A Waypoint-Driven Gradient Descent Solution for a Parallel Robot

by

Gabriel Valdes

Submitted to the
Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology

June 2019

Signature of Author: Signature redacted

Department of Mechanical Engineering
May 10, 2019

Certified by: Signature redacted

Harry Asada
D'Arbeloff Laboratory for Information Systems and Technology Director
Thesis Supervisor

Accepted by: Signature redacted

Maria Yang
Professor of Mechanical Engineering
Undergraduate Officer

A Waypoint-Driven Gradient Descent Solution for a Parallel Robot

by

Gabriel Valdes

Submitted to the Department of Mechanical Engineering
on May 10, 2019 in Partial Fulfillment of the
Requirements for the Degree of

Bachelor of Science in Mechanical Engineering

ABSTRACT

This project aims to introduce a more robust navigation architecture for the Triple Scissor Extender Robot Arm (TSERA) at the d'Arbeloff Laboratory for Information Systems and Technology. TSERA was developed to access a confined area through a narrow channel, commonly known as the last one-foot problem found in final assembly, inspection, and maintenance operations within the aviation, automobile, and industrial equipment industries. Inspired from plant growth mechanisms, the robot is built from a sequence of expandable segments that can each extend and tilt.

The current path planning algorithm computes arm motion by solving a series of inverse kinematic relations for each segment. This requires a user input of a three-dimensional coordinate to a kinematics solver for a robot in a complex and unknown operating space with parasitic displacement characteristics.

This new path-planning design allows users to instead input a desired orientation for an expandable segment, utilizes a gradient ascent algorithm to determine the three-dimensional coordinate that would allow for that desired orientation, and then creates waypoints across the path in order to ensure minimal displacement error and reduce chances of damage to the robot's motors all in real-time. This solution allows for a more intuitive user experience with TSERA and increases robustness of the robot itself.

Thesis Supervisor: Harry Asada
Title: D'Arbeloff Laboratory for Information Systems and Technology Director

Table of Contents

List of Figures

Chapter 1

Introduction

Shop floor artisans are sometimes tasked with reaching specific points through narrow gaps and obstacles during assembly, maintenance, and inspection of complex machinery. Failure to address these situations causes fatigue, low productive, and potential repetitive stress injuries. Other robotic solutions fail to properly address this situation because they either require too much infrastructure or a heavy base actuation system that prevents application to these tight, confined spaces. Other solutions, such as mobile snake robots, require interaction with the environment, which may not be permitted in situations such as aircraft inspection [1].

The Triple Scissor Extender Robot Arm (TSERA) addresses this "last-foot" problem by providing a solution that is "self-supporting ... capable of moving through narrow winding spaces while carrying a significant payload." [1] This paper presents a new feature for the TSERA's Sequential Expansion Algorithm [1]. The previous solution allowed "the arm to traverse winding trajectories through narrow channels with minimal spill out and accurate endpoint positioning..." through "...TSERA's path planning algorithm" which "only requires collision checks for a single rigid body before guaranteeing collision free motion." The solution presented in this paper expands on it by developing the motion planning architecture to take various control inputs, allowing for user to input a desired orientation, rather than a desired XYZ

position. It also applies a gradient descent algorithm to converge on the motor commands necessary to achieve the desired orientation.

This is a critical step towards achieving a fully autonomous motion-planning architecture for positioning of the end effector. Future development on the TSERA will include expanding sensing capabilities in order for the TSERA to map the environment and determine obstacles in its path as well as multi-segment navigation control in order to place the end effector at a desired position while also setting segment length and orientation to optimize between different desired features, such as stiffness or compliance.

1.1 Background

1.1.1 Gradient Descent

Gradient Descent is a mathematical computation done to find the local or global minima of a function. Given a function and a specific coordinate on that function, the slope at that coordinate on the function can be calculated by calculating the partial derivative of the function with respect to each parameter in the function and evaluating at the coordinate. The derivative is then multiplied by a scaling factor, called the "learning rate", and then a step is taken in that direction. [2] The gradient is then calculated again at the new coordinate and the process is repeated until the current coordinate reaches a minima in the function. Each "step" moves the system to a coordinate that reduces the value of the cost function, commonly known as a step "down the slope".[2] The scaling factor can be fine-tuned to the specific needs of the function. For gradient descent computations done on functions with irregularities or for computations that require much more fine calculations, a scaling factor smaller than 1 is used. On the other hand, computations with smooth functions or situations that value speed, a scaling factor of 1 or more is sufficient.

This method is typically used to optimize across a cost surface or cost function,

which is a representation of the system given a set of parameters. The purpose of the gradient descent computation is to find the parameter values that optimize the cost function. In the scope of this project, the cost function is represented by equation 1.20, which is the dot product of the desired orientation, T, with the current orientation, b. Maximizing the cost function in this context physically means aligning the orientation with the desired orientation. Given the goal of maximizing the cost function in this project, the gradient descent operation is negated, thus stepping "up the slope" rather than "down the slope" [2].

