KINEMATIC REDUNDANCY RESOLUTION FOR THE BAXTER RESEARCH ROBOT

by

Khoa Dang Le, B.S.

THESIS

Presented to the Faculty of

The University of Houston-Clear Lake

In Partial Fulfillment

Of the Requirements

For the Degree

MASTER OF SCIENCE

in Computer Engineering

THE UNIVERSITY OF HOUSTON-CLEAR LAKE

MAY, 2020

KINEMATIC REDUNDANCY RESOLUTION

FOR THE BAXTER RESEARCH ROBOT

by

Khoa Dang Le

APPROVED BY

_____

Luong A. Nguyen, Ph.D., Chair

_____

Hakduran Koc, Ph.D., Committee Member

_____

Thomas L. Harman, Ph.D., Committee Member

RECEIVED/APPROVED BY THE COLLEGE OF SCIENCE AND ENGINEERING

_____

David Garrison, Ph.D., Interim Associate Dean

_____

Miguel Gonzalez, Ph.D., Dean

Acknowledgement

I would first like to express my deepest appreciation to my committee chair and Thesis advisor, Dr. Luong A. Nguyen, who has guided me throughout this work. Without his wonderful mentorship and supervision, this Thesis would never have taken shape.

I would also like to extend my sincere gratitude to Dr. Thomas L. Harman, my committee member, for his time, constructive comments, and suggestion.

I am also very grateful to my committee member, Dr. Hakduran Koc, who is also my academic advisor, for his valuable advice and support.

Special thanks to Dr. Liwen Shih for letting me use a San Diego Supercomputer Center (SDSC) allocation to perform some computations which were needed for the Thesis.

Finally, I extend my deep gratitude to my parents for their unconditional support. I owe a great deal to my wife for her sacrifice and encouragement. I also wish to thank my son, who is still too young to understand, for his very existence in this world.

ABSTRACT

KINEMATIC REDUNDANCY RESOLUTION FOR THE BAXTER RESEARCH ROBOT

Khoa Dang Le
University of Houston-Clear Lake, 2020

Thesis Chair: Luong A. Nguyen, Ph.D.

Robotics is a brand of science and engineering which involves many other fields such as mechanical engineering, electrical engineering, and bioengineering. Robotic engineers and scientists develop machines that can substitute for humans in various situations. Nowadays, robots are used in hazardous and dangerous environments, factories or where humans cannot survive (outer space, deep sea, etc.). A lot of tasks can be done from something simple such as object lifting and carrying to something complicated such as bomb deactivation, sample collecting, etc. In order to do such tasks, kinematic redundancy plays an important part, especially in the class of humanoid robots.

A robot manipulator arm is said to be kinematically redundant when it has more degrees of freedom than it is required for a specific task. The redundancy can be used to achieve additional goals which is importance in robot design and planning. In this thesis, kinematic redundancy resolution will be investigated and applied to Baxter robot. This

will include singularity avoidance, collision avoidance, as well as manipulability optimization. As a convenient tool for this research, a method of Baxter's end-effectors manipulation, namely Cartesian velocity control, will also be developed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I:

BAXTER ROBOT

1. Overview

1.1 Hardware

Baxter is an industrial robot built by Rethink Robotics – a company which was founded in 2008 as a startup aiming to create low-cost robots. Baxter was introduced in September 2011 and was intended to be sold to small to medium-sized companies to do dull tasks on the production line. Baxter's key innovations have been recognized with a few awards such as: Time magazine's "Best 25 Inventions of 2012", 2013 "Edison Awards" Gold Winner in Applied Technology Innovation, MIT technology review's "10 Breakthrough Technologies of 2013"…With a base price of just $22,000 Baxter robot packs some nice features which enable it to be a low-cost friendly collaborative pioneer robot.

Baxter is a two-armed humanoid robot with an animated face. It is 3'1" tall without its pedestal and weighs 165 lbs. With an optional adjustable pedestal, it can reach the height of 5'10" – 6'3" which is easy for human to work with. Baxter has a total of 14 degrees of freedom with 7 per arm. Each arm can run a task independently or both arms can do a same task for double capacity.

Figure 1.1: Baxter robot

An interesting feature of Baxter robot is the animated face which shows different emotions representing different states of the robot. This function simplifies the interaction between the robot and the human coworkers/ operators. A sleeping face with its eyes closed showing that Baxter is on standby, concentrating face when it is learning a task, focused face when it is working without a problem, surprised face when it detects a human approaching,… and even a sad face when it gives up a task.

Each arm can lift 5 lbs. (higher payload is possible in limited workspace) with a maximum reach of 1210mm and can perform tasks with an accuracy of about 0.5 to 1 centimeter. The maximum speed of an arm with no payload is 3.3 ft/s and 2 ft/s with rated payload (5 lbs.).

The arms are powered by series elastic actuators. Elastic actuators give up some precision in exchange for safety when the robot is working collaboratively with human. Most industrial robots have rigid actuators which deliver great precision and force. Those robots use sensors to measure the slightest impact and stop the motors. With such

strength and speed, rigid-actuator robots can deliver a high force from a small displacement and a little position error can result in accidents, hence they can be dangerous and unfriendly around humans. Series elastic actuators sacrifice the payload and stiffness for force control. They contain an elastic element that absorb shocks if they hit someone or something, giving the robot additional time to apply the brakes. The actuators doubled as force sensors. By measuring the deformation of the elastic part, the robot can measure the force at the end of the arm. The robot can sense if it hits someone and stop the motion. Force sensing can also be used with cheaper motors (comparing to rigid actuators' motors) to reduce the cost of the robot.

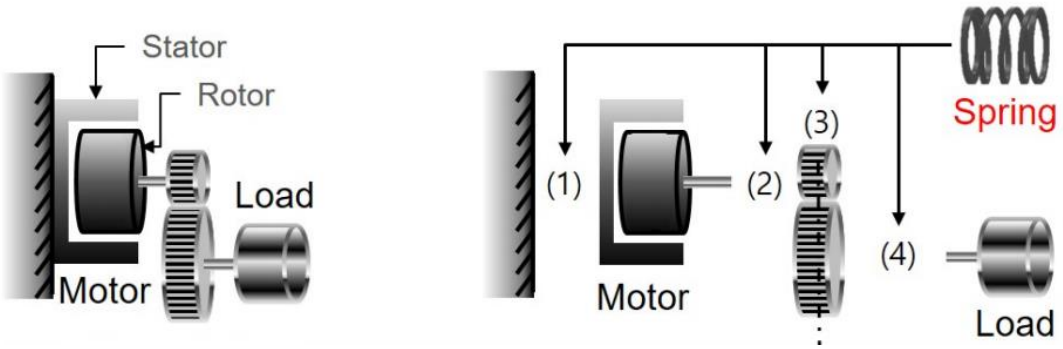

Figure 1.2: Rigid actuator (left) and Elastic actuator (right)

Baxter robot's software is run on a personal computer in its chest with core i7-3770 processor, HD4000 graphics, 4GB DDR3 RAM and 128GB solid state drive.

Baxter has 3 cameras, 1 on its head and 2 at the end-effector on each arm along with 360^{o} sonar sensor, back-drivable motors. Protective covers make it a safe and friendly robot to work among human. To train a Baxter robot, a person physically moves its arms and uses buttons on the arm to make selections. For instance, if you want to program it to pick up and place an object, you would first grab its wrist to engage a training and move its end-effector over the item. The camera on the end-effector will display the item on the screen using computer vision algorithms. You can confirm that it's the right item. After Baxter picks it up, you would position its arm over the

destination and click to confirm. The robot will place the item and the training is finished. Baxter then will continuously pick up the items and place them at the destination (if there are items presented).

Baxter may not be super-fast, super strong and super precise like other industrial robots, but it is smart, safe and easy to work with. The seven-degree-of-freedom arms with force sensing and series elastic actuators help the robot to work effectively and safely. A person does not need to have programing skill to train the robot to do simple tasks. The base price of $22,000 is considerably lower than other industrial robots.

1.2 Baxter's joints

Baxter robot is a humanoid robot equipped with two seven degree-of-freedom arms. Each arm has force, position and torque sensing and control at every joint. There are 3 bend joints and 3 twist joints on each arm.

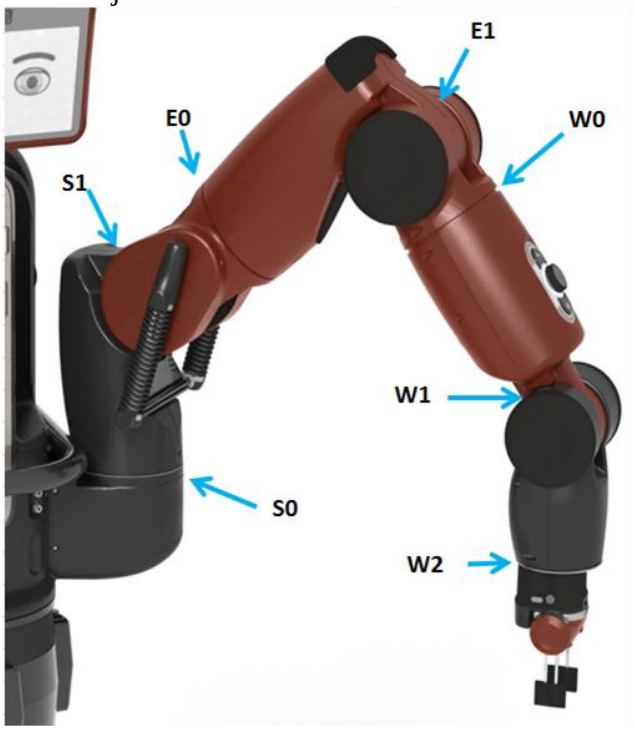

Figure 1.3: Baxter joints

Joint	Motion
S_0	Shoulder roll
S_1	Shoulder pitch
E_0	Elbow roll
E_1	Elbow pitch
W_0	Wrist roll
W_1	Wrist pitch
W_2	Wrist roll

Table 1.1: Joint names

Baxter's link lengths are measured in mm from the center of a joint to its adjacent one.

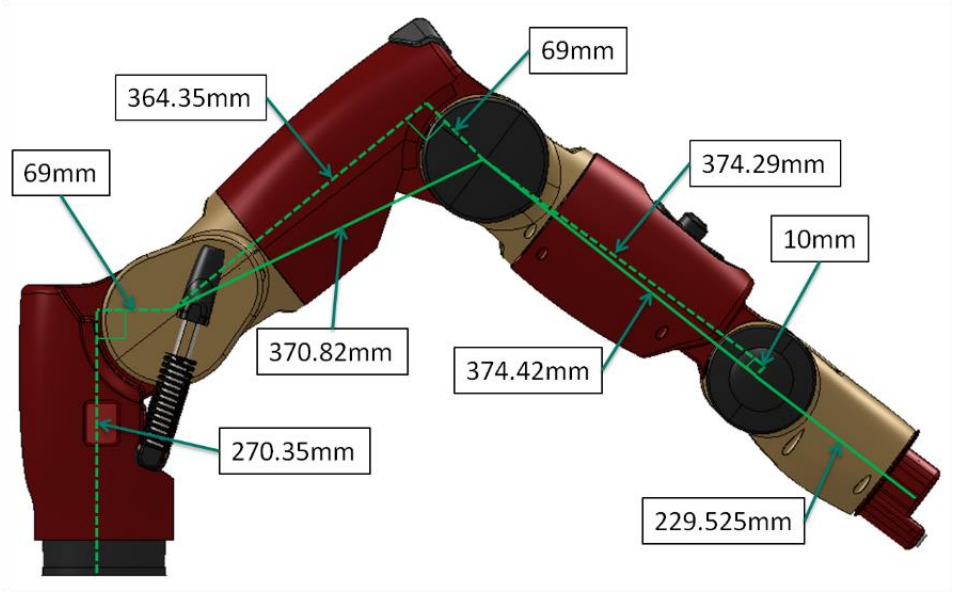

Figure 1.4: Link lengths

Link	Length
L_0	270.35 mm
L_1	69 mm
L_2	364.35 mm
L_3	69 mm
L_4	374.29 mm
L_5	10 mm
L_6	229.525 mm
L_7	10 mm

Table 1.2: Link lengths

The most common limitation for any robot is probably the joint limits, or range of motion, which each joint can archive. Baxter robots have finite range of motion for each joint that they can move.

Joint name	Joint variable	Lower limit (degree)	Upper limit (degree)	Range (degree)	Min (radian)	Max (radian)
S_0	θ_1	-97.494	+97.494	194.998	-1.7016	+1.7016
S_1	θ_2	-123	+60	183	-2.147	+1.047
E_0	θ_3	-174.987	+174.987	349.979	-3.0541	+3.0541
E_1	θ_4	-2.864	+150	153	-0.05	+2.618
W_0	θ_5	-175.25	+175.25	350.5	-3.059	+3.059
W_1	θ_6	-90	+120	210	-1.5707	+2.094
W_2	θ_7	-175.25	+175.25	350.5	-3.059	+3.059

Table 1.3: Joint limits

Other joint specifications of Baxter robot are listed below:

Joint	Maximum Speed (rad/s)
S0	2.0
S1	2.0
E0	2.0
E1	2.0
W0	4.0
W1	4.0
W2	4.0

Table 1.4: Maximum Joint Speeds

Joint	Stiffness
Small Flexures (W0, W1, W2)	3.4deg @ 15Nm (~250Nm/rad)
Large Flexures (S0, S1, E0, E1)	3.4deg @ 50Nm (~843Nm/rad)

Table 1.5: Joint Flexure Stiffness

Description	Spec
Spring Type	Japanese Industrial Standard (JIS) die spring: Extra light duty 35 X 200
Free Length	200 mm
Stiffness (K)	9.6 N/mm
Operating length	101 mm - 154 mm

Table 1.6: S1 Spring Specifications

Joint	Peak Torque
S0, S1, E0, E1	50Nm
W0, W1, W2	15Nm

Table 1.7: Joint Peak Torque

The resolution for the joint sensors is 14 bits (over 360 degrees); 360/ (2^14) = 0.021972656 degrees per tic. All joints have a sinusoidal non-linearity, giving a typical accuracy on the order of +/-0.10 degrees, worst case +/-0.25 degrees accuracy when approaching joint limits. In addition, there may be an absolute zero-offset of up to +/-0.10 degree when the arm is not calibrated properly.

