaves us with c and h as the only two variables that need to be found via optimization in order to fully define the trajectory, which is basically where the Lamé curve should start and where it should end.

3.2 Trajectory Velocity Profile

After the geometry is parametrized with respect to the displacement s along the trajectory using cubic Lamé curves, the velocity profile is then optimized using a 4-5-6-7 polynomial

displacement program, namely [10],

$$p(\tau) = -20\tau^7 + 70\tau^6 - 84\tau^5 + 35\tau^4, \quad 0 \leq \tau \leq 1 \tag{3.5}$$

where $p(\tau)$ is a nondimensional function of a nondimensional argument $\tau = t/T$ as a nondimensional time parameter, t is time, and T is the time elapsed in traversing the trajectory in one direction. The time taken to produce one full cycle is, thus, $2T$. A characteristic of this polynomial is that it provides zero jerk at the start and the end of the motion, which means that the polynomial is C^3-continuous[2].

Notice that $p'(\tau)$ starts with a value of zero at $\tau = 0$, reaches a maximum value of 35/16 at $\tau = 0.5$ and then returns to zero at $\tau = 1$.

Using suitable scaling factors, the displacement $s_a(t)$, velocity, and acceleration of the Peppermill motion program in Cartesian coordinates is related to τ, namely,

$$s_a(t) = 2Sp(\tau) \tag{3.6}$$

$$\dot{s_a}(t) = (S/t_{AD})\frac{dp(\tau)}{d\tau} \tag{3.7}$$

$$\ddot{s_a}(t) = (S/2{t_{AD}}^2)\frac{d^2p(\tau)}{d\tau^2} \tag{3.8}$$

$$S = h + l_c + c \tag{3.9}$$

$$\tau = t/2t_{AD}, \quad 0 < t < t_{AD} \tag{3.10}$$

Where S is the scaling factor for the translational displacement, with a similar scaling factor Φ for the rotational displacement, with $\phi_a(t) = \Phi p(\tau)$, and $\Phi_a = \pi/2$

[2] C^3-continuous polynomials have smooth first-, second-, and third-order derivatives.

3.3 The Optimum Trajectory for the Test Cycle

The trajectory, as defined earlier, is only missing two variables to determine, c and h, which denote where the Lamé curve starts and where it ends. In order to determine these two values, an optimization problem was formulated by Gauthier et al. [10] with the goal of minimizing the payload kinetic energy throughout 1/4 of a cycle[3].

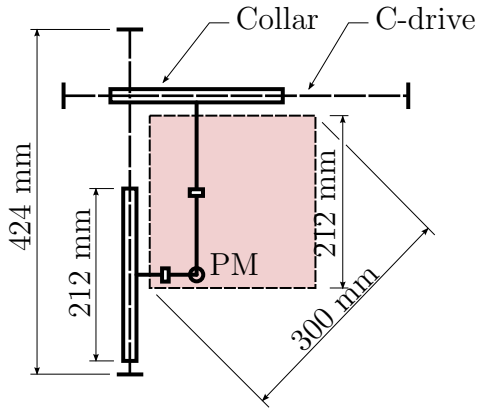

Figure 3.2: Schematic top view of the PMC.

Due to the limited space available for the PMC in the RMSlab, the horizontal segment of the trajectory was diagonally accommodated across the x-y plane, as shown in Fig. 3.2, where the xy workspace of the PM is represented by the shaded square.

The kinetic energy The kinetic energy under minimization is minimal when $h = 0$mm and $c = 120$mm; hence, a closed-form expression was found for the trajectory displacement and velocity profiles, as plotted in Fig. 3.3, and its time history displayed in the plots of Fig. 3.4.

[3]This is all that is needed, given the symmetries of the trajectory.

