©Copyright 2013
Andrew Lewis

Dynamically Evaluated Gravity Compensation for the RAVEN Surgical Robot

Andrew Lewis

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Engineering

University of Washington

2013

Committee:

Blake Hannaford, Chair

Duane Storti

Brian Polagye

Program Authorized to Offer Degree:
Mechanical Engineering

University of Washington

Abstract

Dynamically Evaluated Gravity Compensation
for the RAVEN Surgical Robot

Andrew Lewis

Chair of the Supervisory Committee:
Professor Blake Hannaford
Electrical Engineering

Using an accelerometer on the base of a robot, it is possible to calculate the torque required from each actuator in order to maintain a known pose regardless of base orientation with respect to the direction or magnitude of gravity. A simple and novel method has been developed and implemented for overcoming gravity induced torques on the RAVEN™[1] surgical research robot. This innovation will allow for accurate control of serial robot manipulators with re-orientable bases or for those operating in non-stationary environments such as boats, space stations, or moving vehicles.

[1]The RAVEN is a registered trademark of the University of Washington and Applied Dexterity

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

END EFFECTOR: The manipulator or tool of a robot, usually located at the end of a serial robot; the "business end" of a robot.

THE RAVEN: A surgical research robot designed at the University of Washington and University of California Santa Cruz. The RAVEN is currently manufactured and sold by Applied Dexterity.

PID CONTROLLER: A common control method that determines motor torques from the distance between the current and desired positions and the derivatives of this distance.

DEGREE OF FREEDOM: An axis about which or along which a body can rotate or translate.

TELEOPERATION: The control of a robot or manipulator from a distance.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to his advisor, Blake Hannaford, and his labmates for making this project possible. Sincere gratitude is extended to Hawkeye King for his invaluable help and guidance and to Alicia Clark for all of her support and late night snacks and meals.

DEDICATION

to my family, friends and coworkers that have inspired and encouraged me

Chapter 1

INTRODUCTION

1.1 Motivation

One of the most lauded and world-changing applications for surgical robots is their potential for use in remote locations without specialized surgeons [2]. One can easily imagine the utility of a highly automated or teleoperated mobile operating room in underdeveloped areas, battlefields or areas of natural disaster. In such cases, these operating rooms may be on a rocking hospital boat or in a large trailer on uneven terrain. These dynamic and unpredictable applications introduce challenges in the low-level control of surgical robots. This project aims to provide a solution to the control of a surgical robot with an unknown or changing orientation with respect to gravity.

Gravitational loads on robotic manipulators often present a large, non-linear contribution to static and dynamic control. In many cases, this load can be calculated and compensated for when the positions of the robot links are known with respect to a base frame. Traditionally this base frame is stationary with respect to gravity, such as an industrial robot bolted to a factory floor or a monolithic surgical robot with a heavy, fixed base column. This project aims to provide similar compensation in cases where the base cannot be assumed to be in a fixed orientation with respect to gravity; in such cases as a robotic arm attached to a mobile reconnaissance platform or a surgical robot on an aircraft carrier. It is proposed that actively evaluated gravity vector compensation will greatly improve the control performance of a serial robotic manipulator with a non-stationary base.

1.2 Literature Review

1.2.1 RAVEN Surgical Research Robot

The RAVEN surgical research robot is an open source system developed at the University of Washington and the University of California Santa Cruz as a means of researching and

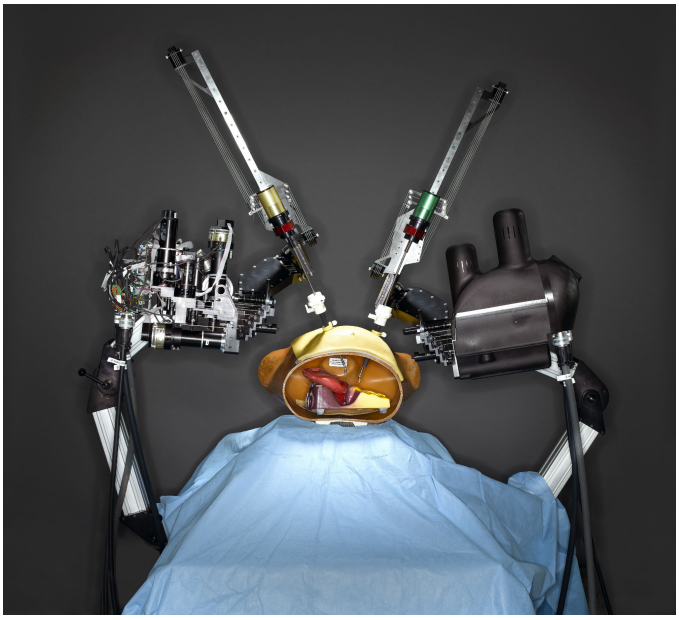

Figure 1.1: The RAVEN I surgical research robot developed at the University of Washington BioRobotics lab.

innovating in the field of surgical robotics. After developing and rigorously testing the limits RAVEN I [10], seen in Figure 1.1, development of RAVEN II began as a common research platform for research labs across the US. In 2012, the first RAVEN II systems were shipped to seven top research institutions as the inaugural members of the RAVEN community [3]. With a common platform built on software standards, the RAVEN community is well suited to collaborating and innovating in the space of surgical robotics and related fields.

The RAVEN II is a compact, cable-driven mechanism with 3 spherical joints and a 4 Degree of Freedom tool. The spherical joints mechanically constrain robot motion to a virtual center, which is a desirable characteristic for minimally invasive surgical robots since they must enter the body through a key-hole incision. Figure 1.2 shows the RAVEN II arms and controller.

Control of the RAVEN II is implemented on a standalone, rack-mounted system consisting of a power supply with safety logic protection, a Linux computer box, and two motor control boxes with custom USB 2.0 interfaces. The Linux computer uses a real-time kernel

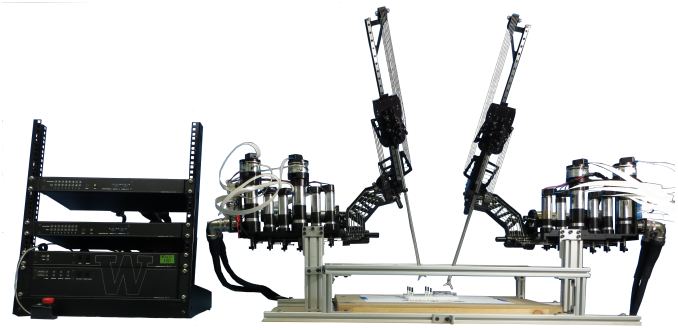

Figure 1.2: The RAVEN II surgical research robot system developed at the University of Washington and University of California Santa Cruz. On the left are the control electronics and emergency stop switch. In this photo, the robot is set up to perform a modified Fitt's law trial to test shared autonomy algorithms.

communications layer to communicate with the USB motor controllers in order to maintain a control loop of 1000 Hz, which is crucial for high fidelity haptic interfaces and control. Low- and high-level control are written in C++ and operate within the Robot Operating System (ROS) framework, which allows for easy communication and interfacing between the robot and other ROS modules. The ROS framework and open source RAVEN software allow for users to modify their own software and easily interface with other ROS tools that are being being developed over a multitude of fields in robotics, such as computer vision and mapping. Teleoperation is achieved via a custom UDP packet interface that was designed for interoperability between master and slave stations [7].

1.2.2 Control with Gravity Compensation

The most common form of low-level control in robotics applications is the Proportional Integral Derivative (PID) controller or some combination of those terms. This family of controllers determine motor torques using the error between the robot's current position and its desired position as the input. Since the early 1990s, it has been known that PID and PD controllers with a gravity compensation term are stable about a reference position for any rigid manipulator [15]. However, it has not yet been proven that PID controllers

can be globally stable in robots with elastic joints. Tomei has shown that PD control with a gravity compensated term at the desired configuration is enough to prove global stability about this reference position. In his proof Tomei notes that since the effects of elasticity and gravitational loading are present in the control law, the stability of the controller may be affected by uncertainties in the modeling of these terms. However, the practical effect of these uncertainties presents itself only in minor steady-state position errors. Therefore accurate knowledge of gravitational effects is necessary for accurate positioning of a robot with elastic joints.

However, using a PID controller for positional control is most effective in linear applications, in which moving an end effector a defined distance requires a relatively constant amount of effort regardless of the initial position of the robot [11]. An example of such a linear application is a 2 Degree of Freedom (DOF) 3D-printer head that requires a relatively equal amount of effort to move 3 inches in any direction. However, a standard industrial robotic arm presents nonlinearity because the moment about the shoulder due to gravity changes in a sinusoidal function characterized by a maximum torque requirement when held out straight and nearly zero torque to hold it straight down. due to the aiding effects of gravity in the latter case. The geometry of the RAVEN and its potentially variable base orientation make positional control similarly non-linear, but in a more complicated manner to compensate for. The explanation and analysis of the effects of gravity on the RAVEN will be presented in the Analysis chapter.

Another challenge to controlling robots like the RAVEN is the elasticity between the joints due to the use of cables, with some cables running around the entire length of the serial links. Alessandro De Luca et al. has contributed a significant amount to the theory of controlling elastically jointed robots. In [9], he points out that exotic, complex control solutions are available for tracking tasks such as linearizing and decoupling nonlinear feedback, an integral manifold approach based on perturbation of the dynamic model, or adaptive, neural network control. He continues to survey the PD plus gravity compensation approaches since [15] first proved that the gravity compensation of a desired position was sufficient for regulation task stability, with the gravity term $g(q_d)$ where q is the robot pose and q_d is the desired robot pose. This approach is termed a constant gravity compensation

term and requires sufficiently high positional PD gains. A nonlinear gravity compensation scheme requires evaluating the required gravity torques for the current position of the robot during its movement, $g(q)$. Alternatively, the gravity term can be replaced with a saturated integral term. Finally, gravity compensation can take the form of an on-line gravity compensation term with torques calculated from the motor positions when it is impractical to sense joint torques, $g(\theta)$. With elasticity between the motors and the joints, on-line gravity compensation will naturally lead to steady state errors because the actual pose of the links is not compensated for. The contribution of [9] is a PD control law with gravity compensation calculated with a new variable $g(\tilde{\theta})$, a gravity-biased motor position that uses a model of joint elasticity to provide a more accurate description of the robot pose. This technique is shown to have smoother control transients in the motor torques and provides a larger range of feasible PD gains over the PD controller with constant gravity compensation proposed in [15].

1.2.3 State of the Art in Gravity Compensation

In the mid 1990s, De Luca, Ma, Hollerbach, and others began investigating iterative methods for determining robot characteristics using the effects of gravity induced torques. In [8], De Luca presents a scheme that learns and updates an online gravity compensation term which has proven stability for rigid robots and robots with flexible links or elastic joints. Ma and Hollerbach present a method in [12] for calibrating torque sensor gains and position offsets at all but the last joints without explicit knowledge of the gravity direction in the base frame. The base gravity vector is determined from the amplitude and phase information of stationary motor torques at automatically coordinated positions of the first two joints. These techniques are then used in [11] to identify mass parameters for gravity compensation. The authors assert that accurate knowledge of link masses and locations of centers of mass are often unknown even to robot manufacturers, thus it is useful to be able to experimentally determine mass parameters without disassembling the robot. Using a model similar to the approach presented in the next chapter, unknown mass parameters are determined at the distal link with now-known gravity parameters and an exploratory phase with the calibrated

torque sensor. Variables that are unable to be solved for are solved recursively by repeating the process towards the most proximal link. These methods were validated in simulations and on a 7 DOF Sarcos Dextrous Arm Slave system.