1.3 Arm control modes

Baxter robot has 3 modes to control the arms' movements: Position, Velocity and Torque. Each arm can be controlled independently with different modes. When a joint position command is published from the development PC, the 'realtime_loop' process which represents a motor control plugin subscribes to this message. This message is then parsed and represented in memory based on the control mode. Depending on the control mode, modifications are made to the input commands. These modifications are typically due to the safety controllers (e.g. arm-to-arm collision avoidance, collision detection, etc.) A control rate timeout is also enforced at this motor controller layer. This states that if a new 'JointCommand' message is not received within the specified timeout (0.2 seconds, or 5Hz), the robot will 'Timeout'. When the robot 'Times out', the current control mode is exited, reverting to position control mode where the robot will command (hold) it's current joint angles. The reason for this is safety. For example, if controlling in velocity control mode where you are commanding 1.0 rad/s to joint S0, and you lose

network connectivity, the robot could result in dangerous motions. By 'timing out', the robot will be safer, reacting to network timeouts, or incorrect control behavior.

The Position mode is the most fundamental control mode of Baxter robot. Target joint angles are published by the user and the internal controller drives the joints to the desired angles. The motor controller ensures the safety while processing the joint command by collision avoidance and detection. The “raw” option can be enabled which will bypass the collision avoidance and allow for very fast motions at the joint limits.

In Velocity mode, the velocity for all 7 joints are specified which we want the joint simultaneously to achieve. The Baxter’s Motor Controller in this mode applies collision avoidance and detection and ensures the expected behavior. If a commanded velocity to any of the joints will result in a joint position which exceed the joint limits, the velocity command is considered invalid and no joints will be commanded.

In Torque mode, joint torques are published by the user and the joints will move at specified torques. Torque mode has the access to the lowest control level, therefore it should be used with precaution since safety feature is disabled in this mode.

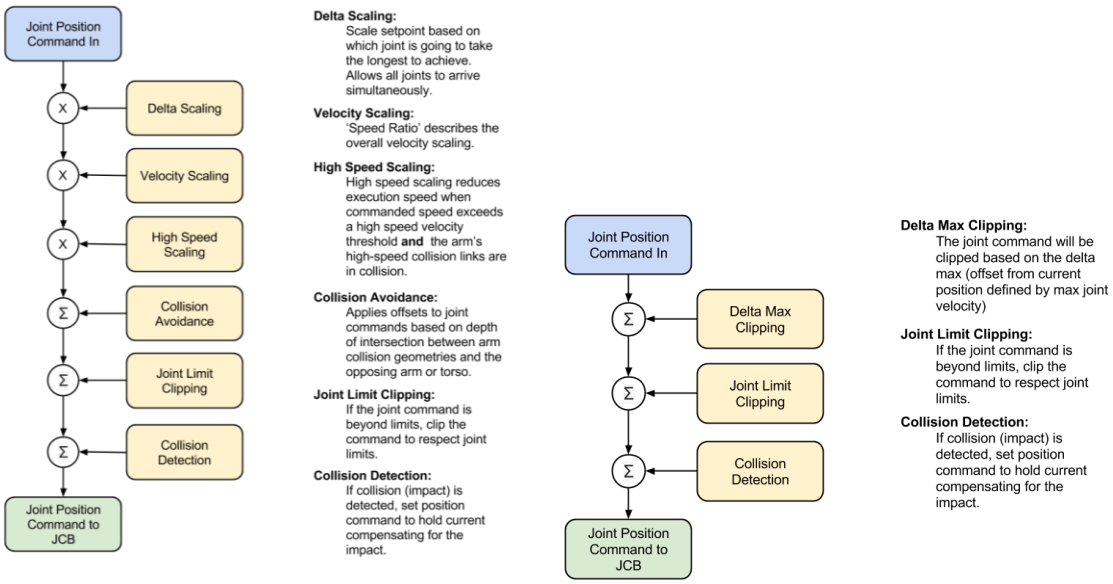

Figure 1.5: Joint position mode (left) with its “raw” function enabled (right)

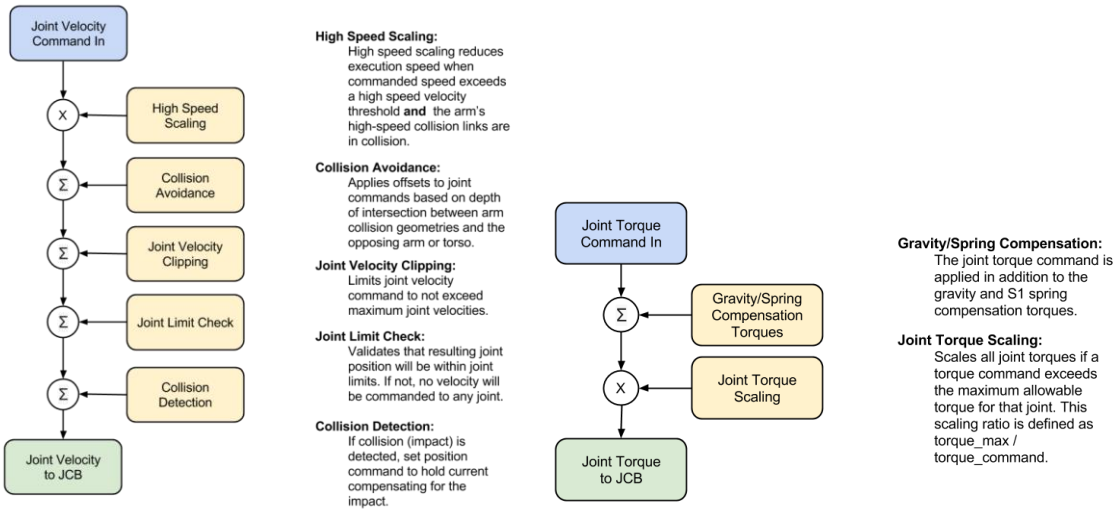

Figure 1.6: Joint velocity mode (left) and Joint Torque mode (right)

1.4 Baxter's degrees of freedom

The Baxter arm is a 7-dof robot arm. It is classified as kinematically redundant since it possesses more degrees of freedom than stricly needed for the end effector six degrees of freedom motion trajectories

All of Baxter's joints are revolute joints. A revolute joint is a one-degree-of-freedom joint which describe rotational movements between two objects. It can be used either as a passive or active joint.

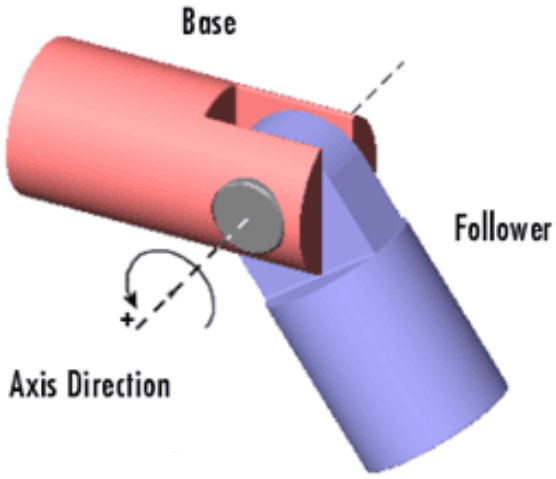

Figure 1.7: Revolute joint

Baxter's arm design consists of a 2-dof shoulder, a 2-dof elbow and a 3-dof wrist. There are no parallel revolute joint axes anywhere on each arm, nor three consecutive revolute joints that share a common origin. The 2-dof head provides the ability to tilt and pan for the camera.

The 2-dof shoulder and 2-dof elbow are offset Universal joints (offset-U-joint). A universal joint is a mechanical device that allows one or more rotating shafts to be linked together, allowing the transmission of torque and/or rotary motion. It also allows for transmission of power between two points that are not in line with each other. In robots a universal joint consists of two revolute joints which circulate around a point. A pair of hinges connected by a cross shaft located close and oriented at 90^{o} to each other allows U joint to have 2 degrees of freedom. In Baxter robot, the U joints have offsets, but they still have 2 degrees of freedom and behave as a regular U joint.

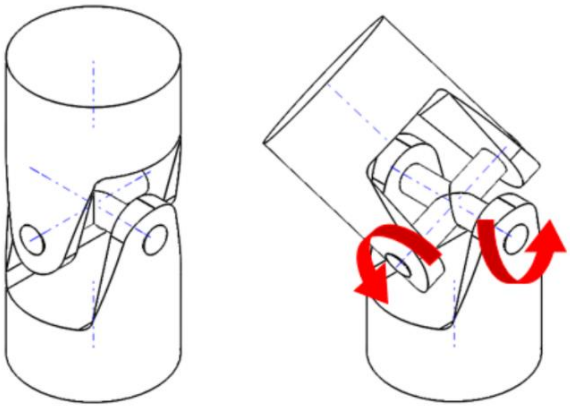

Figure 1.8: Universal joint

The 3-dof wrist is an offset Spherical joint (offset-S-joint). Spherical joint is a 3-dof joint which is used to describe rotation movements between objects. It represents three rotational degrees of freedom at a single pivot point. Its configuration is defined by 3 values, each of them shows the amount of rotation around and axis (x, y or z) of their reference frame. Spherical joints are passive joints. A spherical joint can be a ball joint and it can also be a series of 3 revolute joints connected. The wrist of Baxter is an offset S joint since there is no straight line which can connect 3 axles of the 3 joints.

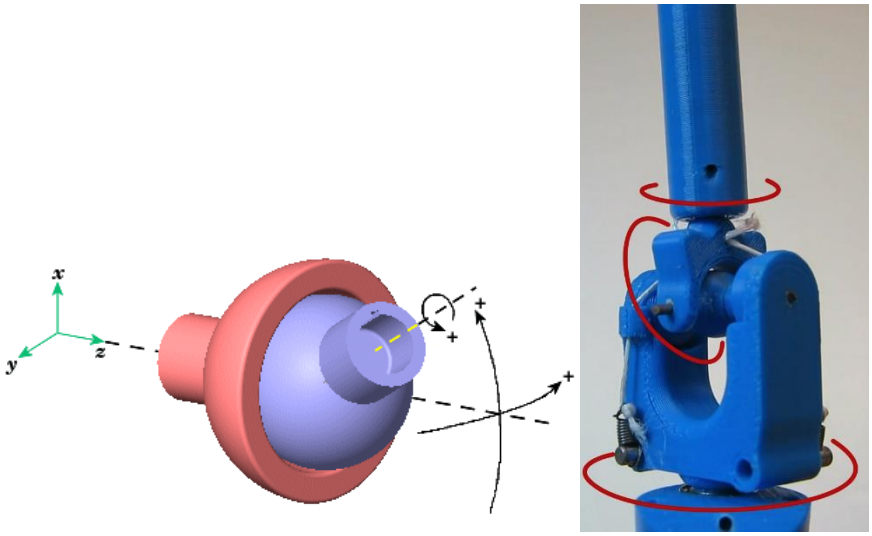

Figure 1.9: Two types of a spherical joint

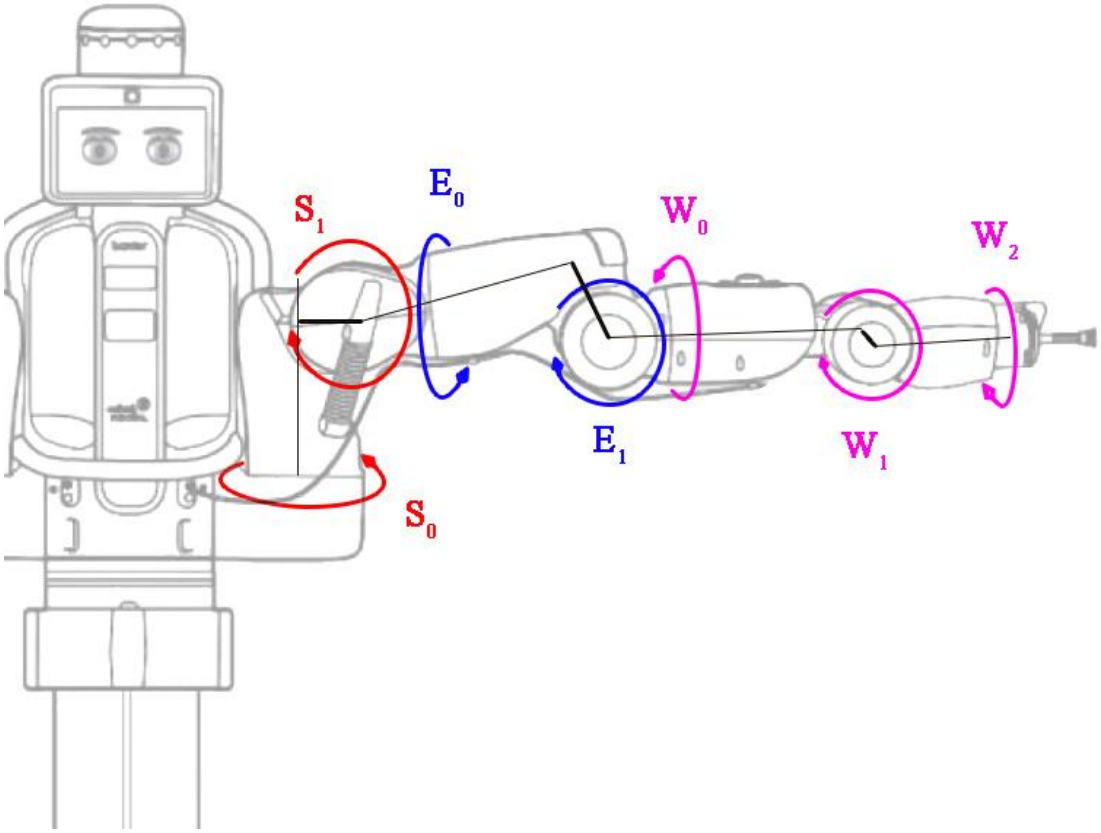

Figure 1.10: Baxter offset U joints and offset S joint

1.5 Baxter Research Robot

In 2013, Rethink Robotics released the Research version of Baxter robot. This version has the same hardware as the Industrial version, but the software is different. Instead of the pre-programmed manufacturing software for industrial work, it came with a software development kit (SDK) installed, based on the Robot Operating System (ROS), which is open source. Some libraries of low-level tasks such as joint control and position were also made open source. Developers can communicate with the Baxter's workstation via Ethernet and program the robot to do various tasks.

