KINEMATICS AND DYNAMICS OF FRUIT PICKING ROBOTIC MANIPULATOR

By

JIAYING ZHANG

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2014

To my Mother, for encouraging me and trusting my ability

ACKNOWLEDGMENTS

I sincerely thank my advisor, Dr. John Schueller, for providing me great support during the process of writing the thesis. Dr. Schueller kindly accepted me as one of his students and tutored me with much patience throughout the two semesters. He is always willing to listen to my opinions and leads me to a right direction. I learnt a lot of academic knowledge from him and am glad to listen to his interesting stories. He is always not hesitating to give me the strongest support that he can provide. This thesis cannot be finished successfully without his kind support and careful check.

I thank University of Florida for providing me a good opportunity to practice the knowledge I have learnt here. University of Florida is the first college where I study in the English environment. Through the process of writing this thesis, my academic knowledge is strengthened and my English writing skills is well practiced. The speed to read and understand relative literature becomes faster than before. Through the thesis, I am able to work close with my advisor, and thus to know the high standard for publishing in a world-class university. The study at Mechanical Engineering Department is joyful and worthwhile throughout my life.

I thank the committee member, Dr. Carl Crane III. I am thankful to Dr. Crane for his course Robot Geometry which is quite useful as the background knowledge for doing the research. All the committee members are very kind to give me advice and support.

I thank the peers who also write the master's theses, such as Tianyu Gu, Zhe Wang, and Jing Zhou. The competition among us is one of the best impetuses for me to finish the thesis on time. I thank them for accompanying me from the very beginning to the end of the defense. The information they gave me, the process they shared with me and the methods they taught me are all very helpful to me.

Finally, I thank my parents for their eternal support and love. They support me both economically and mentally. They are patient to listen to my complaint when I am in trouble, although they don’t know what exactly I am doing. Their encouragement provides me much confidence, which leads me finishing this thesis at the end.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

KINEMATICS AND DYNAMICS OF FRUIT PICKING ROBOTIC MANIPULATOR

By

Jiaying Zhang

May 2014

Chair: John K. Schueller
Major: Mechanical Engineering

The fruit picking robot is ideally suited to fresh fruit harvesting. It not only prevents the fruit tree from damage caused by mass harvesting machines but also saves human labor forces. Through several previous fruit harvester prototypes, some common rules on designing the manipulators are generalized to a representative robot model. Operating kinematics and dynamics studies on the representative robotic manipulator can help to simulate and analyze the fruit picking performance. This thesis studies the joint angles, velocities, and torques for Puma 560 arm to pick fruit individually.

The development of fruit picking manipulator depends largely on the tree size and the fruit distribution. An arm designer graphic user interface (GUI) is introduced to create arms that meet the developers' different requirements on fruit trees. Kinematic and dynamic analyses are integrated into the GUI, which helps the user to understand the picking performance of the generated arm.

CHAPTER 1
INTRODUCTION

Background

Fruit growers in many countries are facing potential problems that could affect their future business. Two significant problems among them are the lack of adequate labor supply and the increase in labor cost. In most countries, fruits are still being picked and packaged by hands, and graded by human inspection. In some advanced countries, scientists already have the awareness of combining automation technology into fruit production system in order to make up the labor shortage. One important stage that has large room to improve the effectiveness is the fruit picking process. Many researchers are continuously making efforts on the harvest mechanization.

Basically, there are three modes of harvest mechanization: Labor-aids, Labor saving machines, Robotics and Automation. Firstly, Labor-aids are aimed at reducing the drudgery of farm labor by reducing the effort and endurance required for the fruit-picking operation. Ladders are one of the most typical labor-aids. As long as the picking is done manually, the potential for increasing productivity is limited. Secondly, Labor saving machines are designed with the concept of mass harvesting. These machines have the advantage of reducing overall harvesting time and human labor costs. For example, Oxbo Corp. has citrus harvester with continuous canopy shake and catch system. Mechanical mass harvesting does not suit to fresh market due to the damage of fruits and tree during the excessive mechanical processing. Thirdly, Robotic fruit harvesting aims to automate the fruit picking process by using a system that emulate the human picker. Conceptually, it should provide an overall faster rate than human labor and a better protection for fruit and tree than mass harvesting. For this reason, many studies have been done on robotic fruit harvesting over the past decades.

A typical robotic fruit harvesting consists of fruit a detecting system to find the position of target fruit, an end-effector to grip and detach the fruit, and a manipulator that carries the end-effector to reach the fruit. A fruit detection system normally has cameras with multiple sensors which needs complex supporting algorithm to find the right fruit. End-effectors are well designed to operate subtle behaviors such as stem-cutting or fruit-twisting. Manipulator arms are core components in computing the workspace and motion trajectory. A well designed manipulator can have verified dexterity, accuracy and static performance. It is necessary to analyze the manipulator before a robotic harvester is built as a real product. The most general analysis methods are kinematics and dynamics, which are widely used in many robotic manipulators. Analysis tools like Matlab Robotics Toolbox (Corke, 1996) and Robotect (Nayar, 2002) are very convenient to do robot simulation analysis.

Problem Statement

As many types of robotic fruit harvesting were designed to pick different fruit, it is easy to find similarities behind their design concept. For most tree fruit, the manipulator is a typical part among the harvesting robots. If a series of typical analysis could be done to simulate the fruit picking process, it will be helpful to design similar fruit harvesting in the future. There exists a need to analysis the fruit picking process for a typical robotic arm, so the analysis can be applied to the design of robotic fruit harvesting with different picking requirements.

Objectives

The first objective of this research is to establish kinematic analysis for a robot manipulator performing fruit picking process. The Puma 560 robotic arm is selected as an example for doing the analysis because of its popularity, data approachability and similarity in structure to previous robotic fruit harvest arms. The Matlab Robotics Toolbox is introduced to do

both forward and inverse kinematics. The goal is to simulate the geometric trajectory of the end-effector for reaching a specified fruit position.

The second objective is to establish dynamic analysis for a robot manipulator performing fruit picking process. Similar to kinematic analysis, I use Puma 560 as an example and use Matlab Robotics Toolbox as the tool. In this part, the research will give the variety of joint velocities and accelerations during a picking trip. The torques of the arm actuators will also come out as the result.

The final objective is to design a graphical user interface (GUI) which helps the users to generate fruit picking arms that meet their requirements. The GUI will allow the users inputting general fruit positions which are greatly determined by fruit types. Arm length as well as the base position will be generated according to the inputted fruit distribution. The GUI will integrate inverse kinematics and dynamics, and apply them to the created fruit picking arm.

CHAPTER 2
LITERATURE REVIEW

Fruit Harvesting Manipulators

Numerous researches on robotic fruit harvesting have been studied over the past decades. Recently, some researchers, such as Burks (2005), Sivaraman (2006), Hannan (2009), have also presented further studies on the base of those developed ones. Studies on harvesting manipulators of apples, citruses, oranges and tomatoes are reviewed in this chapter due to their similar tree structure.

An apple harvesting prototype was first developed in France by Grand D'Esnon et al. (1985). Its mechanical system consisted of a telescopic arm that can move in a vertical framework. The arm was mounted on a barrel that could rotate horizontally. Later, D'Esnon et al. (1987) built a new prototype called MAGALI. It used a spherical manipulator with a camera set at the center of the base rotation axes and a vacuum grasper for fruit picking. Figure 2-1 shows the spherical manipulator that executed a pantographic prismatic movement along with two rotations.

A robotic system for citrus harvesting was development at University of Florida by Harrell (1988). A three degree of freedom manipulator that was actuated by servo-hydraulic drives was designed. Figure 2-2 shows the geometry character of the manipulator. Joint 0 and 1 were revolute and joint 2 was prismatic. The whole picking motion followed a spherical coordinate system. Harrell used a rotating lip mechanism as the end-effector to serve the stem between a stationary and rotating cup. The joint position and velocity control was achieved through potentiometer and tachometer feedback signals. Ultrasonic range sensors and a color charge coupled device (CCD) camera were used to detect the citrus depth information.

Later, new research on citrus fruit harvesting was launched at University of Florida by Sivaraman et al. (2006). A robot manipulator was designed to harvest both surface and inner canopy fruits, and a prototype was built for lab testing. Figure 2-3 shows the prototype of the manipulator and Figure 2-4 shows the joint types in detail. The manipulator had seven degrees of freedom in all and the last four rotational joints were used to study its dexterity. Kinematics and control system work were carried out, but dynamics studies were not included in their research. The end-effector of the prototype was studied by Flood (2006). The end-effector was designed to grasp the citrus with three fingers and to detach it by twist forces.

Another mandarin orange harvesting robot was developed and trialed by Kubota& Co., Ltd in Japan (Sarig, 1993). This robot consisted of a mobile carriage, a boom, and an arm fixed at the end of the boom. Figure 2-5 shows the picking arm with the two main links (shown as Arm 1 and Arm2) of the same length. The Kubota robot had an articulated arm with four degree of freedom, but acted as a spherical coordinate robot. This is because its elbow joint (shown as B in Figure 2-5) and the wrist joint (show as C in Figure 2-5) are interlinked at a speed ratio of 2:1, and then the end-effector can perform the transitional movement as a prismatic joint. This design had the advantage of preventing interference with trees behind the robot. The Kubota robot used an end-effector with rotating stem-cutters that contained a color TV camera and a light source.

A tomato-harvesting robot was studied by Kondo et al. (1996) at Okayama University. Tomatoes are normal cultivated in greenhouses with vertical supports. Figure 2-6 shows the robot prototype and Figure 2-7 shows the joint and link information. It had five rotational joints and two prismatic joints. The seven degree of freedom was designed to reach fruit efficiently and to evade the foliage. The joints were powered by electric motors. A two finger end effector equipped with a suction pad to pull the fruit into the end effector was used for normal sized

tomatoes and a modified end effector with a nipper to cut the peduncle at the fruit-peduncle joint was used for cherry tomato harvesting (Kondo et al., 1996). A photoelectric sensor and a color camera were used for visual sensing in the trials.

Manipulator Kinematics

A typical robot manipulator mechanism is composed of a serial chain of links and joints. One end of the link chain is fixed to the base or the ground, while the other end carries an end effector to perform specific tasks. The links are assumed to be rigid bodies that are connected by neighboring joint axes and each joint has one degree of freedom (DOF), either rotational or translational. The most common joint type is revolute joint and prismatic joint. A method called the Denavit-Hartenberg (D-H) notation (1955) is used to describe a manipulator mechanism by using four parameters. These four joint and link parameters are link length a, link twist α, link offset d, and joint angle θ. The link length and link twist define the relative location of the two axes in space. For the first and last links, these two parameters are arbitrarily selected to be 0 because they are meaningless. The link offset is the distance between neighboring links along the joint axis. The joint angle is the rotation of neighboring links with respect to the joint axis.

We can number the links of a manipulator from 0 to n, with n joints connecting between one and another. Link 0 represents the base of the manipulator and link n carries the end-effector. Joint i connects link i to link i-1. The link i is attached to a coordinate frame i. Denavit and Hartenberg proposed a matrix method of systematically assigning coordinate systems to each link of an articulated chain. The axis of revolute joint i is aligned with z_{i-1}. The x_{i-1} axis is directed along the normal from z_{i-1} to z_i. If z axes are intersecting, x_{i-1} axis is parallel to $z_{i-1} \times z_i$. The y axis follows the right hand rule which satisfies z_i in the direction of $x_i \times y_i$. Therefore the four parameters can be summarized as:

- link length a_i the offset distance between the z_{i-1} and z_i axes along x_i axis;
- link twist α_i the angle from the z_{i-1} axis to the z_i axis about the x_i axis;
- link offset d_i the distance from the origin of frame $i-1$ to the x_i axis along the z_{i-1} axis;
- joint angle θ_i the angle between the x_{i-1} and xi axes about the z_{i-1} axis.

For a revolute joint, the joint angle θ is variable, while the link offset d is constant. For a prismatic joint, the joint angle θ is constant, while the link offset d is variable.

The relationship between two neighboring coordinate frames is expressed in Equation (2-1)

$$^{0}\boldsymbol{T}_i = {}^{0}\boldsymbol{T}_{i-1}{}^{i-1}\boldsymbol{A}_i \tag{2-1}$$

$^{\mathbf{0}}\boldsymbol{T}_i$ is the homogeneous transformation representing the position of coordinate frame i with respect to the world coordinate frame 0. $^{i-1}\boldsymbol{A}_i$ is the transformation matrix that transfers the link coordinate frame i-1 to the next coordinated frame i. It is

$$^{\boldsymbol{i-1}}\boldsymbol{A}_{\boldsymbol{i}} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & \alpha_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & \alpha_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2-2}$$

We can see that this transformation matrix $^{\boldsymbol{i-1}}\boldsymbol{A}_{\boldsymbol{i}}$ **i**s a 4×4 homogeneous matrix. The entries of the matrix are composed of trigonometric functions of θ_i, α_i and d_i. The upper left 3×3 submatrix of $^{i-1}\boldsymbol{A}_i$ represents the rotational relationship from the i-1th link frame to the ith link frame. The fourth column of $^{i-1}\boldsymbol{A}_i$ describes the transitional relationship from the origin of the i-1th link frame to the origin of the ith link frame. The rest entries are zeroes which make the matrix a homogeneous one.

There are two ways to do kinematics analysis, one is forward kinematics and the other is inverse kinematics. A forward analysis of a serial manipulator determines the unique position and orientation of the last link with a specified set of joint variables. It is obtained by a sequence of multiplication of the transformation matrixes

$$^{0}\boldsymbol{T}_{n} = {}^{0}\boldsymbol{A}_{1}\,{}^{1}\boldsymbol{A}_{2}\cdots{}^{n-1}\boldsymbol{A}_{n} \tag{2-3}$$

$^{0}\boldsymbol{T}_{n}$ is the overall manipulator transform, that is the end frame with respect to the base frame. A typical robot manipulator normally has 6 joints because many end-effectors need 6 degree of freedom to reach arbitrary position in the space. For example, the Puma 560 is a 6-axis robot, so the overall transform can be written as $^{0}\boldsymbol{T}_{6}$ or $\boldsymbol{T_6}$ for short.

An inverse kinematic analysis of a serial manipulator determines all possible sets of the joint variables for any specified end-effector location. Each set of joint variables defines a particular arm pose for the given end effector location. The joint variables can be summarized as

$$\mathrm{q_i} = \begin{cases} \theta_i & \text{for a revolute joint} \\ \mathrm{d_i} & \text{for a prismatic joint} \end{cases} \tag{2-4}$$

The inverse kinematic analysis is especially important when the end-effector is scheduled to perform some specified task. For example, a fruit picking robot manipulator is built to reach a number of fruit locations and then collect the picked fruit by placing them in a container. In order to find out the joint variables, we can generalize the overall transform matrix as

$$\mathrm{T_n} = \mathrm{K}(\vec{\mathrm{q}}) \tag{2-5}$$

where $\vec{q} = [q_1\ q_2\ q_3\ \ldots\ q_n]^T$ is the variable we need to solve and T_n is known from the given task location for the end-effector.

$$\vec{\mathrm{q}} = \mathrm{K}^{-1}(\mathrm{T_n}) \tag{2-6}$$

There are many methods to solve the inverse kinematic problem. Matlab Robotics Toolbox is a fast way to compute the solution. Dr. Crane (1998) also introduced a closed loop method to solve out kinematic analysis problems in his book. In most cases, the solution is not unique, and for some classes of manipulator no solution exists. No solution cases may be due to an alignment of joint axes reducing the effective DOF, or the target point being out of workspace.

If the manipulator has more than 6 joints, it is said to be redundant and the solution for joint angles is under-determined.

Manipulator Dynamics

Manipulator dynamics is the study of motion with regard to forces. Motion is described by the displacement of end-effector, joint velocity and acceleration. Forces are exerted as actuators which generate torques to drive the motion. Dynamic analysis is vital for mechanical design, control and simulation. Similar to kinematics, there are two important ways to solve dynamics analysis problems, one is forward (or direct) dynamics and the other is inverse dynamics. Forward dynamics is to compute the motion of manipulator from the given actuation forces and torques. Inverse dynamics is to compute the joint actuation forces or torques from the position of end-effector, velocity, and acceleration.