1.2.4 Dynamics of Robots on Moving Bases

The Mitsubishi PA10-6CE has been used in several investigations into the dynamics of robotic manipulators on moving bases. A model was derived for arbitrary 6-DOF disturbances and implemented for the robot by Wronka and Dunnigan with a 2 DOF rotary base [16]. The derivation assumes that the characteristics of the moving platform are known and it is assumed that the motion of the manipulator has no significant influence on the motion of the base. The Lagrange-Euler method is used for the full dynamic model. This technique uses the total energy of the system and inherently includes a gravity term in the calculation of potential energy. The main goal of the work was to develop an accurate model of the robot for simulation purposes and they found that the model is accurate and the inclusion of a gravity term in the model significantly improves its performance for use in future work with the robot. Sadraei and Moghaddam derive both a Lagrange-Euler model and a Newton-Euler model of the PA10-6CE and simulate the rocking of a ship to test computational performance of the models [14].

In 1999, Iagnemma et al. constructed a low-cost model similar to the JPL Lightweight Survivable Rover with a 6-wheeled rocker-bogie system and a 3 DOF manipulation arm with several end effectors [4]. Global torques on the manipulator were measured with a 6-DOF force-torque sensor at the base of the arm and a gravity compensation term was used to subtract the contribution of gravity from the reading. Instead of using gravity compensation as a term for motor torque output, it is used as a term for compensating torque feedback. It is mentioned that the gravity term is dependent on the orientation of the base, which is measured by an accelerometer.

1.2.5 Surgical Robots, Teleoperation, and Haptic Devices

A recent study investigated the effects of gravity compensation of a haptic master controller under external accelerations [7]. In order to compensate for the biodynamic feedthrough of the limbs of the user, acceleration dependent damping coefficients and mass parameters were used in the control of the Force Dimension omega.7 haptic controllers. Preliminary results from experiments in an elevator and moving van showed increased user performance over constant damping. The tests culminated in a microgravity and variable gravity experiment in which surgeons were asked to perform suturing tasks with the devices with a teleoperated SRI international M7 robot while on board the NASA C-9 airborne parabolic laboratory. Results showed that the variable acceleration compensation improved performance of the user during constant and variable gravity, but had did not appear to help in zero g.

1.3 Project Goals

In the reviewed literature, only a few studies have mentioned the effects of changing base orientation and none have evaluated the use of sensors to study or compensate for these effects. This project proposes to integrate sensor data into the calculation of gravity compensation torques and use this term to implement a PD + gravity compensation controller. Moreover, this project will study the performance of a controller that is able to respond to sensed relative gravity and acceleration data, deemed a PD + dynamic gravity compensation controller, or PD + DG for short. This control and a standard static gravity vector compensation controller (PD + SG) will be implemented on the RAVEN surgical robot for controller performance evaluation. The chapters that follow describe the analysis of the problem, approach to the solution, and a discussion of the results.

1.4 Contributions

The contributions of this project are as follows:

- A simple and novel method for calculating externally induced torques from measured gravity and external accelerations. This method uses a dynamic gravity compensation term, $g(\theta, \bar{g})$.

- A practical gravity compensation method already implemented on several RAVEN surgical research systems at sites across the world.

- A prototype system for easily integrating digital and analog sensors with the RAVEN system.

Chapter 2

ANALYSIS

2.1 General Gravity Compensation Problem

In its simplest form, gravity compensation is the calculation and correction for gravity-induced torques on each robot joint. The general solution proposed in this chapter uses commonly used robot kinematic conventions and a simple frame and point-mass technique for determining gravity torques.

2.1.1 Description of Robot Kinematics

The pose of a robot with rigid links is generally described with a series of coordinate frames attached to each independent joint. With the frames fixed in relationship to their respective joints, the relationships between each frame depend only upon joint parameters: either rotary joint angles or prismatic joint distances. Therefore, with adequate sensing it is possible to know not only the positions of the robot links, but also the transformations between each frame. Frame transformations can be used to represent the known or sensed location of a point in one frame in another frame without needing to re-measure or incorporate another sensor into the system. For instance, frame transformations can be used to locate an object in a robot's base frame when the object's position is sensed by a camera with a known relationship to the robot base, such as in the system configuration in Figure 2.1. In much the same manner, the location of one link's center of mass can be represented in the frame of another link with the use of the appropriate known transformation matrices.

As a means of standardizing the description of serial linkages, the researchers Denavit and Hartenberg developed a methodology for frame and joint variable assignment called DenavitHartenberg parameters, or simply DH parameters. Their method of frame assignment has been in common use in robotics since the 1980's and allows for quickly calculating any transformation between each joint as long as the parameters are correctly determined and

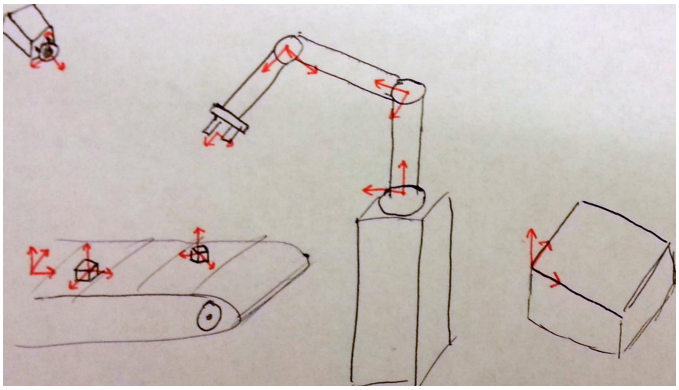

Figure 2.1: An example of a robotic system with various related frames. Performing a pick and place task such as the one depicted, requires knowing the transformations between each frame.

the joint variables are adequately sensed. This standard technique allows us to determine the relative locations of each joint's center of mass in relationship to each other joint.

2.1.2 Derivation of General Gravity Compensation

In this section we will develop a novel, extensible, and general method to compute the torques at each robot joint due to gravity.

First, consider a simplified situation in which a rigid shaft in an arbitrary direction supports a point mass at a distance (Figure 2.2). Suppose a point mass, having mass, m, and located at point P, is rigidly connected to the shaft but offset at some distance from the shaft and subject to the force of gravity, mg.

Assume a frame, $\{1\}$, exists whose origin is on the shaft, and in which Z_1 is pointing along the shaft. The moment generated about the origin of frame 1 is

$$^1\tau = P \times F = m(^1P \times ^1 g)$$

$$^1\tau = \begin{bmatrix} 0 & -Pz & P_y \\ P_z & 0 & -P_x \\ -P_y & P_x & 0 \end{bmatrix} \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = \begin{bmatrix} P_y g_z - P_z g_y \\ P_z g_x - P_x g_z \\ P_x g_y - P_y g_x \end{bmatrix}$$

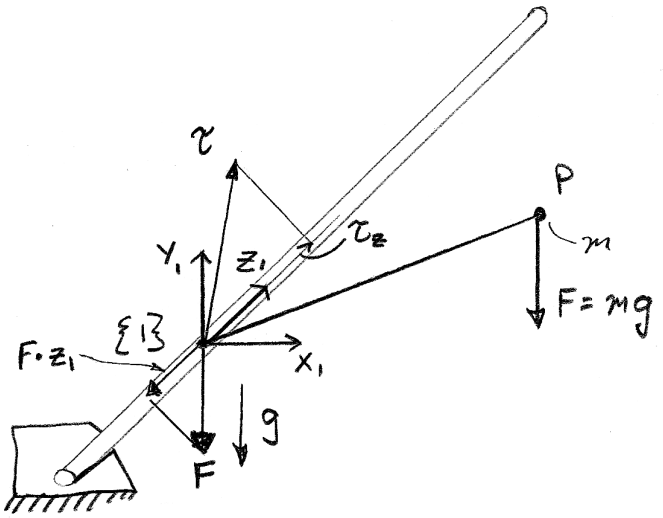

Figure 2.2: A shaft with an off-axis mass generating a torque and a force on the shaft.

When we apply this to robot links, frame $\{1\}$ will be the link frame and Z_1 will be the axis of rotation or translation. Thus the torque we are interested in is the z component of τ

$$\begin{bmatrix} 0 \\ 0 \\ P_x g_y - P_y g_x \end{bmatrix}$$

Now consider the force, $F = mg$ pointing "down". Besides the torque, the force, F, can also be represented in frame $\{1\}$, and it has a projection onto the Z_1 axis:

$$F_z = F \cdot Z_1$$

Applying this to a robot, we assume that the robot is not moving and that the joints are in some pose, θ.

Each link has mass, m_i located at its center of mass: C_i (Figure 2.3). In link i, we wish to compute the z component of torque due to the gravitational force on each center of mass which is more distal to the joint (has a greater subscript, i). The moment on frame i due to the mass in frame j is therefore

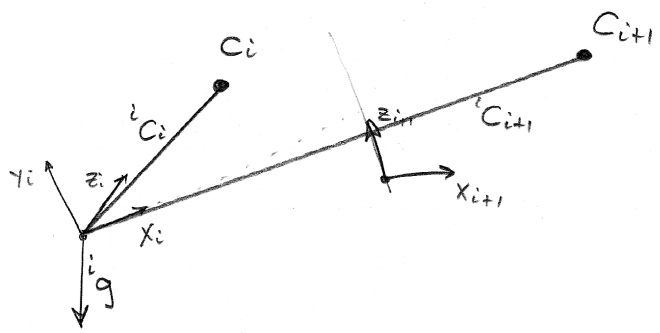

Figure 2.3: Two robot link frames (i, $i+1$) with centers of mass (C_i) and the gravity vector (g) shown for computation of gravity torques.

$$\tau_{ij} = C_j \times m_j g = m_j(C_j \times g)$$

where we assume that all quantities are represented in the same frame. For joint i, the natural frame is frame $\{i\}$:

$$\begin{aligned} {}^i\tau &= \sum_{j=i,n} m_j({}^iC_j \times {}^ig) \\ &= \left[\sum_{j=i,n} m_j {}^iC_j\right] \times {}^ig \end{aligned}$$

The quantity

$$C_{mi} = \sum_{j=i,n} m_j C_j$$

is called the *first moment of mass* which is the center of mass for the entire part of the manipulator distal to joint i and it can be expressed in any frame in general.

The torque in the i'th joint, τ_i is therefore:

$$\tau_i = \hat{z} \cdot {}^i\tau = [0 \quad 0 \quad 1]\, {}^i\tau$$

for rotary joints.

For prismatic joints, we have the analogous equation

$$F_{ij} = \sum_{j=i,n} m_j {}^i g$$

where m_j is the magnitude of the mass of link j and the joint force is just the z component of F_{ij}.

$$F_{Zi} = F_{ij} \cdot [0 \quad 0 \quad 1] = [0 \quad 0 \quad 1]^T F_{ij}$$

2.1.3 Summary

The following is a summary of the method to calculate joint torques due to gravity for rotary joint i. An example of implementation is described in the proceeding chapter.