1.1.2 Description of TSERA Kinematics

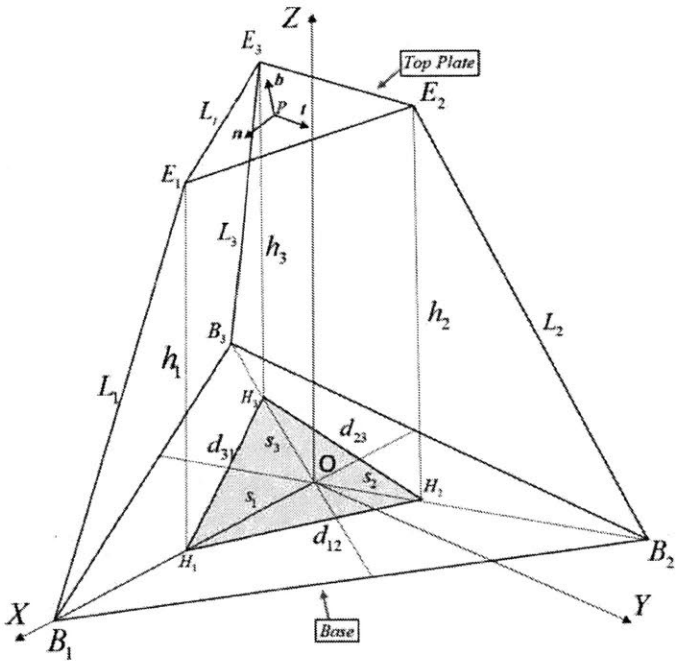

Figure 1-1: Stage Coordinate Basis

To reach an understanding of what the cost function for this robot will be, first the stage's kinematics must be calculated. Below is the vector representation of the

vertices of the top plate [1]:

$$E_1 : \boldsymbol{e_1} = [s_1 \ 0 \ h_1]^T \tag{1.1}$$

$$E_2 : \boldsymbol{e_2} = \left[-s_2 \sin(\frac{\pi}{6}) \ s_2 \cos(\frac{\pi}{6}) \ h_2\right]^T \tag{1.2}$$

$$E_3 : \boldsymbol{e_3} = \left[-s_3 \sin(\frac{\pi}{6}) \ -s_3 \cos(\frac{\pi}{6}) \ h_3\right]^T \tag{1.3}$$

Taking the average of the vertices of the top plate relates the end effector position, $P : \boldsymbol{p} = [x, y, z]^T$, to the internal variables as follows [1]:

$$x = \frac{1}{3}\left(s_1 - \frac{(s_2 + s_3)}{2}\right) \tag{1.4}$$

$$y = \frac{\sqrt{3}}{6}(s_2 - s_3) \tag{1.5}$$

$$z = \frac{1}{3}(h_1 + h_2 + h_3) \tag{1.6}$$

Given the kinematics of a single segment, four equations can be determined to constrain the kinematics of the segment [1]:

$$d_{12}^2 + (h_2 - h_1)^2 = L_t^2 \tag{1.7}$$

$$s_1^2 + s_2^2 + s_1 s_2 + (h_1 - h_2)^2 = L_t^2 \tag{1.8}$$

$$s_2^2 + s_3^2 + s_3 s_2 + (h_2 - h_3)^2 = L_t^2 \tag{1.9}$$

$$s_1^2 + s_3^2 + s_1 s_3 + (h_3 - h_1)^2 = L_t^2 \tag{1.10}$$

For TSERA, L_t is 85.3 mm. Now we can express the orientation of the top plate, vector b, as a function of the top plate position P [1].

$$\boldsymbol{n} = \frac{1}{L_t}(\boldsymbol{e_1} - \boldsymbol{p}) \tag{1.11}$$

$$\boldsymbol{t} = \frac{1}{L_t}(\boldsymbol{e_2} - \boldsymbol{e_3}) \tag{1.12}$$

$$\tag{1.13}$$

And thus, $\boldsymbol{n}\ x\ \boldsymbol{t}\ =\ \boldsymbol{b}$, with the components of $\boldsymbol{b}$ denoted as $\boldsymbol{b} = [b_X, b_Y, b_Z]$. Therefore, the components of $\boldsymbol{b}$ are [1]:

$$b_X = \frac{\sqrt{2(\sqrt{3}L_t + 3x)(r-x) - 6y^2}}{L_t^2} \tag{1.14}$$

$$b_Y = \frac{(r+x)\sqrt{2\sqrt{3}L_t + 3x)(r-x) - 6y^2}}{L_t y} \tag{1.15}$$

$$b_Z = \frac{L_t - 2\sqrt{3}r}{L_t} \tag{1.16}$$

$$\tag{1.17}$$

Where

$$r = \sqrt{x^2 + y^2} \tag{1.18}$$

The optimal position for the segment is when the top plate is positioned in such a way that that, to reach the desired end effector position, only extension from the top plate is required. This optimal position is given by [1]:

$$\boldsymbol{p}^* = \max_{\boldsymbol{p}} J \tag{1.19}$$

Where

$$J = \boldsymbol{b}^T \boldsymbol{T} \tag{1.20}$$

J can be represented as $J = J_X + J_Y + J_Z$, where $J_i = b_i T_i$ for i $\in [x, y, z]$. Then, $\frac{\delta J}{\delta x}$ and $\frac{\delta J}{\delta y}$ are calculated in order to perform the gradient descent calculation and converge on a position P that maximizes J. Given a characteristic singularity at small Y values, since b_Y has a Y term in its denominator which causes b_Y to go to infinity at small Y, b_Y was recalculated as $b_Y = \sqrt{b_X^2 + b_Z^2}$ and J_Y, $\frac{\delta J}{\delta x}$, and $\frac{\delta J}{\delta y}$ were updated accordingly.