Despite being a pioneer in collaborative robots (or co-bots), RethinkRobotics eventually closed its doors in October 2018 and sold their technology, patents, and intellectual property. Rethink's designs failed to meet the needs of its target market. Elastic actuators, while being safe and flexible, have lower accuracy, smaller force bandwidth and lower speed in comparison to other industrial robots with rigid actuators. Baxter and Sawyer (Baxter's one-arm brother) are limited in applications and tasks they can perform in industrial environments. Companies typically want simple, fast, repeatable robot with great precision on the production lines. They do not change or rearrange their production line often, so they care little about the fast training speed and the ease of robot retraining which Rethinks offers. The Baxter's two arm design has little use to the customers. Rethink's robots' end-effectors were also hard to modify. Rethink Robotics is now a part of the HAHN group, a German automation specialist.

While Baxter is quite far from being the best industrial robot, it is still a great research robot. Developers can safely work around it thanks to multiple safety features. Series Elastic Actuators are precise and fast enough for research purposes. The open-source SDK and APIs allow students to study robotic theories and practice robot control. Many projects can be developed with Baxter robot. At UHCL, a research version of Baxter robot is located at the Robotics Lab where it connects to a computer running the Ubuntu operating system.

2. Baxter Kinematics

This section will be devoted to the kinematics of robot manipulators, which is the geometric study of the movement of multi-degree-of-freedom manipulators. Multiple studies have been conducted on robot kinematics [1] [2] [3] [4] [5]. The kinematics of Baxter robot show the relationship of velocity and position between links.

2.1 Coordinate Transformation

One of the basic ideas of robot control is to know where it is, i.e. its position and orientation. In order to mathematically represent an object in 3-dimension space, a 3D coordinate frame, called *object frame* can be attached to it with the position expressed by the frame origin and the orientation by the direction of three axes relative to a given coordinate frame, called *reference frame*.

Consider a reference frame A denoted as $\sum_A$ with origin O_A and 3 axes X_A, Y_A, Z_A. Similarity, there is an object with frame B attached to it and denoted as $\sum_B$ with origin O_B and 3 axes X_B, Y_B, Z_B. ${}^A\boldsymbol{p}_B$ represent a vector from O_A to O_B which is a position vector of O_B relative to O_A, expressed in $\sum_A$. The unit vectors of X_B, Y_B, Z_B expressed in $\sum_A$ is ${}^A\boldsymbol{x}_B, {}^A\boldsymbol{y}_B, {}^A\boldsymbol{z}_B$. The object's orientation is then denoted by $\{{}^A\boldsymbol{x}_B, {}^A\boldsymbol{y}_B, {}^A\boldsymbol{z}_B\}$, expressed in $\sum_A$.

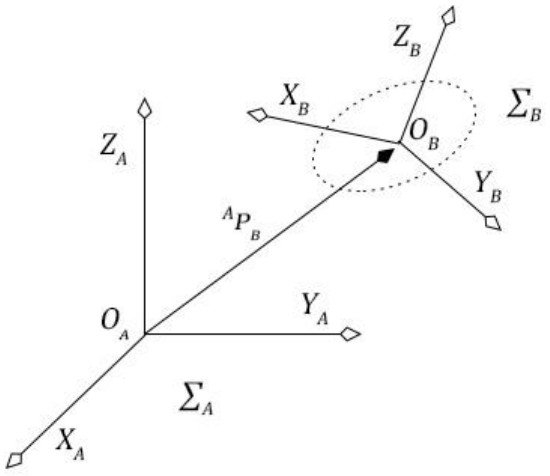

Figure 1.11: reference frame and object frame

A rotation matrix ${}^A\boldsymbol{R}_B$ is used to describe the orientation of $\sum_B$ relative to $\sum_A$. It is defined by

$$ {}^{A}\boldsymbol{R}_{B} = [{}^{A}\boldsymbol{x}_{B}\,{}^{A}\boldsymbol{y}_{B}\,{}^{A}\boldsymbol{z}_{B}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{23} & r_{33} \end{bmatrix} \tag{1.1} $$

The components of any vector of ${}^{A}\boldsymbol{x}_{B}, {}^{A}\boldsymbol{y}_{B}, {}^{A}\boldsymbol{z}_{B}$ in equation (1.1) are the projections of them onto the axes of the reference frame $\sum_A$. Therefore, the rotation matrix can be expressed as a 3×3 matrix with the components are the dot products of the pairs of unit vectors

$$ {}^{A}\boldsymbol{R}_{B} = [{}^{A}\boldsymbol{x}_{B}\,{}^{A}\boldsymbol{y}_{B}\,{}^{A}\boldsymbol{z}_{B}] = \begin{bmatrix} \boldsymbol{x}_B.\boldsymbol{x}_A & \boldsymbol{y}_B.\boldsymbol{x}_A & \boldsymbol{z}_B.\boldsymbol{x}_A \\ \boldsymbol{x}_B.\boldsymbol{y}_A & \boldsymbol{y}_B.\boldsymbol{y}_A & \boldsymbol{z}_B.\boldsymbol{y}_A \\ \boldsymbol{x}_B.\boldsymbol{z}_A & \boldsymbol{y}_B.\boldsymbol{z}_A & \boldsymbol{z}_B.\boldsymbol{z}_A \end{bmatrix} \tag{1.2} $$

Assuming that O_A and O_B are coincident, a vector $\boldsymbol{r}$ in $\sum_B$ is described by

$$ {}^{B}\boldsymbol{r} = [{}^{B}\boldsymbol{r}_{X}\,{}^{B}\boldsymbol{r}_{Y}\,{}^{B}\boldsymbol{r}_{z}]^{\boldsymbol{T}} $$

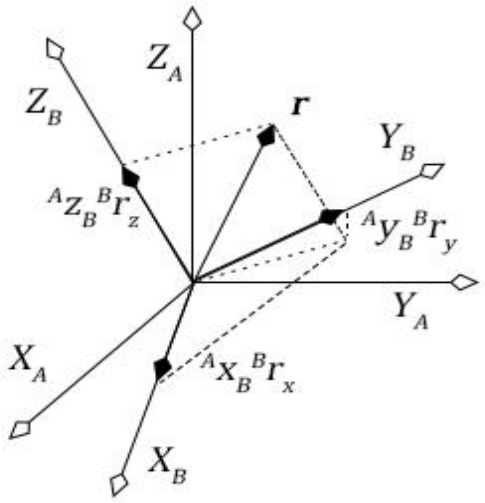

Figure 1.12: Two coincident frames with the same origin

When the origins of two coordinate frames are at the same point, expressing a vector in $\sum_B$ relative to $\sum_A$ is done by applying a linear transformation by the rotation matrix ${}^{A}\boldsymbol{R}_{B}$ from $\sum_B$to $\sum_A$

$$ {}^{A}\boldsymbol{r} = {}^{A}\boldsymbol{R}_{B}\,{}^{B}\boldsymbol{r} \tag{1.3} $$

Similarly, if we have a third coordinate frame $\sum_C$ with origin O_C at the same point of O_A and O_B, a vector ${}^{C}\boldsymbol{r}$ in $\sum_C$ can be expressed in $\sum_A$ and $\sum_B$ below:

$$^{B}\boldsymbol{r} = {}^{B}\boldsymbol{R}_{C}\,{}^{C}\boldsymbol{r} \tag{1.4}$$

$$^{A}\boldsymbol{r} = {}^{A}\boldsymbol{R}_{C}\,{}^{C}\boldsymbol{r} \tag{1.5}$$

From (1.3), (1.4) and (1.5) we have

$$^{A}\boldsymbol{R}_{C} = {}^{A}\boldsymbol{R}_{B}\,{}^{B}\boldsymbol{R}_{C} \tag{1.6}$$

The rotation matrix from $\sum_C$ to $\sum_A$ can be computed by pre-multiplying the rotation matrix from $\sum_C$ to $\sum_B$ with the rotation matrix from $\sum_B$ to $\sum_A$. This can be applied to more frames with the same origin.

Consider the case where $\sum_A$ and $\sum_B$ do not have the same origin but their three axes have the same directions, i.e. X_A, Y_A, Z_A have the same directions as X_B, Y_B, Z_B, respectively. We have a point in $\sum_B$ defined by a vector $^{B}\boldsymbol{r}$ and we want to express it in $\sum_A$. Since the two frames have the same orientation, $\sum_B$ differs to $\sum_A$ only by *translation*. We can calculate the description of point $\boldsymbol{p}$ in $\sum_A$ by vector addition of $^{B}\boldsymbol{r}$ and $^{A}\boldsymbol{p}_{B}$, a vector located at the origin of $\sum_B$ expressed in $\sum_A$ (vector from O_A to O_B)

$$^{A}\boldsymbol{r} = {}^{A}\boldsymbol{p}_{B} + {}^{B}\boldsymbol{r} \tag{1.7}$$

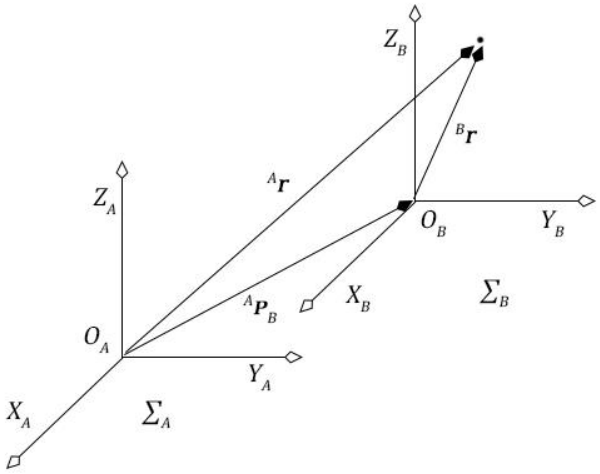

Figure 1.13: Two parallel frames

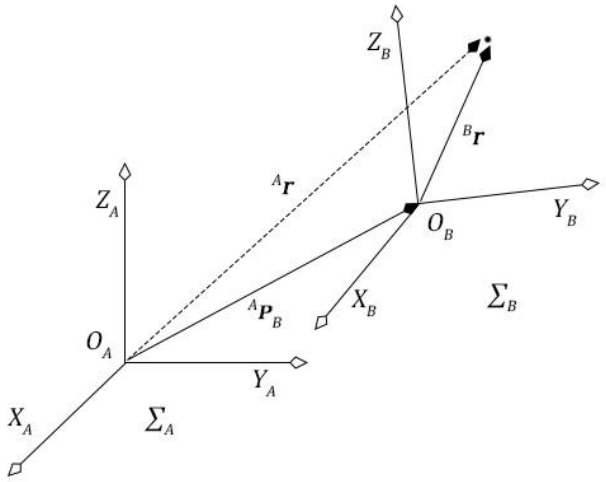

Figure 1.14: General case of mapping between two coordinate frames

Figure 1.14 shows a general case where the origins of $\sum_A$ and $\sum_B$ are not the same and the frames' orientations are different. In order to describe vector ${}^B\boldsymbol{r}$ in $\sum_A$, we first find its description in $\sum_A$ using the rotation matrix from $\sum_B$ to $\sum_A$, then we add it with vector ${}^A\boldsymbol{p}_B$

$$ {}^A\boldsymbol{r} = {}^A\boldsymbol{R}_B\,{}^A\boldsymbol{p}_B + {}^B\boldsymbol{r} \tag{1.8} $$

With ${}^A\boldsymbol{p}_B = [p_{Bx}\ p_{By}\ p_{Bz}]^T$ and ${}^B\boldsymbol{r} = [{}^Br_x\ {}^Br_y\ {}^Br_z]^T$ we have

$$ {}^A\boldsymbol{r} = \begin{bmatrix} r_{11}\,{}^Br_x + r_{12}\,{}^Br_y + r_{13}\,{}^Br_z + p_{Bx} \\ r_{21}\,{}^Br_x + r_{22}\,{}^Br_y + r_{23}\,{}^Br_z + p_{By} \\ r_{31}\,{}^Br_x + r_{23}\,{}^Br_y + r_{33}\,{}^Br_z + p_{Bz} \end{bmatrix} \tag{1.9} $$

Number 1 can be added to both sides of equation (1.9)

$$ \begin{bmatrix} {}^A\boldsymbol{r} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11}\,{}^Br_x + r_{12}\,{}^Br_y + r_{13}\,{}^Br_z + p_{Bx} \\ r_{21}\,{}^Br_x + r_{22}\,{}^Br_y + r_{23}\,{}^Br_z + p_{By} \\ r_{31}\,{}^Br_x + r_{23}\,{}^Br_y + r_{33}\,{}^Br_z + p_{Bz} \\ 1 \end{bmatrix} \tag{1.10} $$

It can then be re-written as

$$ \begin{bmatrix} {}^A\boldsymbol{r} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_{By} \\ r_{21} & r_{22} & r_{23} & p_{By} \\ r_{31} & r_{23} & r_{33} & p_{Bz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^Br_x \\ {}^Br_y \\ {}^Br_z \\ 1 \end{bmatrix} \tag{1.11} $$

or

$$\begin{bmatrix} {}^{A}\boldsymbol{r} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{A}\boldsymbol{R}_{B} & {}^{A}\boldsymbol{p}_{B} \\ 0 \quad 0 & 0 \quad 1 \end{bmatrix} \begin{bmatrix} {}^{B}\boldsymbol{r} \\ 1 \end{bmatrix} = {}^{A}\boldsymbol{T}_{B} \begin{bmatrix} {}^{B}\boldsymbol{r} \\ 1 \end{bmatrix} \tag{1.12}$$

To avoid confusion, $[{}^{A}\boldsymbol{r}, 1]^{T}$ and $[{}^{B}\boldsymbol{r}, 1]^{T}$ can be written as ${}^{A}\boldsymbol{r}$ and ${}^{B}\boldsymbol{r}$, respectively. We then have

$${}^{A}\boldsymbol{r} = {}^{A}\boldsymbol{T}_{B}\,{}^{B}\boldsymbol{r} \tag{1.13}$$

Equation (1.12) is equivalent to (1.8) but it is in a more compact and conceptual form. The 4x4 matrix in (1.11) is called a *homogeneous transformation matrix* and for robots it helps to convert a rotation and translation of a transform to a single matrix.