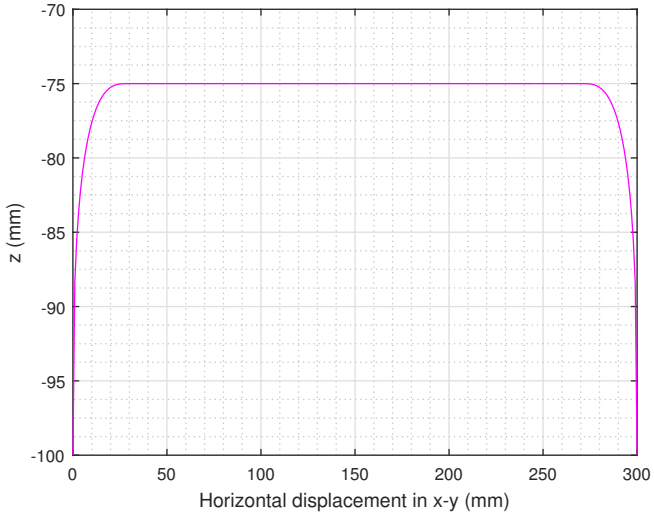

Figure 3.3: Path generated using the optimum trajectory

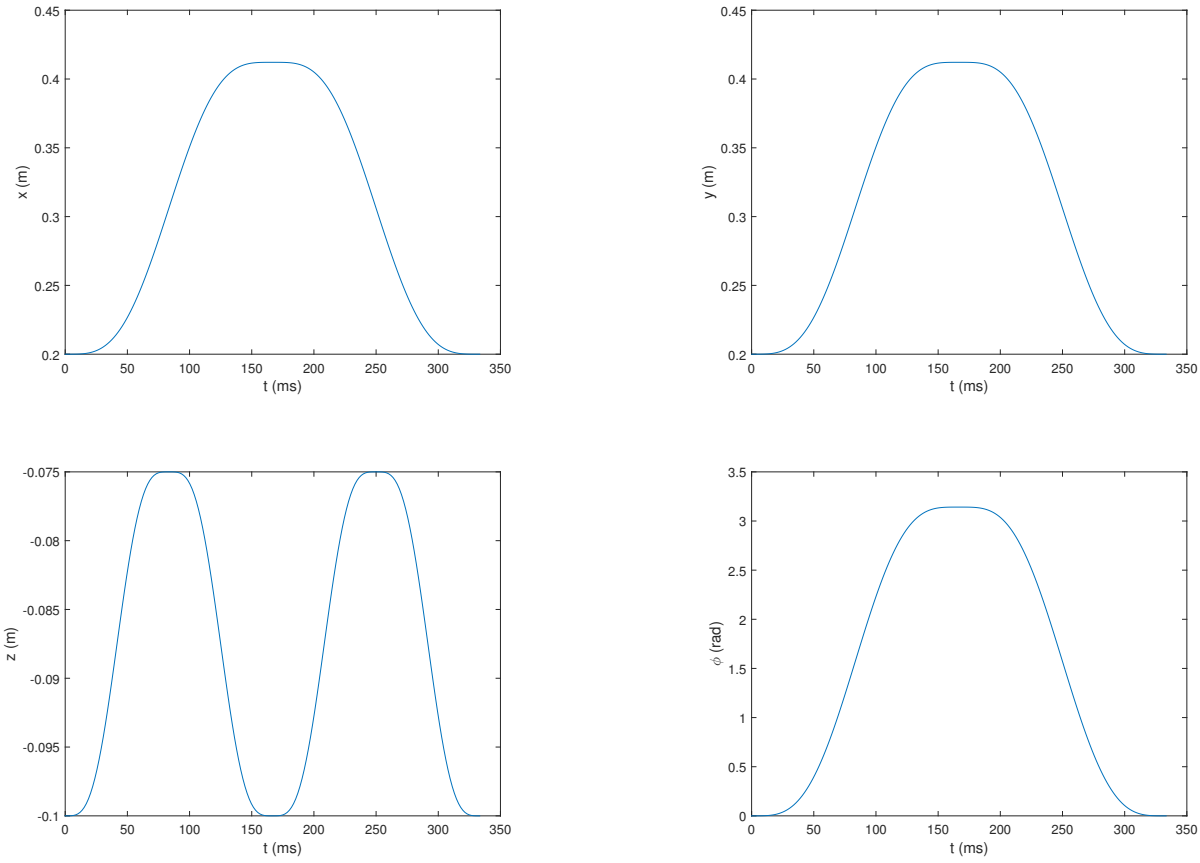

Figure 3.4: Cartesian variables along the test-cycle trajectory.

Chapter 4

Inverse Dynamics

Inverse dynamics is the process of calculating kinetic information (forces and moments) from given kinematic information (position, velocity, and acceleration) on all the moving links. This process is extremely valuable in the fields of robotics and bio-mechanics. In robotics, inverse dynamics algorithms are used to calculate the torques that a robot motors must deliver to make the robot end-point move in the way prescribed by a given task. In our case, a time-history of either the Cartesian or the joint coordinates is given; from knowledge of these histories and the architecture and inertial parameters of the system at hand, the torque or force requirements at the different actuated joints are determined as time-histories as well [9], it is essential for the computed-torque control of robotic manipulators.

The study of the dynamics of systems of multiple rigid bodies is classical, but up until the advent of the computer, it was limited only to theoretical results and a reduced number of bodies [9]. First a method based on the Euler-Lagrange equations of mechanical systems of rigid bodies was used by Uicker (1965) and then Kahn (1969) to simulate the dynamical behaviour of such systems. It wasn't long before Luh et al. came up with a breakthrough in the development of algorithms for dynamics computations in 1980 by proposing a recursive formulation of multibody dynamics, based on the Newton-Euler equations of rigid bodies,

applicable to systems with open kinematic chains. A technique, developed with the aid of a modeling tool known as the *natural orthogonal complement* (OC) [11], was used to obtain these results, where $\mathbf{B} \in \mathbb{R}^{n\times p}$ is said to be *an OC* of $\mathbf{A} \in \mathbb{R}^{m\times n}$ if the columns of $\mathbf{B}$ span the null space of $\mathbf{A}$, i.e., if $\mathbf{AB} = \mathbf{O}_{m\times p} \in \mathbb{R}^{m\times p}$. Notice that any multiple of $\mathbf{B}$, or the matrix resulting from a reshuffling of the columns of $\mathbf{B}$, is also an OC of $\mathbf{A}$, and hence, the concept *is not unique*.

The profiles obtained in Cartesian coordinates in Fig. 3.4 can be transformed into joint space coordinates as defined in eq. (2.15), which are then plotted in Fig. 4.1.

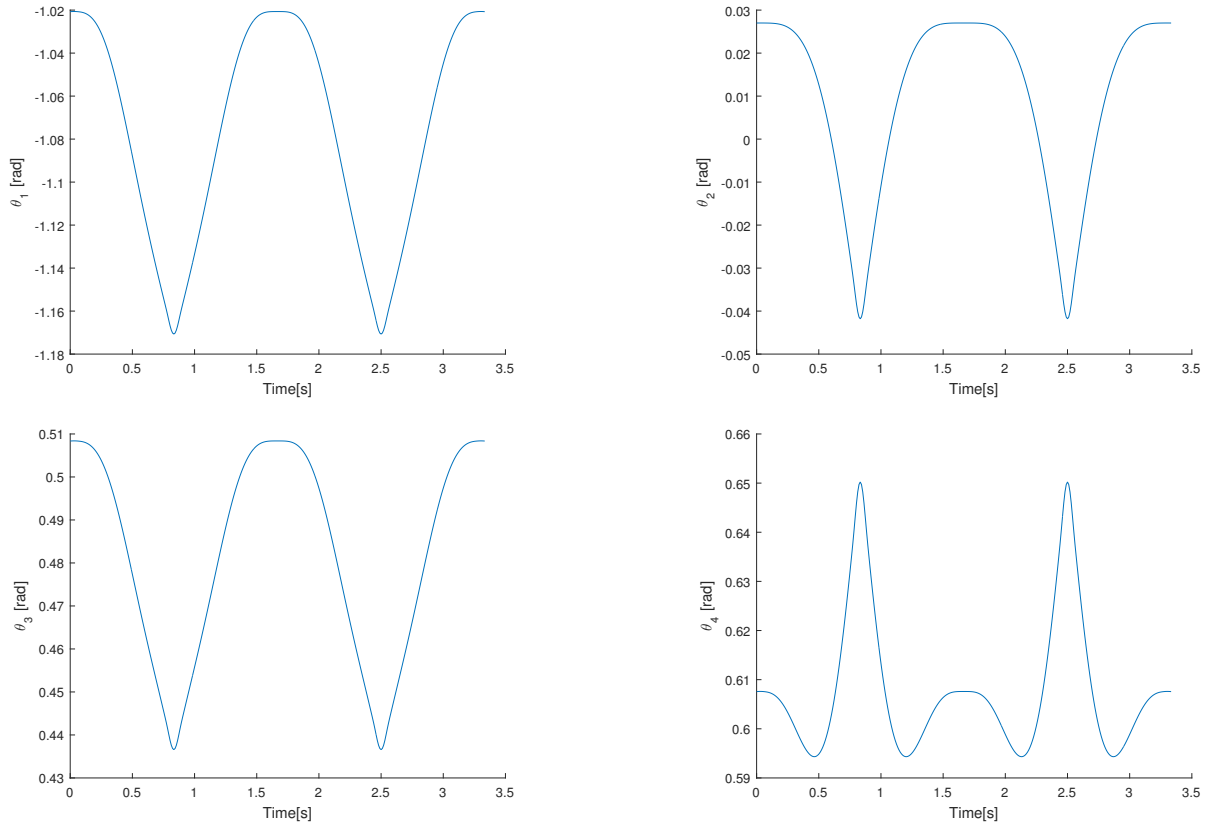

Figure 4.1: Optimum trajectory in joint space

Equation (2.26) can be recast in the form

$$\boldsymbol{\tau} = \mathbf{I}\ddot{\boldsymbol{\theta}}_a + \mathbf{C}\dot{\boldsymbol{\theta}}_a - \boldsymbol{\gamma}, \quad \boldsymbol{\gamma} = \sum_{i=1}^{7} \boldsymbol{t}_i^T \boldsymbol{w}_i^{\boldsymbol{G}} \tag{4.1}$$

$$\boldsymbol{w}_i^{\boldsymbol{G}} = \begin{bmatrix} \tau_{xi} & \tau_{yi} & \tau_{zi} & 0 & 0 & -m_i g \end{bmatrix}^T \tag{4.2}$$

where $\boldsymbol{w}_i^{\boldsymbol{G}}$ represents the gravitational wrench of the i^{th} limb, and $\boldsymbol{\tau}$ is the four-dimensional array of all motor torques required to produce the prescribed trajectory in Cartesian space.