For an *n*-axis rigid-body manipulator, Armstrong (1986) derived a dynamic model which generalized the motion with equation

$$\mathrm{A(q)\ddot{q} + B(q)[\dot{q}\dot{q}] + C(q)[\dot{q}^2] + G(q) = \tau} \qquad (2\text{-}7)$$

where

$A(q)$ is the n × n symmetric inertia matrix or kinetic energy matrix;

$B(q)$ is the n × n(n−1)/2 matrix of Coriolis torques;

$C(q)$ is the n × n matrix of centrifugal torques;

$G(q)$ is the n-vector of gravity torques;

τ is the generalized joint force vector.

q is the n-vector of joint variables describing the pose of manipulator ;

$\dot{q}$ is the n-vector of joint velocities;

$\ddot{q}$ is the n-vector of joint accelerations;

The symbols $[\dot{q}\dot{q}]$ and $[\dot{q}^2]$ are given by:

$[\dot{q}\dot{q}] = [\dot{q}_1\dot{q}_2, \dot{q}_1\dot{q}_3, \cdots \dot{q}_1\dot{q}_n, \dot{q}_2\dot{q}_3, \dot{q}_2\dot{q}_4, \cdots \dot{q}_{n-2}\dot{q}_n, \dot{q}_{n-1}\dot{q}_n]^T$,

$[\dot{q}^2] = [\dot{q}_1^2, \dot{q}_2^2, \cdots \dot{q}_n^2]^T$.

For forward dynamics, we are given the right hand side of the Equation (2-7) and aim at finding $\ddot{q}$. Whilst for inverse dynamics, we are given the left hand side of the Equation (2-7) and aim at finding τ. The equation can be derived via a number of techniques, such as the Lagrange method, Newton-Euler (NE) methodology, and Kane's methodology (1983). These methods compute the matrixes A, B, C and G directly with complex expressions, and they are not recursive. The NE and Lagrange forms can be written in terms of the Denavit-Hartenberg parameters. Kane's method can have lower computational cost for specific manipulators, but cannot be generalized by D-H parameters. Luh (1980) provided a recursive formulation of the Newton-Euler equations with linear and angular velocities referred to link coordinate frames. He suggested a time improvement from the Lagrangian's formulation, and thus it became practical in the implement of computation. Comparing to the non-recursive forms, the recursive forms are more efficient.

The Matlab Robotics Toolbox uses the recursive Newton-Euler (RNE) formulation (1980) to compute the inverse dynamics problems. There are two steps to perform the recursive Newton-Euler techniques, first goes the backward recursion and then the forward recursion. The formulas derived by Hollerbach(1980) and Walker and Orin (1982) included two different coordinate systems which the expressions are based on, one is expressed in the base coordinate frame which is also the inertia frame, and the other is expressed in the link's internal coordinate frame.

Backward Recursion

The backward recursion propagates angular velocities, angular accelerations, linear accelerations, total link forces, and total link torques from the base coordinate frame. Figure 2-8 indicates the position vectors from the base origin to the origins of link frames and the center of mass. The velocity and acceleration parameters are computed through the basic concept of two points fixed on a rigid body.

For a rotational joint, the recursive equations represented in the base coordinate frame are:

$$\omega_i = \omega_{i-1} + z_{i-1}\dot{q}_i \tag{2-8}$$

$$\dot{\omega}_i = \dot{\omega}_{i-1} + z_{i-1}\ddot{q}_i + \omega_{i-1} \times z_{i-1}\dot{q}_i \tag{2-9}$$

$$\dot{p}_i = \dot{p}_{i-1} + \omega_i \times p_i^* \tag{2-10}$$

$$\ddot{p}_i = \ddot{p}_{i-1} + \dot{\omega}_i \times p_i^* + \omega_i \times (\omega_i \times p_i^*) \tag{2-11}$$

where

ω_i is the angular velocity of link i represented in the base coordinate frame

$\dot{\omega}_i$ is the angular acceleration of link i represented in the base coordinate frame

p_i is the position vector from the base coordinate origin to the joint i coordinate origin

p_i^* is the position vector from coordinate origin i - 1 to coordinate origin i

q_i is the joint variable of joint i

z_i is the unit vector in Z direction of joint frame i represented in the base coordinate frame

For effective computation, the Matlab Robotics Toolbox performs the recursive algorithm based on link's internal coordinate frames. Comparing with the base frame form, the local frame form describes the relevant link in its own coordinate frame, and identifies the kinematic perimeters with left superscripts to show which frame it is represented with. In order

to connect different coordinate systems, the local frame recursive formulas combine the rotation matrixes to transfer the coordinate frames.

The recursive equations represented in the link's internal coordinate frames are:

$$^{i}\omega_{i} = {}^{i}R_{i-1}({}^{i-1}\omega_{i-1} + {}^{i-1}z_{i-1}\dot{q}_{i}) \tag{2-12}$$

$$^{i}\dot{\omega}_{i} = {}^{i}R_{i-1}({}^{i-1}\dot{\omega}_{i-1} + {}^{i-1}z_{i-1}\ddot{q}_{i} + {}^{i-1}\omega_{i-1} \times {}^{i-1}z_{i-1}\dot{q}_{i}) \tag{2-13}$$

$$^{i}v_{i} = \dot{p}_{i} = {}^{i}R_{i-1}\,{}^{i-1}v_{i-1} + {}^{i}\omega_{i} \times {}^{i}p_{i}^{*} \tag{2-14}$$

$$^{i}\dot{v}_{i} = \ddot{p}_{i} = {}^{i}R_{i-1}\,{}^{i-1}\dot{v}_{i-1} + {}^{i}\dot{\omega}_{\imath} \times {}^{i}p_{i}^{*} + {}^{i}\omega_{i} \times \left({}^{i}\omega_{i} \times {}^{i}p_{i}^{*}\right) \tag{2-15}$$

where

$^{i}\omega_{i}$ is the angular velocity of link i represented in frame i,

$^{i}\dot{\omega}_{i}$ is the angular acceleration of link i represented in frame i,

$^{i}v_{i}$ is the linear velocity of frame i

$^{i}\dot{v}$ is the linear acceleration of frame i

$^{i}z_{i}$ is the local unit vector in Z direction, obviously $^{i}z_{i} = [0\ 0\ 1]^{T}$

$^{i}p_{i}^{*}$ is the position vector from coordinate origin i-1 to coordinate origin i with respect to frame i.

$^{i}p_{i}^{*} = [a_{i} \quad d_{i}sin\alpha_{i} \quad d_{i}cos\alpha_{i}]^{T}$

$^{i-1}\boldsymbol{R}_{i}$ is the rotation matrix defining frame i orientation with respect to frame i-1. It is the upper left 3×3 submatrix of the link transform matrix, which is

$$^{i-1}R_{i} = \begin{bmatrix} \cos\theta_{i} & -\sin\theta_{i}\cos\alpha_{i} & \sin\theta_{i}\sin\alpha_{i} \\ \sin\theta_{i} & \cos\theta_{i}\cos\alpha_{i} & -\cos\theta_{i}\sin\alpha_{i} \\ 0 & \sin\alpha_{i} & \cos\alpha_{i} \end{bmatrix} \tag{2-16}$$

As this rotation matrix is orthonormal, we have

$$^{i}R_{i-1} = \left({}^{i-1}R_{i}\right)^{-1} = \left({}^{i-1}R_{i}\right)^{T} \tag{2-17}$$

For a translational joint, the recursive equations represented the link's internal coordinate frames are:

$$^{i}\omega_i = {}^{i}R_{i-1}\,{}^{i-1}\omega_{i-1} \tag{2-18}$$

$$^{i}\dot{\omega}_i = {}^{i}R_{i-1}\,{}^{i-1}\dot{\omega}_{i-1} \tag{2-19}$$

$$^{i}v_i = \dot{p}_i = {}^{i}R_{i-1}({}^{i-1}v_{i-1} + {}^{i}z_i\dot{q}_i) + {}^{i}\omega_i \times {}^{i}p_i^* \tag{2-20}$$

$$^{i}\dot{v}_i = \ddot{p}_i = {}^{i}R_{i-1}({}^{i-1}\dot{v}_{i-1} + {}^{i}z_i\dot{q}_i) + {}^{i}\dot{\omega}_\imath \times {}^{i}p_i^* + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}p_i^*) + 2\,{}^{i}\omega_i \times ({}^{i}R_{i-1}\,{}^{i}z_i\dot{q}_i) \tag{2-21}$$

All terms have been defined previously. Boundary conditions are used to accomplish the algorithm. We introduce the effect of gravity as the acceleration of the base link and set the angular parameters as zero.

$\omega_0 = [0\ 0\ 0]^T$

$\dot{\omega}_0 = [0\ 0\ 0]^T$

$v_0 = [0\ 0\ 0]^T$

$\dot{v}_0 = [0\ 0\ -g]^T$

The backward recursion also needs to propagate total link forces, and total link torques. Hollerbach (1980) have derived the formulas as below. All terms can be referred to the base coordinate frame or the internal coordinate frames.

$$\ddot{r}_i = \dot{\omega}_i \times r_i^* + \omega_i \times (\omega_i \times r_i^*) + \ddot{p}_i \tag{2-22}$$

$$F_i = m_i\ddot{r}_i \tag{2-23}$$

$$N_i = I_i\dot{\omega}_i + \omega_i \times (I_i\omega_i) \tag{2-24}$$

Where the undefined terms are

r_i^* is a position vector from coordinate origin i to the link i center of mass

r_i is a position vector from the base coordinate origin to the link i center of mass,

m_i is the mass of link i

F_i is the total external force at the center of mess of link i

N_i is the total external torque at the center of mess on link i

I_i is the inertia tensor of link i about its center of mass. I can be described as a 3×3matrix, in which the diagonal entries are the moments of inertia, and the off-diagonals are products of inertia.

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix} \tag{2-25}$$

Forward Recursion

The forward recursion propagates the forces and moments exerted on link i by link i-1 from the end link of the manipulator to the base.

$$f_i = f_{i+1} + F_i \tag{2-26}$$

$$n_i = n_{i+1} + N_i + (p_i^* + r_i^*) \times F_i + p_i^* \times f_{i+1} \tag{2-27}$$

$$\tau_i = \begin{cases} z_{i-1} \cdot n_i & \text{if link } i+1 \text{ is rotational} \\ z_{i-1} \cdot f_i & \text{if link } i+1 \text{ is translational} \end{cases} \tag{2-28}$$

Where the undefined terms are

f_i is the force exerted on link i by link i-1,

n_i is the moment exerted on link i by link i- 1

τ_i is the input torque exerted by actuator at joint i.

The implicit reference coordinate frame in these formulas is the base coordinate frame. Similar to the backward recursion, we can introduce rotation matrixes and internal coordinate systems to the formulas as:

$$^{i}f_i = {}^{i}R_{i+1}\,{}^{i+1}f_{i+1} + {}^{i}F_i \tag{2-29}$$

$$^{i}n_i = {}^{i}R_{i+1}({}^{i+1}n_{i+1} + {}^{i+1}R_i\ {}^{i}p_i^* \times {}^{i+1}f_{i+1}) + ({}^{i}p_i^* + r_i^*) \times {}^{i}F_i + {}^{i}N_i \tag{2-30}$$

$$\tau_i = \begin{cases} \left({}^{i}n_i \right)^T \left({}^{i}R_{i+1}\, {}^{i-1}z_{i-1} \right) & \text{if link } i+1 \text{ is rotational} \\ \left({}^{i}f_i \right)^T \left({}^{i}R_{i+1}\, {}^{i-1}z_{i-1} \right) & \text{if link } i+1 \text{ is translational} \end{cases} \quad (2\text{-}31)$$

All terms have been defined previously. After finishing the backward and forward recursion processes, the torques or forces from the actuators come out as the solution of inverse dynamics.

The Matlab Robotics Toolbox uses Method 1 in Walker and Orin techniques (1982) for computing the forward dynamics. This method makes use of the recursive Newton-Euler solution of inverse dynamics.

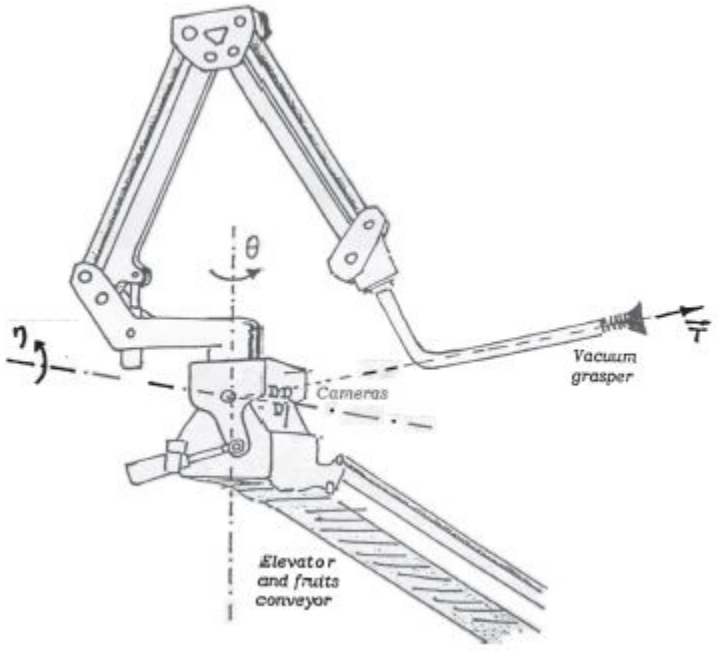

Figure 2-1. MAGALI apple-picking manipulator developed by Grand D'Esnon et al. (1987)

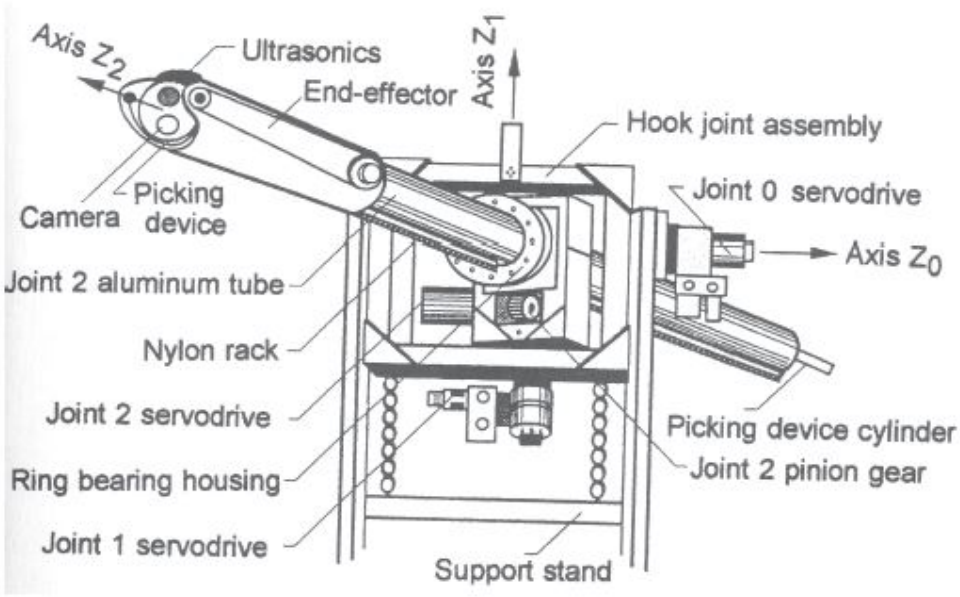

Figure 2-2. Citrus-picking robot developed by Harrell (1988)