As an extension of equation (1.6), if we have three frames $\sum_{A}, \sum_{B}$ and $\sum_{C}$, the homogeneous transformation matrix from $\sum_{C}$ to $\sum_{A}$ can be described below:

$${}^{A}\boldsymbol{T}_{C} = {}^{A}\boldsymbol{T}_{B}\,{}^{B}\boldsymbol{T}_{C} \tag{1.14}$$

2.2 Joint Variables and End-effector's position

The relation between the manipulator arm's joint displacements and end-effector position is analyzed in this section. A manipulator with *n* degrees of freedom has joints numbered 1, 2, …, n from the base. The displacement of the joint (rotational displacement in case of revolute joints or linear displacement for prismatic joints) is denoted q_i for i^{th} joint. q_i is called *joint variable* and is an element of the *joint vector q*

$$\boldsymbol{q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n\,]^{T}$$

The position of the end-effector is denoted by a vector *r* with *m* dimensions

$$\boldsymbol{r} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_m]^{T}$$

Vector $\boldsymbol{r}$ consists the information of the position and/or orientation of the end-effector. For instance, a planar robot arm has a vector $\boldsymbol{r}$ with *m=3* which represents the location of the end point in an XY plane with the orientation of that end point (the angle

from one of the two axes). A manipulator in 3D space will need *m=6*, the first 3 elements represent the 3-dimensional position of the end-effector, the last 3 elements represent the 3 Euler angles, roll, pitch, and yaw.

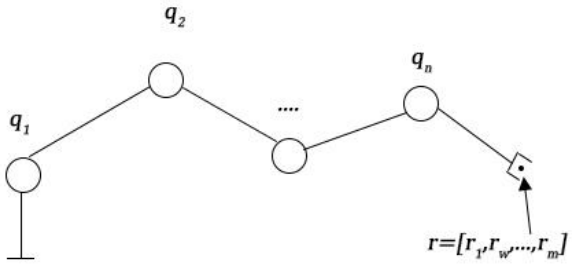

Figure 1.15: n-link manipulator

In general, the relation between $\boldsymbol{q}$ and $\boldsymbol{r}$ is non-linear due to the mechanism of the manipulator. The relation can be denoted by a non-linear function f

$$\boldsymbol{r} = f_r(\boldsymbol{q}) \tag{1.15}$$

This equation is called *kinematic equation* of the manipulator. The problem to find $\boldsymbol{r}$ with a given $\boldsymbol{q}$ is called the *direct kinematics problem* (or *forward kinematics problem*) and is rather simple since $\boldsymbol{r}$ is unique. On the contrary, the problem to find $\boldsymbol{q}$ with a given $\boldsymbol{r}$ called *inverse kinematics problem*, is complicated and is expressed as follows

$$\boldsymbol{q} = f_r^{-1}(\boldsymbol{r}) \tag{1.16}$$

Consider a manipulator with *n+1* links connected by *n* joints in series. Each joint can either be prismatic or revolute and it has one degree of freedom. For each joint *i* in the case of Baxter robot, the joint axis *i* is defined as the rotational axis since all joints of Baxter are revolute joints. The common normal between two adjacent joint axes is the line which is perpendicular with both axes. There are an infinite number of common normals if joint axes *i* and *i+1* are parallel. In this case, we select one common normal arbitrarily.

The size and shape of a link can be described by two variable: the length *a* of the common normal called *link length*, and the angle α between the projections of two adjacent joint axes onto a plane normal to the common normal called *twist angle*. The positional relation between two adjacent links can also be described by two variables: the distance *d* between the two intersecting points of a joint axis and its two common normals, called *link offset,* and the angle θ between the orthonormal projections of these common normals to a plane normal to the joint axis of the latter link, called *joint angle.*

A coordinate frame attached to each link is now defined. The origin O_i of link *i* is set as the endpoint of the mathematical model of link *i* on joint axis *i*, i.e. the foot (on link *i*) of the common normal between link *i* and *i+1*. The Z axis of $\sum_i$, denoted Z_i, is selected along the joint axis of link *i*. The direction of Z_i can be chosen arbitrarily but a good practice is to choose it in such a way that it points to the distal end of the manipulator. The X axis of $\sum_i$, denoted X_i, is selected on the common normal pointing from joint *i* to the next joint *i +1*. Lastly, the Y axis of $\sum_i$, denoted Y_i , is chosen in such a way that $\sum_i$ satisfies the right-hand rule of a three-dimensional coordinate frame.

After defining all coordinate frames of all the links of the manipulator, the four variables of each link and joint are expressed as follows:

a_{i-1} = the distance along the X_{i-1} axis from Z_{i-1} to Z_i

α_{i-1} = the clockwise angle about the X_{i-1} axis from Z_{i-1} to Z_i

d_i = the distance along the Z_i axis from X_{i-1} to X_i

θ_i = the clockwise angle about the Z_i axis from X_{i-1} to X_i

For the four parameters above, if joint *i* is prismatic, θ_i is constant and d_i expresses the translational distance of joint *i*. On the other hand, if joint *i* is revolute, d_i is constant and θ_i expresses the rotational angle of the joint. Therefore, the joint variable q_i is d_i if joint *i* is prismatic and θ_i if joint *i* is revolute.

The notation of the four parameters a, α, d, θ is called Denavit–Hartenberg (DH in short). Jacques Denavit and Richard Hardenberg introduced this convention in 1955 in order to standardize the coordinate frames for spatial linkages.

In this report we focus only on the Direct Kinematics Problem, for Inverse Kinematics please see [6] or [7]. The homogeneous transform of the attached frame of a link relative to the frame of the previous link is described as follows:

$$^{i-1}\boldsymbol{T}_i = \boldsymbol{T}_T(X_{i-1}, a_{i-1})\boldsymbol{T}_R(X_{i-1}, \alpha_{i-1})\boldsymbol{T}_T(Z_i, d_i)\boldsymbol{T}_R(Z_i, \theta_i) \tag{1.17}$$

The homogeneous transform between two adjacent links involves the translation and rotation of the four DH parameters. In equation (1.17), $\boldsymbol{T}_T(Z_i, d_i)$ describes the translation along the *Z* axis for a distance *d*, while $\boldsymbol{T}_R(Z_i, \theta_i)$ describes the rotation about the *Z* axis by an angle θ_i. The same concept applies to $\boldsymbol{T}_T(X_{i-1}, a_{i-1})$ and $\boldsymbol{T}_R(X_{i-1}, \alpha_{i-1})$. We then obtain:

$$\begin{aligned} ^{i-1}\boldsymbol{T}_i = &\begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha_{i-1} & -sin\alpha_{i-1} & 0 \\ 0 & sin\alpha_{i-1} & cos\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &\times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & 0 \\ sin\theta_i & cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{1.18}$$

In equation (1.18) above, all four matrices on the right side are in the form of equation (1.12). The first matrix is a translation along the *X* axis therefore the rotation part is an identity matrix and the *y* and *z* variables of the translation part are both 0. The second matrix a rotation about the *X* axis (yaw angles) hence all variables of the translation part are 0. The third matrix is a translation along the *Z* axis and the last matrix is a rotation about *Z* axis (roll angles). Computing the right side of equation (1.18) we have:

$$ {}^{i-1}\boldsymbol{T}_i = \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & a_{i-1} \\ cos\alpha_{i-1}sin\theta_i & cos\alpha_{i-1}cos\theta_i & -sin\alpha_{i-1} & -sin\alpha_{i-1}d_i \\ sin\alpha_{i-1}sin\theta_i & sin\alpha_{i-1}cos\theta_i & cos\alpha_{i-1} & cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.19) $$

Let the homogenous transform from $\sum_n$ to $\sum_o$ be ${}^0\boldsymbol{T}_n$. According to equation (1.14) we have

$$ {}^0\boldsymbol{T}_n = {}^0\boldsymbol{T}_1 {}^1\boldsymbol{T}_2 \dots {}^{n-1}\boldsymbol{T}_n \quad (1.20) $$

When the values of the link parameters are given, ${}^{i-1}\boldsymbol{T}_i$ is a function of q_i, which is d_i in case of prismatic joint and θ_i in case of revolute joint. Consequently, ${}^0\boldsymbol{T}_n$ is a function of the joint vector $\boldsymbol{q}$.

Let $\sum_E$ be the frame attached to the end-effector, $\sum_R$ is the reference frame and $\sum_0$ is the base frame. We have:

$$ {}^R\boldsymbol{T}_E = {}^R\boldsymbol{T}_0 {}^0\boldsymbol{T}_n {}^n\boldsymbol{T}_E \quad (1.21) $$

${}^RT_0$ is constant since the chosen reference frame is fixed and its homogenous transform to the base frame does not depend on any variable. In many cases the base frame can be chosen as the reference frame. ${}^n\boldsymbol{T}_E$ is also a constant because of the way the end-effector connects to the last link *n*. When $\sum_n$ rotates $\sum_E$ rotates with the same orientation, O_E also moves the same distance in the same direction with O_E if it moves. The *forward kinematic problem* to find $\boldsymbol{r} = f_r(\boldsymbol{q})$ can be solved by equation (1.21) since many elements in ${}^0\boldsymbol{T}_n$ can be a function of $\boldsymbol{q}$ (*d* in case of a prismatic joint and θ for a revolute joint) and ${}^R\boldsymbol{T}_0$ along with ${}^n\boldsymbol{T}_E$ are constant as mentioned above.

2.4 Baxter's Kinematics

In this section, we build the kinematic model of Baxter robot's left arm starting by assigning a coordinate frame to each joint. Baxter's left arm has seven revolute joints numbered from 0 to 7 with corresponding frames attached to them, {0} to {7}. The

reference frame is chosen coincident with frame {0} and a coordinate frame {EE} is attached to the end-effector.

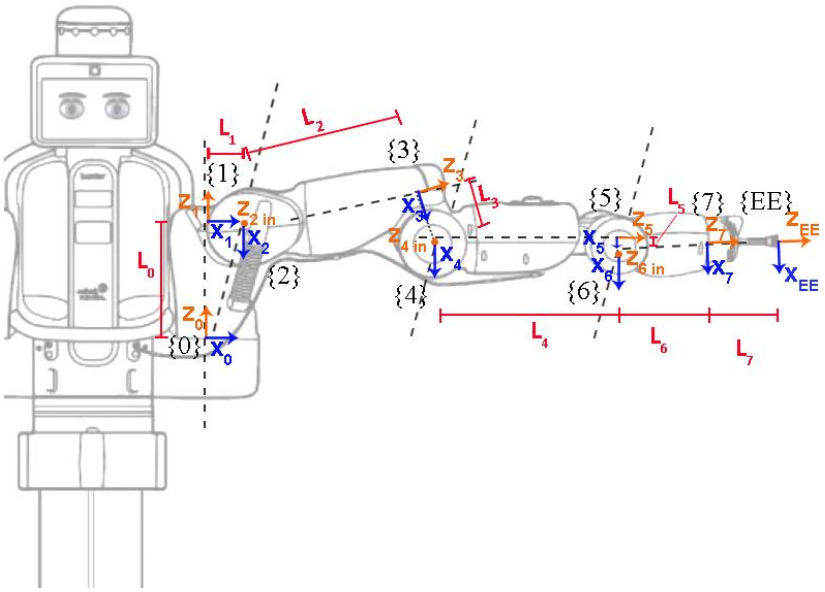

Figure 1.16: Frame assignments of Baxter's left arm

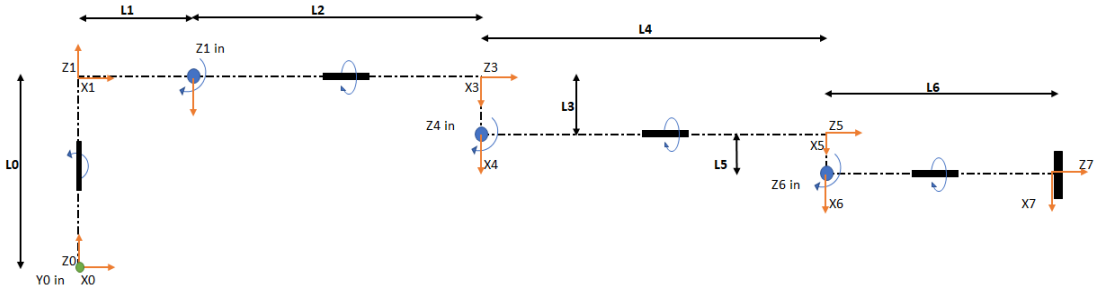

Figure 1.17: Simplified frame assignments of Baxter's left arm

The DH parameter table is then constructed based on the frame assignments. Variable L stands for the link length. Since Baxter's joints are revolute, θ_i is non-constant while other variables are constant. From figure 1.17 X_2 is perpendicular to X_1 hence the 90-degree offset in θ_2.

i	$\boldsymbol{\alpha}_{i-1}$	a_{i-1}	d_i	$\boldsymbol{\theta}_i$
1	0	0	L_0	θ_1
2	-90 deg	L_1	0	$\theta_1 + 90deg$
3	90 deg	0	L_2	θ_3
4	-90 deg	L_3	0	θ_4
5	90 deg	0	L_4	θ_5
6	-90 deg	L_5	0	θ_6
7	90 deg	0	L_6	θ_7
EE (from joint 6)	90 deg	0	$L_6 + L_7$	θ_7

Table 1.8: D-H parameter table of Baxter's left arm

3. Cartesian Velocity Control

3.1 Jacobian Matrix

In the last section, the relation between the joint variables and the end-effector position was analyzed. The forward kinematic problem can be solved using the homogenous transformation matrices. The position and orientation of an object can be expressed by the relation between its attached frame to the reference frame. We now discuss the Jacobian matrix, which relates the joint velocities to the end-effector velocities [8]. A Jacobian (or Jacobian matrix) defines the relationship between two different representations of a system. In the case of a robot arm, Jacobian matrix relates the joint velocity to the end-effector velocity (which includes the translational and rotational velocity).