The inverse dynamics of a robotic system is purely algebraic, which makes it easier to address than the forward dynamics problem. thus Equation (4.1) was implemented in Mathematica as per the code in the Appendix, the resulting torques being plotted in Fig. 4.2.

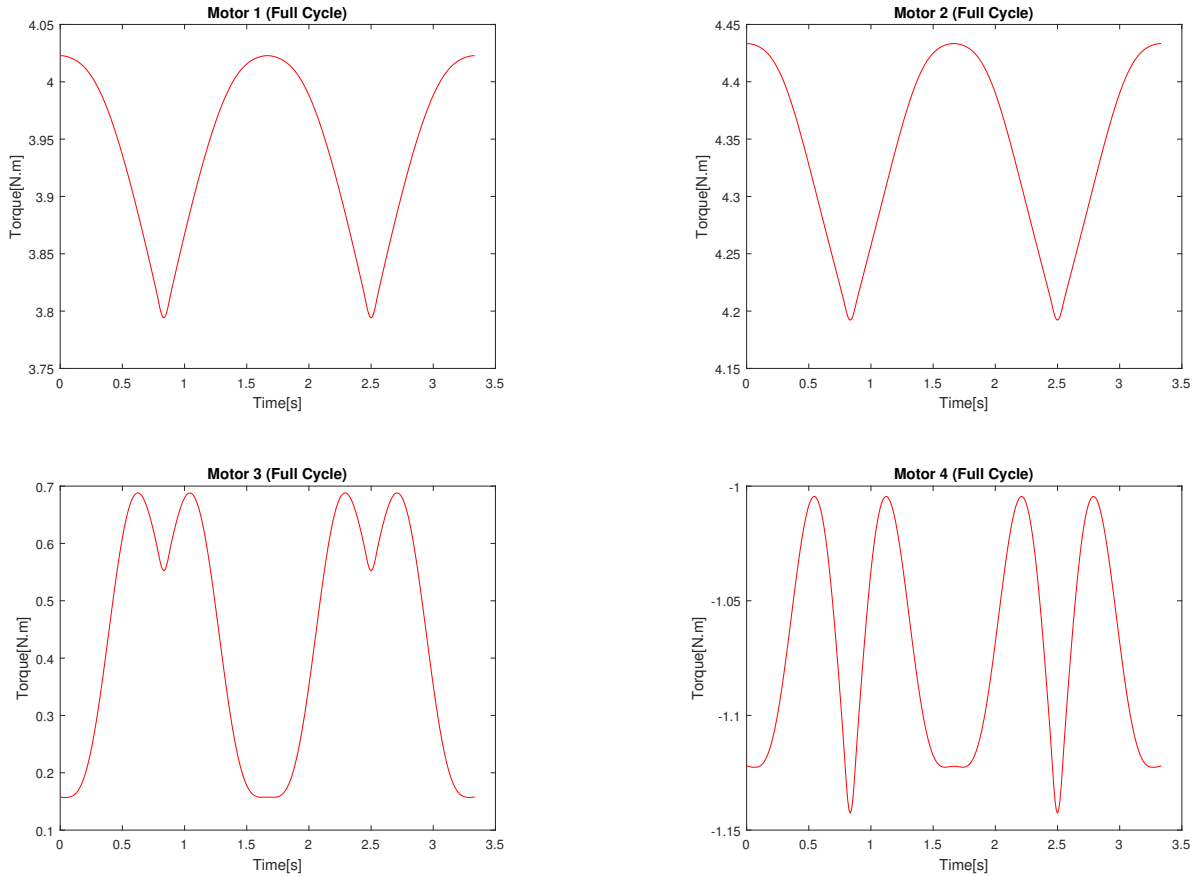

Figure 4.2: Torque profile for a full standard cycle 0.3 Hz

The torque profiles shown in Fig. 4.2 were calculated for the PMC running an optimum test cycle with a frequency of 0.3 Hz, which is the current speed of the PMC. This allows for the results to be verified by comparing them to the experimental torque profiles generated by the PMC shown in Fig. 4.3. The experimental torque data will need to be imported and filtered to be able to provide a meaningful comparison. The control system of the PMC will need to be modified to be able to import the torques and analyze them, but at this point it is sufficient to compare them based on the torque range, which shows a good match, knowing that the simulated results don't account for the dissipative forces and signal noise that are unavoidable in any system.

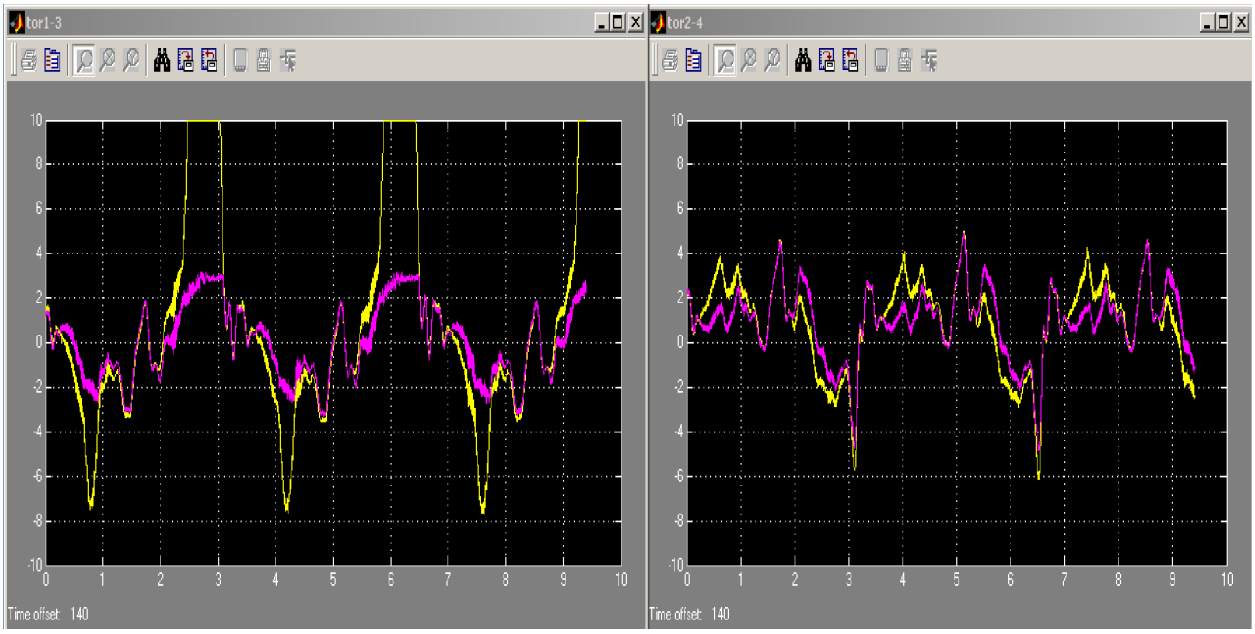

Figure 4.3: Experimental torque results for 0.3 Hz

The inverse dynamics simulation algorithm will help improve the performance of the PMC by providing a valuable tool that can calculate the torques needed for the PMC to operate at higher speeds.