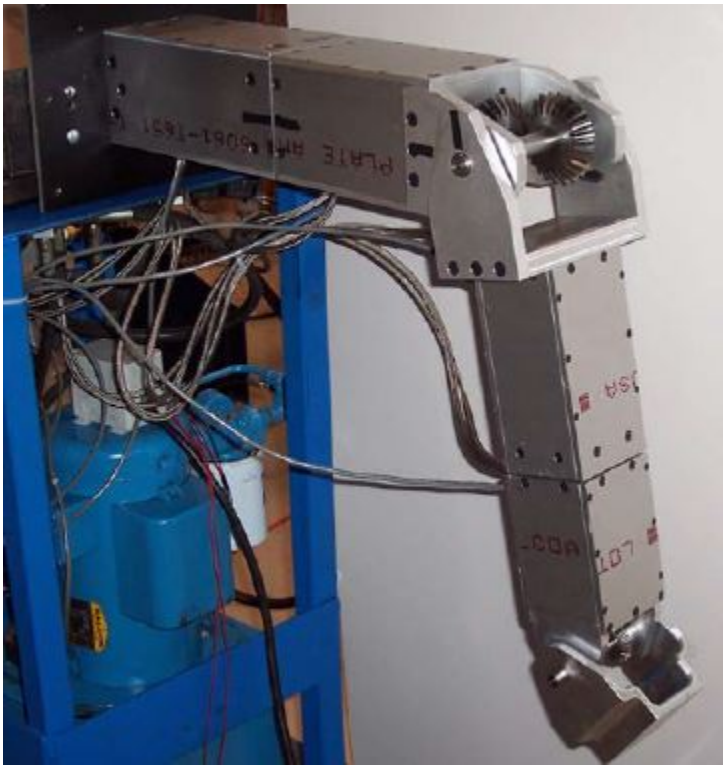

Figure 2-3. The prototype of citrus harvesting robot manipulator (Source: Babu Sivaraman. *Design and Development of a Robot Manipulator for Citrus Harvesting*. Dissertation for the degree of doctor of philosophy at University of Florida. 2006)

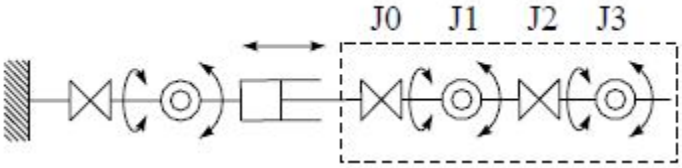

Figure 2-4. The joint types of the manipulator (Sivaraman et al., 2006)

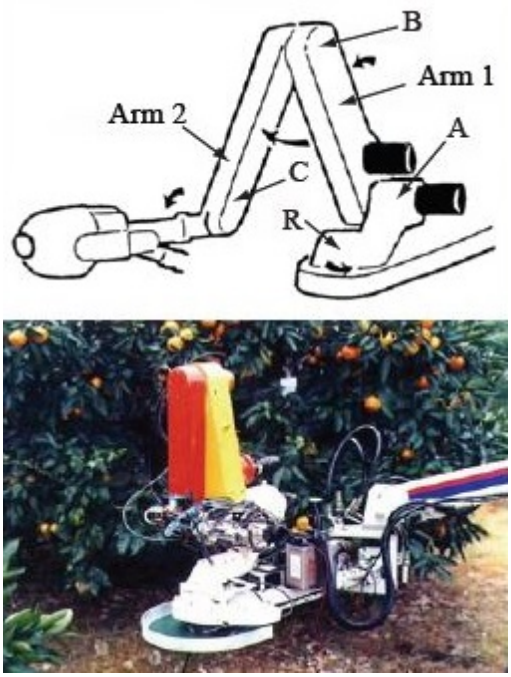

Figure 2-5. Kubota mandarin orange harvesting robot (Source: Naoshi Kondo, Mitsuji Monta & Noboru Noguchi. *Agricultural Robots Mechanisms and Practice.* Kyoto University Press. 2006)

Figure 2-6. Tomato harvesting robot designed by Okayama University. Source: Reprinted with permission from (Source: Naoshi Kondo, Mitsuji Monta & Noboru Noguchi. *Agricultural Robots Mechanisms and Practice.* Kyoto University Press. 2006)

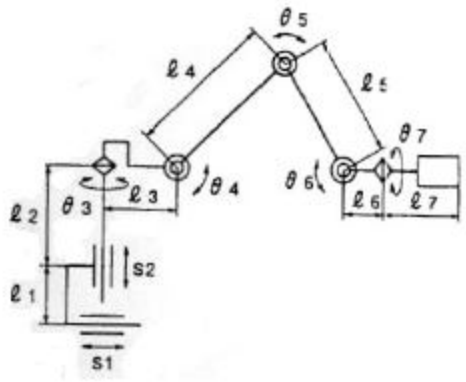

Figure 2-7. Tomato harvester manipulator (Kondo et al., 1996)

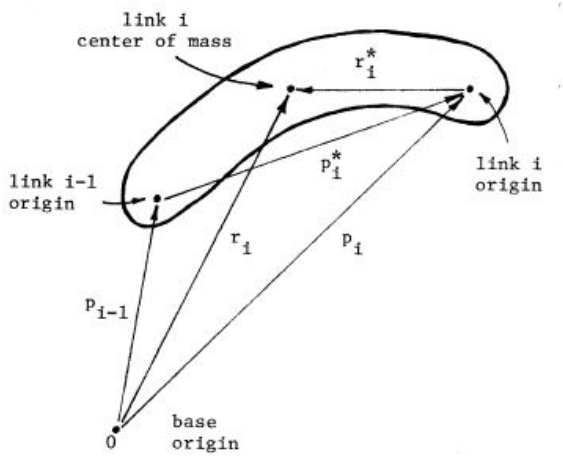

Figure 2-8. Position vectors between the base origin and the link origins and the center of mass (Kurfess, 2005)

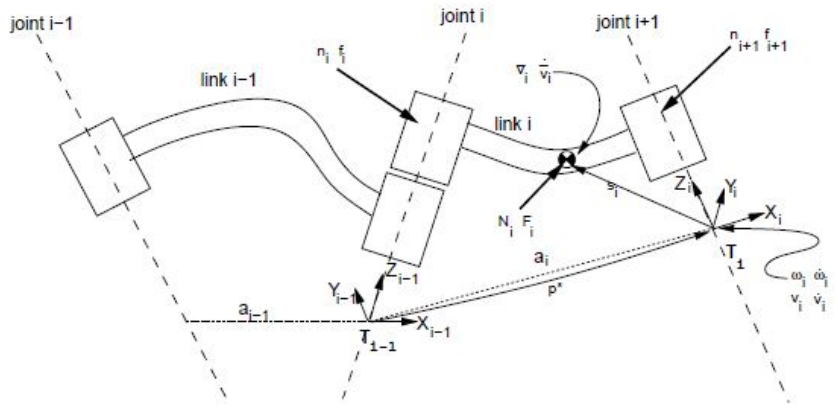

Figure 2-9. The notation used for inverse dynamics (Corke, 2002)

CHAPTER 3
KINEMATICS FOR ROBOTIC MANIPULATOR TO PICK FRUIT

Puma Robot Manipulator

Puma robot is one of the numerous mature robots that are popular in many industries. It is easy to do kinematic and dynamic analysis for Puma, because the data are available in many robotics research studies. In this research, the Puma 560 robotic arm is selected as a representative fruit picking manipulator, because the arm structure covers the characters from many other fruit picking robots.

The Puma Robot from the Unimation Company is a widely-used model of electrically driven robots. The series 500 is designed for applications requiring high degrees of flexibility and reliability. The Puma 560 is a six degrees of freedom robot manipulator with 6 rotational joints, shown in Figure 3-1. The end-effector of the robot arm can reach a point within its workspace from any direction. The six degrees of freedom are controlled by six brushed DC servo motors. The power requirement is 110-130VAC, 50-60Hz, 1500Watts. Its load capacity is 2.5kg (5.5 lbs.), and the maximum straight line velocity is 0.5m/sec. The arm of Puma 560 weighs 54.5kg (120 lbs.) The robot can determine its global position from the given feedback information. The robot can be controlled remotely through a network.

Matlab Robotics Toolbox

The Matlab Robotics Toolbox (Corke, 2002) can perform many useful functions for robotics analysis such as kinematics, dynamics, and trajectory generation. The creation of a serial-link robot manipulator is based on the D-H parameters. The Toolbox provides functions for converting between vectors, homogeneous transformations matrixes and quaternions which are necessary to represent 3-dimensional position and orientation. The Toolbox is also capable of plotting the 3D graphical robot and allows users to drive the robot model.

Forward Kinematics for Puma 560

According to Denavit and Hartenberg notation, we can number each link and joint as show in Figure 3-2. The location of the end-effector is dominated by the angles of joints 1, 2 and 3, while joints 4, 5 and 6 play important roles in changing the orientation of the end-effector. Generally speaking, the workspace of the end-effector is determined mostly by the lengths of links 1, 2 and 3.

The standard coordinate frames and the D-H parameters are showed in Figure 3-3. Frame 0 is attached to link 0 and z_0 is aligned with the axis of joint 1. Frame 1 is attached to link 1 and z_1 is aligned with the axis of joint 2, and so on. Table 3-1 lists the values of link twist, link length, link offset of each link and the joint angle is variable.

The forward kinematic analysis of Puma 560 is done by using Matlab Robotics Toolbox. Several functions are used to do the analysis conveniently. First, create the six links and set the D-H parameters as given in Table 3-1. The joint variable is set to be 0 at the beginning. The first three links have rotation limits, which are defined as link properties. As each link is well defined, use “robot” function to set up the puma model. Second, define the time step and generate a time vector. A few useful joint angle vectors are generated to define the starting position, ready position or reach position. We can get the trajectory between any of these two joint angle vectors with respect to time. After that, we can easily perform the forward kinematic analysis by using “fkine” function, which returns a homogeneous transformation for the final link of the manipulator. A 3-dimensional matrix is returned, the first two dimensions homogeneous transformation and the third dimension is time. Finally, a few figures are plotted to present the result. Matlab Robotics Toolbox allows users to see the movement of the robot in 3D space, which makes the trajectory visible. The whole function returns the coordinates of the end-effector if the user input the angle of every joint. Here is the source code in Matlab.

```
function [preach]= puma560ForwK(qreach)
close all;
%create links using D-H parameters
%L = link([alpha, a, theta, d], convention)
L{1} = link([ pi/2  0       0   0      0], 'standard');
L{2} = link([ 0     .4318   0   .15005 0], 'standard');
L{3} = link([-pi/2  .0203   0   0      0], 'standard');
L{4} = link([pi/2   0       0   .4318  0], 'standard');
L{5} = link([-pi/2  0       0   0      0], 'standard');
L{6} = link([0      0       0   0      0], 'standard');
%assign joint rotation limit
L{1}.qlim=[deg2rad(-160) deg2rad(160)];
L{2}.qlim=[deg2rad(-125) deg2rad(125)];
L{3}.qlim=[deg2rad(-270) deg2rad(90)];
%build up the robot model
puma560 = robot(L, 'Puma 560');
qzero=zeros(1,6);
qready = [0 -pi/4 pi/4 0 0 0]; % ready position
% generate a time vector
t=[0:0.056:2];
% compute the joint coordinate trajectory
q = jtraj(qready, qreach, t);
% forward kinematics for each joint coordinate
T = fkine(puma560, q);
preach=T(:,4,36);

%figure plotting
figure,title('Puma 560 Forward Kinematics');
plot(puma560_ready,qready,'noname','erase'),
hold on
plot(puma560,q,'noname','noerase');
figure,plot3(squeeze(T(1,4,:)),squeeze(T(2,4,:)),
squeeze(T(3,4,:)));
xlabel('X (m)'),ylabel('Y (m)'),zlabel('Z (m)'),
title('End-effector 3D trajectory'),grid;
figure,plot(t,squeeze(T(1,4,:)),'-',t,squeeze(T(2,4,:)),'--',t,
squeeze(T(3,4,:)),'-.'),
xlabel('Time(s)'),   ylabel('Coordinate (m)'),grid,
legend('X','Y','Z'),
title('End-effector Position');
figure,plot(t,rad2deg(q(:,1)),'-',t,rad2deg(q(:,2)),'--',t,
rad2deg(q(:,3)),'-.'),
xlabel('Time(s)'),   ylabel('Joint Angle (Deg)'),grid,
legend('Joint 1','Joint 2','Joint 3'),
title('Joint Angle Variation');

end
```

For example, if we input the joint angles as

```
qreach=[1.0694 0.0637 -0.9054 0.0000 0.8417 -1.0694]
```

which means the robot needs to reach a position where θ_1=1.0694, θ_2=0.0637, θ_3=−0.9054, θ_4=0, θ_5=0.8417, θ_6=−1.0694, the function returns the coordinate vector as

```
ans =
    0.5000
    0.6000
    0.3000
    1.0000
```

which means the final position of the end-effector is [0.5 0.6 0.3] with respect to the origin of the base coordinated frame of Puma robot.

Figure 3-5 to Figure 3-8 are the outputs of Puma 560 forward kinematics programming. Figure 3-5 shows the variation of the first three joint angles with respect to time. A 7^{th} order polynomial is used to fit the values. Figure 3-6 is the 3D simulation of Puma 560, and it reflects the motion in the space of every link gradually. The initial pose is shown in blue and the final pose in black. It takes 2 seconds for the Puma 560 to perform the motion from the initial pose to the final through every time step. Figure 3-6 is the trajectory showing the space displacement of the end-effect, which provides a visualized viewpoint on the performance. Figure 3-8 shows the change of X, Y, Z coordinates of the end-effector. In this case, the end-effector moves farther along y direction than along x direction, and the z span is the shortest.

Inverse Kinematics for Puma 560

The inverse kinematic analysis of Puma 560 is done by using Matlab Robotics Toolbox. As in forward kinematics, we create the six link robot model and a time vector first. As the ready pose is given by joint angles, we can use forward kinematics function to generate the transform matrix of the ready pose with respect to the base of the robot. The goal position is given by coordinate value, we can use “transl” function to create goal transform matrix. The “ctraj”

function is used to produce the trajectory from the two transform matrix in the Cartesian coordinate. It returns a Cartesian trajectory with straight line motion from the starting point to the finishing point represented by homogeneous transforms. After that, we can easily perform the inverse kinematic analysis by using “ikine” function. This function returns a serial of vectors and each line of vector represents the six joint angle values in a step of time. Finally, a few figures are plotted to present the result. The whole function returns the final angle of every joint if the user input the position of the end-effector. Here is the source code in Matlab. The robot building up part is omitted because it is the same as forward analysis.

```
function [qreach] = puma560InverK(preach)
%create links using D-H parameters
……
%build the robot model
……

t=[0:0.056:2];
qready = [0 -pi/4 pi/4 0 0 0];

%form a homogeneous transformation matrix from the ready joint
angles using forward kinematic analysis
T0 = fkine(puma560, qready);
%form a homogeneous transformation matrix from goal position
T1= transl(preach);

%compute a Cartesian path
TR01 = ctraj(T0, T1, length(t));   % to reach the goal position
TR10 = ctraj(T1, T0, length(t));   %return to the ready position

%inverse kinematic analysis
q=ikine(puma560, TR01);            %reach the goal only
qq=ikine(puma560,TR10);            %return to initial pose
qreach=q(36,:);
……
%figure plotting
……

end
```

For example, if we input the goal position as

```
puma560InverK([0.414, -0.203, 0.597])
```

which means the robot needs to reach a position has the coordinate values x=0.414, y=−0.203, z=0.597. The program returns the joint angle vector as

```
ans = -0.1244    0.3955   -0.4354   -0.0000    0.0399    0.1244
```

which means in order to reach the final position, the 6 joint angles should have the values of θ_1=−0.1244, θ_2=0.3955, θ_3=−0.4354, θ_4=0, θ_5=0.0399, θ_6=0.1244

From the output figures, we can observe the movement of robot manipulator. It starts from a ready position and then move to the goal position which is inputted by user. After reaching the goal position, it pauses for a second to allow the end-effector finishing its tasks. Finally, the manipulator returns to its starting position and ready for another motion. The program also produces joint angle values showing the change of each joint angle with respect to time. See Figure 3-9. In this case, the joints 2 and 3 rotate by larger angles than joint 1, and all of the joints rotate within their joint limits. Figure 3-10 is the 3D trajectory of the end-effector from the invers kinematics. A straight line motion is created rather than a polynomial fitting curve because the use of invers kinematics is to find a fastest path to reach the goal.