When Y is greater than 0.1,

$$\frac{\delta J}{\delta x} = T_X \Bigg(\frac{2(\sqrt{3}L_t + 3x)(-1 + \frac{x}{r}) + 6(-x + r)}{2L_t \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{(x + r)\big(2(\sqrt{3}L_t + 3x)(-1 + \frac{x}{r}) + 6(-x + r)\big)}{2L_t y \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{(1 + \frac{x}{r})\sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}}{L_t y} - \frac{2\sqrt{3}x}{L_t r} \Bigg) \quad (1.21)$$

$$\frac{\delta J}{\delta y} = T_Y \Bigg(\frac{(-12y + \frac{2(\sqrt{3}L_t + 3x)y}{r})}{2L_t \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{(-12y + \frac{2(\sqrt{3}L_t + 3x)y}{r})(x + r)}{2L_t y \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{\sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}}{L_t r} - \frac{(x + r)\sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}}{L_t y^2} - \frac{2\sqrt{3}y}{L_t r} \Bigg) \quad (1.22)$$

When Y is smaller than 0.1,

$$\frac{\delta J}{\delta x} = T_X \Bigg(\frac{2(\sqrt{3}L_t + 3x)(-1 + \frac{x}{r}) + 6(-x + r)}{2L_t \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{\sqrt{3}L_t(x + r) - 3(2x^2 + y^2 + 2xr)}{\sqrt{2}L_t r \sqrt{\sqrt{3}L_t(x + r) - 3(r^2 + xr)}} + \frac{2\sqrt{3}x}{L_t r} \Bigg) \quad (1.23)$$

$$\frac{\delta J}{\delta y} = T_Y \Bigg(\frac{(-12y + \frac{2(\sqrt{3}L_t + 3x)y}{r})}{2L_t \sqrt{-6y^2 + 2(\sqrt{3}L_t + 3x)(-x + r)}} + \frac{y(\sqrt{3}L_t - 3(x + 2r)}{\sqrt{2}L_t r \sqrt{\sqrt{3}L_t(x + r) - 3(r^2 + xr)}} + \frac{2\sqrt{3}y}{L_t r} \Bigg) \quad (1.24)$$

Chapter 2

Experimental Design

In this project, all design and testing was performed on the second-generation TSERA stage, which has improved stiffness through higher quality materials and increased range of extension through longer linear guides. Given that the navigation architecture is currently centered around orientation control of one segment at a time, the design and testing was done on solely one segment of the TSERA.

Figure 2-1 is a diagram of the ROS architecture, with the nodes labelled and the topics highlighted. The user interfaces with the joystick and outputs a message of type Joy to topic 1, /joy. joyInterpreter receives the Joy message and converts the array of sensor data into a list of the desired T-vectors and publishes it to topic 2, /des_ort. Grad_calc receives the list of T-vectors and determines whether they are different from the previous T-vectors. If so, ten new waypoints are calculated from vector b to vector T and gradient descent is performed on each one in order to determine the position that maximizes the cost function for that T-vector. The optimal position coordinate is then published to topic 3, /des_ort_xyz. Node ik receives this coordinate and calculates the required number of revolutions in the output shaft of the stage necessary to achieve the desired scissor linkage extension. A list of required widths are then sent to the arduinoInterface via topic 4, /ik. The arduinoInterface node then uses an On-Off Controller to achieve a desired error given the desired motor setpoint.

Once the error falls below the tolerance level of 0.07 revolutions, the arduinoInterface outputs a Boolean message of True to topic 5, /continueWaypoint, in order for the following waypoint to be published. When /grad_calc receives the message, it then outputs the next coordinate to /ik. This cycle continues for all ten waypoints until the next desired orientation is inputted through the joystick by the user.

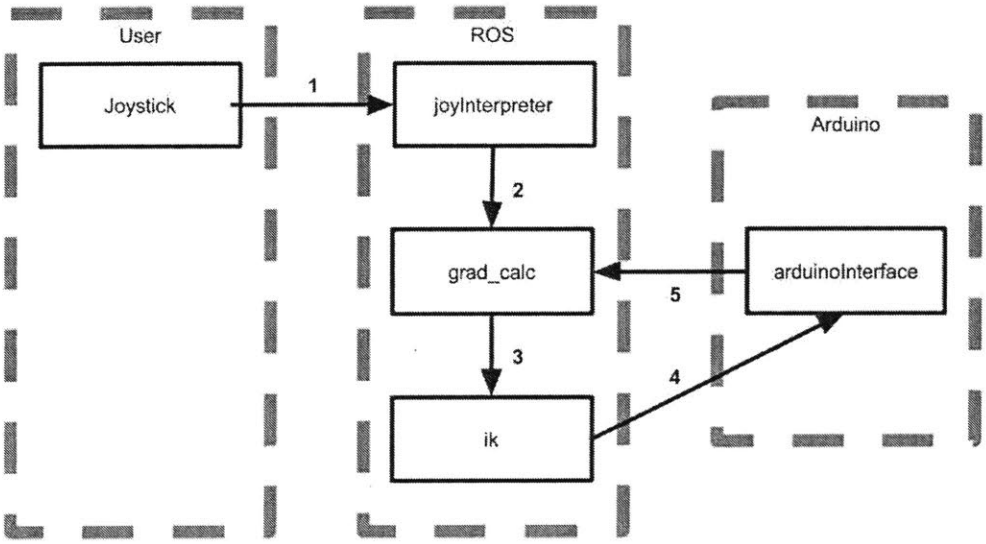

Figure 2-1: TSERA-ROS Architecture

2.1 Gradient Descent Implementation

The initial steps of designing the navigation architecture involved first calculating the derivative of the J function with respect to X and Y, as seen in equations 1.21, 1.22, 1.23, and 1.24, given . Then, ten waypoints are created between the current orientation and the desired orientation. Then, we use this derivative and a scaling

factor of 1 to implement a gradient descent algorithm. After the algorithm converges on a coordinate that maximizes the J function, then the position is saved in a list and ready to be sent later to /ik. The new position is not sent to the Arduino until the system receives a message that the error in the motors falls below the desired tolerance of 0.07 revolutions of the output shaft.