1. Transform and sum all center of mass locations of following links into frame i to find first moment of mass in the i frame.

$$^i C_{mi} = \sum_{j=i,n} m_j {}^i C_j$$

2. Take the cross product of this vector and gravity in i.

$$^i \tau = C_{mi} \times {}^i g$$

3. Find the projection onto the z-axis. This is the torque on the ith joint due to gravity. The dynamic gravity compensation term, $g(\theta, \bar{g})$, output by the controller is the opposite of the torque induced by gravity.

$$\tau_i = [0 \quad 0 \quad 1] \, {}^i \tau$$

$$g_i(\theta, g_i) = -\tau_i$$

This should be repeated for all joints where gravity loading is a significant term in control. This is not the case, for example, in the four tool joints of the RAVEN since the cable friction is much higher than the gravity loading of the relatively small grasper components.

2.2 The RAVEN II Robot

2.2.1 Kinematics

The derivation of the RAVEN II DH parameters, forward kinematics and inverse kinematics are detailed in [6]. Figure 2.4 shows the base frames and the frames of the first 3 gross positioning joints obtained with the Craig convention for DH frame assignment. The left and right arm DH parameters are shown in Tables 2.1 and 2.2 from [6]. The variables are defined as such: α_{i-1} is the angle between z-axes of $i-1$ and i about x_{i-1}; a_{i-1} is the distance between z-axes about x_{i-1}; d_i is the distance between the origins of $i-1$ and i measured along z_i; θ_i is the angle between the x-axes of $i-1$ and i measured about z_i. These parameters fully define a transformation between two successive frames defined within the conventions described by Denavit and Hartenberg.

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	La_{12}	0	0	θ_2
3	$\pi - La_{23}$	0	d_3	$\pi/2$

Table 2.1: Left (Gold) Arm parameters for the positioning joints.

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	π	0	0	θ_1
2	La_{12}	0	0	θ_2
3	La_{23}	0	d_3	$-\pi/2$

Table 2.2: Right (Green) Arm parameters for the positioning joints.

These DH parameters and corresponding serial transformation matrices from the base frame to the third line are listed below. In the matrices below let $s_n = sin(\theta_n), c_n =$

$cos(\theta_n), k_1 = cos(La_{12}), k_2 = sin(La_{12}), k_3 = cos(La_{23})$, and $k_4 = sin(La_{23})$.

$$ {}^{0}_{1}T_{L/R} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1/-s_1 & c_1/-c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1} $$

$$ {}^{1}_{2}T = \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ s_2k_1 & c_2k_1 & -k_2 & 0 \\ s_2k_2 & c_2k_2 & k_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2} $$

$$ {}^{2}_{3}T_{L/R} = \begin{bmatrix} 0 & -1/1 & 0 & 0 \\ k_3 & 0 & -k_4 & -k_4d_3 \\ (-1/1)k_4 & 0 & (-1/1)k_3 & (-1/1)k_3d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3} $$

These equations are used in the RAVEN control code to calculate the position of the robot at every time step. The resulting matrices are available to be used in other parts of the robot code as well.

2.2.2 Center of Mass Locations

The most reasonable method for determining the approximate locations for the Centers of Mass (COM) of each link was to use the CAD models of each link. First the frames needed to be added to each link to match those in [6]. Physical measurements of the COM locations was deemed unfeasible since the links consist of complicated geometry and have few virtual frame locations. Therefore, the mass properties were analyzed by the CAD software to determine the location of the COM of the assembly in the newly defined reference frame. The results for the first two link models are shown in Figures 2.5 and 2.6. The third frame is defined at the very center of the tool head, and since the majority of the mass are the carriage, interface and toolhead, it was assumed that the center of mass would be close enough to this point as to not cause significant errors in calculations. The software model

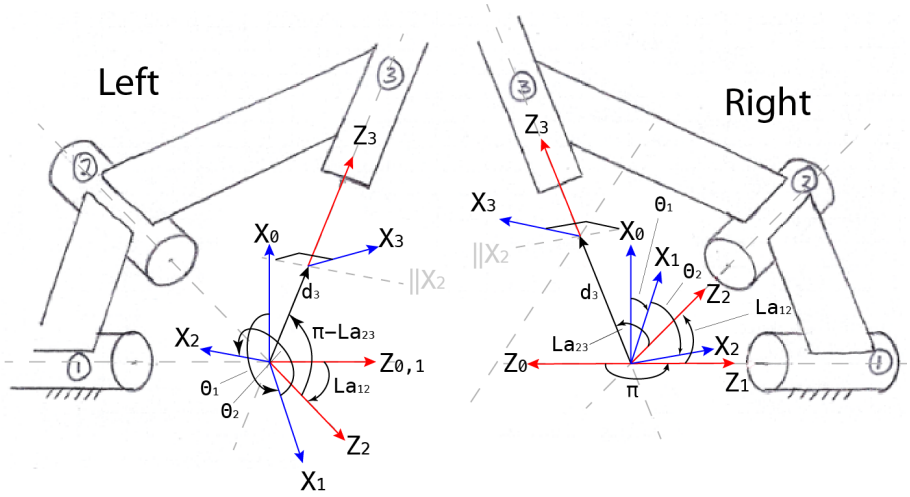

Figure 2.4: The base and gross positioning frames of the RAVEN II robot as described in [6].

Link i	Mass (g)	Left Arm Location in i (m)	Right Arm Location in i (m)
1	494	[0.007, 0.097, -0.138]	[-0.007, -0.097, 0.138]
2	750	[-0.002, -0.183, -0.231]	[0.002, -0.183, 0.231]
3	231	[0 , 0, 0]	[0, 0, 0]

Table 2.3: Masses and locations of link COMs in their respective frames.

of the COM locations is subject to some error because it assumes that all of the included components are of uniform density. This could lead to some errors, but the locations appear to be correct upon visual inspection.

Link masses were determined empirically from measurements taken on links that were under construction for the latest build of RAVENs. Included in the measurements are nearly all of the components of each link. Very little was changed for the second RAVEN II build, and so it can be reasonably assumed that the link masses are accurate. The link masses are shown in Table 2.3.

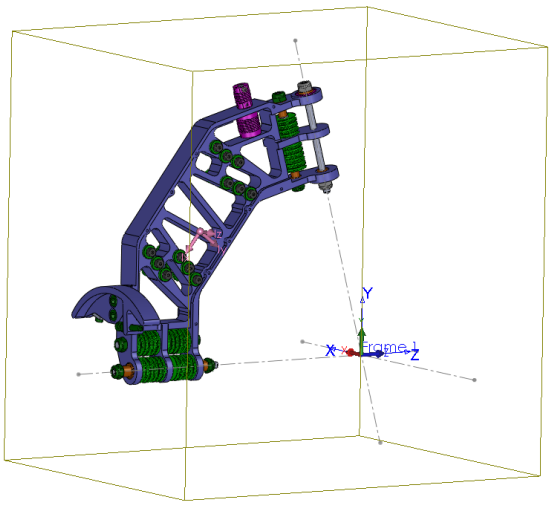

Figure 2.5: Link 1 with frame and Center of Mass (pink frame) as determined from the CAD assembly model.

2.3 Sensor Interfaces to the RAVEN System

In order to implement dynamic gravity compensation, the direction and magnitude of gravity relative to the base must be sensed and communicated to the robot control software. Since the only sensing capabilities of the RAVEN relate to motor positions and currents, a new sensor must be added to the system and a new sensor interface must be implemented. The available hardware input interfaces to the robot control computer are: ethernet, USB, parallel port, microphone port or ps2 port. Each of these options would require a custom hardware and software solution to interface directly with an embedded sensor. This leads to a solution that uses a secondary processor capable of communication with both the embedded sensor and Ubuntu control computer via hardware inputs or a network connection. Possible network interfaces include: hosting a regularly updated html website with sensor data; opening a UDP socket to communicate sensor data; or using Robotic Operating System (ROS) functionality to constantly publish sensor messages or host a client service to respond to queries with sensor data. ROS is particularly adept at integrating dispersed and networked computers into one robotic system.

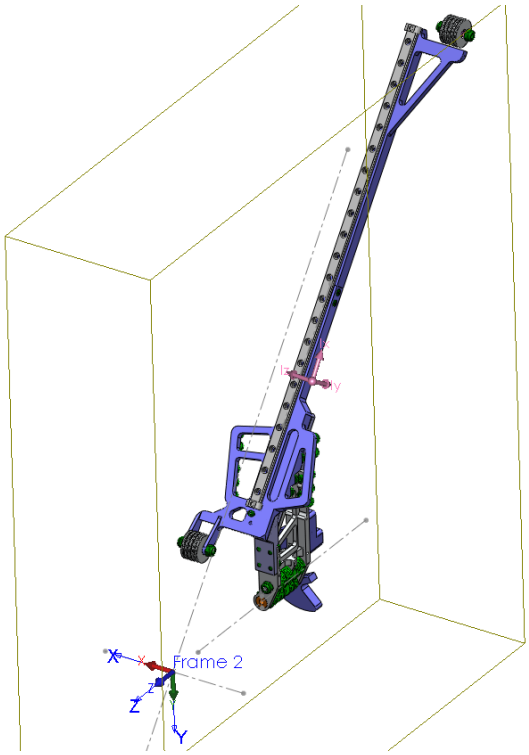

Figure 2.6: Link 2 with frame and Center of Mass (pink frame) as determined from the CAD assembly model.

2.4 Summary

A novel method of gravity compensation calculations has been derived. This method is applicable to any serial manipulator and uses standard DH conventions to simplify calculations and utilize kinematics information that is already used in most robot controllers. The generalities of this method allow for any of the quantities to be time-varying, including the locations of the link COMs, the pose of the robot, and most interestingly, the direction and magnitude of the relative acceleration due to gravity. Since the acceleration due to gravity is indistinguishable from external accelerations, this method could be more accurately described as a general method for compensation of base accelerations. When this method is used for gravity compensation with real-time, sensed acceleration vectors, it is termed a dynamic gravity compensation term.

Mass parameters for the RAVEN II robot have been analyzed and described in a way that facilitates implementation of the general method for compensation of base accelerations on the RAVEN.

Possible interfaces with the RAVEN system have also been briefly discussed.

Chapter 3

METHODS

3.1 Hypothesis

This project proposes that the use of a dynamic gravity compensation term in addition to the PD controller already implemented on the RAVEN II robot will improve robot performance in regulation and low speed tracking at any base angle.

3.2 Proposed Gravity Compensation System Architecture

Figure 3.1 provides a high level view of the proposed sensor integration. The acceleration sensor (accelerometer) communicates with a small computer or microprocessor that communicates over a network to the ROS master process. ROS will then pass the sensor information into the control software which will use it in its calculation of the dynamic gravity compensation. This term will be added to the PD control torque to directly control the robot which feeds back sensor information from the motor current and position.

It was decided to use the ROS networking capabilities for this project in order to explore and implement proof-of-concept designs for adding networked sensors to the RAVEN system. The RAVEN community is very interested in further integrating the RAVEN system with the capabilities of ROS.

3.2.1 Sensing

Initial Prototype

The sensing subsystem was the first to be analyzed and implemented. After exploring the available sensors, microprocessors and embedded computers, the first prototype was built using a Bosch BMA180 three-axis accelerometer breakout board from Sparkfun Electronics (sparkfun.com) and an Ethernet Arduino microprocessor. The components were chosen

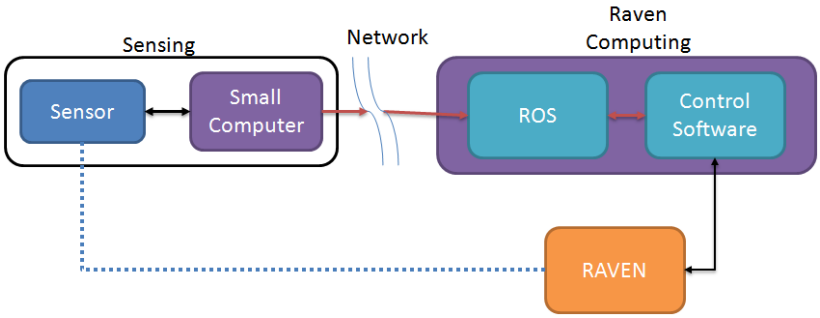

Figure 3.1: High level architecture of proposed sensor integration system.

because of their low cost, easy access to support from the hobbyist community, and ease of use.