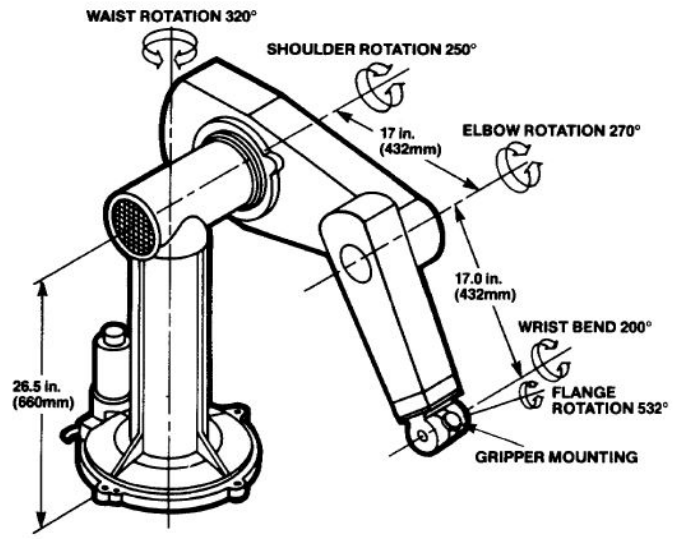

Figure 3-1. PUMA 560 robot manipulator (Unimation, 1984).

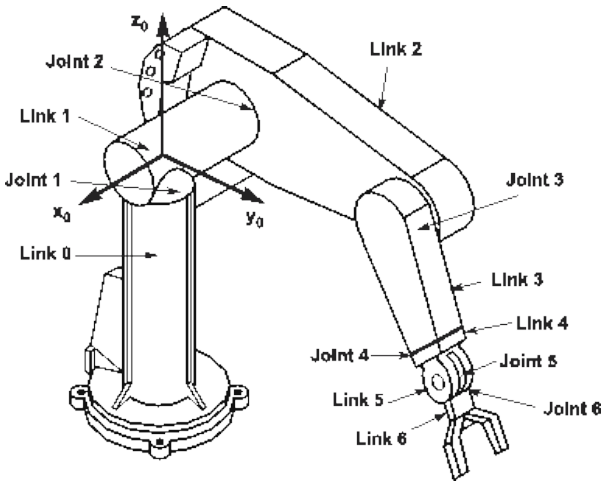

Figure 3-2. Links and joints of Puma 560 (Benitez, et al., 2012).

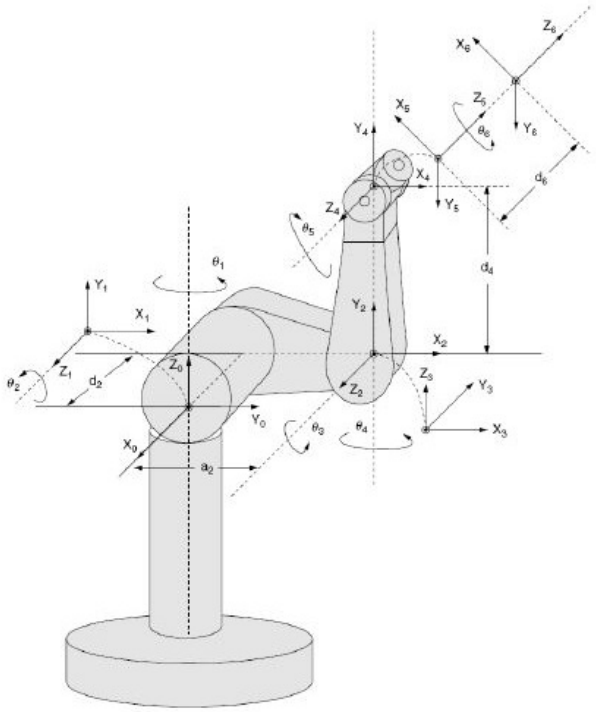

Figure 3-3. Coordinate frames of Puma 560 (Kurfess, 2005).

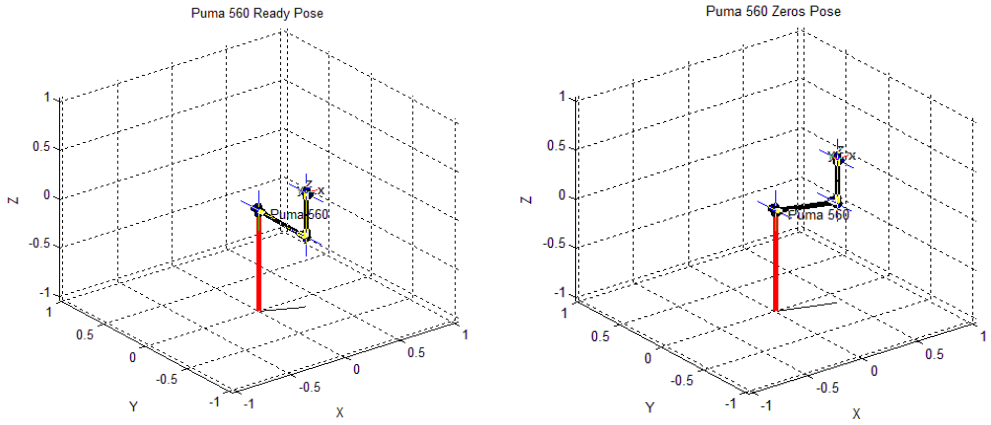

Figure 3-4. Puma 560 ready pose and zero pose.

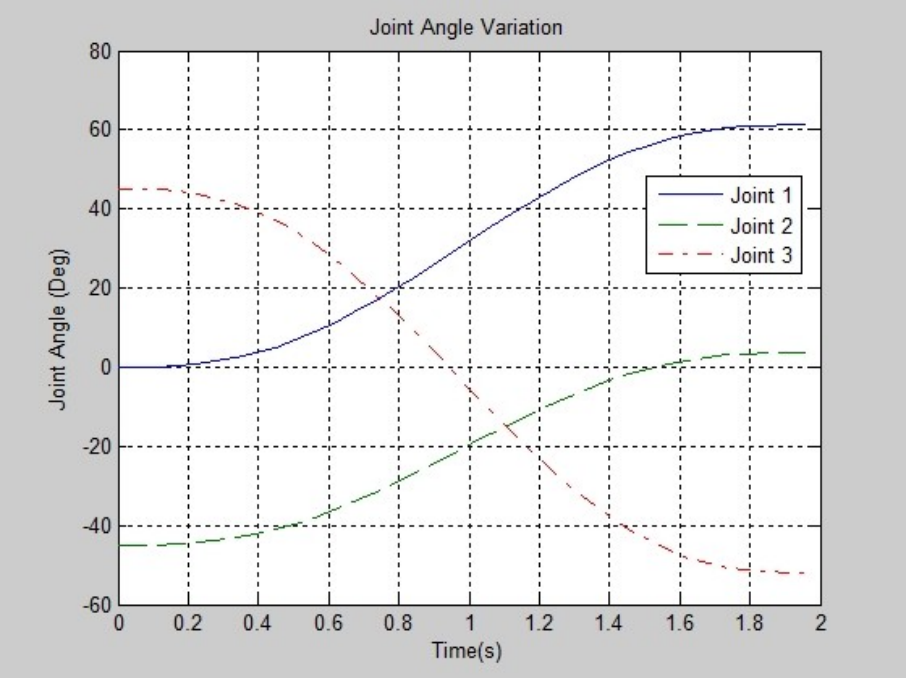

Figure 3-5. Puma 560 forward kinematics joint angles.

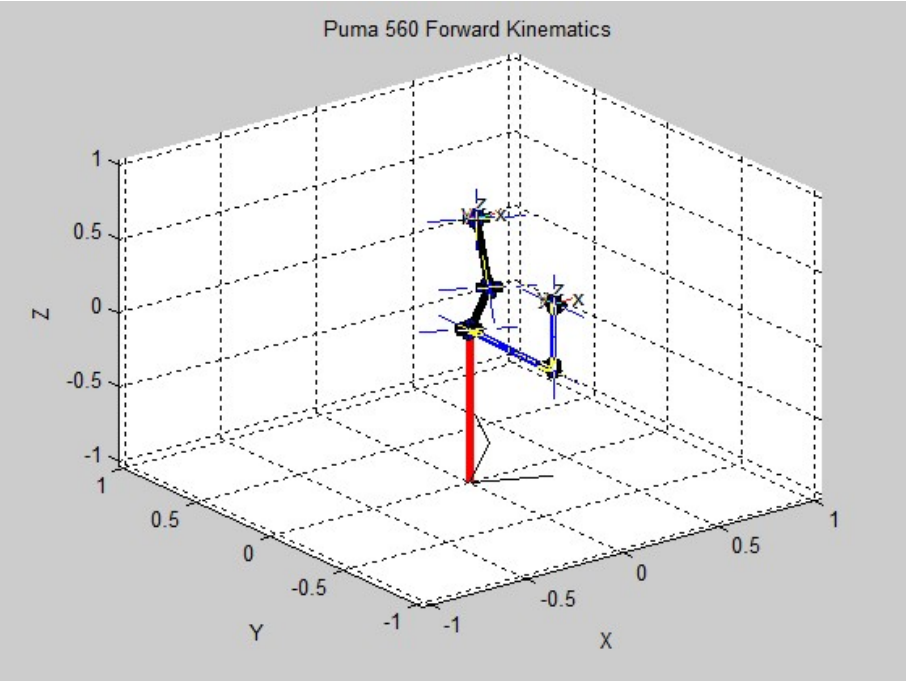

Figure 3-6. Puma 560 forward kinematics simulation.

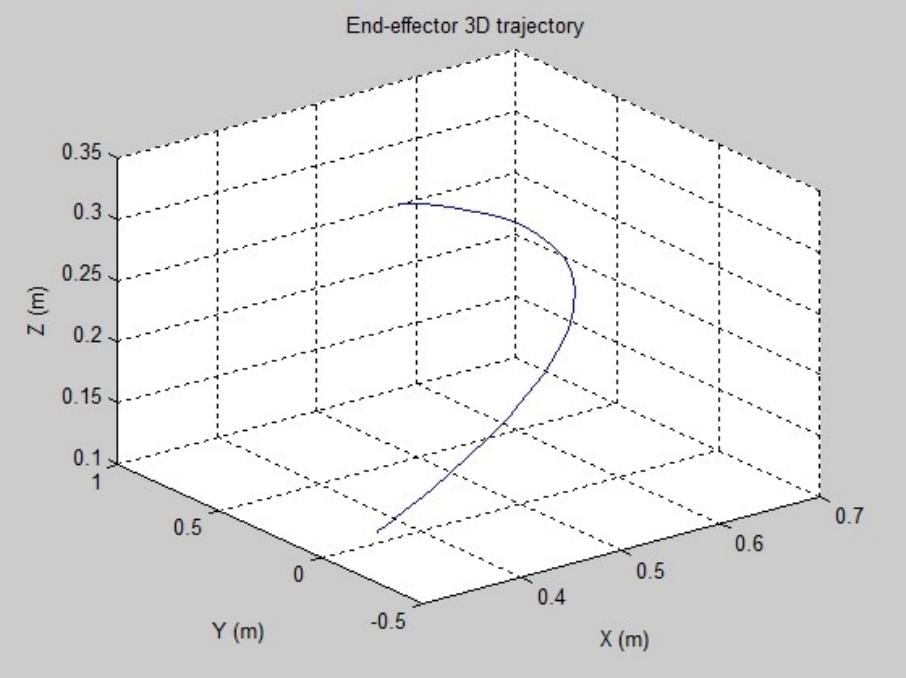

Figure 3-7. Puma 560 forward kinematics end-effector trajectory.

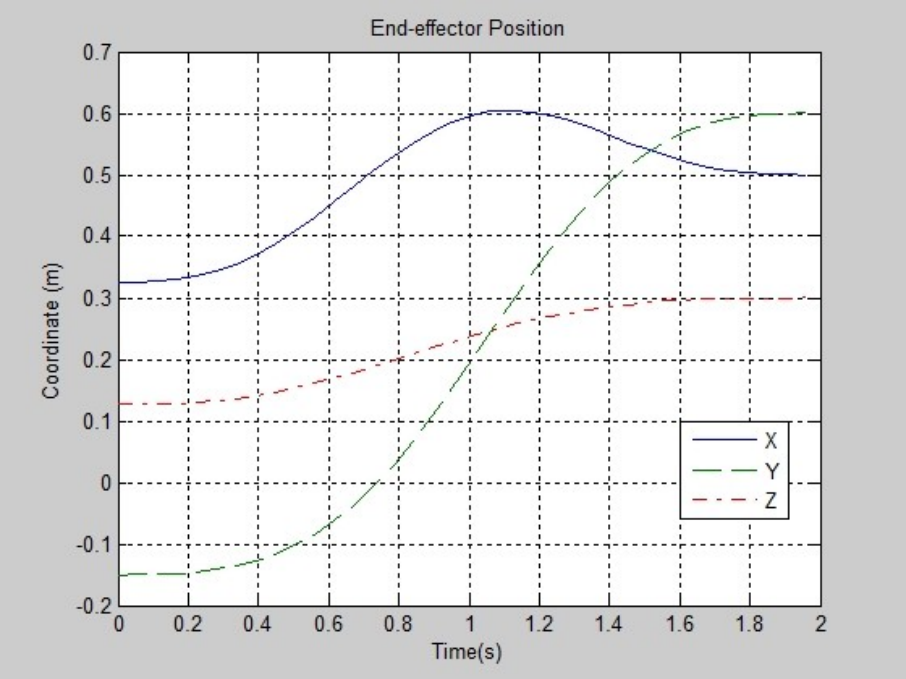

Figure 3-8. Puma 560 forward kinematics end-effector X, Y, Z coordinates.

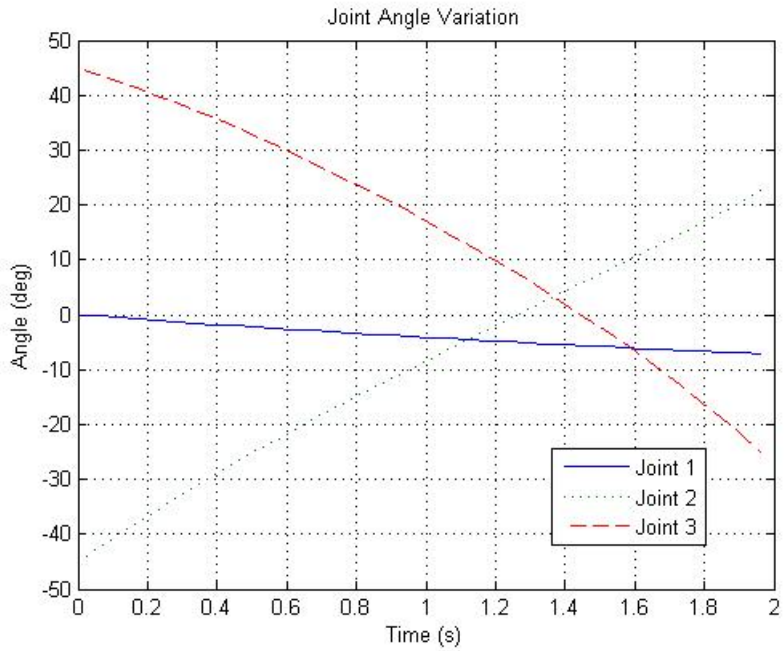

Figure 3-9. Puma 560 inverse kinematics joint angles.