Chapter 5

Forward Dynamics

In the forward problem, values of the joint coordinates and their first time-derivatives are known at a given instant, the time-histories of the applied torques or forces being also known, along with the architecture and the inertial parameters of the manipulator at hand. With these data and initial conditions $\boldsymbol{\theta}_0 = \boldsymbol{\theta}(0)$ and $\dot{\boldsymbol{\theta}}_0 = \dot{\boldsymbol{\theta}}(0)$, the time-histories of the joint coordinates and their time-derivatives are computed at $t > 0$ by integration of the underlying system of nonlinear ordinary differential equations [11].

Forward dynamics is needed either for purposes of simulation or for the model-based control of manipulators, and hence, a fast calculation of the joint-variable time-histories $\boldsymbol{\theta}(t)$ is needed [11]. The purpose of having a forward dynamics simulation for the PMC is to enable us to predict the PMC behavior under given initial conditions, applied torques, and applied loads. The ability of predicting this behavior is important for several reasons: for example, we may want to know whether, with a given selection of motors, the PMC will be able to perform a certain task in a given time frame; also in designing a feedback controller for the PMC, where stability is a major concern, we cannot risk a valuable piece of equipment by exposing it to untested control strategies. Hence, a facility capable of predicting the behaviour of the PMC becomes imperative. To this end, the model of eq. (4.1) is recast in

the form

$$\mathbf{I}\ddot{\boldsymbol{\theta}}_a = -\mathbf{C}(\boldsymbol{\theta}_a, \dot{\boldsymbol{\theta}}_a)\dot{\boldsymbol{\theta}}_a + \boldsymbol{\tau}(t) + \boldsymbol{\gamma}(\boldsymbol{\theta}_a), \quad \boldsymbol{\theta}_0 = \boldsymbol{\theta}(0), \quad \dot{\boldsymbol{\theta}}_0 = \dot{\boldsymbol{\theta}}(0), \quad t \geq 0 \tag{5.1}$$

whose right-hand side can be calculated with the aid of the Newton-Euler recursive algorithm, which is known to have a complexity of $O(n^3)$ [11]. Although in real-time calculations, we would like to have a computational scheme of $O(n)$, and there are some methods that offer such a privilege, we decided to use the method of the natural orthogonal complement as its derivation is straightforward and gives, for a four-axis manipulator, a computational cost similar to that of other methods with $O(n)$ complexity.

Since we start with a second-order four-dimensional system of *ordinary differential equations* (ODEs) in the joint variables of the PMC, the system has to be integrated in order to determine the time-histories of all joint variables. Now eq. (5.1) is recast in state-space form, with the state vector $\boldsymbol{x}$ and the input vector $\boldsymbol{u}(t)$ defined as

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\theta} \\ \\ \dot{\boldsymbol{\theta}} \end{bmatrix}, \quad \boldsymbol{u}(t) = \boldsymbol{\tau}(t) \tag{5.2}$$

and the state-variable equation then taking the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\tau}), \quad \boldsymbol{x}(0) = [\boldsymbol{\theta}_0, \dot{\boldsymbol{\theta}}_0] \tag{5.3a}$$

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\tau}) = \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \\ -\boldsymbol{I}(\boldsymbol{\theta})^{-1}[\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} - \boldsymbol{\gamma}(\boldsymbol{\theta})] + \boldsymbol{I}(\boldsymbol{\theta})^{-1}\boldsymbol{\tau}(t) \end{bmatrix} \tag{5.3b}$$

thereby obtaining a system of 8 first-order nonlinear ODEs in the state-variable vector $\boldsymbol{x}$ defined above, that were then solved numerically in MATLAB using ODE45, which is an explicit technique using a combination of 4^{th}-and 5^{th}-order *Runge-Kutta* methods, the results being shown in Fig. 5.1 plotted along with the prescribed trajectory in joint-space coordinates, as shown in Fig. 4.1.