The BMA180 accelerometer has a wide range of measurable acceleration ranges from $\pm 1g$ to $\pm 16g$ with varying degrees of precision. An I^2C digital serial communication protocol is used for communication with the chip, which makes it possible to use multiple sensors on the same serial bus. Onboard signal processing includes configurable filters, slope sensing, and tap detection. While it was not envisioned for these processing features to be used in the gravity compensation system, they could be easily integrated into the robot software using the final sensing system developed for this project.

An Arduino was chosen over other microprocessor kits(TI MSP430 LaunchPad, TI Stellaris LaunchPad) or embedded computers (Raspberry Pi, Beagle Bone, Gumstix) because the Arduino family has a large community of active hobbyists and an extensive amount of freely available example code. In fact, the Sparkfun product description site for the BMA180 breakout board includes a link to example code for using the chip with an Arduino. In addition, the Power over Ethernet (PoE) was purchased with the Ethernet Arduino so that the system could be easily implemented with only one cable for power and communications. The ethernet capabilities of the Arduino were considered advantageous because it could be simple to set up and use from anywhere with an internet connection. The ethernet capabilities also allowed for both traditional internet data transfer and ROS integration.

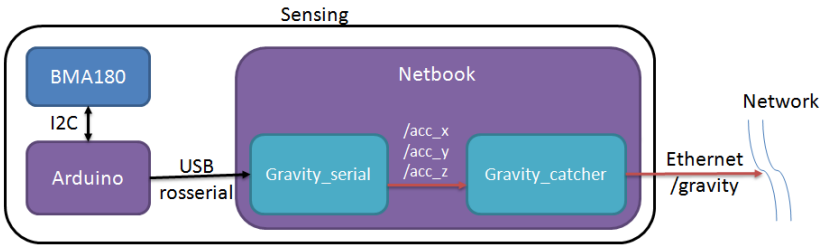

Figure 3.2: The second and most recent iteration of the sensing subsystem.

Second Prototype

However, after attempting to develop software on the Arduino to publish sensor data on the internet, it was discovered that the security settings of the network in the EE building were not conducive to allowing microprocessors to be communicated with. Therefore a second solution was developed with a more capable computer in the form of a Lenovo Ideapad netbook. The netbook computer was already loaded with Ubuntu, the most common OS for running ROS. Once ROS was installed, the `rosserial` package was installed to allow easy communication with the Arduino via the netbook's USB port. The system diagram for the sensing subsystem can be seen in Figure 3.2.

The Arduino's program initializes and retrieves acceleration data from the BMA180 at 100 Hz. The `rosserial` package allows for interfacing serial connections in a ROS node; it provides a library for using ROS commands and constructs in the Arduino code. The data from the sensor is directly placed into ROS messages and published at 100 Hz on board the Arduino. However, a node from the `rosserial` package, `serial_node`, must be running on the netbook in order for the messages to be available to other nodes, such as the RAVEN control node. Notably, the `rosserial` package does not support array messages, so a separate node was written to subscribe to the acceleration messages published by the Arduino and republish them as a message structure. The structure is a custom ROS message with two fields: an array with acceleration components, and a magnitude field that is calculated in the node. This republisher node is called `ard_ catcher`.

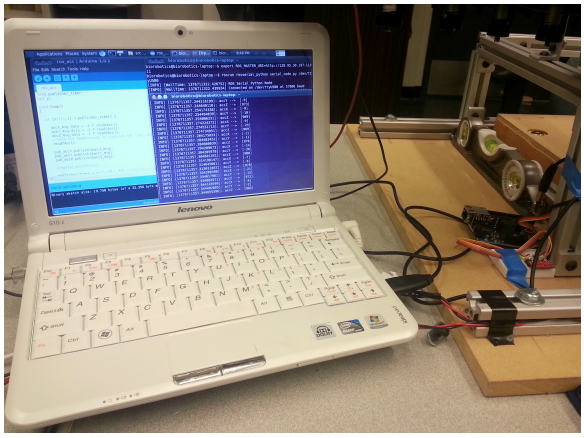

Figure 3.3: The netbook, ehernet arduino (right, dark blue), and accelerometer (rght, red) used in final system architecture. On the netbook screen are the Arduino development environment and the terminals necessary for retrieving ta from the arduino and for repackaging and filtering the data.

3.2.2 Computing

The RAVEN II software, implemented in C++, had files and a structure in place for gravity compensation as an unimplemented feature that was carried over from the RAVEN I code. The structure of this was difficult to read and understand in addition to being inflexible to changes in the direction of gravity or the kinematics of the robot. The main loop of the robot control thread is comprised of a switch statement that allows for different behaviors based on the control modes. For instance, in the cartesian space teleoperation mode a PD function is called to determine the desired torques given the current and desired states of the robot. Gravity compensation is determined in a similar manner with the `getGravityTorque()` function.

Inside the `getGravityTorque()` function , a `getCurrentG()` function is called to update the get the current gravity vector from a globally accessible variable. In this case, the gravity data is taken from the parameter passing object which is where a separate ROS handling thread puts the data whenever there is new information published on the `gravity` topic. In the event that the topic is not published when the robot starts up, the vector defaults

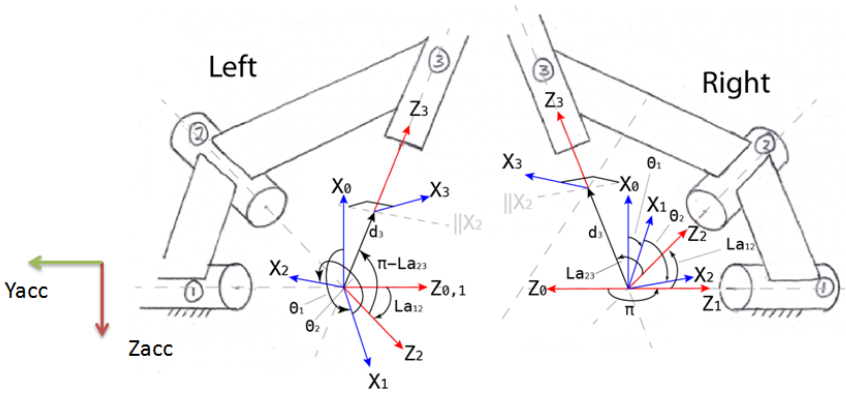

Figure 3.4: The base and gross positioning frames of the RAVEN II robot with the frame of the attached accelerometer.

to the case where the robot is on a perfectly level surface under nominal Earth gravity. The gravity vector is then transformed into the base frame of the arm that is currently being considered using the accelerometer frame and respective frame 0 shown in Figure 3.4. Similarly, the correct COMs are defined as constants, although these could easily be variables if the occasion called for it. A new function was written to recursively calculate the transforms between any two frames at the current configuration. This function is used to obtain the appropriate transforms to find the first 3 COMs in each of the previous frames as described in Chapter 2. Once the gravity vector and all of the COMs have been transformed to the necessary frames, the dynamic gravity compensation method is implemented for the first three joints.

The dynamic gravity compensation method returns desired joint torques, which has been learned to be quite different than motor torques. After extensive testing of the inner workings of the torque chain, it was determined that the calculated joint torques need to be transformed into the torque required at the output shaft of the motor gearbox. For this reason, the `getMotorTorqueFromJointTorque()` function was written and is called at the end of the `getGravityTorque()` function. In its current state, this function compensates for the cable coupling and mechanical advantages of the cable system. When the modeling

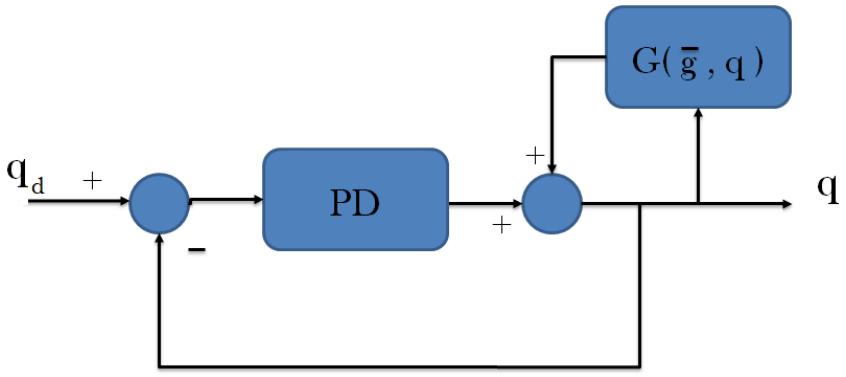

Figure 3.5: A diagram of the PD + DG controller, in which a desired orientation q_d is achieved through negative position feedback (from the robot pose q) into a PD term which is summed with a feed-forward gravity compensation term G. The function G is dependent on the robot pose q and the gravity vector $\bar{g}$.

of cable coupling and, perhaps, cable frictions are improved, this function will need to be revisited. Once the motor torques have been calculated, the `getGravityTorque()` function updates the desired gravity compensation torque variable in the robot state object.

Back in the main control thread, after the `getGravityTorque()` function has returned, the desired gravity torque parameter can be added to the desired torque variable in the robot state object, as seen in the controller diagram in Figure 3.5. If a PD torque is also implemented, then the control mode will operate as a PD controller with a dynamic gravity compensation term. However, if just the gravity compensation term is implemented in the control mode, it will act as a passive stable pose controller at any pose and at any base orientation (provided the accelerometer is working properly). This mode allows for outside disturbances to change the pose of the robot by only overcoming various frictions and motor inertias and the robot will hold this pose.

3.2.3 Summary

The ability to include a dynamic gravity compensation term into the RAVEN control mode has been described and implemented. By publishing acceleration data from a BMA180

accelerometer via an Arduino and netbook system as a ROS message, up-to-date external acceleration information can be included in the RAVEN software. These data are used to implement the method for dynamic gravity compensation described in Chapter 2. A dynamic gravity compensation term can be used by itself for a passive pose compensator. When this term is used in conjunction with a PD controller, the controller is expected to achieve better performance gains than those mentioned for other types of PD plus gravity compensation controllers mentioned in Chapter 1.2.2.

PD plus gravity compensation provides an alternative to a PID controller as a controller with zero steady state error. In trajectory following applications as in teleoperation and telesurgery, the I term is known to decrease trajectory tracking performance. A traditionally implemented I term will continue to compensate for previous errors when a new way-point is introduced, which is detrimental to tracking performance. Additionally, the I term introduces another pole in the controller, which makes the controller inherently less stable. Finally, in regulation tasks a PD controller requires an error to enable a control torque, whereas a gravity compensation term will inherently maintain a pose without error. In theory, a PD plus gravity compensation controller has advantages of greater stability, regulation accuracy, and trajectory tracking performance over a traditional PID controller without losing the steady state advantage of the I term.