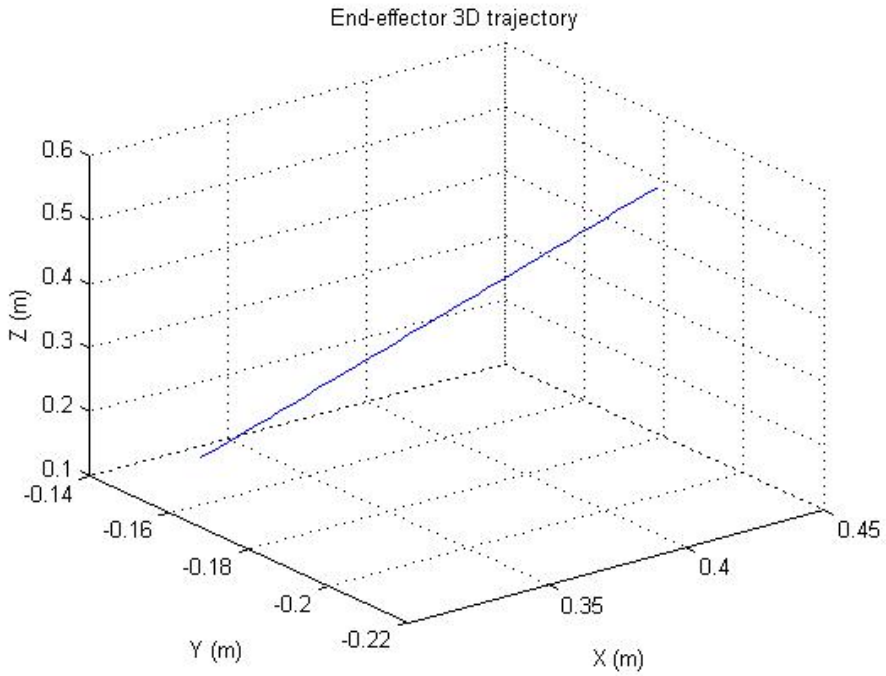

Figure 3-10. Puma 560 inverse kinematics end-effector trajectory.

Table 3-1. D-H parameters of Puma 560.

Link No.	Link Twist α (rad)	Link Length a (m)	Link Offset d (m)	Joint Angle θ (rad)	Joint Type
Link 1	1.570796	0	0	Variable θ_1	Revolute
Link 2	0	0.4318	0.15005	Variable θ_2	Revolute
Link 3	-1.570796	0.0203	0	Variable θ_3	Revolute
Link 4	1.570796	0	0.4318	Variable θ_4	Revolute
Link 5	-1.570796	0	0	Variable θ_5	Revolute
Link 6	0	0	0	Variable θ_6	Revolute

CHAPTER 4
DYNAMICS FOR ROBOTIC MANIPULATOR TO PICK FRUIT

Dynamic Parameters of the Puma 560

Many studies have been working on the examination of the dynamic parameters for Puma robots. Some of the well-known studies are from Armstrong (1986), Paul (1981), Lee (1983) and Tarn(1985). Corke (1994) have also did research on the comparison among those studies. The main dynamic parameters for a robot manipulator can be separated into two aspects, one is regarding the link's inertia and the other is regarding the actuator or motor. For the link's inertia, we mostly consider:

- Link mass
- Link center of gravity
- Link moments of inertia

Comparing to the other studies, Paul's (1981) research is more fundamental and follows the standard D-H notation. The data I use in this thesis refers to Paul's experiment because it contains detail values of the first link of Puma 560. Moreover, Paul assumes the link mass with uniform distribution, which is useful for my research in the next section. Table 4-1, Table 4-2, and Table 4-3 list the link's dynamic parameters of Puma 560

For the actuators, the dynamic parameters we mostly consider are:

- Motor inertia
- Gear ratio
- Friction

There are two types of Puma 560, one is built by Unimation and the other is by Kawasaki (Japan). The two types are similar in most respects but use different servo motors. Typically, the motors used for the first 3 joints are larger than for the last 3 joints due to the purpose of the links. The motor inertia combining with the link inertia is used to learn the total rotational inertia at

each joint. The motor friction reported in previous studies is Coulomb and viscous friction. The motor data used in the following analysis is based on the Unimation type.

As discussed, the code to build a Puma 560 robot using Matlab Robtics Toolbox can be defined as below.

```
%the dynamics parameters refer to Paul's study
%motor data refers to the Unimation Puma 560
%define link mass
L{1}.m = 4.43;
L{2}.m = 10.2;
L{3}.m = 4.8;
L{4}.m = 1.18;
L{5}.m = 0.32;
L{6}.m = 0.13;
%define center of gravity
L{1}.r = [ 0    0      -0.08];
L{2}.r = [ -0.216  0   0.026];
L{3}.r = [ 0    0      0.216];
L{4}.r = [ 0    0.02    0];
L{5}.r = [ 0    0       0];
L{6}.r = [ 0    0       0.01];
%define link inertial as a 6-element vector
%interpreted in the order of [Ixx Iyy Izz Ixy Iyz Ixz]
L{1}.I = [  0.195    0.195   0.026   0   0   0];
L{2}.I = [  0.588    1.886   1.470   0   0   0];
L{3}.I = [  0.324  0.324     0.017   0   0   0];
L{4}.I = [  3.83e-3  2.5e-3  3.83e-3  0  0   0];
L{5}.I = [  0.216e-3 0.216e-3  0.348e-3   0  0   0];
L{6}.I = [  0.437e-3 0.437e-3  0.013e-3  0   0   0];
%define motor inertia
L{1}.Jm =  200e-6;
L{2}.Jm =  200e-6;
L{3}.Jm =  200e-6;
L{4}.Jm =  33e-6;
L{5}.Jm =  33e-6;
L{6}.Jm =  33e-6;
%define gear ratio
L{1}.G =  -62.6111;
L{2}.G =  107.815;
L{3}.G =  -53.7063;
L{4}.G =  76.0364;
L{5}.G =  71.923;
L{6}.G =  76.686;
% motor viscous friction
```

```
L{1}.B =   1.48e-3;
L{2}.B =   .817e-3;
L{3}.B =    1.38e-3;
L{4}.B =   71.2e-6;
L{5}.B =   82.6e-6;
L{6}.B =   36.7e-6;
% motor Coulomb friction
L{1}.Tc = [ .395    -.435];
L{2}.Tc = [ .126    -.071];
L{3}.Tc = [ .132    -.105];
L{4}.Tc = [ 11.2e-3 -16.9e-3];
L{5}.Tc = [ 9.26e-3 -14.5e-3];
L{6}.Tc = [ 3.96e-3 -10.5e-3]
%build the robot model
puma560 = robot(L, 'Puma 560');
```

After defining the link mass, center of gravity, link inertial, motor inertia, gear ratio and Coulomb friction for the six links, the Puma 560 model is successfully built. The dynamic analysis will be performed on the base of this model.

Inverse Dynamics for Puma 560

The first several steps to do inverse dynamic analysis of Puma 560 in Matlab are similar to kinematics. Create the robot model with dynamics parameters which have been listed above, and then set up the time vector and useful pose vectors. To find a joint space trajectory between two joint posed, the Toolbox has a function “jtraj”. It uses a 7th order polynomial with default zero boundary conditions to compute the velocities and accelerations. It also allows the user to add the boundary conditions for velocity. The built in function “rne” is used to computer inverse dynamics via recursive Newton-Euler formulation. This function returns the torque from the actuator of each link. Here is the Matlab code to use these functions:

```
function [taufmax,tau0fmax] = puma560InverD(preach)
%create links using D-H parameters
……
%build the robot model
……
qready = [0 -pi/4 pi/4 0 0 0];
```

```
%use inverse kinematics solution to computer the joint angle
qreach=puma560InverK(preach);
% create time vector
t = [0:.056:2];
[q,qd,qdd]=jtraj(qready,qreach,t);
% compute joint coordinate trajectory
%compute inverse dynamics using recursive Newton-Euler algorithm
tauf = rne(puma560, q, qd, qdd);
% compute the joint torque with no motor friction
tau0f=rne(nofriction(puma560), q, qd, qdd);
% compute the joint torque from the gravity load only
taug = gravload(puma560, q);
taufmax=[max(abs(tauf(:,1))) max(abs(tauf(:,2)))
max(abs(tauf(:,3)))];
tau0fmax=[max(abs(tau0f(:,1))) max(abs(tau0f(:,2)))
max(abs(tau0f(:,3)))];
……
%figure plotting
……
end
```

To use this inverse dynamics function, I input a goal position for the end-effector as an example:

```
puma560InverD([0.5, 0.6, 0.3])
```

A few figures are plotted to show the trend and values of joint velocities, accelerations and torques.

Observing the Figure 4-1, the angular velocity of all the joints have similar trend. This is because the built-in curve fitting function to compute the velocity uses the same order of polynomial. The angular velocities increase from zero to the middle of time vector and then decrease gradually to zero for finishing. The curves are smooth with no singular point. For angular accelerations, all joints have their curve cross the zero line which means a trend switch of velocity. The crossing point implies the peek time of absolute velocity. In this case, the joint 3 has comparatively larger absolute values of angular velocity and acceleration among the joints.

Observing the Figure 4-2, the motor torque exerted on link 2 has the larger value than on link 1 and 3. This is because much of the torque on joints 2 is due to gravity. The torque on joint 2 keeps increasing while the ones on joint 1 and 3 switch direction during the motion. Joint 4-6 do not have much torque due to their type of motor.

When the motor's inner friction is added to analysis the model, we can find the result in Figure 4-3. The motor friction affects only the beginning and the end of the motion. It increases the absolute value of the motor torque in a rapid speed when the robot starts moving, and drops to the initial value fast when the robot reach the stop position. The torque trend within the process is just identical to the non-friction analysis.

Figure 4-4 shows the torques on joint 2 and 3 caused by the links, gravity loads. As we can see, the torque affected by gravity load keeps a large percentage of the total motion torque, while the torque from the pure motion occupies a comparatively small portion. The initial torque on joint 2 is already very large before it starts to move. This fact implies some suggestion on reducing the weight when we need to design a robot.

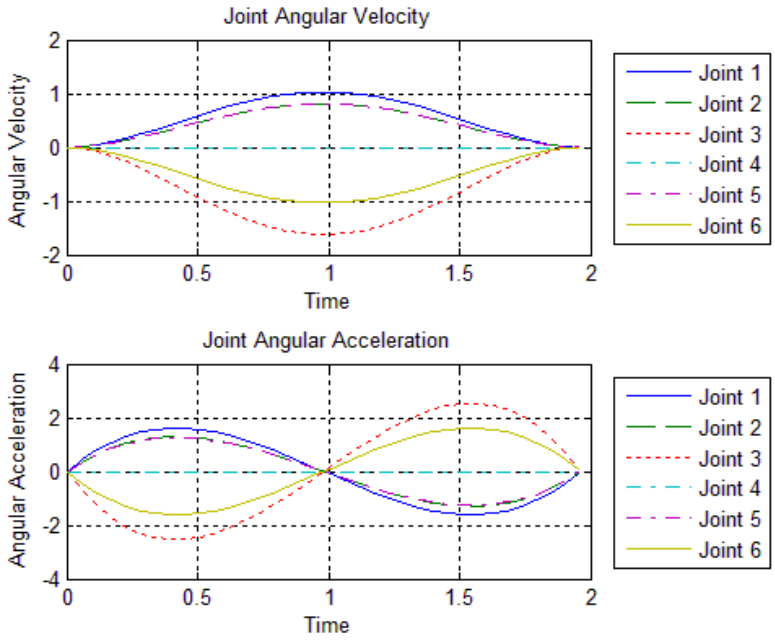

Figure 4-1. Link angular velocity and acceleration.

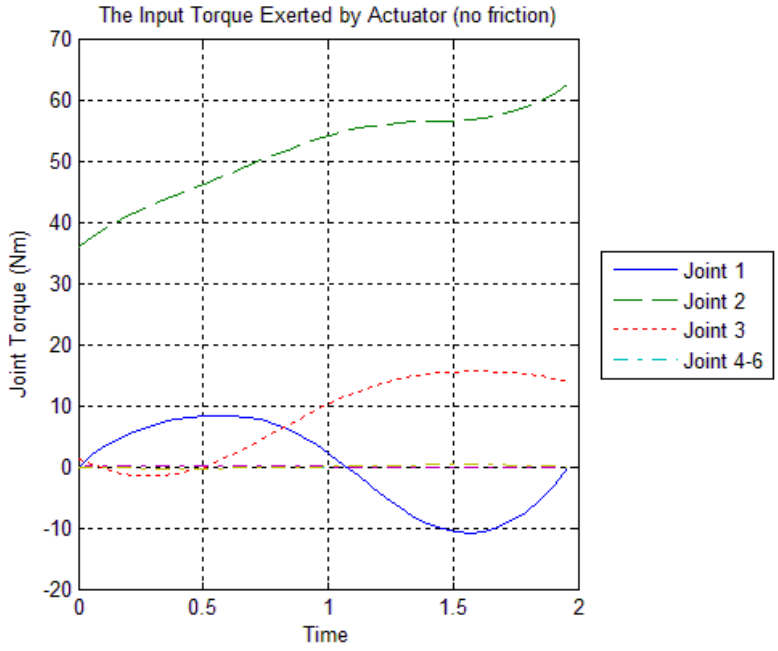

Figure 4-2. Link motor torque for the robot model without friction.

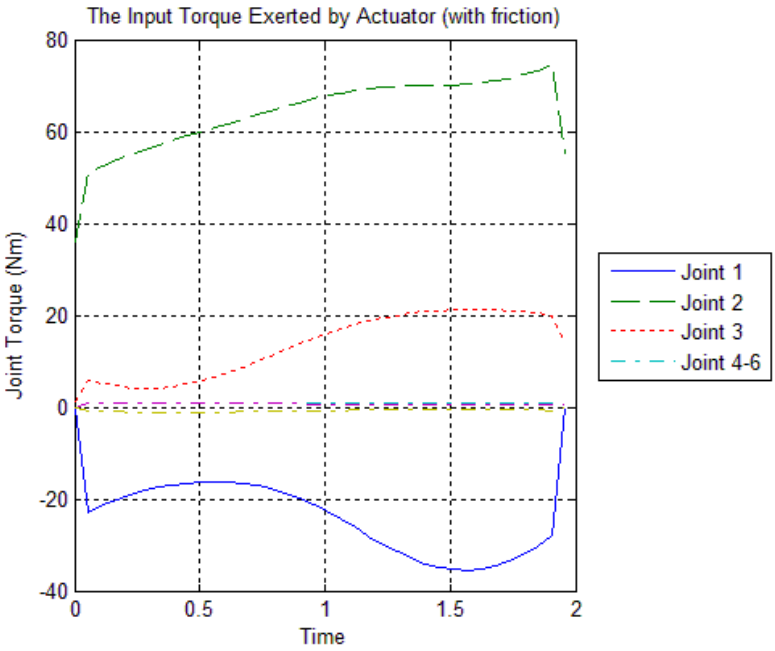

Figure 4-3. Joint motor torque for the robot model with friction.

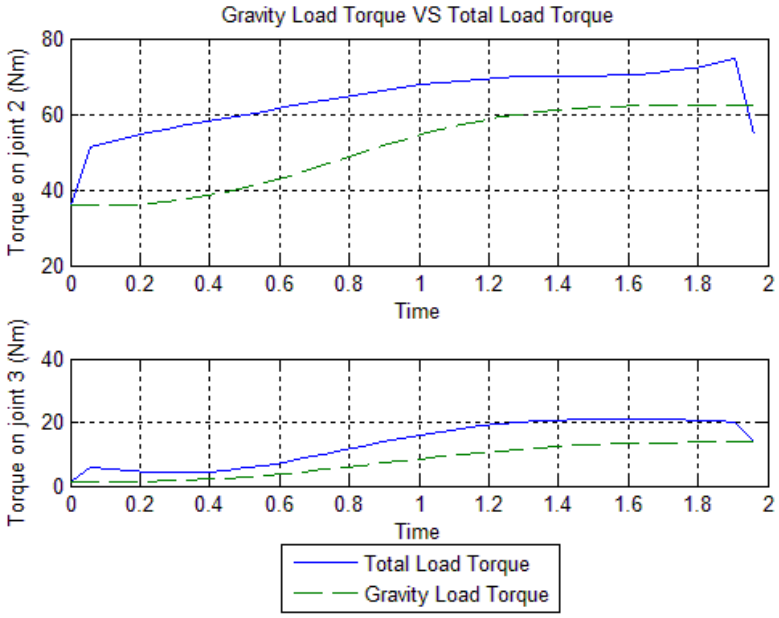

Figure 4-4. The comparisons between total load torque and gravity load torque.