The motivation of creating waypoints as opposed to just converging on the coordinate of the desired orientation is because in different orientations, some motors may be more loaded than others, especially if the end effector is carrying any load. Therefore, because some motors may be more loaded then others, then those motors will move slower compared to the others. This presents a danger to the stages and scissor linkages because at the end effector does not have an unlimited tilt angle range. At extreme tilt angles, the end effector can crash into the scissor linkages, causing damage to the TSERA. On the other hand, not using waypoints introduces the issue that if some motors are moving faster than others, then the end effector is not necessarily moving in the desired trajectory, and potentially crash into the environment and obstacles it was supposed to avoid.

Below is a representation of the cost surface, J, in figure 2-2 and the cost contour in figure 2-2 at the desired T-vector of $T = [0.579, 0.579, 0.573]$. The singularity at small Y is visible as a vertical plane in figure 2-2 and as a solid line along the positive +x axis in figure 2-3. The cost surface is seen to be smooth in the majority of the workspace, which allows for a learning rate of 1 or greater.

2.1.1 Challenges with Gradient Descent Implementation

Equations 1.21 and 1.22 go to infinity when Y is close to 0 and at X greater than or equal to 0. This irregularity in the B-vector space is due to the 2π rotation identity in which a rotation of 2π radians around the origin is the same position. For example, for the cost surface, this coordinate has a low J value and is at an angle of 350°. For the robot however, this coordinate is at the max J value and

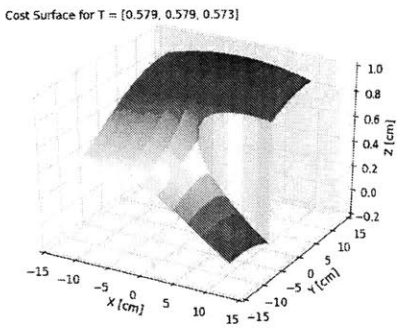

Figure 2-2: Cost Surface for Desired T-vector

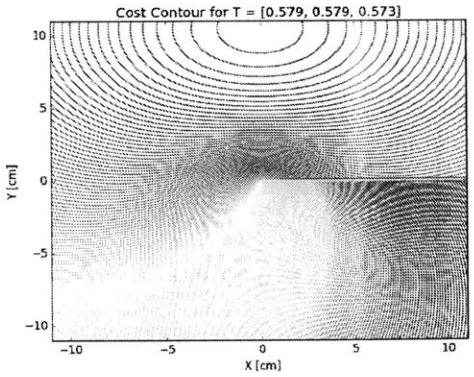

Figure 2-3: Cost Contour for Desired T-vector

is at an angle of -10°. Since the calculated cost surface only represents a range of [0°, 360°], there are orientations and coordinates that the computation cannot converge to. Instead, the solution proposed introduces a multiple of 120 degrees to the system, moving the current b-vector and desired b-vector counter-clockwise around the origin in order to bring the global maximum into the range of [0°, 360°]. This will result in a coordinate within the rotated frame. Since the angle between the different motors is 120°, the only change downstream necessary is to rotate the order of the message from /ik to /arduinoInterface in the topic /ik. This shift in the order of messages returns the desired coordinate to the original base frame. In the rotation implementation, gradient descent is performed on the rotated frame. If the gradient descent computation converges, then the final coordinate is saved and the computation continues. If the gradient descent computation does not converge, then the frame is rotated again. The process is repeated up to an additional 360°of rotation for each waypoint convergence step. Once a waypoint requires rotation of the frame, that rotation frame is kept for all future waypoint calculations and updated as needed. If a coordinate still does not converge after an additional 360°of rotations, then the waypoint is skipped and the system moves on to converge on the following waypoint. Since the system creates ten waypoints between the current orientation

and the desired orientation, skipping one waypoint will not introduce danger to the system.

2.2 Hardware

The current system is run on an Ubuntu 16.04.6 operating system with ROS Kinetic Kame. ROS interfaces with an Arduino Mega 2560 microcontroller. The Arduino Mega relays Pulse Width Modulation (PWM) signals to Cytron Dual Channel Motor Drivers, which uses a 15V switching power source to drive three 116 RPM Planetary Gear Motors per TSERA segment. These motors come with encoders which, connected to a Quadrature Encoder Buffer, provides feedback to the Arduino on angular position and speed. The controller layout is depicted in figure 2-4. The Arduino is a

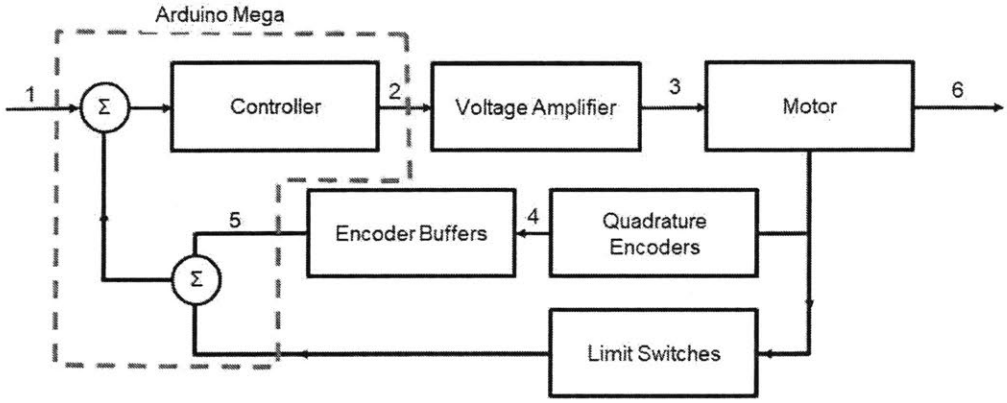

Figure 2-4: Arduino Controller System

low-level controller that receives the desired motor positions from the /ik topic and uses an On-Off Controller to drive the individual motors to a position within tolerance of the desired motor position. The On-Off Controller is designed to drive the motors