Chapter 4

TESTING

4.1 Objective

The overall purpose in the testing phase was to test if PD plus dynamic gravity compensation control improves performance over PD only control and PD plus static gravity compensation control (SG) where it is assumed that the gravity vector is at a constant relation to the robot. First, the PD plus dynamic gravity compensation (DG) controller was tested to show that it performs as expected and that it has similar or improved performance to the other commonly used controllers mentioned. Second, trajectory following performance was tested at increasing static base angles to show improved controller performance in cases where a robot is on a stationary base but may be set to different configurations. Finally, the controller was tested with a continuously varying base angle.

4.2 Additional Tools and Capabilities for Tests

In order to perform the tests and collect the necessary data, several software and hardware modifications were made to the standard set up of the RAVEN. Figure 4.1 shows the robot set up with long handles to allow for base angle adjustment. Angles were measured with an adjustable angle bubble level and markings were drawn on a vertical board where the handle intersected it as seen in Figure 4.2. Wooden blocks were placed underneath the base of the frame to keep the robot at a constant base angle. C-clamps were affixed to the edge of the desk surface to keep the robot safe and to provide a constant center of rotation.

In addition to the standard PD control mode, a PD + DG control mode and a PD + SG control mode were added to the software. These new modes are identical to the standard PD controller except for the addition of a gravity compensation term. The PD gains were not changed so that comparisons could be made between controllers without the additional effects of different PD gains. In all modes, the DG and SG terms are calculated, but are

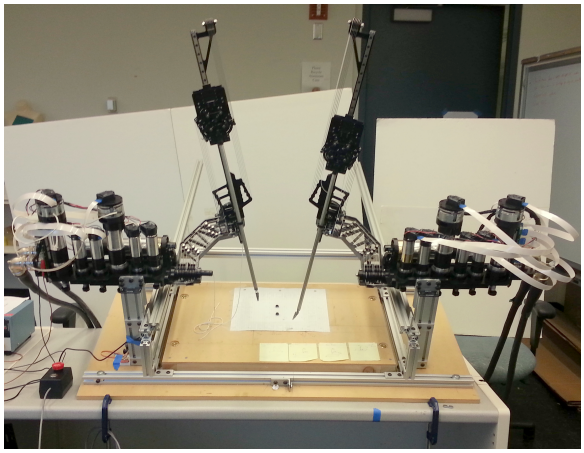

Figure 4.1: The RAVEN 2 with frame handles (rear) for base angle adjustment. The robot is shown at a base angle of 20 degrees.

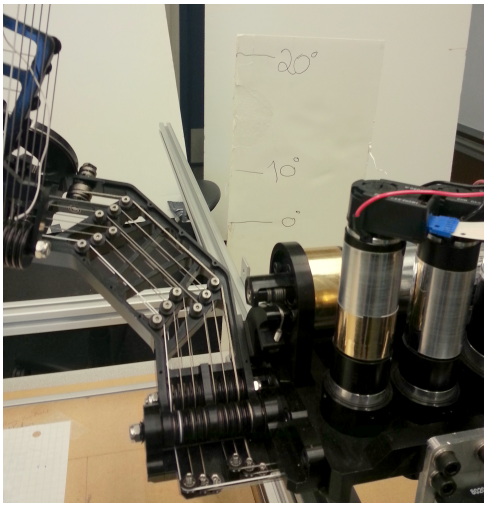

Figure 4.2: Angle markings used for base angle adjustment and rocking tests.

only used in controller torque when appropriate. A simple Bash script was implemented for initiating data collection from a custom ROS topic that output relevant portions of the robot state at 50 Hz.

4.3 Pose Validation Test

4.3.1 Comparison to PD control

The purpose of the pose validation test is first to validate that the calculations of the required gravity compensation torques are correct. This is achieved by an experimenter moving an arm through its range of motion on one joint but holding the other joints in a constant pose. Engaging the PD controller at increments throughout this range will provide data indicating the joint torque required at each incremental pose. This results in a sampled function of required gravity torques with only one independent variable since the other joint positions are held constant. This function can be repeated and compared to validate calculations. Autonomous movement of the joint was not used because joint-level commands have not yet been implemented on the RAVEN.

First, the robot is initialized with its homing procedure. Then the robot is manually moved, with brakes released, to one end of the first joint's travel. From here the data collection is started and the experimenter manually moves the robot through its travel, stopping incrementally to enable the PD controller at each pose. In this way the torque required to hold the robot at each increment pose can be implemented and recorded. This test was performed at the following base angles to show that the DG term will follow the required PD torque at each base angle: 0 degrees, 10, degrees, 20 degrees, and 25 degrees. This test was repeated during development of the controllers until the performance is as expected. For instance, this test will indicate if the transformation between the accelerometer frame and the robot base frame is correct if the sign and overall behavior of the gravity compensation term is correct.

4.3.2 Controller Validation

After validation of the gravity torque calculations and measurements, the gravity compensated controllers were evaluated with the same test procedure as above. It was expected that the PD term would drop significantly because the gravity compensation term will provide nearly all of the required torque at each position. However, at non-horizontal base angles the PD controller should be required to compensate for the inaccurate calculated gravity torque in the PD + SG controller.

4.4 Trajectory Following with Static Base Angles

Trajectory tracking performance was tested by performing a repeatable autonomous trajectory. At the base angles of 0, 10 and 20 degrees, the autonomous trajectory started at the initial, homed position. Data collection started when the trajectory starts and lasted for two repetitions of the cycle. Each of the 3 controllers was tested at each angle. It was expected that tracking performance would increase marginally with the PD + DG controller, which would also show a decreased effort from the PD term over the same effort of the PD + SG controller.

The autonomous trajectory used in the trajectory tracking tests was generated in the teleoperation master software. A simple function was implemented to piggy-back on the normal operation of the master to send the trajectory as a series of movement increments whenever a previously-unused foot pedal is pressed down. The trajectory is a series of sinusoidal motions in each Cartesian direction starting from the homed point and having an amplitude of about 3.1 inches. The period of each sinusoid is 4 seconds. The path can be seen in 3D and as X, Y and Z signals in Figure 4.3. The shape and frequency of the trajectory was chosen as a safe shape that would explore a significant fraction of the workspace and not induce extra dynamic control components that would need to be compensated for. The shape provides data that is easy to visualize and analyze.

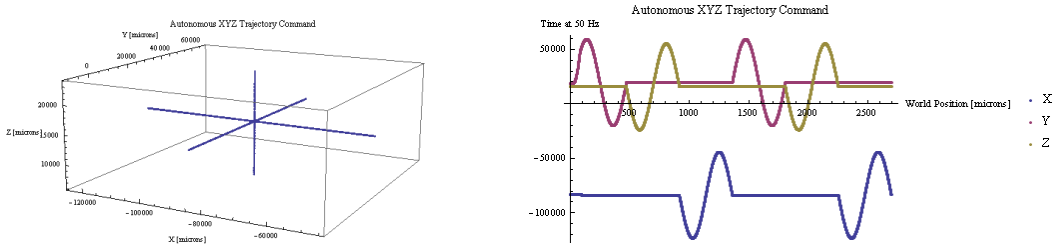

Figure 4.3: The autonomous trajectory used for trajectory tracking tests. The amplitude in each direction is 78000 microns, 3.1 inches. Each test is run for two cycles.

4.5 Trajectory Following with Rocking Base

A rocking base test was used to test the performance of the PD + DG controller under conditions where the the relative gravity or acceleration vector is changing during operation. A 1 Hz sinusoidal rotation of the base angle between 10 and 20 degrees was used as a simple and repeatable base trajectory. The experimenter used the frame handles to move the base between the two angles as indicated on the measurement board, hitting the max and min on each beat of a 120 beats per minute signal from the "Slick Metronome" application for Android. The rocking started when the data collection and autonomous trajectories started.

4.6 Summary

In summary, the planned tests were intended to validate and evaluate the performance of a PD + DG controller over a PD + SG controller and the standard PD controller on the RAVEN surgical research robot. To that end, the implementation of gravity control was tested on the first joint of the robot by manually moving it through its range of motion and sampling the PD torques necessary to hold at each incremental position. Then, the performance of the controllers was evaluated while following a repeatable autonomous trajectory at various static base angles and during a 1 Hz continuous variation of base angles.

Chapter 5

RESULTS AND EVALUATION

5.0.1 Evaluation Metrics

Several metrics are used in the following results in order to measure the effects of gravity compensated control against PD control. The data used in each of the tests are simply the end effector position, desired end effector position, joint angles, desired joint angles, control torques, and the individual terms of the PD, PD + SG and PD + DG controllers.

In the validation of the gravity calculations, the calculated, but not implemented, gravity compensation terms are compared with the PD torques at each angle. The PD values have been filtered with a moving average filter in the figures for this step to compensate for the incremental data collection procedure. The errors between the PD torque and gravity terms are averaged for each trial and displayed as a bar graph. In the next step, the validation tests are repeated with the gravity compensation terms implemented. In these tests, the PD term is considered to be an error in the gravity compensation since the gravity compensation term is supposed to provide exactly the torque necessary to maintain the robot at its current position. In order to compare the error in compensation between the two gravity compensated controllers, the root-mean-square (RMS) averaging technique is used to determine the average efforts of the controllers and of the individual PD terms. Thus, the effort of the PD term can be expressed as a percentage of the total control effort. In validation, a percentage of zero indicates that no extra effort was required to maintain the robot's pose. Thus, a higher pd percentage indicates greater inaccuracies in the gravity compensation model.

Both trajectory tracking tests use the PD percentage in addition to analysis of position errors. In these tests, the PD percentage indicates how useful the term is in control during trajectories. Ideally gravity compensation will eliminate the PD effort that is normally used to overcome gravity torques, leaving only friction and inertia for the PD term to overcome.

In order to distill the PD percentage to a single value for each controller the PD percentages for each motor are simply averaged. Position and joint errors are analyzed as an RMS of the difference between desired position and the time-corresponding actual position.

5.1 Pose Validation Tests

5.1.1 Comparison to PD control

Initial validation testing ensured that the calculated values of the gravity terms are as expected. When the robot base is at its nominal horizontal orientation, both the dynamic gravity and static gravity calculated terms match the PD values used at incremental positions. Figure 5.1 shows this result at zero degrees with PD values that have been smoothed using a moving average filter to account for the incremental nature of the test. Furthermore, this figure demonstrates the necessity for a DG term over SG because the SG term becomes less accurate at increasing angles. At extreme angles, this could potentially have a significant adverse effect on the performance of the controller. The RMS values of the differences can be seen in Figure 5.2, which shows that the RMS error of the SG calculated torque is more than half of the actual PD torque value at a base angle of 25 degrees.

5.1.2 Control Validation

As expected, the PD + DG performs more accurately with changing base angles. Figure 5.3 shows that the DG controller accounts for nearly all of the total control torque and the PD does very little correcting since the PD can be seen to have zero torque at almost all times and the sum of the PD and DG terms follows very closely to the DG term. However, the PD + SG controller requires more effort from the PD term in order to hold a pose with increasing base angles. The stark contrast in performance can be seen in Figure 5.4, which shows the RMS value of the PD term as a percentage of the RMS of the total torque throughout the trajectory. This figure implies that the PD term does not need to contribute more overall effort due to inaccuracies in the gravity compensation in the PD + SG controller at non-horizontal angles.