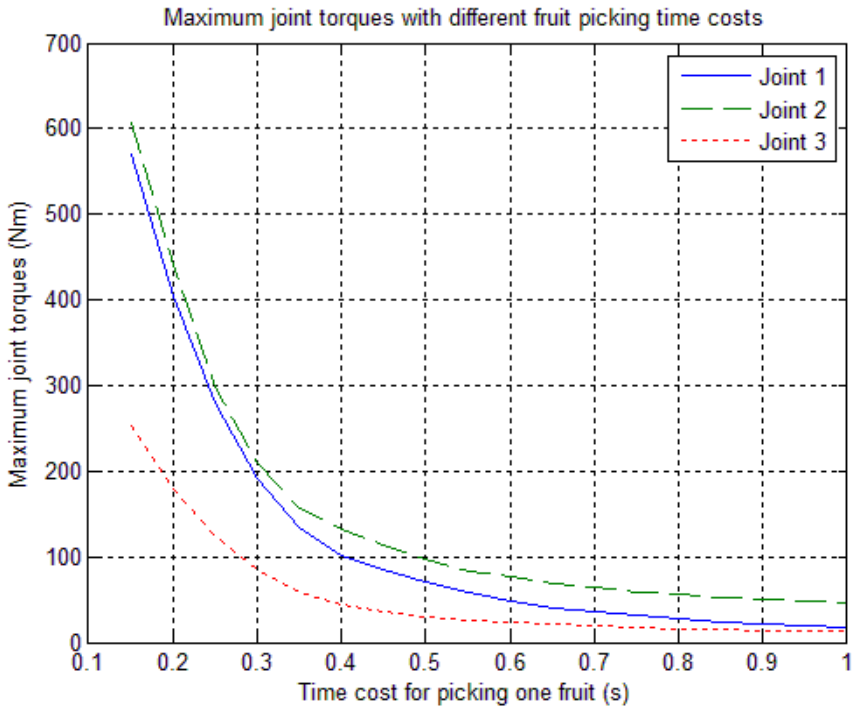

Figure 4-5. Maximum joint torques for different fruit picking duration.

Table 4-1. Link mass of Puma 560.

Link Index	Mass (kg)
1	4.43
2	10.20
3	4.8
4	1.18
5	0.32
6	0.13

Table 4-2. Center of gravity of Puma 560.

Link Frame	Direction	Coordinate Value(m)
1	X_1	0
	Y_1	0
	Z_1	-0.08
2	X_2	-0.216
	Y_2	0
	Z_2	0.026
3	X_3	0
	Y_3	0
	Z_3	0.216
4	X_4	0
	Y_4	0.02
	Z_4	0
5	X_5	0
	Y_5	0
	Z_5	0
6	X_6	0
	Y_6	0
	Z_6	0.0

Table 4-3. Moments of inertia about COG.

Link Index	Moments of Inertia	Value ($kg \cdot m^2$)
1	I_{xx1}	0.195
	I_{yy1}	0.195
	I_{zz1}	0.026
2	I_{xx2}	0.588
	I_{yy2}	1.886
	I_{zz2}	1.470
3	I_{xx3}	0.324
	I_{yy3}	0.324
	I_{zz3}	0.017
4	I_{xx4}	3.83e-3
	I_{yy4}	2.53e-3
	I_{zz4}	3.83e-3
5	I_{xx5}	0.216e-3
	I_{yy5}	0.216e-3
	I_{zz5}	0.348e-3
6	I_{xx6}	0.437e-3
	I_{yy6}	0.437e-3
	I_{zz6}	0.013e-3

CHAPTER 5
FRUIT PICKING ARM DESIGNER GUI

Assumption

In this chapter, a graphic user interface (GUI) is designed to generate fruit picking arms that satisfies the target plant type. In Chapter 2, I have listed a few previous harvesting robots that can pick tree fruit like apple, citrus and tomato. Comparing with one another, most of them have wheeled vehicles that carry picking manipulators and fruit collecting baskets, manipulators that send end-effects to reach the fruit, and end-effectors that perform the picking and detaching. Figure 5-1 shows Kubota orange harvesting robot (Sarig, 1993). This design concept can be used as an illustration and the GUI developed in this chapter is based on it as an assumption.

If we inspect the manipulator aims, all of them have several degrees of freedom, some dominate the picking workspace and others dominate the approaching angle of the end-effector. Normally, the workspace dominating links have actuators with larger torque requirements due to their longer link length, while the approaching angle dominating links have smaller motors with accurate controllers. Therefore, there exist some common concepts among the fruit picking arm developments as long as the plant structures are similar. These common concepts can be generalized in Figure 5-2. Three rotational joints and two relatively long links are the main body connecting the base and the end-effector. From the base, it first has a joint with vertical revolution axis, and then two revolute joints with horizontal axis carrying two dominant links. At the end of the second dominant link, there connects an end-effector which might contain 2 to 3 degrees of freedom. This robotic manipulator framework is right what Puma 560 has. Therefore, I assume the fruit picking arm we want to develop is similar to the first three links of the Puma 560 manipulator. The rest links of Puma 560 are of the end-effector which will not be considered in this research. The two workspace dominated links are assumed to have equal length.

Some more assumption is regarding with the dynamic parameters. As discussed in Chapter 4, the link's inertia parameters that we mostly consider are link mass, center of gravity and moments of inertia. The dynamic analysis of the fruit picking arm generated in this chapter refers to Puma 560 parameters. The link masses are assumed to be proportional to link lengths of Puma 560 with the same material density. Link center of gravities are also proportionally computed. The calculation of link moments of inertia is more complicated and refers to Armstrong's (1986) research on Puma 560. Armstrong presented a suspension model to measure the moments of inertia, see Figure5-3. The rotational inertia was concluded as

$$I = \frac{Mg * r^2}{\omega^2 * l} \tag{5-1}$$

where I is the inertia about the axis of rotation

Mg is the weight of the link

r is the distance from each suspension wire to the axis of rotation

ω is the oscillation frequency in radians per second

l is the length of the supporting wires

Applying this inertia computation, I assume the oscillation frequency and the length of the support wires are fixed for the picking arm generated in GUI. Therefore, the moment of inertia will be proportional to the link mass and link length square. This assumption is only valid for limited link length due to the way it is established. The motor internal frictions are not considered in the dynamic analysis here because the users have their own choices when designing a manipulator.

Other assumptions in this chapter are regarding with the orchard and manipulator behaviors. I assume the orchards or the greenhouses have enough room to place the robotic manipulator. A specified tree type in the same orchard is assumed to have similar fruit

distribution. Extreme cases are not considered and analyzed. The manipulator does not need to pick every fruit on a tree during each working period but can reach almost every fruit within its designed workspace. The detailed behavior of the end-effector is not considered in this research. The fruit positions are assumed to be detected by reliable sensors.

All the assumptions are valid because the GUI is developed for future researchers to get an overall fruit picking scheme before they design and build the arm. The users need to do further studies if they have specific limits and special conditions.

Graphical User Interface Development

The fruit picking arm designer GUI is created by using Matlab GUI due to its compatibility with Matlab Robotics Toolbox. Figure 5-4 shows the fruit picking arm designer GUI. It allows the user to input five typical fruit positions with respect to the tree coordinate frame. The instruction for entering the fruit coordinates is indicated in the upper left image in the window. Also see Figure 5-5.

The five typical fruit positions are the highest, the lowest, the right most, the left most and the front most which might be the closest to the robot arm. The first four positions determine the manipulator workspace and the last one determines the manipulator base location. If the user enters these five fruit positions from a specified tree, the generated picking arm is designed for this specified case. The user might collect the five typical fruit positions from the trees in one orchard and then enter their average coordinate values in GUI. In this way, the generated picking arm will be suitable to most trees in that orchard.

In Figure 5-5, a coordinate frame is set to trunk of a fruit tree with its origin to the ground. The x vector of this coordinate frame points to the base of the manipulator and the z vector is pointing up vertically. For easy computation, I located the base of the manipulator on the positive x vector. The five typical fruit cases are highlighted in orange with numbers from 1 to 5.

In order to reach this five fruit positions with a shortest arm length, the base of the manipulator need to be lifted to the middle of the height difference between the highest fruit and the lowest one. Let b represents the height of the base, then $b = \frac{z_1 - z_2}{2} + z_2$, where z_i is the height of fruit *i* in Figure 5-5. The Pythagorean Theorem is used to build the relationship between the arm length *a* and the horizontal offset from the tree to the base *d*. If we assume the highest fruit is reached when the arm is fully extended, the relationship could be $(2a)^2 = \left(\frac{z_1 - z_2}{2}\right)^2 + d^2$. Figure 5-6 shows the extended arm reaching the two typical fruit positions.

In general, the arm length a, the base height b, and base distance d are the outcome of the GUI, representing the basic characters of the fruit picking arm. These three characters identify the shortest arm length needed to be in order to reach the typical fruits, and the relative position the base need to be located with respect to the fruit tree. The computation of these three characters depends only on the inputted coordinates in green boxes in Figure 5-4. They are the height of the highest and the lowest fruit, the side offsets of the left most and the right most fruit, and the x coordinated of the front most fruit.

After the arm model is created, the user can easily do inverse kinematic and dynamic analysis on it. The analysis compiles the program using Matlab Robotics Toolbox which has been presented in previous chapters. If the user clicks the “Inverse Kinematics” button, a joint angle variation image will show up in the lower-left corner of the window. The image will include the joint angle variations of joint 1 to 3 with respect to time for the arm to pick the typical fruit by sequence. If the user clicks the “Inverse Dynamics” button, a motor torque variation image will show up in the lower-right corner of the window. The image will include the torque variations of joint 1 to 3 with respect to time for the arm to pick the typical fruit by sequence. Note that the arm is assumed to return to its initial pose for fruit collection before it

starting to make a new pick. Because the system only has limited target fruit positions, the kinematic and dynamic analysis cannot represent all the fruit picking cases. The typical fruit picking process is just analyzed as illustration to help the user understanding the arm's performance or selecting the motor. Figure 5-7 generalizes the work process of the GUI in a flow chart.

Fruit Tree Test Cases

To test the Fruit Picking Arm Designer, I use two special cases, one is a peach tree and the other is a citrus tree. Both of the fruit trees are cultivated in an orchard at University of Florida.

Test Case One: Peach Tree

Figure 5-8 is a photograph of a typical peach tree taken in 2013 summer. The tree is about 2 m in height and 3 m in width with a few mature peaches on it. The five peaches that the GUI requires to consider are highlighted in red circles with index numbers in order. A coordinate frame is attached to the peach tree, and the coordinate of the five specified peaches are measured as:

1. The highest peach: x_1=0.374 m, y_1=−0.104 m, z_1=2.012 m
2. The lowest peach: x_2=0.589 m, y_2= 0.018 m, z_2=0.356 m
3. The left most peach: x_3=0.298 m, y_3=−1.672 m, z_3=1.123 m
4. The right most peach: x_4=0.461 m, y_4= 1.720 m, z_4=1.505 m
5. The front most peach: x_5=0.603 m, y_5=−0.419 m, z_5=1.812 m

After entering these coordinate values in the input panel, the user just click the "generate" button, and then the character parameters of the fruit picking arm will be calculated and present in the output panel. In this case, the designed arm has

- Arm length a=1.651 m
- Base height b=1.184 m
- Base distance d=1.770 m

Therefore, the fruit picking arm designed for this peach tree should be located at 1.770m away from the tree root with the base lifted to 1.184m high. The manipulator is supposed to have revolute joints for the first three joints, and the two links connecting neighboring joints should have 1.651 m in length equally in order to reach the peaches within the workspace.

Next is to analyze the picking performance of the designed arm. Figure 5-9 shows the outcome of this test case. The inverse kinematics function produces a figure of joint angle vs time. During 25 seconds, the arm picks the five peaches in sequence and each pick spends almost 5 seconds. For instance, in period 0 to 5, the arm first moves forward to reach the highest peach and then returns to its initial pose to drop the fruit for collection. We can see that joint 1 does not rotate much when picking the highest and the lowest peaches, while picking the left and the right peaches it rotates by 50 degree at most. Joint 2 and 3 angles have steps when the arm finishing picking the lowest peach. The steps are due to the joint angles range. From Figure 5-9, we can see that joint 2 rotates by larger angles for picking the highest and the lowest peaches than the others. The joint 3 rotates largely when picking the left and the right peach.

The inverse dynamics function produces a figure of joint torque vs time. From Figure 5-9, we can find that the joint 2 has the largest torque with its maximum of around 600 Nm. This means when the user wants to select motors for the arm, the motor torque for joint 2 should be no less than 600 Nm. The maximum torques of joint 1 and 3 are around 200 Nm.

Test Case Two: Citrus Tree

Figure 5-10 is the side view of a citrus tree taken in 2013 summer. The tree is about 3.5 m in height and 2 m in width with some citrus on it. A coordinate frame is attached to the citrus tree, and the coordinate of the five target citruses are measured as:

1. The highest peach: $x_1=0.187$ m, $y_1=0.475$ m, $z_1=2.642$ m
2. The lowest peach: $x_2=0.305$ m, $y_2=-0.268$ m, $z_2=0.832$ m
3. The left most peach: $x_3=0.267$ m, $y_3=-0.758$ m, $z_3=1.583$ m

4. The right most peach: x_4=0.393 m, y_4= 0.697 m, z_4=1.881 m
5. The front most peach: x_5=0.684 m, y_5=−0.099 m, z_5=2.077 m

The character parameters of the fruit picking arm computed in the output panel are:

- Arm length a=0.869 m
- Base height b=1.737 m
- Base distance d=1.298 m

Therefore, the fruit picking arm designed for this citrus tree should be located at 1.298 m away from the tree root with the base lifted to 1.737 m high. The lengths of two main links are both need to be 0.869 m.

Figure 5-11 shows the outcome of this test case. The produced kinematics image indicates that the rotations of the three joints are quite smooth if we ignore the step during time period from 7 to 10 s. The dynamics image in this citrus case is similar to the one of peach tree. The torque variation pattern is close to the peach case, but the absolute values of the joint torque are comparatively smaller. It is because the arms have shorter length in this case than in the first case. Joint 1 has still larger torques than that of joint 2 and 3, whose maximum value is around 200 Nm.

Result and Conclusion

For the two test cases of real fruit trees, the GUI works and the picking process can be analyzed. To make things more general, a sketch of a typical tree with four key fruits is shown in Figure 5-12. The tree is supposed to be in a standard sphere shape and the four fruits are supposed to be centrosymmetric and uniform distributed. Several fruit trees with different canopy diameters are tested by the GUI, and then the required arm lengths and joint torques for picking the sample fruits are obtained.

The results of these test cases can be concluded in Table 5-1. The joint torques are increasing when the arm length become longer. The torque values for joint 1 to pick the highest

and the lowest fruit are zeroes because the fruits are assumed to be in the vertical center line of the tree so that joint 1 has no rotations. The torque values for joint 2 are larger than those of joint 3 due to gravity forces. For each arm case, the maximum torque for joint 1 is from the picking process of the left or right most fruit. While the maximum torque for joint 3 is from the picking of the lowest fruit. The torques of joint 2 for picking the different fruits are close to each other, but when the arm length is larger than 0.7 m, it happens to the highest fruit.

The italic values in Table 5-1 are marked as the maximum torques for each joint in each arm length case. Figure 5-13 shows the variation of joint maximum torques with respect to the arm length. With the arm length increasing from 0.5 to 2 m, the joint torques raises smoothly in a curve.

Nowadays, in well-planned orchards or greenhouses, fruit plants are cultivated in lines for easy management and harvesting. With the growth of fruit trees, it is easy for them to form hedges, which make the robotic harvesting become more realizable. Figure 5-14 is an example of a hedge. The fruits in the hedge tend to grow into a vertical surface plane, which is especially convenient for robots to detect and harvest the fruit. Under this situation, the robot vehicle can just slowly move parallel alone the hedge, letting the harvesting arm pick the nearest fruit continuously. In extreme cases, the fruit picking arm just needs to pick the fruit with in a narrow vertical strip in the hedge. For hedge fruit harvesting, therefore, it is more important to study arm joints with horizontal axes than those with vertical axes. For the articulated arm presented in this research (Figure 5-2), we need to consider more on joints 2 and 3 than joint 1. For instance, a 3m high hedge has mature fruit with different heights from ground to top. An arm for this hedge can be created by the Fruit Picking Arm Designer GUI. The length of the arm is 1.44 m with its base 2.018 m away from the hedge and 1.5 m lifted from the ground. Table 5-2 and Figure 5-15

shows the torques of joints 2 and 3 for this arm to pick the fruit with different height. The maximum torque of joint 2 happens to pick the highest fruit, while the maximum torque of joint 3 happens to pick the lowest one. This result agrees with the study of typical fruit tree picking.