Consider the case of a *k*-dimensional vector $\boldsymbol{\xi} = [\xi_1, \xi_2, \dots, \xi_k]^T$ and an *l*-dimensional vector $\boldsymbol{\eta} = [\eta_1, \eta_2, \dots, \eta_l]^T$ and their relation as follows:

$$\eta_i = f_i(\xi_1, \xi_2, \dots, \xi_k) \; with \; i = 1,2, \dots, l \tag{1.22}$$

The *Jacobian matrix* of $\boldsymbol{\eta}$ with respect to $\boldsymbol{\xi}$ is a $l \times k$ matrix:

$$\boldsymbol{J}_{\eta}(\boldsymbol{\xi}) = \frac{\partial_{\boldsymbol{\eta}}}{\partial_{\xi^T}} = \begin{bmatrix} \frac{\partial \eta_1}{\partial \xi_1} & \frac{\partial \eta_1}{\partial \xi_2} & \cdots & \frac{\partial \eta_1}{\partial \xi_k} \\ \frac{\partial \eta_2}{\partial \xi_1} & \frac{\partial \eta_2}{\partial \xi_2} & \cdots & \frac{\partial \eta_2}{\partial \xi_k} \\ \vdots & \vdots & & \vdots \\ \frac{\partial \eta_l}{\partial \xi_1} & \frac{\partial \eta_l}{\partial \xi_2} & \cdots & \frac{\partial \eta_l}{\partial \xi_k} \end{bmatrix} \tag{1.23}$$

Differentiating equation (1.22) with respect to time we have:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_{\eta}(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} \tag{1.24}$$

Similarity, we can also use the Jacobian matrix to express equation (1.15) when taking its derivative:

$$\dot{\boldsymbol{r}} = \boldsymbol{J}_{r}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{1.25}$$

with

$$\boldsymbol{J}_{r}(\boldsymbol{q}) = \begin{bmatrix} \frac{\partial r_1}{\partial q_1} & \frac{\partial r_1}{\partial q_2} & \cdots & \frac{\partial r_1}{\partial q_n} \\ \frac{\partial r_2}{\partial q_1} & \frac{\partial r_2}{\partial q_2} & \cdots & \frac{\partial r_2}{\partial q_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial r_m}{\partial q_1} & \frac{\partial r_m}{\partial q_2} & \cdots & \frac{\partial r_m}{\partial q_n} \end{bmatrix} \tag{1.26}$$

To simplify, we can denote $\boldsymbol{J}_{r}(\boldsymbol{q})$ as $\boldsymbol{J}$**.** For a manipulator arm, the Jacobian matrix can be used to calculate the end-effector velocity with respect to the joint velocity using formulas (1.25) and (1.26). However, for a robot arm with many degrees of freedom, such as the Baxter robot with seven joints, its Jacobian matrix is a system of six equations and seven variables hence the calculation may be highly complicated.

A simpler method of calculating the Jacobian matrix was mentioned in [9] based on the geometry of the manipulator. For a robot arm with $n + 1$ joints we have the following notations:

${}^{j}\boldsymbol{p}_{i}$:$Vector\ from\ the\ origin\ of\ joint\ j\ to\ the\ origin\ of\ joint\ i$

${}^0\boldsymbol{p}_{E,i}$:$Vector\ from\ the\ origin\ of\ joint\ i\ to\ the\ origin\ of\ the\ end-effetor\ frame, expressed\ in\ frame\ \{0\}$

${}^0\boldsymbol{z}_i$: $the\ unit\ vector\ of\ Z\ axis\ expressed\ in\ frame\ \{0\}$

We have

$$\boldsymbol{J} = \begin{bmatrix} {}^0\boldsymbol{z}_1 \times {}^0\boldsymbol{p}_{E,1} & {}^0\boldsymbol{z}_2 \times {}^0\boldsymbol{p}_{E,2} & \cdots & {}^0\boldsymbol{z}_n \times {}^0\boldsymbol{p}_{E,n} \\ {}^0\boldsymbol{z}_1 & {}^0\boldsymbol{z}_2 & \cdots & {}^0\boldsymbol{z}_n \end{bmatrix} \tag{1.27}$$

We can then calculate $\boldsymbol{J}$ from ${}^0T_i$ by using the following relations:

$${}^0\boldsymbol{T}_i = \begin{bmatrix} {}^0x_i & {}^0y_i & {}^0z_i & {}^0\boldsymbol{p}_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, i = 1,2,\dots,n+1 \tag{1.28}$$

$${}^0\boldsymbol{p}_{E,i} = {}^0\boldsymbol{p}_E - {}^0\boldsymbol{p}_i \tag{1.29}$$

3.2 Baxter's Jacobian

The joint angles of Baxter's arm are denoted as $\theta_i,\ i = 1,2,\dots,7$. The end-effector position and orientation are denoted as $r_j, j = 1,2,\dots,6$. According to equation (1.15) we have:

$$\boldsymbol{r} = f(\boldsymbol{\theta}) \tag{1.30}$$

with

$$\boldsymbol{r} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \dots, \boldsymbol{r}_6]^T$$

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_7]^T$$

In order to calculate the velocity of the end-effector with the given joint velocity, we take the derivative of equation (1.49) with respect to time:

$$\dot{\boldsymbol{r}} = \boldsymbol{J}\dot{\boldsymbol{\theta}} \tag{1.31}$$

where $\dot{\boldsymbol{r}} = d\boldsymbol{r}/dt$, $\dot{\boldsymbol{\theta}} = d\boldsymbol{\theta}/dt$ and $\boldsymbol{J} = \partial f / \partial \boldsymbol{\theta}$ is the Jacobian matrix. $\boldsymbol{J} \in \boldsymbol{R}^{6\times7}$

As mentioned in the previous subsection, the Jacobian matrix can be calculated using equation (1.26). However, it is highly complicated especially in the case of Baxter robot with seven joints. The geometric formula (1.27) can be used instead. Figure 1.18

describes a Baxter's manipulator. The Jacobian matrix of the end-effector frame {EE} with respect to frame {0} is calculated as follows

$$ {}^{0}\boldsymbol{J}_{E} = \begin{bmatrix} {}^{0}\boldsymbol{z}_1 \times {}^{0}\boldsymbol{p}_{E,1} & {}^{0}\boldsymbol{z}_2 \times {}^{0}\boldsymbol{p}_{E,2} & \cdots & {}^{0}\boldsymbol{z}_7 \times {}^{0}\boldsymbol{p}_{E,7} \\ {}^{0}\boldsymbol{z}_1 & {}^{0}\boldsymbol{z}_2 & \cdots & {}^{0}\boldsymbol{z}_7 \end{bmatrix} \tag{1.32} $$

To compute the Jacobian matrix of the end-effector with a different *reference frame*, for instance the body frame {B} of the Baxter robot, we can use the formula below

$$ {}^{B}\boldsymbol{J}_{E} = \begin{bmatrix} {}^{B}\boldsymbol{R}_0 & 0 \\ 0 & {}^{B}\boldsymbol{R}_0 \end{bmatrix} {}^{0}\boldsymbol{J}_{E} \tag{1.33} $$

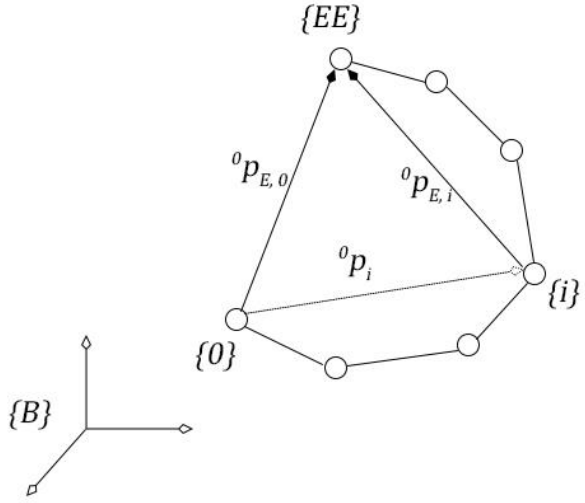

Figure 1.18: Presentation of a seven-joint manipulator

In summary, for the task of finding the Baxter's end-effector velocity corresponding to the joint velocity, the homogenous transforms between links can be computed using equations (1.19) and (1.20) based on the link frame assignment in Figure 1.16 and the D-H parameter Table 1.8. The position vectors from the origin of the joint frames to the end-effector with respect to the reference frame can be found using the equations (1.28) and (1.29). Then the Jacobian matrix is computed using equation (1.32).

3.3 Cartesian Velocity Control

Baxter robot has 3 modes for arm control which are Joint Position, Joint Velocity and Torque. In all three modes, data is input once, the movement is then carried out by applying joint angle, velocity or torque on all joints simultaneously. In this subsection, Cartesian Velocity Control mode is developed for the Baxter robot. The velocity of the end-effector is defined, and the joints will move to achieve the desired end-effector trajectory. The main idea of this mode is to use the Joint Velocity arm control mode to give sets of joint angle values to the robot over a period of time so that the end-effector moves with the desired velocity.

From previous sections we know how to find the position of the end-effector for a given joint position and vice versa. We also know how to find the end-effector velocity with given joint velocity. We now consider the case of finding the joint velocity that results in a desired end-effector velocity. This can be viewed as a problem of inverse kinematic in a broader sense. From equation (1.50) we have

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{r}} \tag{1.34}$$

Hence, in principle we can find $\dot{\boldsymbol{\theta}}$ by calculating the inverse of J and multiplying it with $\dot{\boldsymbol{r}}$. This, however, only works in the case of $n = m$, i.e. when the number of joint variables equals to the number of end-effector variables. $\boldsymbol{J}$ is a squared matrix and there exists $\boldsymbol{J}^{-1}$.

Consider the case when $n \geq m + 1$ and the rank of $\boldsymbol{J}$ is m, we have the general solution for equation (1.50)

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^{+}\dot{\boldsymbol{r}} + (\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})\boldsymbol{k} \tag{1.35}$$

where J^{+} is the Pseudo inverse of the Jacobian matrix [10]

$$\boldsymbol{J}^{+} = \boldsymbol{J}^{T}(\boldsymbol{J}\boldsymbol{J}^{T})^{-1} \tag{1.36}$$

and $\boldsymbol{k}$ is an arbitrary n-dimensional constant vector. This $\boldsymbol{k}$ implies that there are an infinite number of solutions to equation (1.34), i.e. there are an infinite number of ways to move the end-effector while tracking the same trajectory. If the task is only to position and orient the end-effector, multiple solutions to (1.34) also imply that there is some redundancy to the manipulator and additional goals can be achieved using the second term of the right side of equation (1.35). This redundancy will be discussed in later sections.

CHAPTER II:

MANIPULARITY OF BAXTER ROBOT

1. Measure of Manipulability

Quantitative measure of manipulating ability of robot arm in positioning and orienting the end-effectors is beneficial for the design of the robots. Manipulability is the ability of an end-effector to move from its current position and orientation. The easier it can move the better (larger) the measure of manipulability.

Consider a manipulator with *n* degrees of freedom, the joint variables can be denoted as $\theta_i, i = 1,2, \dots, n$. We define a class of *m* tasks we are interested in as $r_j, j = 1,2 \dots, m$ $(m \leq n)$. In the scope of this report $\boldsymbol{r}_j$ is the position and orientation of the Baxter's end-effector. The choice of joint variable symbol θ_i is convenient since Baxter's joints are all revolute. We have the following relation

$$\boldsymbol{r} = f(\boldsymbol{\theta}) \tag{2.1}$$

where $\theta = [\theta_1, \theta_2, \dots, \theta_n]^T$ is the joint vector and $r = [r_1, r_2, \dots, r_m]^T$ is the manipulator vector. Differentiating the equation above with respect to time yields

$$\dot{\boldsymbol{r}} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{2.2}$$

The Jacobian $J(\theta)$ is a $m \times n$ matrix. We have the following condition

$$\max_{\theta} rank\, \boldsymbol{J}(\boldsymbol{\theta}) = m \tag{2.3}$$

When the condition (2.3) is satisfied, we say that the manipulator is kinematically redundant [11], and the degree of kinematic redundancy is $(n - m)$. If for some $\boldsymbol{\theta}$

$$rank\, \boldsymbol{J}(\boldsymbol{\theta}) < m$$

which means that there are one or more dependent columns in $\boldsymbol{J}$. The number of independent columns is less than m and $\boldsymbol{J}$ transforms $\dot{\boldsymbol{\theta}}$ to a lower level of dimension. We say that the manipulator is in a singular state. In this state, the manipulator vector $\boldsymbol{r}$ cannot move in certain directions therefore this state is undesirable.