Informal validation has shown that controlling with only a DG term allows the robot

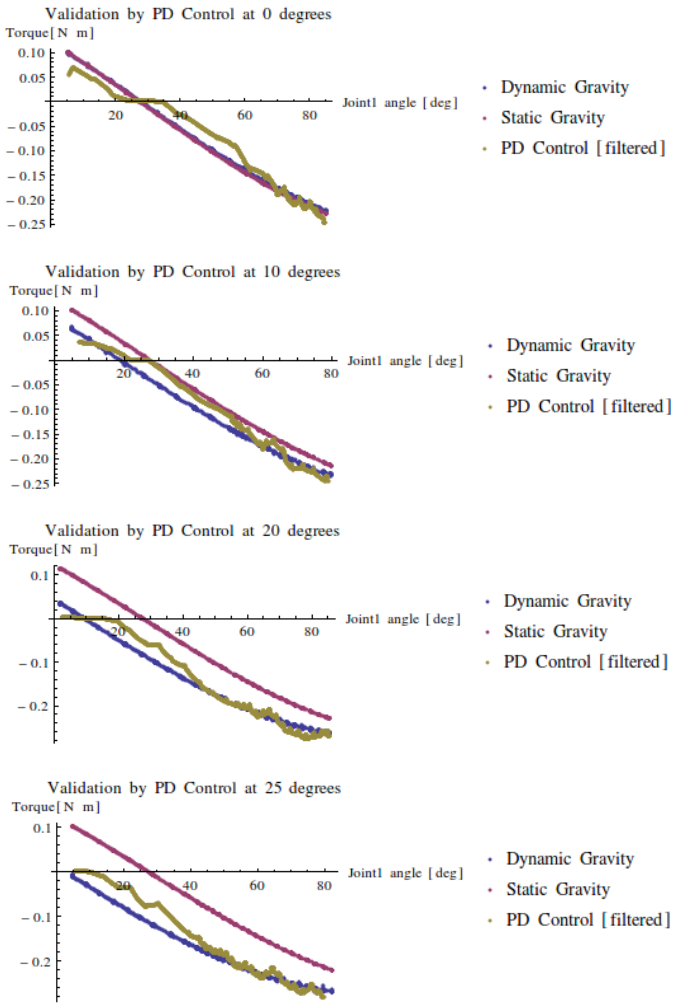

Figure 5.1: Calculated gravity compensation terms compared to PD torques at increasing base angles.

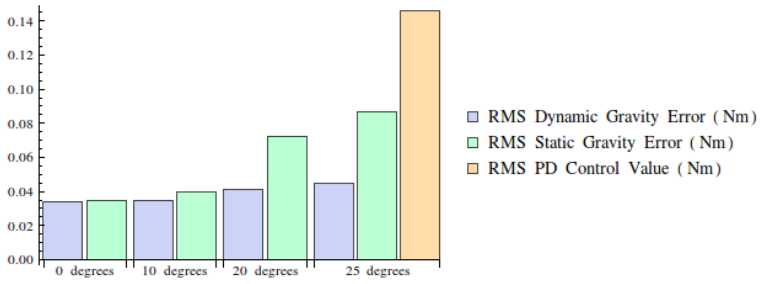

Figure 5.2: RMS of error between PD torque at a static pose and the calculated gravity terms compared to RMS value of total control torque. The RMS PD control value is nearly constant at all degrees and is shown as a comparison for the magnitude of calculated error.

to be easily manipulated by hand and maintain any new position. The majority of the resistance to motion in this mode is thought to be motor inertia and cable friction terms. This effect is seen at all angles for the DG term, but only at or near horizontal with the SG term. Unlike the DG only controller, when the SG only controlled robot is tipped more than a few degrees the robot will drift due to the error in gravity compensation.

5.2 Trajectory Following with Static Base Angles

5.2.1 Position Error and Control Effort

As expected from the literature, gravity compensation does seem to help tracking error. Although Figure 5.5 shows that the robot's tracking error may not follow closely in Cartesian space, Figure 5.6 shows that joint tracking error is improved by gravity compensation for joints 1 and 2. At a base angle of 20 degrees, the PD + DG controller's RMS joint 1 error is around 12% less than the error of the PD only controller. Improvements with the PD + SG controller disappear at 20 degrees, but are noticeable at smaller angles. At 20 degrees, the inaccuracy of the gravity compensation term in the PD + SG controller negates any improvements that it may have imparted. Joint 3 is shown to have higher errors with gravity compensated control over PD only control which may be due to the different nature of compensating for linear rails.

Figure 5.5 shows the RMS error between the Cartesian position of the robot end effector

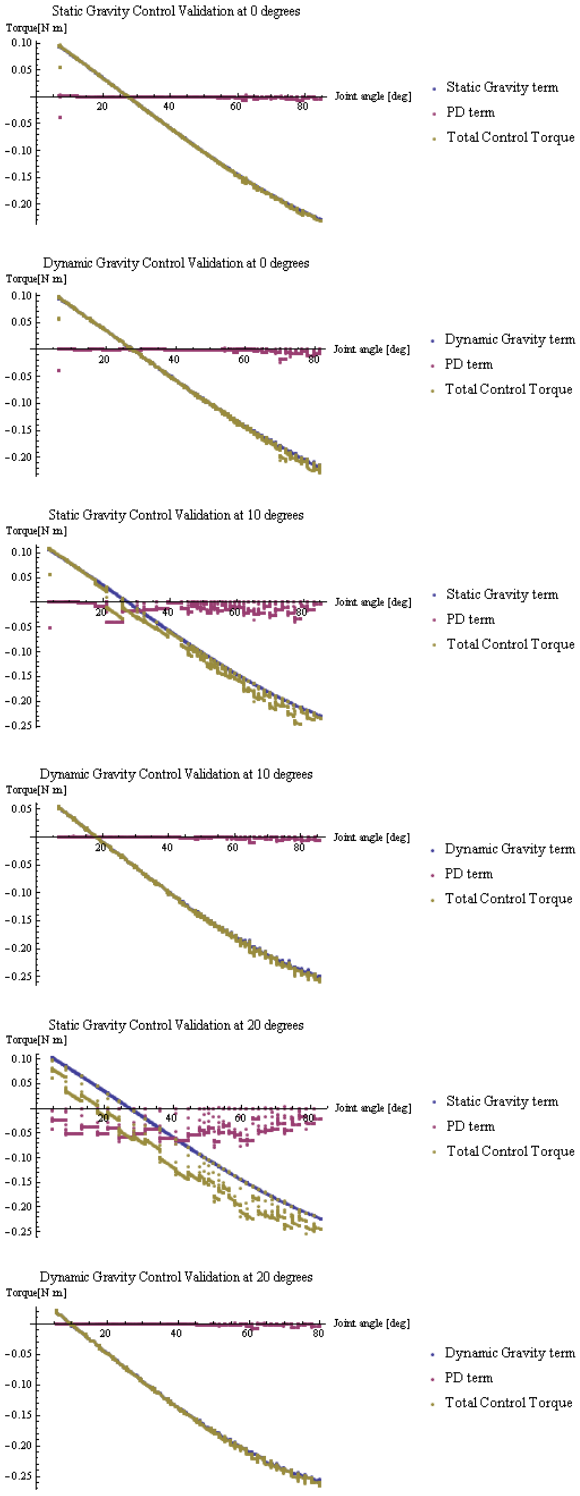

Figure 5.3: Control torques of motor 1 with PD + SG (left) and PD + DG (right) at increasing base angles.

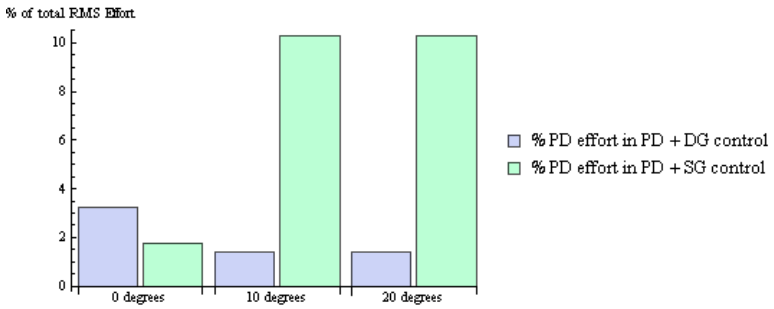

Figure 5.4: PD term as a percentage of RMS torque in gravity compensated controllers at increasing base angles during validation trajectory.

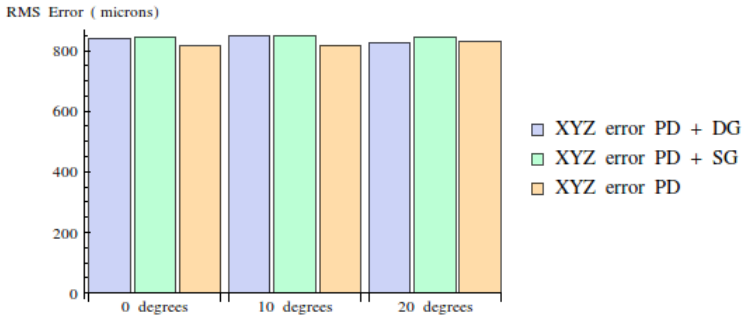

Figure 5.5: RMS of total position error for the three controllers at increasing base angles during autonomous trajectory.

and its desired position at every time step. The error is nearly constant across controllers and angles at about 0.03 inches (800 microns). Similarly, the RMS control efforts seen in Figure 5.7 are nearly constant across controllers at a single angle, except for the PD + SG controller at 20 degrees. Again, this exception is likely due to the modeling inaccuracies of the PD + SG controllers at base angles more than 10 degrees.

5.2.2 PD Effort

Similar to the results of the validation tests (Figures 5.1 through 5.4), PD effort is reduced during trajectory tracking. Figure 5.8 shows very interesting results where PD effort is greater than 100% and PD + SG at 20 degrees requires less PD effort than PD + DG. Figure

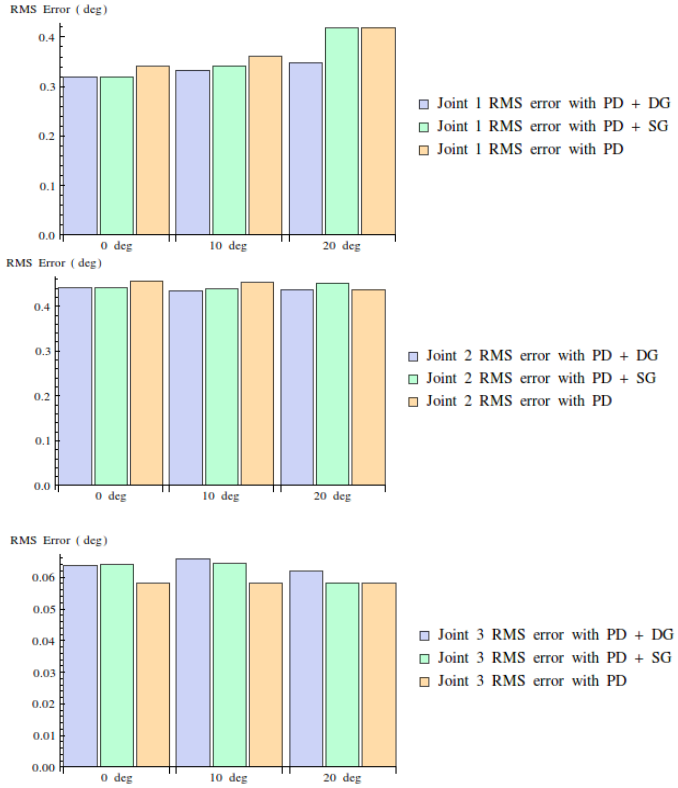

Figure 5.6: RMS joint angle errors in degrees for each controller at each angle.