at half speed, approximately 17 rad/s, until the error falls within 0.14 rotations of the output shaft. After that, the controller drives the motor at one-fourth speed, approximately 8 rad/s, until error falls within the desired tolerance of 0.07 rotations of the output shaft. Additionally, the Arduino has a sampling frequency of 100 Hz, calculating error one hundred times every second. The voltage supplied by the On-Off Controller was reduced in order to reduce the speed of the motors so as to ensure that the waypoints are generated before the robot reaches its desired position, thus allowing for a much smoother traversal across the work space. The top plate, base frame, and lead nuts were all 3D printed on a Stratasys Dimension 1200es printer using ABS as seen in figures 2-5 and 2-6. The scissor linkages were waterjet cut from 6061 aluminum. Each DC motor drives a lead screw via the timing belt and pulley system seen in figure 2-7 [1].

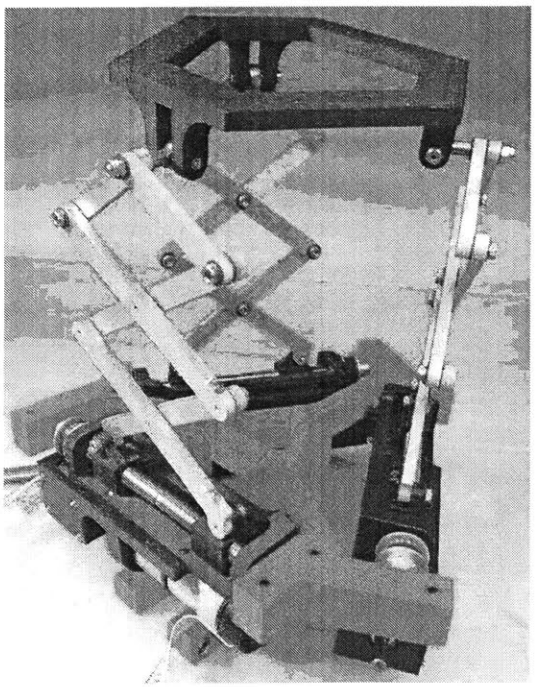

Figure 2-5: TSERA Gen. 2 Prototype

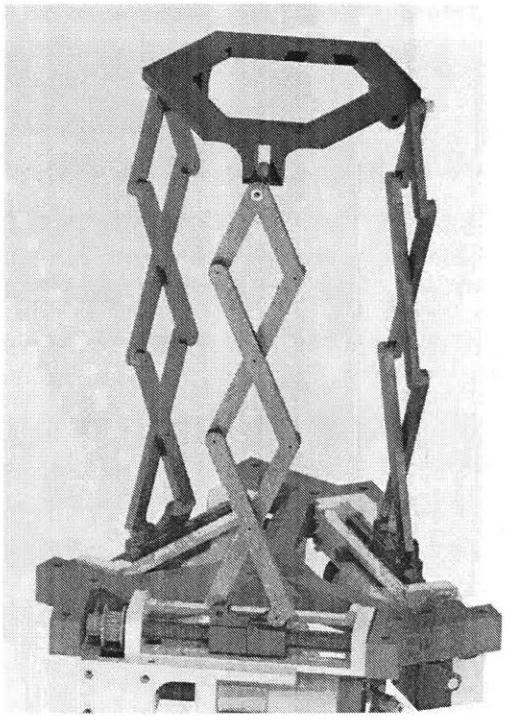

Figure 2-6: CAD Design for TSERA Gen. 2

Figures 2-7 and 2-8 depict the actuator assembly and the CAD model of the actuator assembly.

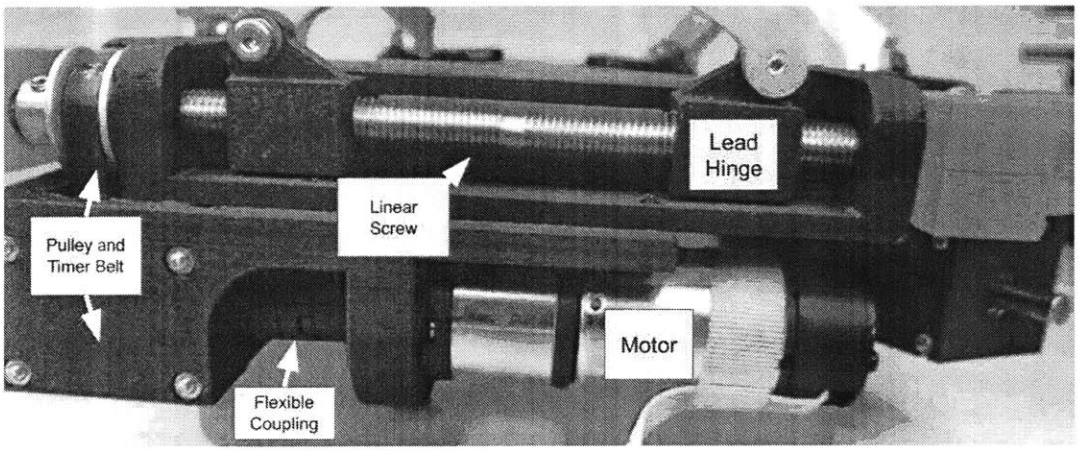

Figure 2-7: Actuator Assembly

2.2.1 Testing

To determine the prototype's workspace, theoretical estimates of the workspace are created based off Singularity Analysis [3]. This theoretical estimate is then compared to experimental results. These experimental results are collected by moving the TSERA to extreme points of its orientation and height, while still preventing damage to the motors and linkages. The range of orientations and heights for the TSERA should match with the theoretical results. Figure 2-9 is the system and the relevant parameters used to determine the workspace of the TSERA. Figure 2-10 represents the theoretical workspace of the TSERA, with a maximum tilt angle of over 110°. [3]