We define a scalar value w as the *measure of manipulability* of state θ with respect to manipulator vector $\boldsymbol{r}$. w is given by

$$w = \sqrt{\det\ \boldsymbol{J}(\boldsymbol{\theta})\boldsymbol{J}^T(\boldsymbol{\theta})} \tag{2.4}$$

A physical presentation of this measure can be given in the following. For any matrix $\boldsymbol{J} \in \boldsymbol{R}^{m\times n}$ there exist orthogonal matrices $\boldsymbol{U} \in \boldsymbol{R}^{m\times m}$ and $\boldsymbol{V} \in \boldsymbol{R}^{n\times n}$ [12] such that

$$\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{2.5}$$

with

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & & 0 & \\ & \sigma_2 & & 0 \\ 0 & & \ddots & \\ & & & \sigma_m \end{bmatrix} \in \boldsymbol{R}^{m\times n}\ and\ \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \geq 0$$

The equation (2.5) is called the *singular value decomposition* [13]. From (2.4) and (2.5) we have

$$w = \sigma_1.\sigma_2 \ldots \sigma_m$$

The subset of the realizable velocity $\dot{r}$ in $\boldsymbol{R}^m$ is an ellipsoid with principal axes $\sigma_1 u_1, \sigma_2 u_2, \ldots, \sigma_m u_m$ using joint velocity $\dot{\boldsymbol{\theta}}$ such that $\left\|\dot{\boldsymbol{\theta}}\right\| = \dot{\theta}_1^2 + \dot{\theta}_2^2 + \cdots + \dot{\theta}_n^2 \leq 1$, u_i is the i^{th} column vector of $\boldsymbol{U}$. This ellipsoid can be called *manipulability ellipsoid*. The volume of this ellipsoid is given by

$$d\sigma_1.\sigma_2 \ldots \sigma_m \tag{2.6}$$

where d is constant and is given by

$$d = \frac{(2\pi)^{\frac{m}{2}}}{2\cdot 4\cdot 6\cdots(m-2)\cdot m} when\ m\ is\ even$$

and

$$d = \frac{(2\pi)^{\frac{m-1}{2}}}{1\cdot 3\cdot 5\cdots(m-2)\cdot m} when\ m\ is\ odd$$

For simplicity, we can call w as the volume of the manipulability (volume of the ellipsoid)

2. Analysis of Baxter's manipulability

For the task moving the end-effector, $m = 6$ corresponding with the velocity of three cartesian angles (X,Y,Z) and three Euler angles (roll, pitch, yaw) while Baxter has 7 degrees of freedom per arm $n = 7$. Therefore, Baxter manipulator arm is kinematically redundant for the task of moving the end-effector

Since m is even, we have

$$d = \frac{(2\pi)^{\frac{m}{2}}}{2 \cdot 4 \cdot 6 \cdots (m-2) \cdot m} = \frac{(2\pi)^3}{2 \cdot 4 \cdot 6} = \frac{(2\pi)^3}{48}$$

We then have the formula to calculate the volume of manipulability

$$w = \frac{(2\pi)^3}{48} \cdot \sigma_1 \cdot \sigma_2 \cdots \sigma_m \tag{2.7}$$

with $\sigma_1, \sigma_2, \ldots, \sigma_m$: singular values of vector Σ

To find w, we used MATLAB to compute the homogenous transformation matrices between links and calculate the Jacobian matrix, then we apply formula (2.7). To show the result, Baxter simulation was used and Python code ik_service_client.py from RethinkRobotics was modified to calculate different arm postures for the same end-effector position.

The maximum and minimum volume of manipulability and their joint configurations were also investigated. For an increment of 1 degree, all the combinations of the joint angles are 4.971×10^{19} combinations which is 49710 trillion or forty-nine quadrillion seven hundred ten trillion cases. The calculation may take weeks on a personal workstation. To reduce the calculation time, an increment of 20 degrees was used, then several offsets were added to the joint angles to perform the calculation. An estimate of 410,572,800 cases were tested. To help speeding up the run time, a supercomputer was used. Comet is a dedicated XSEDE cluster designed by Dell and SDSC delivering ~2.0 petaflops, featuring Intel next-gen processors with AVX2,

Mellanox FDR InfiniBand interconnects and Aeon storage. The total cases were divided into several smaller sets. Each set is a “job” which gets submitted to the cluster and is handled by a node consisting of 20 CPU cores. The maximum and minimum volume of manipulability along with their joint angle configurations of each set are then downloaded to the local computer to compare and produce the results.

Figure 2.1 shows four different joint configurations with the same end-effector position. They have different volume of manipulability ellipsoids. Manipulability volume W1 is largest hence it is easy for the end-effector to move from its current position. W4 is smallest therefore it is harder for it to move around comparing to the first joint configuration. We can even see the first posture looks more natural than the last one.

Figure 2.2 shows the difference of the ellipsoid volume between two different joint movements with the same trajectory. We can see that Baxter has better manipulability in picture 2 of Figure 2.3.

Figure 2.3 shows the volume of manipulability over time of the two different movements from Figure 2.3. The robot arm has better manipulability moving as picture 2 of Figure 2.3.

Figure 2.4 shows the joint configurations of the maximum and minimum ellipsoid volume. Maximum W is 0.8199 when minimum is almost zero (1.7855×10^{-6} . The location of the end-effector in the right picture is at the boundary which explains the almost zero volume of the ellipsoid. The ellipsoid at this posture squeezes into a line.

Figure 2.5 shows the method used to find the minimum and maximum manipulability. The total number of sets of joint angles was divided into smaller batches and they were uploaded to multiple nodes of a super computer running MATLAB.

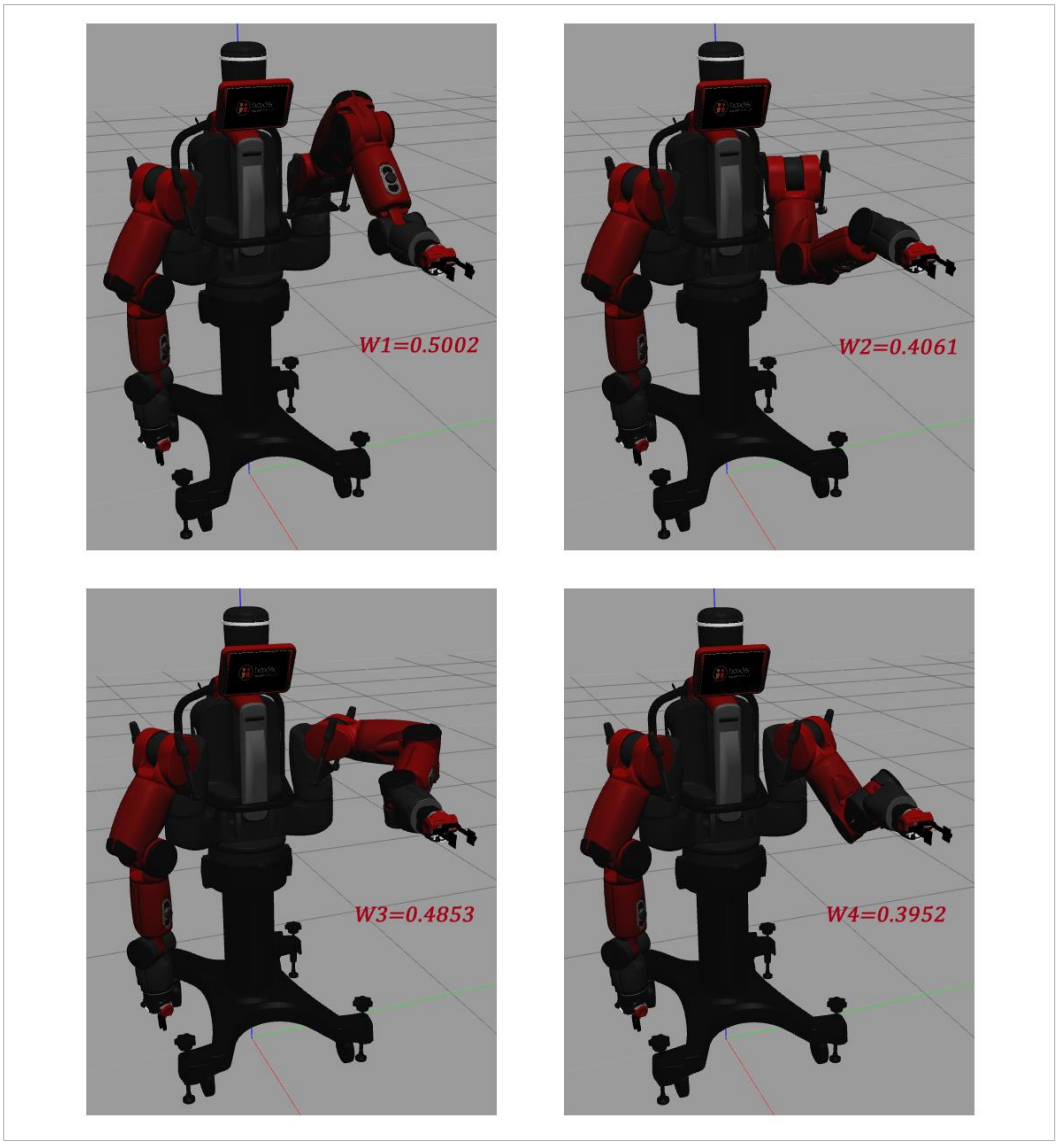

Figure 2.1: Different joint configurations for the same end-effector position

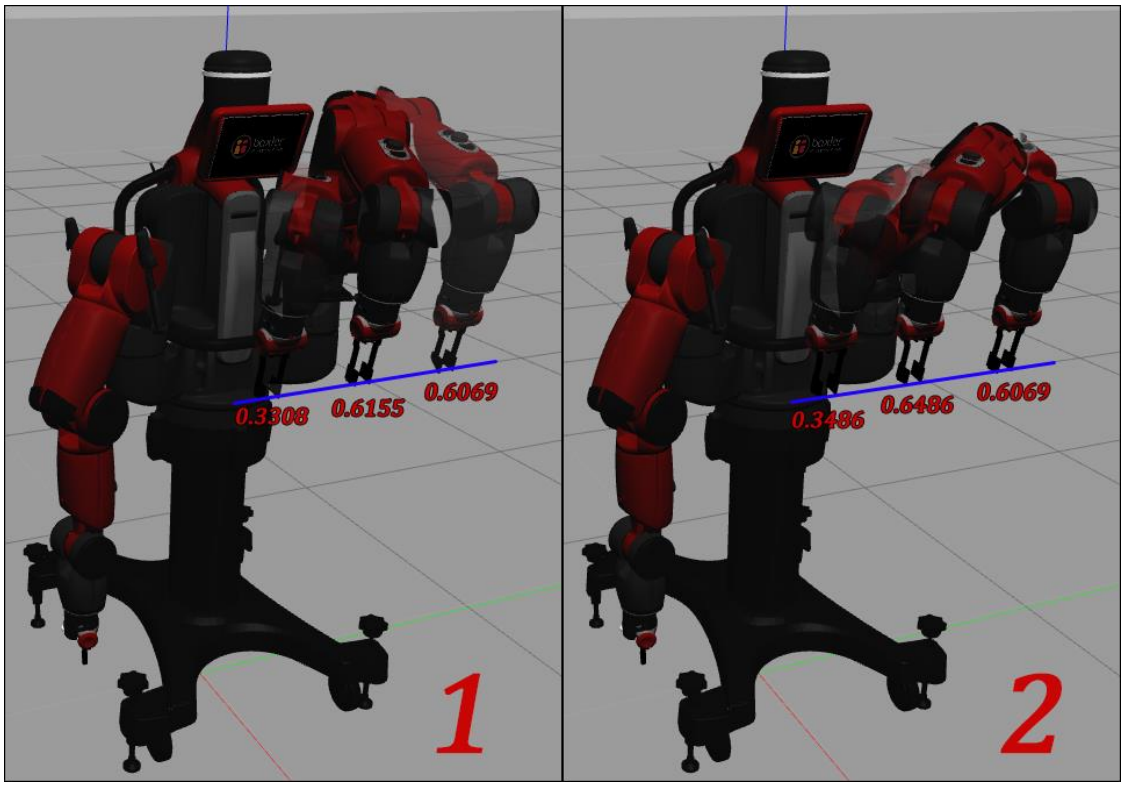

Figure 2.2: Different joint movements for the same trajectory with volume manipulability