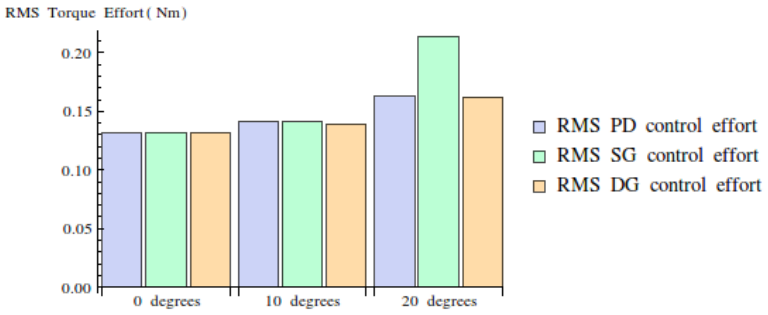

Figure 5.7: RMS of total effort for the PD, PD + DG, and PD + SG controllers at increasing base angles during autonomous trajectory.

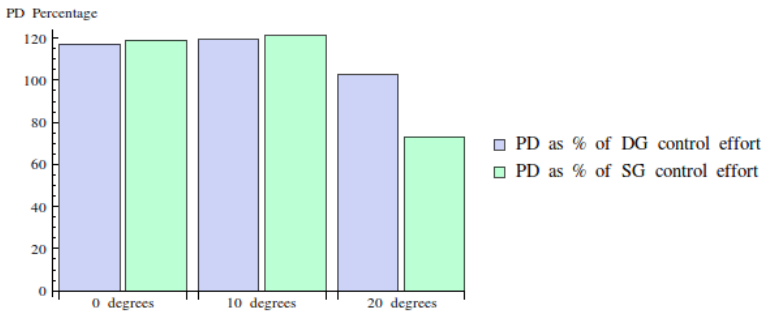

Figure 5.8: PD term as a percentage of RMS torque in gravity compensated controllers at increasing base angles during autonomous trajectory.

5.9 indicates that the cause for these unexpected results is due to the gravity compensation of the third joint. For all but PD + SG at 20 degrees, the percent of total effort for the PD term is over 150%. However, the individual contributions to this effort (Figure 5.9) show that it joint 3 that pulls the average PD effort above 100%. Figure 5.10 shows that total motor torque and gravity terms for each controller at 20 degrees, where the PD effort is shown to have an effort of less than 50% when all other tests show more than 150% at the same joint.

In the rotary joints 1 and 2, PD effort as a percentage of total torque behaves as expected and is not higher than 100%. Not only is the percentage smaller for PD + DG at each angle, it also decreases at each angle for motor 1.

5.3 Trajectory Following with Rocking Base

5.3.1 Position Error and Control Effort

Under constantly varying base angles, the gravity compensated controllers show little collective effect on Cartesian tracking performance, as seen in Figure 5.11. However, Figure 5.12 shows that joint 1 tracking performance increases by the same 12% as in the static base angle tests. Also similarly to the last test, gravity compensation does not seem to affect the control effort required to perform the trajectory (Figure 5.13).

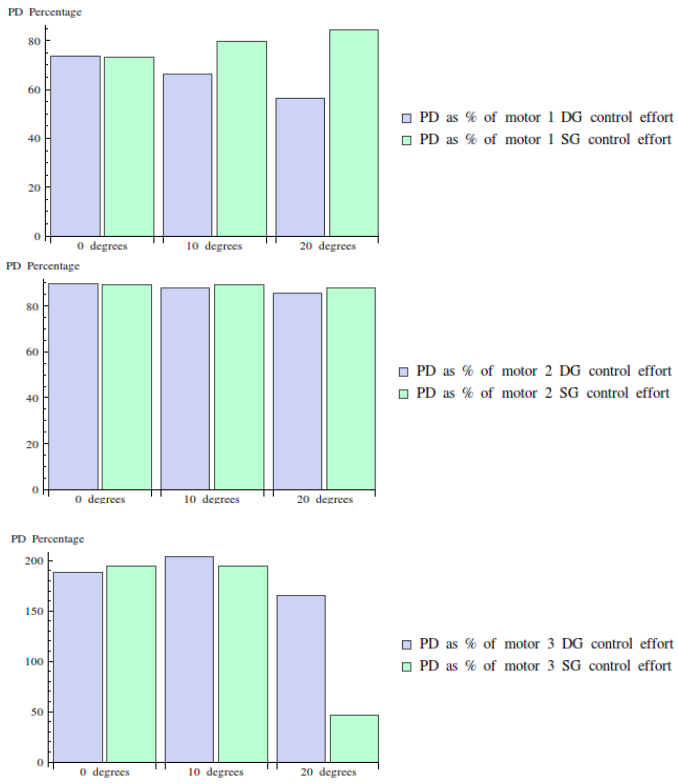

Figure 5.9: PD term as a percentage of individual motor torques in gravity compensated controllers at increasing base angles during autonomous trajectory. The increase in PD percentage over Figure 5.4 is due to the additional PD effort required to move the robot.

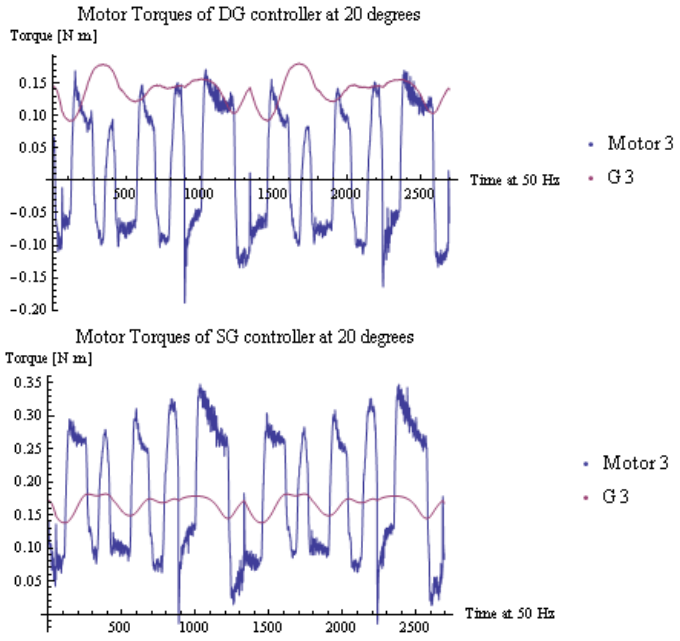

Figure 5.10: Motor torques and gravity terms

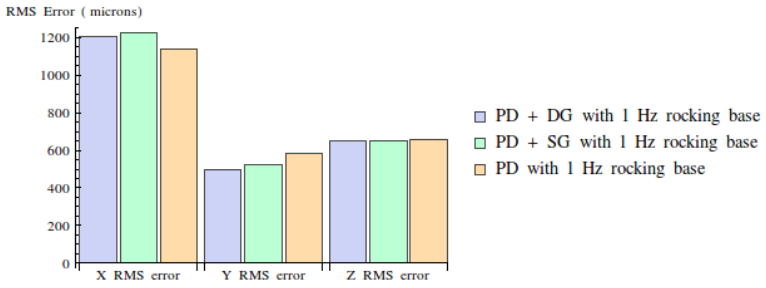

Figure 5.11: RMS of X, Y, and Z position error for the three controllers during autonomous trajectory with rocking base.

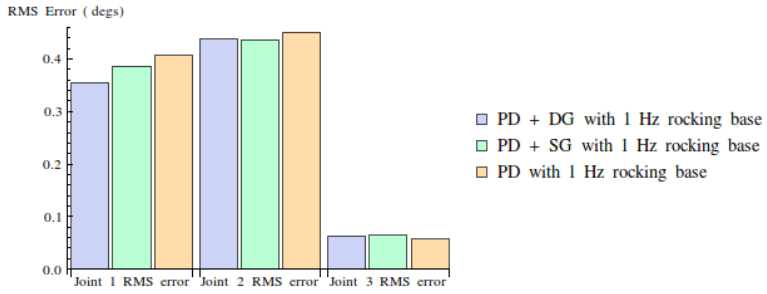

Figure 5.12: RMS of joint angle error for the three controllers during autonomous trajectory with rocking base.

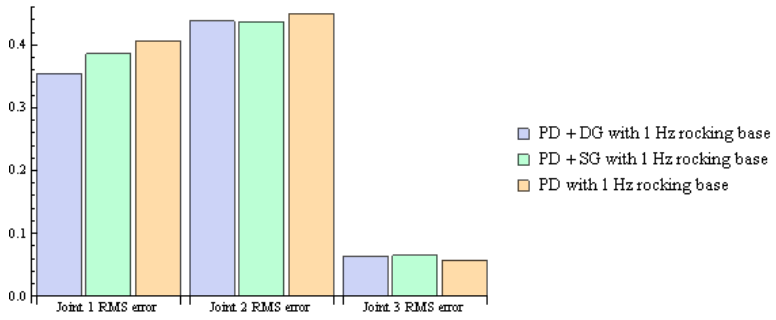

Figure 5.13: RMS of total effort for the three controllers at increasing base angles during autonomous trajectory with rocking base.

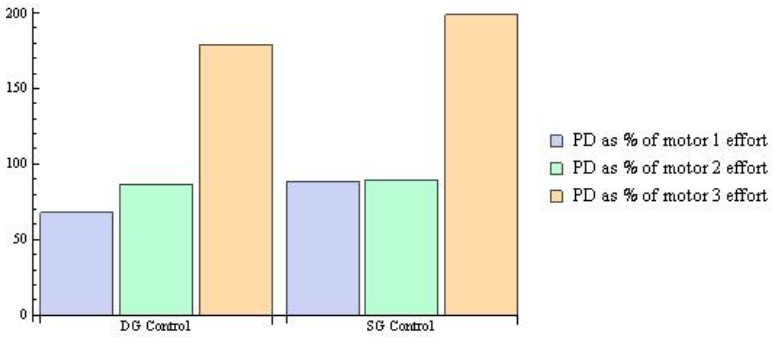

Figure 5.14: PD term as a percentage of RMS motor torques in gravity compensated controllers during autonomous trajectory with rocking base.

5.3.2 PD Effort

PD effort shows a decrease in the PD + DG controller compared to PD + SG control, although both gravity compensation terms require over 175% of the total motor 3 effort from the PD term.

Chapter 6

DISCUSSION

In the validation step, is is possible that the unmodelled friction terms that the PD controller is compensating for could skew the results of the tests. This means that the gravity compensation term validated and used in testing could be larger than the actual gravity torque applied to the robot. Further analysis and testing will be required in order to fully understand the contributions of friction on the system. However, it is clear from the results of the validation tests that gravity compensation can significantly reduce the required PD effort in regulation tasks(Figure 5.2). In fact the DG term accounts for more than 98% of the regulation effort at varying base angles. Although less effective, the SG term accounts for 90% of the regulation effort up to a 20 degree base angle.

In the results from trajectory following at static joint angles, joint 3 is shown to have higher errors with gravity compensated control with gravity compensated control (Figure 5.6) which may be due to the different nature of compensating for linear rails. Furthermore, movement of the linear carriage requires additional modeling of the cable coupling from the tool spindles on the carriage. Moving the carriage invariably requires the tool motors to rotate in conjunction, otherwise, the carriage will have added drag from the motor inertia and friction of the four tool cables. This modeling is not yet fully implemented in the gravity compensation software.