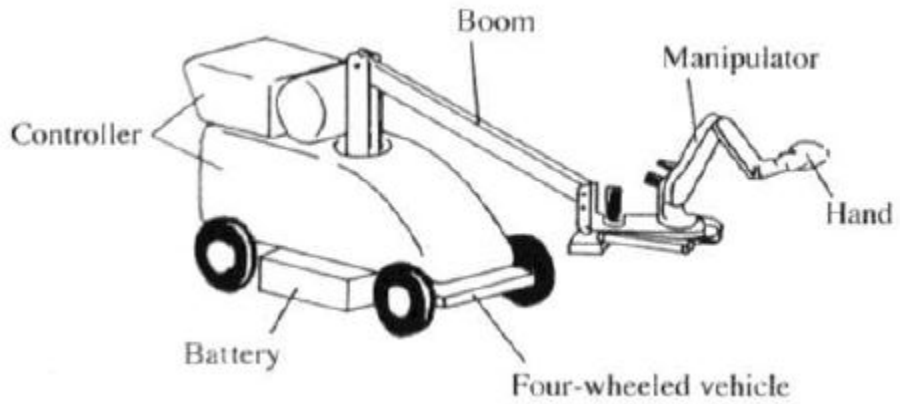

Figure 5-1. Fruit harvesting robot concept (Kubota, 1993).

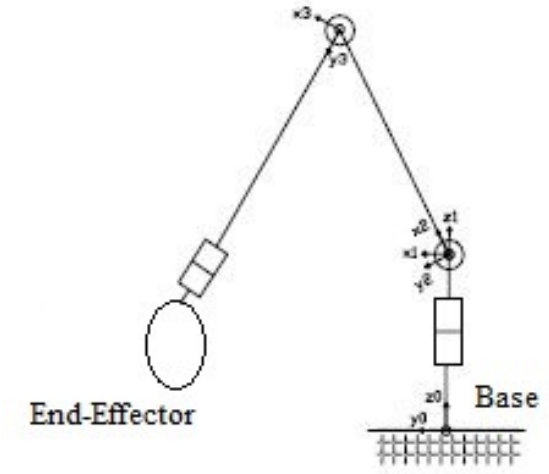

Figure 5-2. Articulated arm concept.

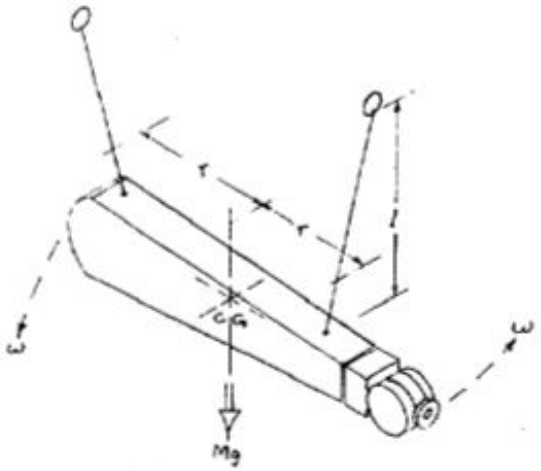

Figure 5-3. The suspension model for rotational inertia measurement (Armstrong, 1986).

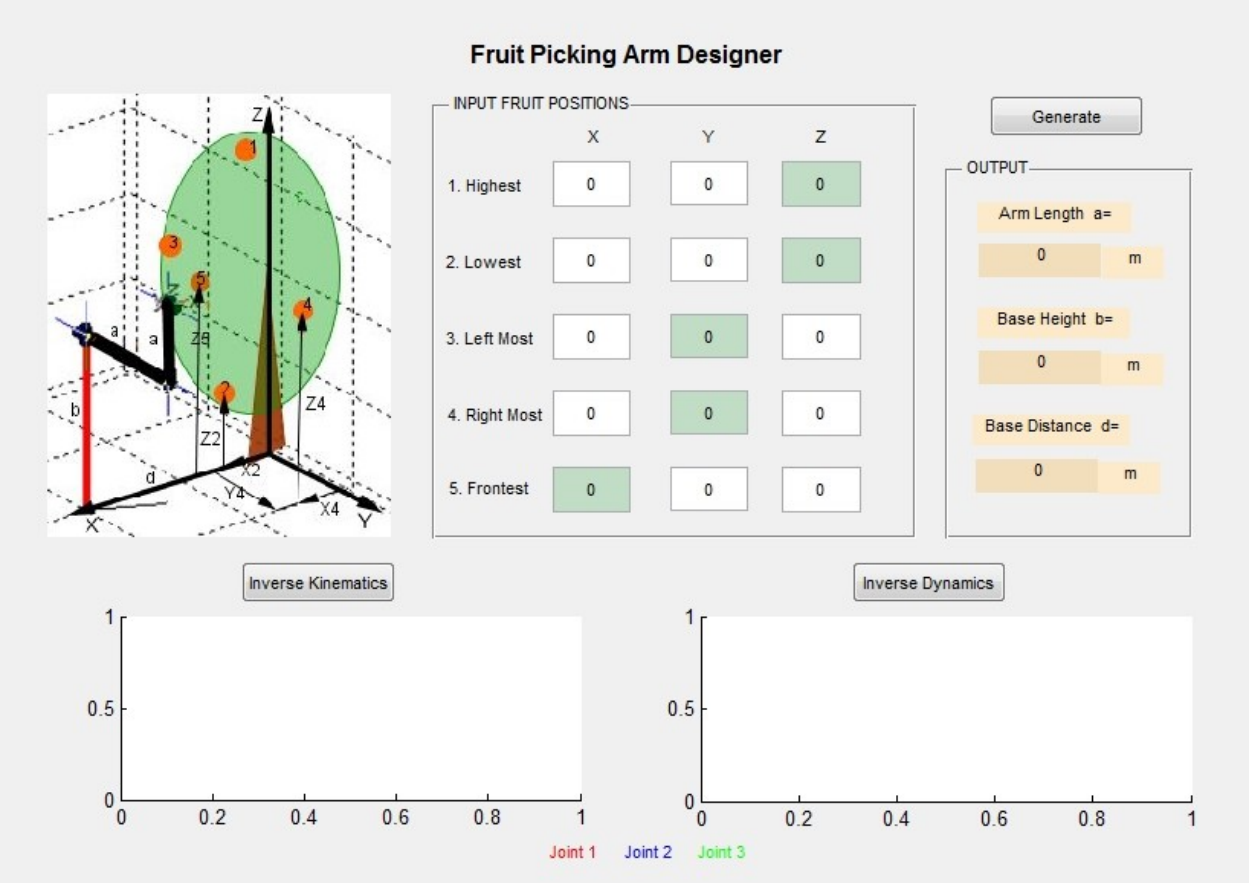

Figure 5-4. Fruit picking arm designer GUI.

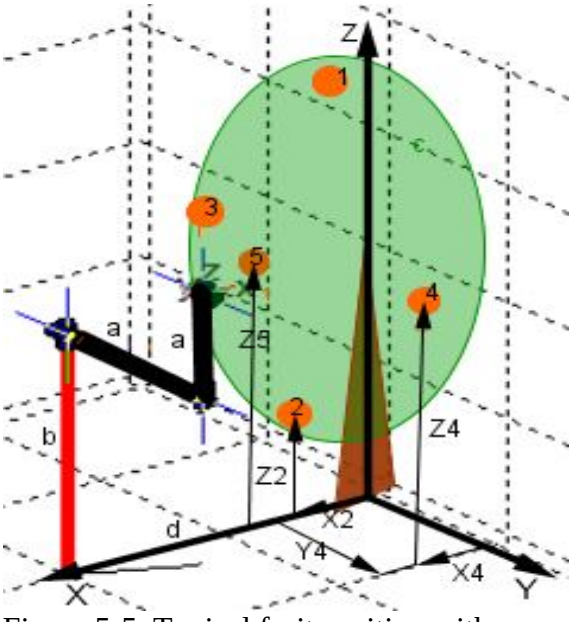

Figure 5-5. Typical fruit position with respect to the tree coordinate frame.

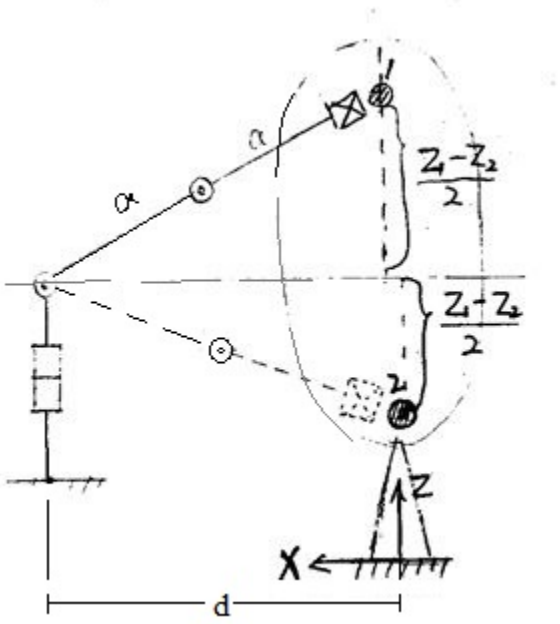

Figure 5-6. Fully extended arm pose.

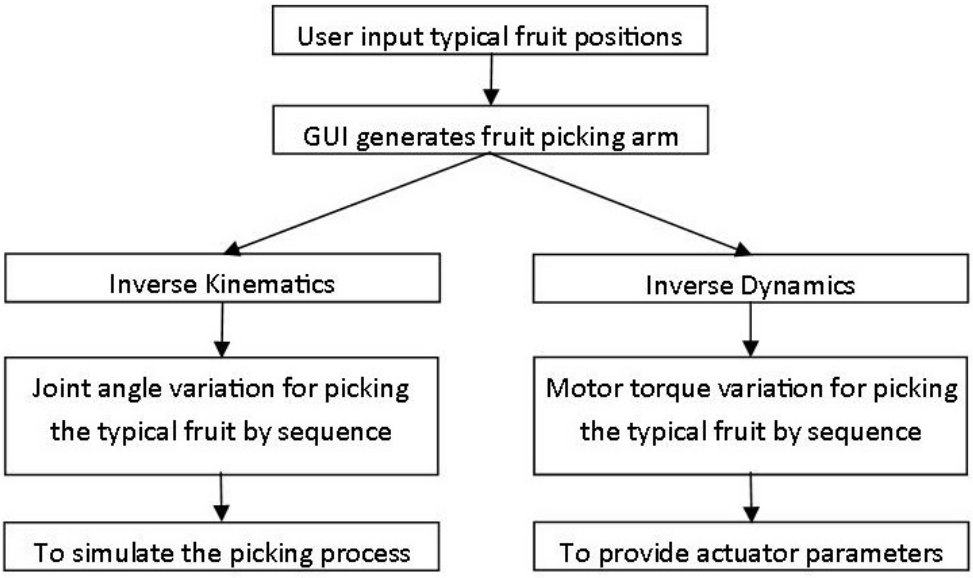

Figure 5-7. Flow chart of the Fruit Picking Arm Designer GUI.

Figure 5-8. A typical mature peach tree at University of Florida. Jiaying Zhang. May 6, 2013. Gainesville, FL.

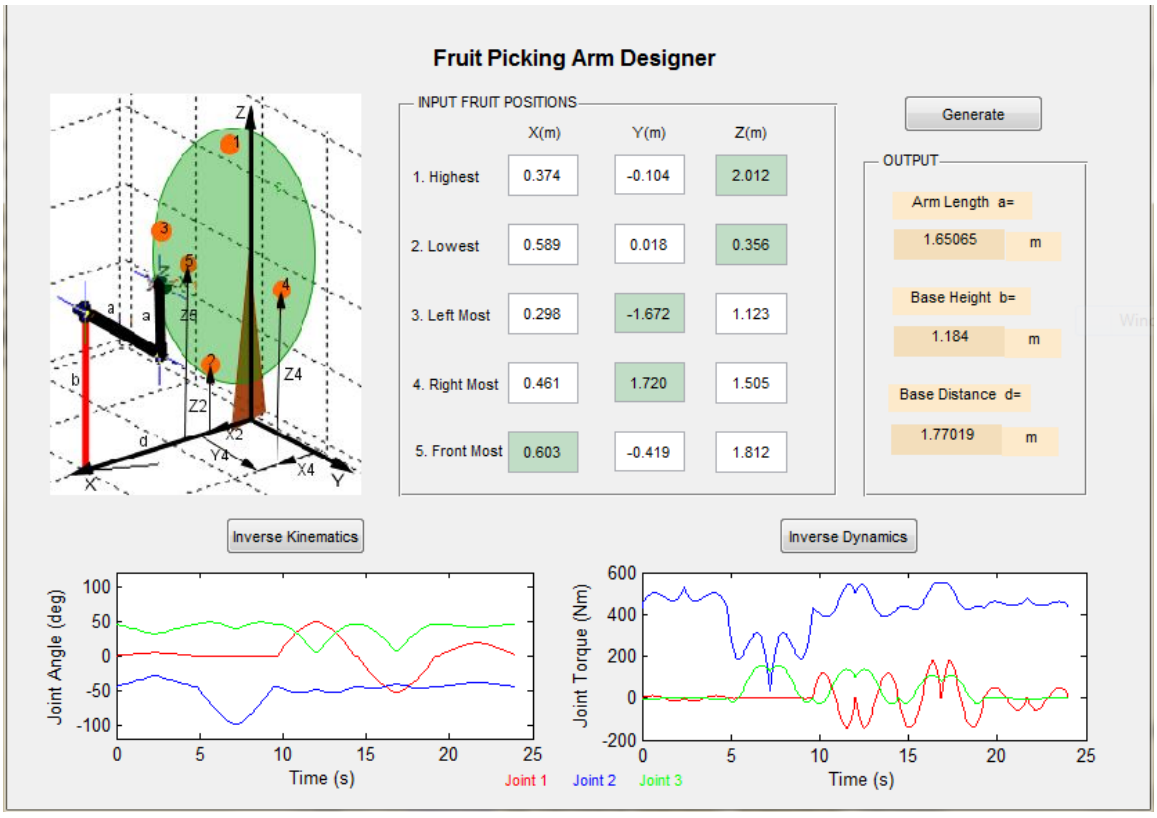

Figure 5-9. GUI test results of the peach tree.

Figure 5-10. A typical citrus tree at University of Florida. Jiaying Zhang. May 6, 2013. Gainesville, FL.

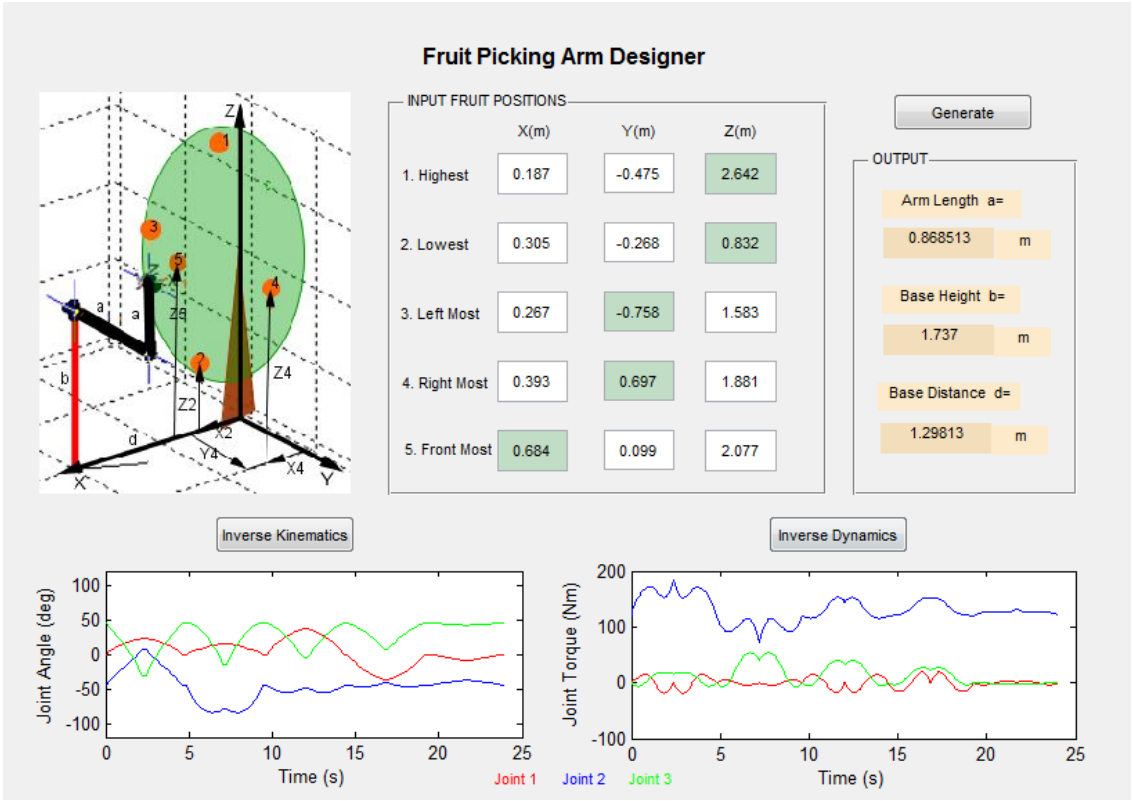

Figure 5-11. GUI test results of the citrus tree.