Typical robotic arms are also tested for their load-bearing capacity. This can be found by placing a load on the end effector and collecting motor currents in order to estimate the torque the motor is applying. However, in the case of the TSERA, this process will not deliver accurate results due to the efficiency of the system itself and the ability of the actuation mechanism. Due to friction at each of the joints of the

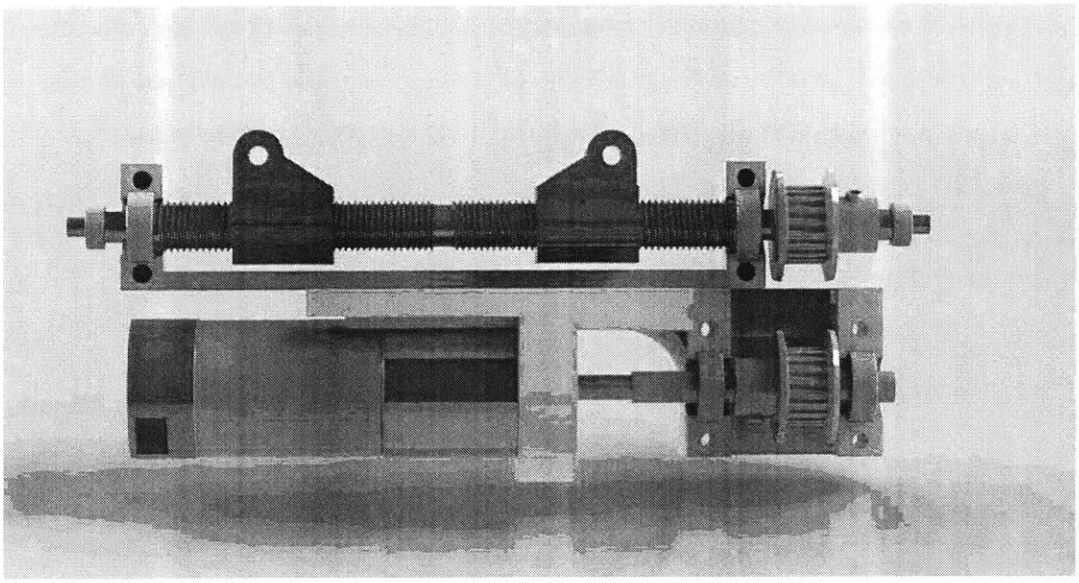

Figure 2-8: CAD of Actuator Assembly

scissor linkages and between the lead hinge and the linear guide, the motor's torques are not representative of the actual load-bearing capacity of the TSERA. Instead, the force-generating capacity can be calculated by placing a scale underneath the robot and place a rigid surface above the robot. The robot is then commanded to exert maximum torque on the motors, thus pressing the end effector against the rigid surface. The scale will then read a "weight", which is equal to the maximum force the actuators can apply and thus, the maximum load the robot can carry. The setup is depicted in figure 2-11.

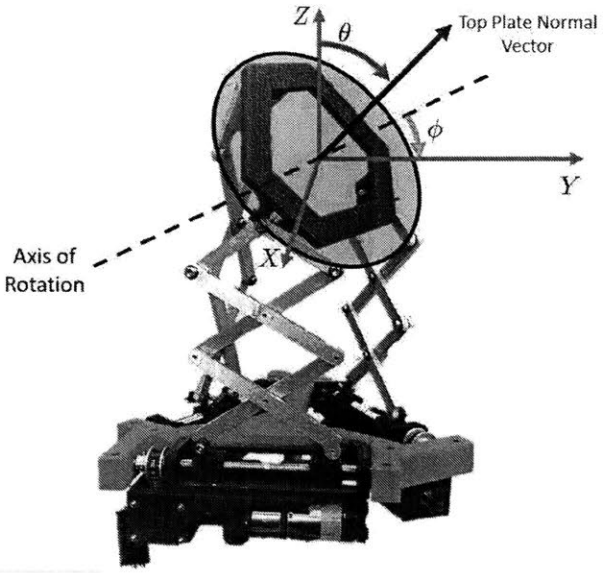

Figure 2-9: Workspace System Definition

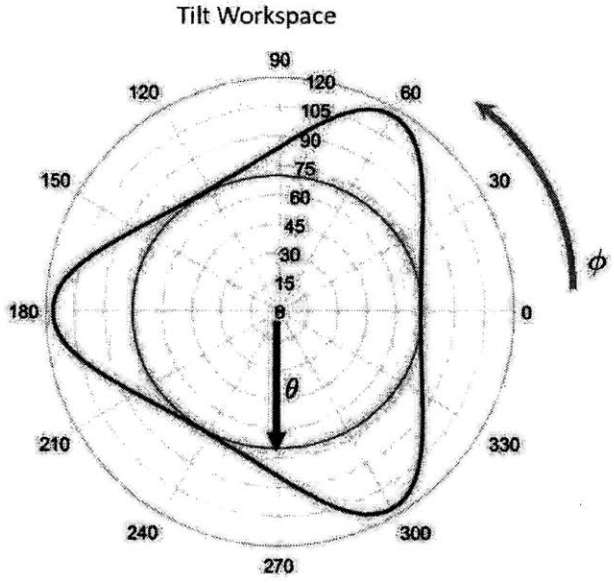

Figure 2-10: Theoretical Workspace [3]