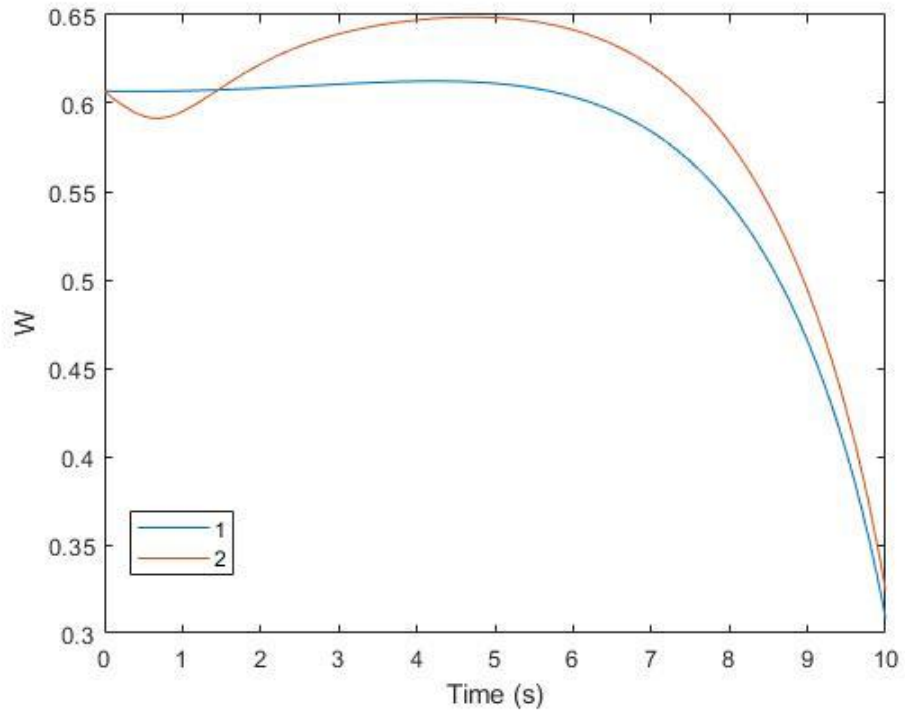

Figure 2.3: Manipulability ellipsoid volume over time of Figure 2.3

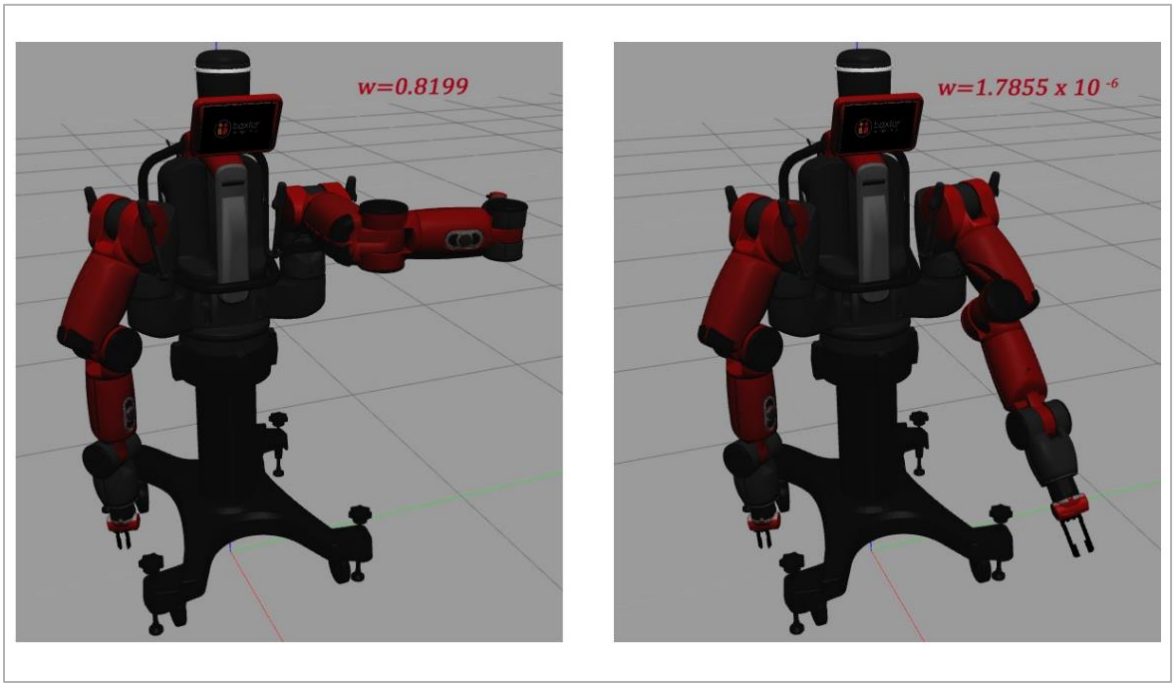

Figure 2.4: Maximum (left) and minimum (right) volume of manipulability

```
lek5294@comet-ln3:~/matlabcode/20degoffset13
[lek5294@comet-ln3 20degoffset13]$ squeue -u lek5294
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          27496657   compute  matlab1  lek5294  R      18:50      1 comet-23-69
          27496658   compute  matlab5  lek5294  R      18:50      1 comet-07-43
          27496659   compute  matlab9  lek5294  R      18:50      1 comet-07-54
          27496660   compute matlab13  lek5294  R      18:50      1 comet-07-68
          27496661   compute matlab17  lek5294  R      18:50      1 comet-15-70
          27496662   compute matlab21  lek5294  R      18:50      1 comet-15-71
          27496663   compute matlab25  lek5294  R      18:50      1 comet-15-72
          27496664   compute matlab29  lek5294  R      18:50      1 comet-22-10
          27496665   compute matlab33  lek5294  R      18:50      1 comet-22-15
          27496666   compute matlab37  lek5294  R      18:50      1 comet-22-71
          27496667   compute matlab41  lek5294  R      18:50      1 comet-13-02
          27496669   compute matlab45  lek5294  R      18:50      1 comet-13-03
          27496670   compute matlab49  lek5294  R      17:50      1 comet-03-50
          27496671   compute matlab53  lek5294  R      17:50      1 comet-13-38
          27496672   compute matlab57  lek5294  R      17:50      1 comet-13-44
          27496673   compute matlab61  lek5294  R      17:50      1 comet-20-63
          27496674   compute matlab65  lek5294  R      17:50      1 comet-20-66
          27496675   compute matlab69  lek5294  R      17:50      1 comet-20-69
          27496676   compute matlab73  lek5294  R      17:50      1 comet-20-71
          27496677   compute matlab77  lek5294  R      17:50      1 comet-20-72
          27496678   compute matlab81  lek5294  R      17:50      1 comet-25-45
          27496679   compute matlab85  lek5294  R      17:50      1 comet-25-64
          27496680   compute matlab89  lek5294  R      17:50      1 comet-25-65
[lek5294@comet-ln3 20degoffset13]$
```

Figure 2.5: Jobs submitted to the supercomputer to find the minimum and maximum volume of manipulability

CHAPTER III:

SINGULARITY AVOIDANCE

1. Order of Priority

The concept of dividing a task into multiple subtasks and the control algorithm to utilize the redundancy for optimizing given performance criterion were mentioned in [14]

Task decomposition is the basic idea for the development of the redundancy control algorithm. Most of the task for a multi-degree-of-freedom robot arm can be divided into different subtasks with different priorities. For instance, for a robot arm with a simple given task to press a button, this task can be divided into two subtasks: moving the end-effector to the location of the button and pressing the button. The first subtask is more important since the second subtask cannot be done without the success of the first one. The first subtask can also be divided into two subtasks which are hand position and orientation, the former is more important than the latter since the rotation of the hand is not as significant as the position. In the case of Baxter robot, a second subtask can be added to the task of moving the end-effector which is the singularity avoidance, or the obstacle avoidance, or both.

For these tasks, it is natural to try to perform the first subtask with the highest priority. If there is any ability left for the manipulator, the second subtask with the next priority can be performed. The third subtask then can be picked up if there is still ability left after achieving the first two subtasks, and so on. The existence of the remaining ability in any stages means that the manipulator is redundant for the subtask up to that stage.

2. Control Algorithm for Redundancy

We assume that the first subtask is to track a desired trajectory $\boldsymbol{r}^*(t)$ of the manipulation vector $\boldsymbol{r}$. We also assume that a scalar performance criterion for the second subtask is given by

$$\boldsymbol{p} = q(\boldsymbol{\theta})$$

The general solution of $\dot{\boldsymbol{r}} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}$ is given by (1.54). As mentioned in section 3.3 of chapter II, the second term $(\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{k}$ can be used to achieve additional subtasks.

The time derivative of p is given by

$$\dot{\boldsymbol{p}} = \boldsymbol{\xi}^T\dot{\boldsymbol{\theta}} \tag{3.1}$$

where

$$\boldsymbol{\xi} = [\xi_1, \xi_2, \dots, \xi_n]^T$$
$$\xi_l = \frac{\partial q(\boldsymbol{\theta})}{\partial \theta_l}, \qquad l = 1,2, \dots, n$$

If the first subtask is perfectly performed, from (1.54) and (3.1) we have

$$\dot{\boldsymbol{p}} = \boldsymbol{\xi}^T\boldsymbol{J}^+\dot{\boldsymbol{r}}^* + \boldsymbol{\xi}^T(\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{k} \tag{3.2}$$

To achieve the second subtask, we select k based on a constant k_1

$$\boldsymbol{k} = \boldsymbol{\xi}k_1$$

The basic equation for the control algorithm is given as follows

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^+\dot{\boldsymbol{r}}^* + (\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{\xi}k_1 \tag{3.3}$$

The gradient vector $\boldsymbol{\xi}$ can be computed so perform the redundancy task. In order to prevent $\dot{\boldsymbol{\theta}}$ from becoming excessive, k_1 should be chosen within a limit. A condition can be given as follows

$$k_1 \leq [\boldsymbol{\xi}^T(\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{\xi}]^{-\frac{1}{2}}k_3\dot{\boldsymbol{\theta}}_H \tag{3.4}$$

where k_3 is a constant $(0 \leq k_3 \leq 1)$ and $\dot{\boldsymbol{\theta}}_H$ is the hardware limit for the joint angle rates.

3. Singularity Avoidance

A robot Singularity is a configuration in which the end-effector of a robot loses the ability to move along certain directions. Near singularity, the joint velocity becomes exceptionally large and the end-effector deviates from its expected trajectory. A robot at its singularity can be unpredictable and dangerous. Damage can be done to itself and any humans/objects around it. A redundant manipulator can utilize its redundancy to try to avoid singularity [15].

For the case of singularity avoidance, we try not only to avoid singularity but also to keep the measure of manipulability as large as possible. A technique to calculate the gradient vector $\boldsymbol{\xi}$ was developed in [16]. Let

$$\boldsymbol{G} = \boldsymbol{J}\boldsymbol{J}^T = \left[g_{ij}\right], i, j = 1,2, \dots, m$$

The performance criterion is then $p = \sqrt{detG}$. We have

$$\xi_l = \frac{1}{2\sqrt{det\boldsymbol{G}}} \sum_{i,j=1}^{m} \Delta_{i,j} \cdot {}_l g'_{ij} = \frac{1}{2}\sqrt{det\boldsymbol{G}} \sum_{i,j=1}^{m} [\boldsymbol{G}^{-1}]_{ij} \left({}_l\boldsymbol{J}'_i\boldsymbol{J}_j^T + {}_l\boldsymbol{J}'_j\boldsymbol{J}_i^T\right) \qquad (3.5)$$

where

$$\Delta_{i,j} = cofactor\ of\ a_{ij}\ for\ G$$

$${}_l g'_{ij} = \frac{\partial\left(g_{ij}(\theta)\right)}{\partial\theta_l}$$

$$[\boldsymbol{G}^{-1}]_{ij} = the\ (i,j) element\ of\ the\ inverse\ of\ G$$

$$\boldsymbol{J}_i = the\ ith\ row\ of\ J$$

$${}_l\boldsymbol{J}'_i = the\ derivative\ of\ J_i\ with\ respect\ to\ \theta_l$$

Substitute (3.5) to (3.3) and choose a k_1 we have the joint velocity $\dot{\boldsymbol{\theta}}$. Using the Cartesian Velocity Control mode, we can move the end-effector of Baxter along the desired trajectory while avoiding the singularities.

4. Singularity avoidance of Baxter robot

As shown in figure 1.10, Baxter robot has three offsets in its kinematic structure which complicates the analytical kinematic equations. Baxter has no three consecutive joints that share a common origin. According to Pieper's principle [17], if there are three consecutive coordinate frames anywhere on a 6-dof robot, an analytical solution to a non-linear inverse kinematic problem is guaranteed to exist, i.e. for an end-effector position and orientation the joint variables can be found. With 7-dof Baxter robot, the task to find joint variables based on end-effector variables is highly complicated therefore its singularity is challenging to find. Research has been carried out in [18] which resulted in the conclusion that it is essentially impossible to analyze the inverse kinematic problem analytically. Due to the offsets there is no closed-form analytical solution to the problem of finding the singularity configuration.

A control method for equation (3.3) generally keeps the measure of manipulability as large as possible so that it will prevent the arm from coming close to the singularity where the volume of manipulability ellipsoid is close to zero. It also helps the arm to move away quickly from the singularity.