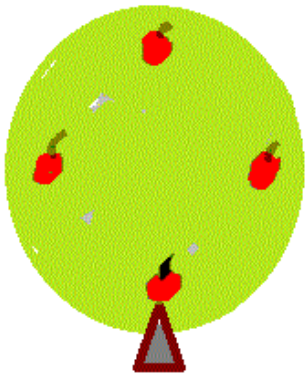

Figure 5-12. A typical fruit tree sketch.

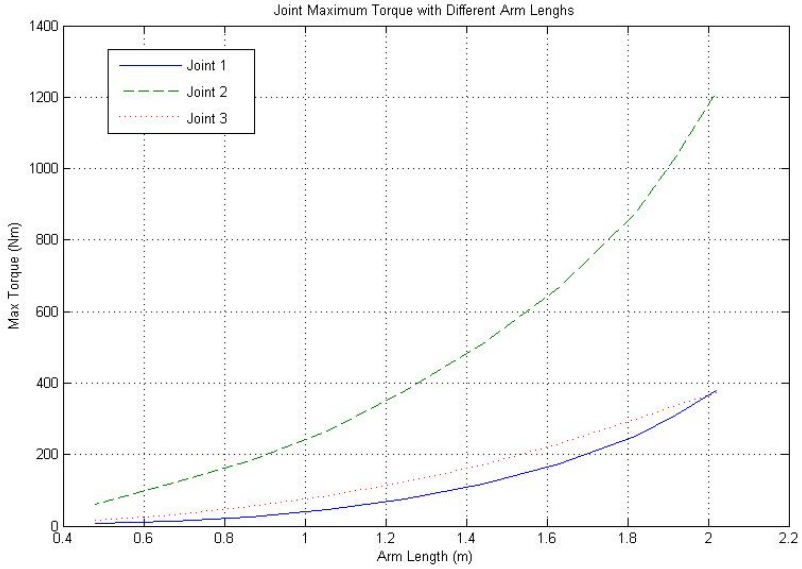

Figure 5-13. Joint maximum torques for different arm cases.

Figure 5-14. A hedge with fruits. Source: Reprinted with permission from Edible-landscape-design, http://www.edible-landscape-design.com/privacy-hedges.html (Feb 2014).

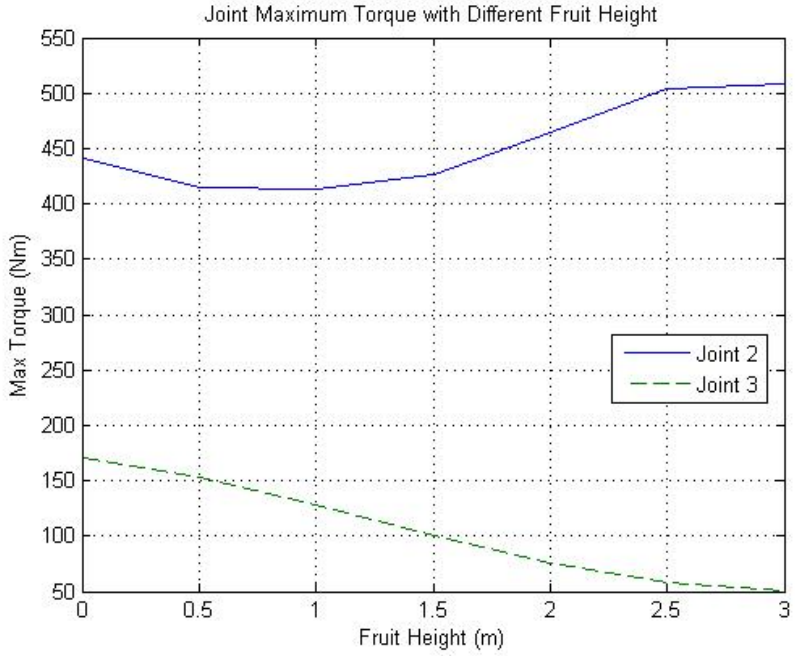

Figure 5-15. Joint torque variation for hedge harvesting.

Table 5-1. Test results for typical fruit trees.

Arm Length a (m)	Base Distance d (m)	Joint Torque (Nm)								
		Highest Fruit			Lowest Fruit			Side Most Fruit		
		Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
2.020	3.426	0	*1214*	188	0	1160	*374*	*379*	1132	302
1.920	3.157	0	*1032*	149	0	1002	*338*	*312*	1003	235
1.820	2.789	0	*876*	110	0	829	*299*	*252*	861	227
1.630	2.350	0	*667*	73	0	606	*229*	*175*	663	171
1.440	2.018	0	*509*	51	0	441	*171*	*118*	506	127
1.248	1.682	0	*379*	33	0	305	*123*	*76*	372	91
1.056	1.346	0	*267*	19	0	196	*84*	*45*	258	61
0.864	1.211	0	*183*	18	0	142	*54*	*26*	182	42
0.672	1.076	0	114	17	0	101	*32*	*14*	*121*	28
0.480	0.739	0	58	8	0	47	*16*	*6*	*60*	14

Table 5-2. Joint torques for hedge harvesting.

Fruit Height(m)	Joint 2 (Nm)	Joint 3(Nm)
3	509	51
2.5	504	58
2	465	76
1.5	427	101
1	414	128
0.5	415	153
0	441	171

CHAPTER 6
SUMMARY AND DISCUSSION

Summary

This thesis studies the manipulator of robotics fruit harvesting. After comparing some previous fruit picking manipulators, a Puma 560 robot is selected as a representative fruit picking arm for analysis. Puma 560 has 6 degrees of freedom with 6 revolute joints, and its arm structure covers the characters from many previous fruit picking robots. The two aspects to analyze a manipulator are kinematics and dynamics. Kinematics emphasizes on end position with regard to joint angles, while the dynamics is a study of motion with regard to forces or torques. It is convenient to these analyses by using the Matlab Robotics Toolbox.

In Chapter 3, both forward kinematics and inverse kinematics are established to analyze the performance of Puma 560 to pick fruits. The four parameters of Puma 560 are specified by applying D-H notation. Some programming is done by using the Toolbox, and the outcomes are clear to predict the performance. 3D figures are plotted to simulate the geometric trajectory of the end-effector from ready pose to reaching the fruits. For inverse kinematics, the joint angles are solved if given the target fruit position. While for forward kinematics, the reaching position is verify by the given joint angles.

In Chapter 4, forward dynamics and inverse dynamics are studied for the actuator torques of Puma560 during the fruit picking process. The Toolbox uses Newton-Euler Recursive technique to solve dynamics problems. The dynamic parameters of Puma 560 refer to previous studies. One outcome of the programming function is the variation of joint velocities and accelerations during a picking trip. A few figures are plotted to show the joint torques information under different situations, such as non-friction motor case or only considering gravity forces.

The outcomes of the kinematics and dynamics are helpful if using Puma 560 to pick fruits, but this robot will not be applicable to every fruit tree cases. For different fruit trees, fruit picking robot designer might need different arms that depend on the fruit distribution. If future researchers want to build fruit picking arms that have similar link structure to Puma robot, they will probably need the GUI created in Chapter 5.

The fruit picking arm designer GUI generates an arm that satisfies the user's requirements on the objective fruit tree. It allows the user to input five typical fruit positions with respect to the tree coordinate frame. These five positions are dominant in the arm's workspace, and are used to compute the arm length. The user will get three parameters of the generated arm, which identify the character of the arm they want. Kinematics and dynamics for the arm to pick the five given fruits are also included in the GUI. The user will have a clear view on the performance of the arm picking the required fruits. Two test cases are illustrated, and they have proved that the GUI is helpful for tree fruit arm design.

Discussion

The robotic fruit harvestings are complex machines with mechanical and control systems. The manipulator is only a small part of the machine, so more studies regarding the cooperation between fruit picking arms, end-effectors, detecting sensors and the control systems will need future efforts. The main problem on selective robotics fruit harvesting is how to improve the picking effectiveness.

The research in this thesis is based on many assumptions that need to be proved. The Puma 560 is assumed to be able to a fruit harvesting, and reliable fruit detecting sensors can provide true fruit position information. The end-effectors are assumed to grasp and detach the fruit successfully so that the arm won't waste time on duplicated motion trips. The kinematic and

dynamic analysis for every picking process is from a specified reach pose, but in some cases, returning to initial pose is not necessary. Therefore, more detailed different fruit picking cases will need to be studied in future.

Comparing to kinematics, dynamics is much more complex and needs more data. The dynamics study in this research is based on previous puma robot studies by other researchers. The reliability of the dynamics parameters such as link mass and center of gravity are supposed to be correct. Moreover, in the fruit picking arm designer GUI, an arm is generated to pick the given fruit. Dynamic analysis is available for the new arm, but the dynamics parameters are assuming to be proportional to puma 560. It is difficult to measure the moments of inertia for every new arm due to limit experiment conditions. If the assumption cannot be proved to be valid, more experiment to measure the moments of inertial is necessary for future work. Motor internal factors are also important in dynamics computation. As the researchers often have different motor choices when they are designing a new manipulator, the motor internal frictions are not considered in this research.

Even if a sample robotic arm is computed through the GUI, more detailed design works needs to be operated. The arm generated by GUI gives the designer a general concept and very basic level information of a fruit picking arm, so there still exist much room to improve the arm. Finally, Matlab Robotics Toolbox is one of the many tools for analyzing robot manipulators. If some work from other analysis tools can be used to make a comparison, it will be a good way to check the results in this research.

LIST OF REFERENCES

Armstrong, B., Khatib, O., & Burdick, J. (1986). The explicit dynamic model and inertial parameters of the Puma 560 arm. *Proc. IEEE Int. Conf. Robotics and Automation*, Washington, USA (1), 510–18.

Benitez, A, Huitzil1, I., Casiano, A., Calleja, J. D. L. &Medina, M. A. (2012). PUMA 560: Robot Prototype with Graphic Simulation Environment. *Advances in Mechanical Engineering*. 2, 1, 15-22.

Burks, T., Villegas, F., Hannan, M., Flood, S., Sivaraman, B., Subramanian, V. & Sikes, J. (2005). *HorTechnology*. 15 (1), 79-87.

Crane, C. D. III and Duffy J. (1998). *Kinematic analysis of robot manipulators.* ISBN 978-0-521-57063-3, Cambridge University Press

Corke, P. I. (1996). A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine* 3: 24-32.

Corke, P. I. & Armstrong, B (1994). A search for consensus among model parameters reported for the PUMA 560 robot. *Proc. IEEE Int. Conf. Robotics and Automation.*

Craig, J. J. (1989). *Introduction to Robotics: Mechanics and Control.* 2nd ed. Reading, MA.: Addison-Wesley.

Edible-landscape-design. Retrieved Feb 2014. from http://www.edible-landscape-design.com/privacy-hedges.html

Flood, S. J. (2006). *Design of a Robotic Citrus Harvesting End Effector and Force Control Model Using Physical Properties and Harvesting Motion Tests.* Dissertation for the degree of doctor of philosophy at University of Florida.

Grand D'Esnon, A. (1985). Robotic harvesting of apples. *Proc. Agri- Mation*, 1, 210-214.

Grand D'Esnon, A., Rabatel, G., Pellenc,R., Journeau, A., and Aldon, M. J. (1987). MAGALI: A self-propelled robot to pick apples. *ASAE Paper* No. 87-1037. St. Joseph, MI.

Hannan, M. W., Burks, T. F. & Bulanon, D. M. (2009). A Machine Vision Algorithm for Orange Fruit Detection. *Agricultural Engineering International: the CIGR Ejournal.* Manuscript 1281. Vol. XI.

Harrell, R. (1987). Economic Analysis of Robtic Citrus Harvesting in Florida. *American Society of Agricultural Engineers*. 30(2), 298-304.

Hartenberg, R. S. and Denavit, J. (1955). A kinematic notation for lower pair mechanisms based on matrices. *Journal of Applied Mechanics*, 77, 215–221.

Hollerbach, J. (1980). A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. *IEEE Trans. Syst. Man Cybern.*, vol. SMC-10, 730–736.

Hollerbach, J. (1982) Dynamics. *Robot Motion - Planning and Control* (M. Brady, J. M. Hollerbach, T. L. Johnson, T. Lozano-Perez, and M. T. Mason, eds.), 51–71

Kurfess, T. R. (2005). *Robotics and Automation Handbook.* CRC Press LLC. ISBN 0-8493-1804-1 (alk. paper)

Kane, T. and Levinson, D. (1983). The use of Kane's dynamical equations in robotics. *Int. J.Robot. Res.* 2: 3–21.

Kondo, N., Monta, M. and Fujiura. T. (1996). Basic constitution of a robot for agricultural use. *Advanced Robotics.* 10(4): 339-353.

Kondo, N., Monta, M. & Noguchi, N. (2006). *Agricultural Robots Mechanisms and Practice.* Kyoto University Press.

Lee, C. S. G., Lee, B., and Nigham, R. (1983) Development of the generalized D'Alembert equations of motion for mechanical manipulators. *Proc. 22nd CDC*, (San Antonio, Texas), 1205–1210.

Luh, J. Y. S., Walker, M. W. & Paul, R. P. C. (1980). On-line computational scheme for mechanical manipulators. *ASME Journal of Dynamic Systems, Measurement and Control*, 102: 69–76.

Nayar, H. D. (2002). Robotect: serial-link manipulator design software for modeling, visualization and performance analysis. Proc. 7th Intl. Conf. Control, Automation, Robotics and Vision, 3: 1359-1364. Piscataway, NJ: IEEE.

Oxbo Citrus Harvesters. Retrieved Dec 2013 from http://www.oxbocorp.com/Products/Citrus.aspx

Paul, R. P. (1981). *Robot Manipulators: Mathematics, Programming, and Control*. Cambridge, Massachusetts: MIT Press, 1981.

Paul, R and Zhang, H. (1986). Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation. *The International Journal of Robotics Reaserch.* 5, 2, 32-44.

Sarig, Y. (1993). Robotics of Fruit Harvesting: A State-of-the-art Review. *J. Agric. Eng.Research* 54: 265-280.

Sivaraman, B. (2006). *Design and Development of a Robot Manipulator for Citrus Harvesting*. Dissertation for the degree of doctor of philosophy at University of Florida.

Unimate Puma Series 500 Industrial Robot (1084), Unimation Incorporated.

BIOGRAPHICAL SKETCH

Jiaying Zhang was born in 1987, and was born and grew up in Shanghai, China. She was enrolled into Shanghai Jiaotong Universty (SJTU) in September 2006, and received a Bachelor of Science in Engineering degree with an industrial engineering major in Jul 2010. She took a job immediately after graduation from SJTU, and worked as a mechanical engineer in Siemens Ltd., China until 2012. She came to the United States of America for graduate education in 2012 fall. She was admitted by the Department of Mechanical and Aerospace Engineering and anticipates obtaining a Master of Science degree in May 2014.