However, an outlier in Figure 5.4 shows that joint 3 under PD + DG control requires less than 50% PD effort at a 20 degree base angle. This result is unexpected because it indicates that the gravity term is more accurate for joint 3 when there is a 20 degree error from the actual gravity vector. A reason for this could lie in the motor torques shown in Figure 5.10. It can be seen that the DG compensation term is located along the peaks of the motor torques at 20 degrees, while the SG term is located at what appears to be the average of the torque command. This means that the PD term is spending less effort compensating

for the SG term than the DG term. This leads to an almost identical performance between PD and PD + SG in tracking the desired joint position of joint 3 at 20 degrees, and this is also better performance than the PD + DG controller.

Although improvements in performance seem to be relatively limited in this application, this effort in modeling of the RAVEN is just one of a series of steps for accurate understanding and control of the robot. Furthermore, when implemented on the RAVEN I, gravity compensation was anecdotally shown to greatly increase robot performance. The large performance difference is thought to be due to different approaches in robot design between the first and second RAVEN robots. In RAVEN I, the mechanisms were designed for backdrivability. This requirement was thought to be necessary for accurate haptic loops and requires low gear ratios and powerful motors. The RAVEN II mechanism foregoes these requirements in favor of high gear ratios and less powerful motors. Thus, it is thought that the gravity torques were a much larger portion of RAVEN I control than in RAVEN II control, which has much higher gearing inertia. From this experience, it is thought that gravity compensation is especially important for robots in which gravity loads are a large component of control. A simple experiment to test this hypothesis is to add a known load to one or all of the links and repeat the above performance tests.

6.1 Summary

From these results, the following assertions can be made about gravity compensation on the RAVEN surgical research robot:

- As shown in the literature, gravity compensation demonstrates incremental improvements in robot performance.

- Using a real-time value for the gravity vector in gravity compensation calculation will decrease the PD effort in regulation and tracking tasks in applications where the robot base angle is adjustable or subject to movement during operation. This is due to the improved model accuracy as demonstrated in Figure 5.1. This figure and 5.2 demonstrate that modelling errors require the PD term to "fight" the inaccurate term in order to achieve the desired state.

- Dynamic gravity compensation shows the same improvements over static gravity compensation with a constantly varying base angle as with statically increasing base angles. Further testing is required for statistical significance, but preliminary testing is promising.

- Joint 1 tracking performance is shown to improve the most with either type of gravity compensation implemented. This is likely due to this joint having the highest imposed gravity torques since it has to compensate for the whole robot. This joint also acts in parallel with the imposed rotations in testing, so it has the most to gain from an accurate dynamic gravity compensation term in these tests.

- Static gravity compensation provides similar improvements to PD effort at static base angles below 10 degrees.

- Gravity compensation is found to contribute very little to the control of the prismatic joint at link 3. In fact, it seems to be detrimental to robot performance.

Chapter 7

CONCLUSIONS

A PD + dynamically evaluated gravity compensation controller has been analyzed, implemented and tested on the RAVEN surgical research robot. By utilising frame transformations that are often calculated in robot controllers for forward kinematics, a novel method for calculating gravity torque has been derived and implemented. By using high level data structures, this method is capable of compensating for changing relative acceleration vectors in addition to real-time varying COM locations, robot configurations, and even the number of links. A new method was implemented for interfacing low level sensor data with the RAVEN with the use of an Arduino and netbook computer networked with the robot using the Robot Operating System. The PD + dynamic gravity controller was implemented as an additional term summed with the existing PD controller and compared against the PD controller and the new controller with a constant gravity vector at $[0, 0, -9.8]m/s^2$. After validating the calculations and expected performance of the controller, tests were performed to discover if the PD + DG controller has noticeable effects on trajectory tracking performance during static and dynamic base orientations. As in regulation performance, PD effort as a percentage of total control effort is decreased in trajectory following. Slight improvements have been shown in tracking error under constantly varying base angle with the PD + DG controller. Either due to modeling errors or the nature of the third link, the prismatic tool cart, gravity compensation is detrimental to trajectory tracking on the RAVEN robot. It has also been shown that assuming a constant gravity vector will provide sufficiently useful results for small amounts of base angle error. The nature of the RAVEN community and its open source software makes it possible and practical to develop on improvements and innovations made by other RAVEN users. The following section details extensions of this project and possible research areas for improving the modeling and control of the robot.

7.1 Future Work

More testing is required to fully explore the benefits of dynamic gravity compensation. First, control transients should be studied to see if step response performance is improved with varying base angles. A large area of study is the performance of dynamic gravity compensation under external accelerations and significant reductions in gravity accelerations. A study of these characteristics will demonstrate the utility of dynamic gravity compensation in or on moving vehicles. Potential improvements allowed by PD + DG control could be decreased response time and overshoot and increased tracking and regulation performance. It is expected that the PD + DG controller will provide superior canceling of sustained disturbances and may be beneficial when introduced with transient accelerations. Analysis will be required to determine the relationship between the frequency of the filtered acceleration sensor data and the frequency of disturbance that the controller is effective at canceling. Additional testing is also required to determine if different PD gains would improve the control of the RAVEN for its intended purposes. Furthermore, performance of the PD + DG controller for the RAVEN robot could be improved by more accurate and more detailed modeling of the robot and environment.

7.1.1 Improvements in Modeling

Several inaccuracies exist in the implemented PD + DG controller due to an unknown amount of modeling inaccuracy. Firstly, the COMs used in the gravity calculations are based on computer models with homogeneous density which does not account for the variety of materials used in the construction of each joint. A more detailed computer model, a rigorous empirical test, or the gravity loading algorithm presented in [11] could find the locations of the locations of the COMS more accurately. As mentioned in the discussion of joint 3 errors, cable coupling and frictions of the carriage and tool DOFs need to be improved for accurate control of the robot.

In [9], an online gravity compensation controller is presented in which the robot joint positions are estimated from gravity-biased models of the sensed motor positions. Using the known direction of induced gravity torques and models of joint elasticity, an accurate

model of joint positions can be built without the need to add more sensors to the robot. This would be very useful on the RAVEN but would require very careful analysis of the complex cable system or a very capable autonomous method for learning the cable stiffness properties of each link.

Further improvements could also be found in calculating gravity compensation based on the desired robot configuration, discussed earlier as constant gravity compensation. This may be especially useful for incremental trajectory tracking, which is implemented on the RAVEN for teleoperation. In these cases, small changes in the desired pose would cause a shift in the gravity compensation towards the new pose. This small effort from the gravity compensation term could assist the PD term in movement instead of simply maintaining the current position of the robot.

The new sensor integration system developed for this project could be used to improve modeling of the robot and its environment. For instance, a gyro on the robot base could allow for modeling centripetal and Coriolis forces from externally induced base rotations in much the same way that external accelerations are modeled by the accelerometer in this project. Placing additional accelerometers on the links of the RAVEN would allow for estimation of joint angles based on the angular differences between the sensed acceleration of the accelerometers on the links and on the base of the robot. A Raspberry Pi embedded computer is currently being developed to replace and improve the sensing system used for this project. The Raspberry Pi is small enough to be able to communicate with embedded sensors and powerful enough to run a version of ROS that should integrate with the RAVEN system just as easily as the current system. Once this is complete, a whole world of sensors will be supported for use with the RAVEN.

BIBLIOGRAPHY

[1] J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Trans. ASME E, Journal of Applied Mechanics*, 22:215–221, June 1955.

[2] Pablo Garcia, Jacob Rosen, Chetan Kapoor, Mark Noakes, Greg Elbert, Michael Treat, Tim Ganous, Matt Hanson, Joe Manak, Chris Hasser, David Rohler, and Richard Satava. Trauma pod: a semi-automated telerobotic surgical system. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 5(2):136–146, 2009.

[3] B. Hannaford, J. Rosen, D.W. Friedman, H. King, P. Roan, Lei Cheng, D. Glozman, Ji Ma, S.N. Kosari, and L. White. Raven-ii: An open platform for surgical robotics research. *Biomedical Engineering, IEEE Transactions on*, 60(4):954–959, 2013.

[4] Karl Iagnemma, Robert Burn, Eric Wilhelm, and Steven Dubowsky. Experimental validation of physics-based planning and control algorithms for planetary robotic rovers. In *Experimental Robotics VI*, pages 319–328. Springer, 2000.

[5] H Hawkeye King, Thomas Low, Kevin Hufford, and Timothy Broderick. Acceleration compensation for vehicle based telesurgery on earth or in space. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1459–1464. IEEE, 2008.

[6] H Hawkeye King, Sina Nia Kosari, Blake Hannaford, and Ji Ma. Kinematic analysis of the raven-ii research surgical robot platform. Technical report, University of Washington and U.C. Santa Cruz, 2012.

[7] H.H. King, B. Hannaford, Ka-Wai Kwok, Guang-Zhong Yang, P. Griffiths, A. Okamura, I. Farkhatdinov, Jee-Hwan Ryu, G. Sankaranarayanan, V. Arikatla, K. Tadano, K. Kawashima, A. Peer, T. Schauss, M. Buss, L. Miller, D. Glozman, J. Rosen, and T. Low. Plugfest 2009: Global interoperability in telerobotics and telemedicine. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1733–1738, 2010.

[8] Alessandro De Luca and Stefano Panzieri. Learning gravity compensation in robots: Rigid arms, elastic joints, flexible links. *International Journal of Adaptive Control and Signal Processing*, 7(5):417–433, 1993.

[9] Alessandro De Luca, Bruno Siciliano, and Loredana Zollo. {PD} control with on-line gravity compensation for robots with elastic joints: Theory and experiments. *Automatica*, 41(10):1809 – 1819, 2005.

[10] MitchellJ.H. Lum, DianaC.W. Friedman, HawkeyeH.I. King, Regina Donlin, Ganesh Sankaranarayanan, TimotyJ. Broderick, MikaN. Sinanan, Jacob Rosen, and Blake Hannaford. Teleoperation of a surgical robot via airborne wireless radio and transatlantic internet links. In Christian Laugier and Roland Siegwart, editors, *Field and Service Robotics*, volume 42 of *Springer Tracts in Advanced Robotics*, pages 305–314. Springer Berlin Heidelberg, 2008.

[11] Donghai Ma and J.M. Hollerbach. Identifying mass parameters for gravity compensation and automatic torque sensor calibration. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 661–666 vol.1, 1996.

[12] Donghai Ma, J.M. Hollerbach, and Yangming Xu. Gravity based autonomous calibration for robot manipulators. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 2763–2768 vol.4, 1994.

[13] Renaud Ott, M. Gutierrez, D. Thalmann, and F. Vexo. Improving user comfort in haptic virtual environments through gravity compensation. In *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005. First Joint*, pages 401–409, 2005.

[14] E. Sadraei and M.M. Moghaddam. On a moving base robotic manipulator dynamics. In *Robotics and Mechatronics (ICRoM), 2013 First RSI/ISM International Conference on*, pages 165–170, 2013.

[15] P. Tomei. A simple pd controller for robots with elastic joints. *Automatic Control, IEEE Transactions on*, 36(10):1208–1213, 1991.

[16] C.M. Wronka and M.W. Dunnigan. Derivation and analysis of a dynamic model of a robotic manipulator on a moving base. *Robotics and Autonomous Systems*, 59(10):758 – 769, 2011.