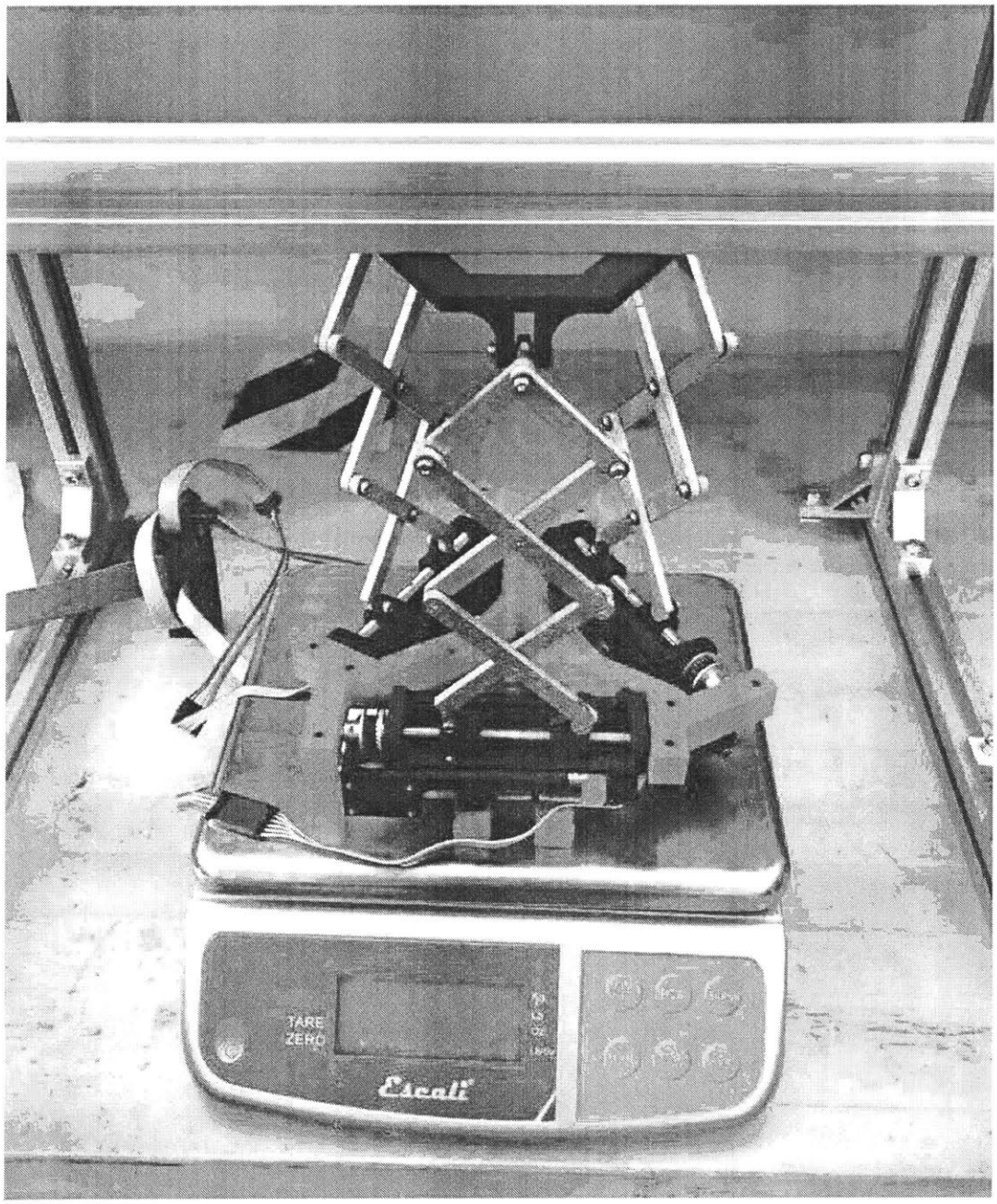

Figure 2-11: Force-Generating Capacity Experiment

Chapter 3

Results and Discussion

3.1 Gradient Descent Results

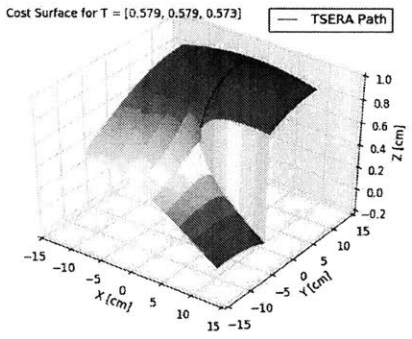

Figure 3-1: Cost Surface Traversal

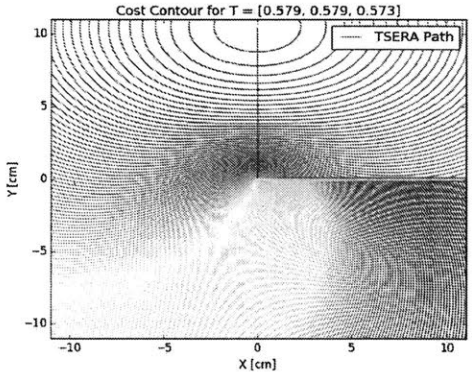

Figure 3-2: Cost Contour Traversal

As seen above in figures 2-11 and 3-1, the gradient descent solution produces a smooth cost surface for the stage to traverse, only failing for orientations that fall near the singularity range, as visible in the contour plot with a solid horizontal line and in the surface plot as a vertical plane. After the rotation solution was implemented, however, orientations within the workspace that were previously unreachable are now attainable. As seen below in figures 3-3 and 3-4, the navigation solution has proven

capable of traversing the entire surface, visible through the blue line seen traversing the entire surface in figure 3-3 and the entire contour in figure 3-4.