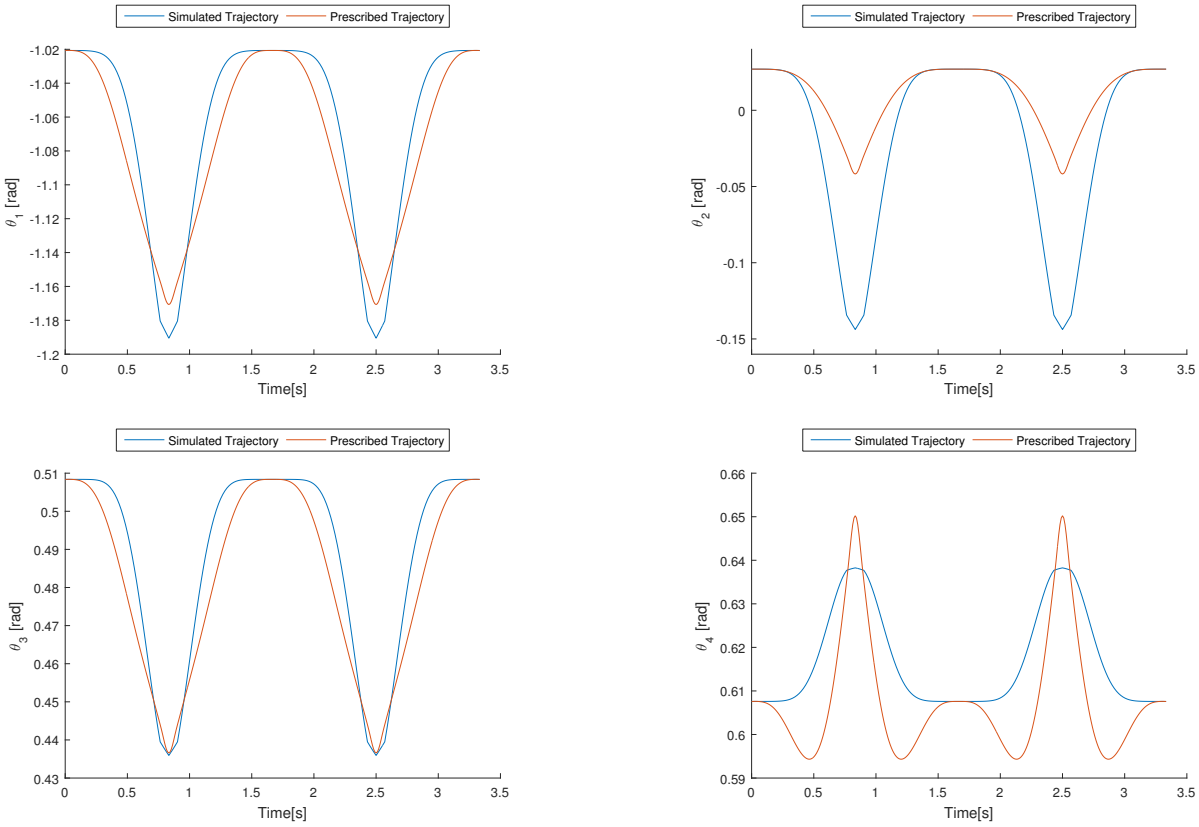

Figure 5.1: Simulated trajectory in joint space

As can be seen in Fig. 5.1, the simulated trajectory tracks fairly well that of the prescribed one for θ_1 and θ_3, but significant errors arise in the case of θ_2 and θ_4. It is not surprising that errors arise, since the forward dynamics problem relies on the numerical integration of a system of ODEs, which incurs not only round-off, but also truncation errors. The inverse dynamics is a purely algebraic problem, which incurs only round-off errors. Before the results

reported here for the forward dynamics can be verified and used to design a better controller for the PMC, an in-depth error analysis to eliminate the source of these errors and identify their significance, is mandatory.

A drawback of Runge-Kutta methods is their inability to deal with what is known as *stiff systems*, i.e., systems of ODEs that are known to be stable, but whose linear part—i.e., the linear part of the series expansion of $\boldsymbol{f}$, evaluated at the current instant—comprises a coefficient matrix that is ill-conditioned [12]. If the mathematical model of the PMC turns out to be stiff, then an integration scheme suitable for this kind of systems will have to be adopted.

Chapter 6

Conclusions and Recommendations for Future Work

Solutions to the inverse and forward dynamics of a CRRHHRRC Schönflies-motion generator, dubbed the Peppermill-carrier, were discussed in this thesis. First, the mathematical model derived by Eden Klingenberg was verified. Then an optimized version of the industry standard test cycle was recalled. From there, we were able to solve the inverse followed by the forward dynamics problems for the PMC while following a full standard test cycle at 0.3 Hz.

The completion of this work represents another step towards developing a commercial robot with a novel architecture that can compete with existing pick-and-place robots on the market.

Some recommendations for further work on this project include:

- Verify the forward dynamics results reported in Ch. 5 with an in-depth error analysis.
- Once the inverse and forward dynamics results are verified, the disturbances that are accompanied with the PMC motion need to be modelled and included into the simulation to better reflect the robot behaviour.

- Designing a control system for the robot, making use of the mathematical model and the simulation tools developed here. Understanding the torques required from the motors will be key in optimizing the robot control, thereby allowing the implementation of faster movements.

Bibliography

[1] Makino, H., Kato, A., and Yamazaki, Y., "Research and commercialization of scara robot-the case of industry-university joint research and development", in *International Journal of Automation Technology*, vol. 1, pp. 61-67, 2007.

[2] Angeles, J., "Design Challenges in the Development of Fast Pick-and-place Robots", *in V. Padois, P. Bidaud, O. Khatib (editors), Romansy 19 – Robot Design, Dynamics and Control, CISM International Centre for Mechanical Sciences* DOI:10.1007/978-3-7091-1379-0_8

[3] Lee, C.C., Lee, P.C., "Isoconstrained mechanisms for fast pick-and-place manipulation" *in Lou, Y. and Li,Z (editors), Geometric Methods in Robotics and Mechanism Research, Lambert Academic Publishing, Saarbrücken, Germany*, pp. 85-99, 2010.

[4] Lee, P.-C., Lee, J.-J., and Lee, C.-C., "Four novel pick-and-place isoconstrained manipulators and their inverse kinematics," in *ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers*, pp. 1079- 1088, 2010.

[5] Lee, P.-C. and Lee, J.-J., "Forward kinematics and numerical verification of four novel parallel manipulators with schöenflies motion," in *Proceedings of the 1st IFToMM Asian Conference on Mechanical Machine Science, Taipei* , 2010.

[6] Harada, T. and Angeles, J., "Kinematics and singularity analysis of a CRRHHRRC parallel Schönflies-motion generator," *Proc. CCToMM Symposium on Mechanisms, Machines, and Mechatronics*, May 30–31, Montreal, Paper 12, 10 pp., 2013.

[7] Friedlaender, T. and Angeles, J., "Design, Control and Testing of a Pick-and-Place Robot and its Novel Actuators," *Masters Thesis*, 2015.

[8] Klingenberg, E. and Angeles, J., "The Mathematical Model of a Pick-and-Place Robot," *Honours Thesis*, 2015.

[9] Angeles, J., *Fundamentals of Robotic Mechanical Systems, Theory, Methods and Algorithms*, 4th ed. New York: Springer, 2013.

[10] Gauthier, J.-F., Angeles, J, and Nokleby, S, "Optimization of a test trajectory for scara systems," in *Advances in Robot Kinematics: Analysis and Design*, Springer, pp. 225-234, 2008.

[11] Angeles, J, "Form, Structure and Reciprocity in Multibody-systems Models," in *The* 4^{th} *Joint International Conference on Multibody System Dynamics*, 2016.

[12] Shampine, Lawrence F. "Numerical solution of ordinary differential equations," Vol. 4. *CRC Press*, 1994.

[13] von Mises, R., "Motorrechnung, ein neues Hilfsmittel der Mechanik," *Z. Angewandte Mathematik und Mechanik* 4, No. 2, pp. 155–181, 1924.

[14] Harada, T., Friedlaender, T., Angeles, J., *The Development of an Innovative Two-DOF Cylindrical Drive: Design, Analysis and Preliminary Tests*, Department of Mechanical Engineering, Faculty of Science and Engineering, Kinki University, Higashiosaka, Osaka, Japan, Department of Mechanical Engineering and Center for Intelligent Machines, McGill University, Montreal, Quebec, Canada, 2013.

[15] Spong, M.W., Hutchinson, S., and Vidyasagar, M., *Robot Modeling And Control*, John Wiley & Sons, Inc. Hoboken, NJ., 2006.

[16] Lee, P.-C., Lee, J.-J., and Lee, C.-C., "Four novel pick-and-place isoconstrained manipulators and their inverse kinematics," in *ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, 2010*, pp. 1079- 1088.

[17] Harada, T. and Angeles, J., "Kinematics and singularity analysis of a CRRHHRRC parallel schönflies motion generator," *Transactions of the Canadian Society for Mechanical Engineering*, vol. 38, no. 2, pp. 173-183, 2014.