Figure (3.1) shows the ellipsoid volumes of a movement of the Baxter's left arm along the Y-axis of the base frame from the initial joint angles $\boldsymbol{\theta} = [-\frac{\pi}{4},\ -\frac{\pi}{4},\ 0,\ \frac{\pi}{2}, 0,\ -\frac{\pi}{4}, 0]^T$ with different k_1 . With the un-controlled movement with $k_1 = 0$, the algorithm does not try to keep the volume of manipulability as large as possible. The control algorithm keeps the volume of manipulability of as large as possible while tracking the desired trajectory. The larger the constant k_1 the more aggressive the robot arm tries to avoid the singularity

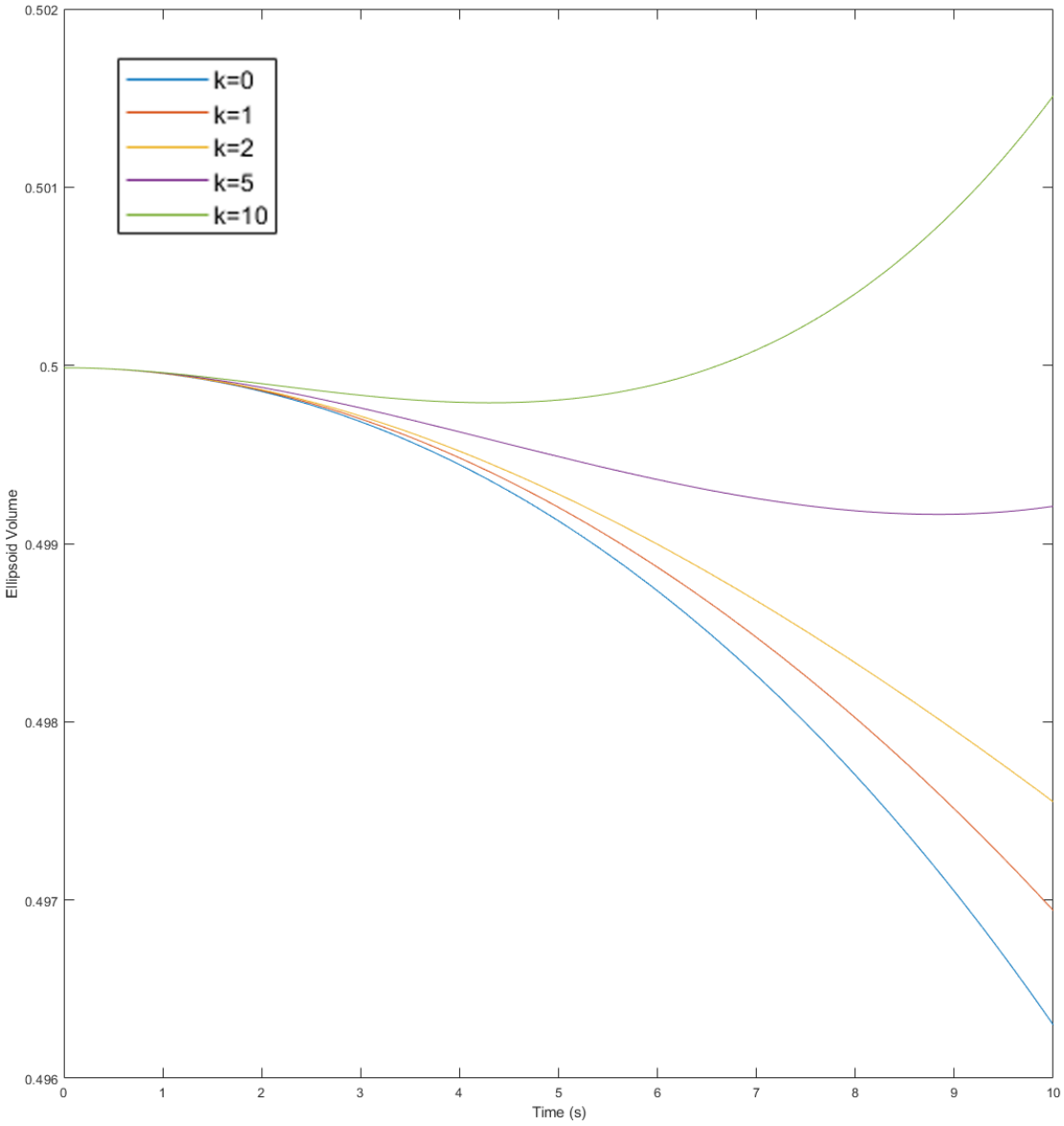

Figure 3.1: Controlled algorithm with different k

CHAPTER IV :

OBSTACLE AVOIDANCE

1. Obstacle Avoidance Algorithm

A control algorithm was shown in [19] where the first subtask is to move along a trajectory and the second subtask is to avoid an obstacle. The former is more important than the latter. The velocity of the end-effector is the manipulator vector for the task of tracking a trajectory. For the obstacle avoidance algorithm, the main idea is to take $\boldsymbol{\theta}$ as the second manipulation vector and to teach in advance an arm posture $\boldsymbol{\theta}_r$ which avoids collision. Our goal is to move the arm as close as possible to the arm posture $\boldsymbol{\theta}_r$ while tracking the desired trajectory. Let the performance criterion for the obstacle avoidance subtask be

$$p = g(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_r)^T H_2(\boldsymbol{\theta} - \boldsymbol{\theta}_r) \tag{4.1}$$

where

$$\boldsymbol{H}_2 = diag(h_{2i}) \in \boldsymbol{R}^{n\times n} \text{and } h_{2i} > 0 \text{ are constants}$$

The performance criterion p in equation (4.1) should be as large as possible. The arm should try to come as nearly as possible to the arm posture $\boldsymbol{\theta}_r$ while satisfying the first subtask with the top priority which is tracking the desired end-effector tracjectory.

From (4.1) we have

$$\boldsymbol{\xi} = -\boldsymbol{H}_2(\boldsymbol{\theta} - \boldsymbol{\theta}_r) \tag{4.2}$$

From (3.3) and (4.2) we obtain

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^+\dot{\boldsymbol{r}}^* + (\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{H}_2(\boldsymbol{\theta} - \boldsymbol{\theta}_r)k_1$$

$\dot{r}^*$ is then replaced by a modified desired velocity $\dot{r}_M^*$ given by

$$\dot{\boldsymbol{r}}_M^* = \dot{\boldsymbol{r}}^* - \boldsymbol{H}_1(\boldsymbol{r} - \boldsymbol{r}^*) \tag{4.3}$$

where

$$\boldsymbol{H}_1 = diag(h_{1i}) \in \boldsymbol{R}^{n\times n} \text{ and } h_{1i} > 0$$

Equation (4.3) is used to cope with any error in the first subtask using the error feedback $H_1(\boldsymbol{r} - \boldsymbol{r}^*)$. Therefore, $\dot{\boldsymbol{r}}_M^*$ replaces $\dot{\boldsymbol{r}}^*$ in the equations (4.2) and (4.3)

2. Obstacle Avoidance on Baxter robot

The control law in the previous section was applied to the Baxter robot. From the initial state $\boldsymbol{\theta}$, the end-effector was given a commanded velocity to move along a desired trajectory. An obstacle was placed in the way and without control law the arm would collide with it. We then taught the arm the posture $\boldsymbol{\theta}_r$ at the same end-effector position where the robot arm hits the object. $\boldsymbol{\theta}_r$ was set in a way that it would not collide with the obstacle. The control law allowed the Baxter arm to come to the same posture while keeping the desire trajectory and was able to avoid the obstacle.

Figure 3.2 and 3.3 shows the two test cases in which the robot arm moved along the Y-axis and Z-axis of the body frame. In both cases, without the control algorithm the arm would collide with the black ball. The Baxter robot avoided the obstacle with the control algorithm while keeping the same trajectory (which was the line in blue).

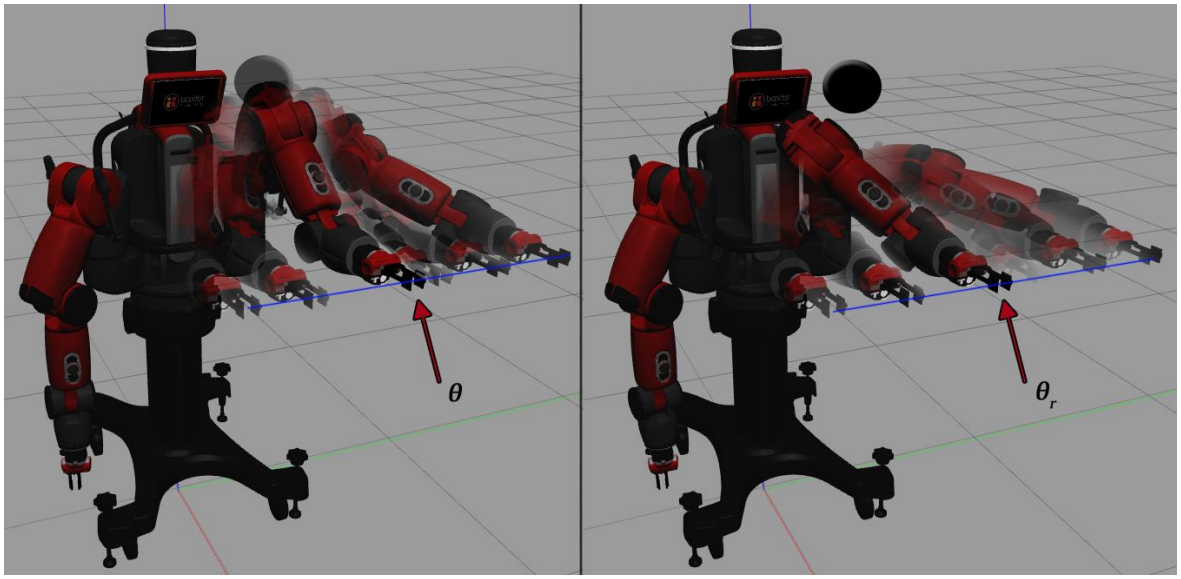

Figure 3.2: Obstacle avoidance – uncontrolled (left) versus controlled (right)

Figure 3.3: Obstacle avoidance – uncontrolled (top) versus controlled (bottom)

CHAPTER V:

CONCLUSION

Kinematic redundancy plays an important part in robot design and planning. A manipulator is termed kinematically redundant when it has more degrees of freedom than it is strictly needed to execute a given task. The Baxter research robot has two manipulator arms, each arm possesses seven degrees of freedom. Baxter arm is kinematically redundant for the task of moving the end-effector which requires six degrees of freedom. The redundancy can be used to achieve additional goals.

In this research, some fundamental aspects of Robotics were studied. The relationship between manipulator links can be showed by using Denavit-Hartenberg convention. The homogeneous transformation matrix between two links can be calculated using the DH parameter table. The Jacobian matrix relates the joint velocity and the end-effector velocity is computed using the geometric formula. The Cartesian Velocity Control mode was developed for the Baxter robot using its Joint Velocity arm control mode.

The focus of this research was the kinematic redundancy resolution of Baxter robot. The manipulability, which is the ability of the end-effector to move from a certain position and orientation, was investigated for Baxter. Different joint configurations of the same end-effector position and orientation were shown to have different volumes of manipulability ellipsoid.

A manipulator arm can perform several subtasks with different priorities. In this thesis, the redundancy was used to perform additional subtasks after completing the first subtask of moving the end-effector along a desired trajectory. The second subtask can be the task to avoid the singularity while trying to keep the volume of manipulability as large as possible. The control algorithm was applied on Baxter robot to prevent the

manipulability ellipsoid volume of Baxter's end-effector from getting too small, therefore it helps to avoid the singularity. The algorithm also helps Baxter's end-effector to move away quickly from the singularity.

Obstacle avoidance was also performed as a second subtask in this thesis. Baxter robot came with a function which allows it to stop the movement when it detects a collision. However, Baxter cannot avoid the object without a control method. In this work, a control algorithm was used to help Baxter to avoid the obstacle while tracking the desired trajectory. Baxter would try to come as close as possible to a pre-taught arm posture so that it would not collide with the obstacle while staying on the given trajectory of its end-effector.

REFERENCES

[1] M. W. Spong, S. Hutchinson and M. Vidyasagar, Robot Modeling and Control, 2nd ed., Wiley, 2020.

[2] S. Niku, Introduction to Robotics: Analysis, Control, Applications, 3rd ed., Wiley, 2019.

[3] T. Yoshikawa, Foundations of Robotics: Analysis and Control, MIT Press, 1990.

[4] J. J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed., Pearson Education, 2005.

[5] B. Sicialano and O. Khatib, Springer Handbook of Robotics, Springer, 2016.

[6] T. Yoshikawa, in *Foundations of Robotics: Analysis and Control*, MIT Press, 1990, pp. 45-53.

[7] J. J. Craig, in *Introduction to Robotics: Mechanics and Control*, 3rd ed., Pearson Education, 2005, pp. 101-128.

[8] A. Somasundar and G. Yedukondalu, "Robotic Path Planning and Simulation by Jacobian Inverse for Industrial Applications," *Procedia Computer Science*, vol. 133, pp. 338-347, 2018.

[9] T. Yoshikawa, in *Foundations of Robotics: Analysis and Control*, MIT Press, 1990, pp. 53-66.

[10] C. A. Klein and C.-H. Huang, "Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, Vols. SMC-13, no. 2, pp. 245-250, 1983.

[11] L. Hou and L. Baron, "The Joint-limits and Singularity Avoidance in Robotic Welding," *Industrial Robot: An International Journal,* vol. 35, 2008.

[12] B. Noble and J. Daniel, Applied Linear Algebra, 2nd ed., Prentice-Hall, Inc., 1977.

[13] G. Strang, "The Singular Value Decomposition (SVD)," in *Introduction to Linear Algebra*, 5th ed., Wellesley-Cambridge Press, 2016.

[14] T. Yoshikawa, "Analysis and Control of Robot Manipulators with Redundancy," *Robotics Reseach The First International Symsosium,* pp. 735-747, 1984.

[15] F.-T. Cheng, J.-S. Chen and F.-C. Kung, "Study and Resolution of Singularities for a 7-DOF Redundant Manipulator," *IEEE Transactions on Industrial Electronics,* vol. 45, no. 3, pp. 469-480, 1998.

[16] T. Yoshikawa, "Analysis and Control of Robot Manipulators with Redundancy," *Robotics Research The First International Symsosium,* pp. 741-742, 1984.

[17] D. L. Pieper, *The Kinematics of Manipulators Under Computer Control,* Stanford University, 1968, pp. 19-64.

[18] R. L. Williams II, "Baxter Humanoid Robot Kinematics," in *Internet Publication*, 2017.

[19] H. Hanafusa, T. Yoshikawa and Y. Nakamura, "Analysis and Control of Articulated Robot Arms with Redundancy," *8th IFAC World Congress on Control Science and Technology for the Progress of Society,* vol. 14, no. 2, pp. 1930-1931, 1981.