Appendices

Appendix A

Mathematica Code

```
ClearAll[tau];
freq = 0.3; time = (1 / freq) / 2;
(*CONSTANTS*)
B0 = 0.300;
x0 = 0.2; y0 = 0.2; z0 = -0.1; phi0 = 0;
P = 0.06; P3 = 0.02; r1 = 0.3; r2 = 0.3; l1 = 0.3; l2 = 0.3;
m1 = 0.9655;
Ix1 = 0.006338; Iy1 = 0.001535; Iz1 = 0.007503;
Rho1 = 0.1015;
m2 = 0.9655;
Ix2 = 0.006338; Iy2 = 0.001535; Iz2 = 0.007503;
Rho2 = 0.1015;
m3 = 0.1000;
Ix3 = 0.001012; Iy3 = 0.0004403; Iz3 = 0.0005897;
Rho3 = 0.1320;
m4 = 0.1000;
Ix4 = 0.001012; Iy4 = 0.0004403; Iz4 = 0.0005897;
Rho4 = 0.1320;
m5 = 0.4173;
Ix5 = 0.0001302; Iy5 = 0.0001350; Iz5 = 0.0001475;
m6 = 0.4173;
Ix6 = 0.0001302; Iy6 = 0.0001350; Iz6 = 0.0001476;
m7 = 0.61226;
Ix7 = 0.0006475; Iy7 = 0.0007796; Iz7 = 0.0007936;
M1 = {{Ix1 , 0, 0, 0, 0, 0},
    {0 , Iy1, 0, 0, 0, 0},
    {0 , 0, Iz1, 0, 0, 0},
    {0 , 0, 0, m1, 0, 0},
    {0 , 0, 0, 0, m1, 0},
    {0, 0, 0, 0, 0, m1}};
M2 = {{Ix2 , 0, 0, 0, 0, 0},
    {0 , Iy2, 0, 0, 0, 0},
    {0 , 0, Iz2, 0, 0, 0},
    {0 , 0, 0, m2, 0, 0},
    {0 , 0, 0, 0, m2, 0},
    {0, 0, 0, 0, 0, m2}};
M3 = {{Ix3 , 0, 0, 0, 0, 0},
    {0 , Iy3, 0, 0, 0, 0},
    {0 , 0, Iz3, 0, 0, 0},
    {0 , 0, 0, m3, 0, 0},
    {0 , 0, 0, 0, m3, 0},
    {0, 0, 0, 0, 0, m3}};
```

```
M4 = {{Ix4 , 0, 0, 0, 0, 0},
    {0 , Iy4, 0, 0, 0, 0},
    {0 , 0, Iz4, 0, 0, 0},
    {0 , 0, 0, m4, 0, 0},
    {0 , 0, 0, 0, m4, 0},
    {0, 0, 0, 0, 0, m4}};
M5 = {{Ix5, 0, 0, 0, 0, 0},
    {0 , Iy5, 0, 0, 0, 0},
    {0 , 0, Iz5, 0, 0, 0},
    {0 , 0, 0, m5, 0, 0},
    {0 , 0, 0, 0, m5, 0},
    {0, 0, 0, 0, 0, m5}};
M6 = {{Ix6, 0, 0, 0, 0, 0},
    {0 , Iy6, 0, 0, 0, 0},
    {0 , 0, Iz6, 0, 0, 0},
    {0 , 0, 0, m6, 0, 0},
    {0 , 0, 0, 0, m6, 0},
    {0, 0, 0, 0, 0, m6}};
M7 = {{Ix7, 0, 0, 0, 0, 0},
    {0 , Iy7, 0, 0, 0, 0},
    {0 , 0, Iz7, 0, 0, 0},
    {0 , 0, 0, m7, 0, 0},
    {0 , 0, 0, 0, m7, 0},
    {0, 0, 0, 0, 0, m7}};
(* Trajectory *)
(*4,5,6,7 polynomial*)
s[tau_] = (-20 * tau^7) + (70 * tau^6) - (84 * tau^5) + (35 * tau^4);
sdot = D[s[tau], tau];
sddot = D[sdot[tau], tau];
x = 0.5 * 0.3 * Cos[Pi / 4] * (s[tau / time]);
y = 0.5 * 0.3 * Sin[Pi / 4] * (s[tau / time]);
z = 0.025 * (s[2 * tau / time]);
phi = (Pi / 2) * (s[tau / time]);
p1z = z + B0 + (P * phi);
p2z = z - B0 - (P * phi);
(* Kinematics *)
theta1s =
  2 * ArcTan[-p1z - (0.5 * Sqrt[(4 * (y^2 + p1z^2)) - (((y^2 + p1z^2) ^2) / (r1^2))])];
theta2s = 2 * ArcTan[(((-y^2 + p1z^2) ^2) / (2 * r1)) - y];
theta3s =
  2 * ArcTan[-x + (0.5 * Sqrt[(4 * (x^2 + p2z^2)) - (((x^2 + p2z^2) ^2) / (r2^2))])];
theta4s = 2 * ArcTan[(((-x^2 + p2z^2) ^2) / (2 * r2)) - p2z];
thetadot1s = D[theta1s, tau];
```

```
thetadot2s = D[theta2s, tau];
thetadot3s = D[theta3s, tau];
thetadot4s = D[theta4s, tau];
thetaddot1s = D[thetadot1s, tau];
thetaddot2s = D[thetadot2s, tau];
thetaddot3s = D[thetadot3s, tau];
thetaddot4s = D[thetadot4s, tau];
(*Mathematical Model*)
(*Symbolic Differentiation of T's*)
T1s = {{0.5, 0.5, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {P / (4 * Pi), -P / (4 * Pi), 0, 0},
    {(-Rho1 / 2) * Sin[(theta1s + theta2s) / 2] ,
     (-Rho1 / 2) * Sin[(theta1s + theta2s) / 2] , 0, 0},
    {(Rho1 / 2) * Cos[(theta1s + theta2s) / 2] ,
     (Rho1 / 2) * Cos[(theta1s + theta2s) / 2], 0, 0}};
diffT1s = {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0},
    {(-Rho1 / 2) * ((thetadot1s + thetadot2s) / 2) * Cos[(theta1s + theta2s) / 2] ,
     (-Rho1 / 2) * ((thetadot1s + thetadot2s) / 2) * Cos[(theta1s + theta2s) / 2] , 0, 0},
    {(-Rho1 / 2) * ((thetadot1s + thetadot2s) / 2) * Sin[(theta1s + theta2s) / 2] ,
     (-Rho1 / 2) * ((thetadot1s + thetadot2s) / 2) * Sin[(theta1s + theta2s) / 2], 0, 0}};
T2s = {{0, 0, 0, 0},
    {0, 0, 0.5, 0.5},
    {0, 0, 0, 0},
    {0, 0 , (Rho2 / 2) * Cos[(theta3s + theta4s) / 2] ,
     (Rho2 / 2) * Cos[(theta3s + theta4s) / 2]},
    {0, 0, -P / (4 * Pi), P / (4 * Pi)},
    {0, 0, (-Rho2 / 2) * Sin[(theta3s + theta4s) / 2] ,
     (-Rho2 / 2) * Sin[(theta3s + theta4s) / 2]}};
diffT2s = {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0},
    {0, 0, (-Rho2 / 2) * ((thetadot3s + thetadot4s) / 2) * Sin[(theta3s + theta4s) / 2],
     (-Rho2 / 2) * ((thetadot3s + thetadot4s) / 2) * Sin[(theta3s + theta4s) / 2]},
    {0, 0, 0, 0},
    {0, 0 , (-Rho2 / 2) * ((thetadot3s + thetadot4s) / 2) * Cos[(theta3s + theta4s) / 2],
     (-Rho2 / 2) * ((thetadot3s + thetadot4s) / 2) * Cos[(theta3s + theta4s) / 2]}};
lamda1s = (ArcCos[(((P / (4 * Pi)) * (theta4s - theta3s)) -
          (r1 * Cos[(theta1s + theta2s) / 2])) / l1]) - ((theta1s + theta2s) / 2);
vartheta1s = (theta1s + theta2s) / 2;
T3s = {{(-r1 * Sin[vartheta1s]) / (2 * l1 * Sin[lamda1s + vartheta1s]),
     (-r1 * Sin[vartheta1s]) / (2 * l1 * Sin[lamda1s + vartheta1s]),
     P / (4 * Pi * l1 * Sin[lamda1s + vartheta1s]),
     -P / (4 * Pi * l1 * Sin[lamda1s + vartheta1s])},
    {0, 0, 0, 0}, {0, 0, 0, 0}, {P / (4 * Pi), -P / (4 * Pi), 0, 0},
    {((Rho3 * r1 * Sin[vartheta1s]) / (2 * l1)) - ((r1 / 2) * Sin[vartheta1s]),
     ((Rho3 * r1 * Sin[vartheta1s]) / (2 * l1)) - ((r1 / 2) * Sin[vartheta1s]),
```

```
    -P * Rho3 / (4 * Pi * l1), P * Rho3 / (4 * Pi * l1)},
   {((-Rho3 / (16 * Pi^2)) * Sin[vartheta1s] *
       (2 * P * Pi * r1 * (theta4s - theta3s) - 8 * (r1^2) * (Pi^2) * Cos[vartheta1s]) /
        ((l1^2) * Sin[lamda1s + vartheta1s])) +
     (r1 / 2) * Cos[vartheta1s], ((-Rho3 / (16 * Pi^2)) * Sin[vartheta1s] *
       (2 * P * Pi * r1 * (theta4s - theta3s) - 8 * (r1^2) * (Pi^2) * Cos[vartheta1s]) /
        ((l1^2) * Sin[lamda1s + vartheta1s])) +
     (r1 / 2) * Cos[vartheta1s], ((-Rho3 / (16 * Pi^2)) *
      ((P^2) * (theta3s - theta4s) + 4 * P * r1 * Pi * Cos[vartheta1s]) /
       ((l1^2) * Sin[lamda1s + vartheta1s])), ((-Rho3 / (16 * Pi^2)) *
      ((P^2) * (theta4s - theta3s) - 4 * P * r1 * Pi * Cos[vartheta1s]) /
       ((l1^2) * Sin[lamda1s + vartheta1s]))}};
diffT3s = D[T3s, tau];
lamda2s =
  (ArcCos[(((P / (4 * Pi)) * (theta1s - theta2s)) - (r1 * Cos[(theta3s + theta4s) / 2])) /
      l1]) - ((theta3s + theta4s) / 2);
vartheta2s = (theta3s + theta4s) / 2;
T4s = {{0, 0, 0, 0},
   {P / (4 * Pi * l2 * Sin[lamda2s + vartheta2s]),
    -P / (4 * Pi * l2 * Sin[lamda2s + vartheta2s]),
    (-r2 * Cos[vartheta2s]) / (2 * l2 * Cos[lamda2s + vartheta2s]),
    (-r2 * Cos[vartheta2s]) / (2 * l2 * Cos[lamda2s + vartheta2s])},
   {0, 0, 0, 0},
   {P * Rho4 / (4 * Pi * l2), -P * Rho4 / (4 * Pi * l2) ,
    ((Rho4 * r2 * Cos[vartheta2s]) / (2 * l2)) - ((r2 / 2) * Cos[vartheta2s]) ,
    ((Rho4 * r2 * Cos[vartheta2s]) / (2 * l2)) - ((r2 / 2) * Cos[vartheta2s])},
   {0, 0, -P / (4 * Pi), P / (4 * Pi)},
   {((Rho4 / (16 * Pi^2)) *
      ((P^2) * (theta2s - theta1s) + 4 * P * r2 * Pi * Sin[vartheta2s]) /
       ((l2^2) * Cos[lamda2s + vartheta2s])), ((Rho4 / (16 * Pi^2)) *
      ((P^2) * (theta1s - theta2s) - 4 * P * r2 * Pi * Sin[vartheta2s]) /
       ((l2^2) * Cos[lamda2s + vartheta2s])) ,
    ((Rho4 / (16 * Pi^2)) * Cos[vartheta2s] * (2 * P * Pi * r2 * (theta1s - theta2s) - 8 *
          (r2^2) * (Pi^2) * Cos[vartheta2s]) / ((l2^2) * Cos[lamda2s + vartheta2s])) -
     (r2 / 2) * Sin[vartheta2s], ((Rho4 / (16 * Pi^2)) * Cos[vartheta2s] *
       (2 * P * Pi * r2 * (theta1s - theta2s) - 8 * (r2^2) * (Pi^2) * Cos[vartheta2s]) /
        ((l2^2) * Cos[lamda2s + vartheta2s])) - (r2 / 2) * Sin[vartheta2s]}};
diffT4s = D[T4s, tau];
T5s = {{0, 0, 0, 0},
   {0, 0, 0, 0}, {0, 0, 0, 0}, {P / (4 * Pi), -P / (4 * Pi) , 0, 0},
   {0, 0, -P / (4 * Pi) , P / (4 * Pi)},
   {((-1 / (16 * (Pi^2))) * Sin[vartheta1s] *
       (((2 * P * Pi * r1) * (theta4s - theta3s)) - 8 * (r1^2) * (Pi^2) * Cos[vartheta1s]) /
```

```
          (l1 * Sin[lamda1s + vartheta1s])) + (r1 / 2) * Cos[vartheta1s],
     ((-1 / (16 * (Pi^2))) * Sin[vartheta1s] * (((2 * P * Pi * r1) * (theta4s - theta3s)) -
           8 * (r1^2) * (Pi^2) * Cos[vartheta1s]) / (l1 * Sin[lamda1s + vartheta1s])) +
      (r1 / 2) * Cos[vartheta1s], (-1 / (16 * (Pi^2))) *
      (((P^2) * (theta3s - theta4s)) + 4 * r1 * Pi * P * Cos[vartheta1s]) /
       (l1 * Sin[lamda1s + vartheta1s]), (-1 / (16 * (Pi^2))) *
      (((P^2) * (theta4s - theta3s)) - 4 * r1 * Pi * P * Cos[vartheta1s]) /
       (l1 * Sin[lamda1s + vartheta1s])}};
diffT5s = D[T5s, tau];
T6s = {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0},
   {P / (4 * Pi), -P / (4 * Pi) , 0, 0},
   {0, 0, -P / (4 * Pi), P / (4 * Pi)},
   {(1 / (16 * (Pi^2))) *
      (((P^2) * (theta2s - theta1s)) + 4 * r2 * Pi * P * Sin[vartheta2s]) /
       (l2 * Cos[lamda2s + vartheta2s]) , (1 / (16 * (Pi^2))) *
      (((P^2) * (theta1s - theta2s)) - 4 * r2 * Pi * P * Sin[vartheta2s]) /
       (l2 * Cos[lamda2s + vartheta2s]),
     ((1 / (16 * (Pi^2))) * Cos[vartheta2s] * (((2 * P * Pi * r2) * (theta1s - theta2s)) -
           8 * (r2^2) * (Pi^2) * Cos[vartheta2s]) / (l2 * Cos[lamda2s + vartheta2s])) -
      (r2 / 2) * Sin[vartheta2s], ((1 / (16 * (Pi^2))) * Cos[vartheta2s] *
        (((2 * P * Pi * r2) * (theta1s - theta2s)) - 8 * (r2^2) * (Pi^2) * Sin[vartheta2s]) /
         (l2 * Cos[lamda2s + vartheta2s])) - (r2 / 2) * Sin[vartheta2s]}};
diffT6s = D[T6s, tau];
T7s = {{0, 0, 0, 0}, {0, 0, 0, 0},
   {(-1 / (2 * P3)) * (T5s[[6, 1]] - T6s[[6, 1]]) ,
    (-1 / (2 * P3)) * (T5s[[6, 2]] - T6s[[6, 2]]) , (-1 / (2 * P3)) *
     (T5s[[6, 3]] - T6s[[6, 3]]) , (-1 / (2 * P3)) * (T5s[[6, 4]] - T6s[[6, 4]])},
   {P / (4 * Pi), -P / (4 * Pi), 0, 0},
   {0, 0, -P / (4 * Pi), P / (4 * Pi)},
   {(T5s[[6, 1]] + T6s[[6, 1]]) / 2 , (T5s[[6, 2]] + T6s[[6, 2]]) / 2,
    (T5s[[6, 3]] + T6s[[6, 3]]) / 2 , (T5s[[6, 4]] + T6s[[6, 4]]) / 2}};
diffT7s = D[T7s, tau];
(*Wrenches W's*)
W1 = {{0, 0, 0, 0, 0, 0},
   {0, 0, -(thetadot1s + thetadot2s) / 2, 0, 0, 0},
   {0, (thetadot1s + thetadot2s) / 2, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}};
W2 = {{0, 0, 0, 0, 0, 0},
   {0, 0, -(thetadot3s + thetadot4s) / 2, 0, 0, 0},
   {0, (thetadot3s + thetadot4s) / 2, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0},
```

```
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}};
W3 = {{0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, -(1 / (l1 * Sin[lamda1s + vartheta1s])) *
       (((P / (4 * Pi)) * (thetadot3s - thetadot4s)) +
          ((r1 * Sin[vartheta1s]) * ((thetadot1s + thetadot2s) / 2)))},
    {0, (1 / (l1 * Sin[lamda1s + vartheta1s])) *
       (((P / (4 * Pi)) * (thetadot3s - thetadot4s)) +
          ((r1 * Sin[vartheta1s]) * ((thetadot1s + thetadot2s) / 2))) , 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}};
W4 = {{0, 0, 0, 0, 0,
     (1 / (l2 * Cos[lamda2s + vartheta2s])) * (((P / (4 * Pi)) * (thetadot1s - thetadot2s)) -
        ((r2 * Cos[vartheta2s]) * ((thetadot3s + thetadot4s) / 2)))},
    {0, 0, 0, 0, 0, 0},
    {(-1 / (l2 * Cos[lamda2s + vartheta2s])) *
       (((P / (4 * Pi)) * (thetadot1s - thetadot2s)) -
          ((r2 * Cos[vartheta2s]) * ((thetadot3s + thetadot4s) / 2))), 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}};
W5 = {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
W6 = {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
W7 = {{0, (1 / (2 * P3)) * ((T5s[[6, 1]] - T6s[[6, 1]]) + (T5s[[6, 2]] - T6s[[6, 2]]) +
          (T5s[[6, 3]] - T6s[[6, 3]]) + (T5s[[6, 4]] - T6s[[6, 4]])), 0, 0, 0, 0},
    {(-1 / (2 * P3)) * ((T5s[[6, 1]] - T6s[[6, 1]]) + (T5s[[6, 2]] - T6s[[6, 2]]) +
          (T5s[[6, 3]] - T6s[[6, 3]]) + (T5s[[6, 4]] - T6s[[6, 4]])), 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}};
I1 = (Transpose[T1s].M1).T1s;
I2 = Transpose[T2s].M2.T2s;
I3 = Transpose[T3s].M3.T3s;
I4 = Transpose[T4s].M4.T4s;
I5 = Transpose[T5s].M5.T5s;
I6 = Transpose[T6s].M6.T6s;
I7 = Transpose[T7s].M7.T7s;
C1 = (Transpose[T1s].M1.diffT1s) + (Transpose[T1s].W1.M1.T1s);
C2 = (Transpose[T2s].M2.diffT2s) + (Transpose[T2s].W2.M2.T2s);
```

```
C3 = (Transpose[T3s].M3.diffT3s) + (Transpose[T3s].W3.M3.T3s);
C4 = (Transpose[T4s].M4.diffT4s) + (Transpose[T4s].W4.M4.T4s);
C5 = (Transpose[T5s].M5.diffT5s) + (Transpose[T5s].W5.M5.T5s);
C6 = (Transpose[T6s].M6.diffT6s) + (Transpose[T6s].W6.M6.T6s);
C7 = (Transpose[T7s].M7.diffT7s) + (Transpose[T7s].W7.M7.T7s);
Itot = I1 + I2 + I3 + I4 + I5 + I6 + I7;
Ctot = C1 + C2 + C3 + C4 + C5 + C6 + C7;
WG = {{0}, {0}, {0}, {0}, {0}, {(m1 + m2 + m3 + m4 + m5 + m6 + m7) * 9.81}};
Ttot = T1s + T2s + T3s + T4s + T5s + T6s + T7s;
Torque = Itot.{{thetaddot1s}, {thetaddot2s}, {thetaddot3s}, {thetaddot4s}} + Ctot.
     {{thetadot1s}, {thetadot2s}, {thetadot3s}, {thetadot4s}} + Transpose[Ttot].WG;
```