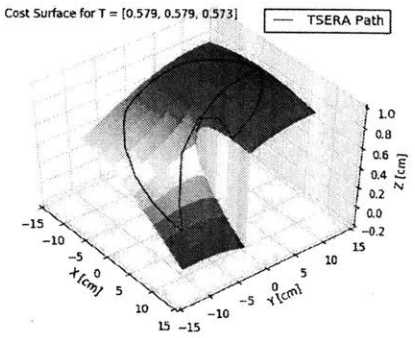

Figure 3-3: Cost Surface Traversal for Various Desired T-vectors

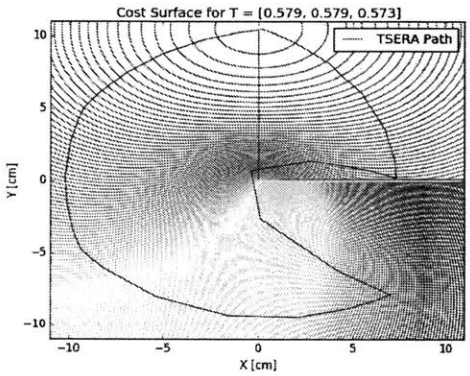

Figure 3-4: Cost Contour Traversal for Various Desired T-vectors

Traversal of the cost surface for T-vectors that require rotation are omitted due to the fact that the TSERA paths for the rotated T-vectors will not represent a smooth line, but rather would have a sharp detour at the point that is rotated to a point exactly 120°around the origin, and then continue on its trajectory. However, with the rotation implementation, the robot is able to reach the desired orientations of [0, 0.819, 0.573], [0.5, 0.5, 0.707], [0.819, 0, 0.573], [0.5, -0.5, 0.707], [0, -0.819, 0.573], [-0.5, -0.5, 0.707], [-0.819, 0, 0.573], and [-0.5, 0.5, 0.707]. This list of orientations correspond to the steepest orientations in a 360°rotation around the Z-axis of the base stage, which demonstrates the TSERA's ability to traverse its entire workspace, regardless of the singularity caused by the 2π rotation identity.

3.2 Hardware Testing Results

The workspace was experimentally determined by 1. extending one scissor linkage while the other two are held at their retracted state and 2. extending two scissor linkages while holding one at it's retracted state. Procedure 1 will demonstrate max-

imum tilt angle achievable where the axis of rotation runs parallel to the extended scissor linkage as seen in figure 3-5, resulting in a maximum tilt angle of 85°. Procedure 2 will demonstrate the maximum tilt angle achievable where the axis of rotation runs parallel to the retracted scissor linkage as seen in figure 3-6, resulting in a maximum tilt angle of 45°. Compared to the theoretical workspace calculated through Singularity Analysis, the robot has a reduced workspace. This is due in part to the use of the ball joints between the scissor linkages and the end effector. Although a ball joint does offer large ranges of rotation, the end effector's tilt angle is limited by the collision of the end effector stage and the retracted scissor linkages. This limits the theoretical workspace of the robot, yet still allows for a workspace that makes it effective for solving the last-foot problem.

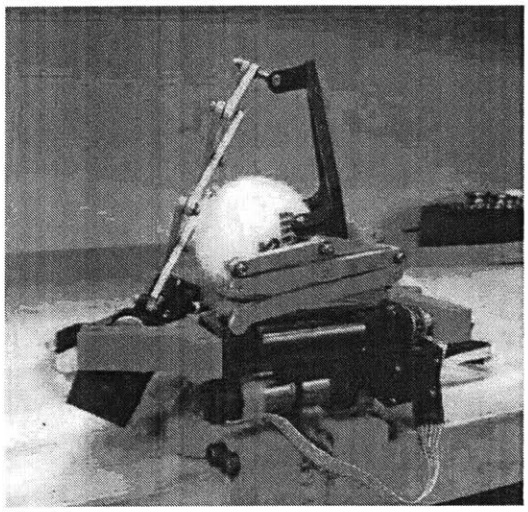

Figure 3-5: Maximum Tilt Angle in Operation 1

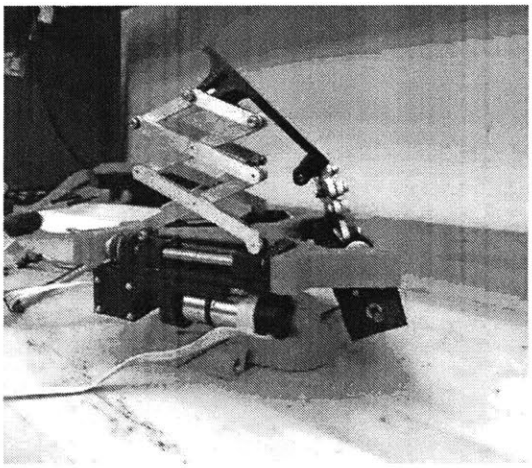

Figure 3-6: Maximum Tilt Angle in Operation 2

The maximum load the TSERA can carry is 1.5kg, which is nearly twice the weight of a single segment of TSERA itself. This was determined by applying the maximum torque possible on the motors within the previously defined setup in 2-11 and recording the output value on the scale. This maximum value is equivalent to 14.715N of Actuating Force generated. Additionally, when the Force-Generating

Capacity test was performed, it was found that the average efficiency between Applied Force from the motors to the Measured Force on the scale was 2.4% with a standard deviation of 0.3% as seen in figure 3-7.

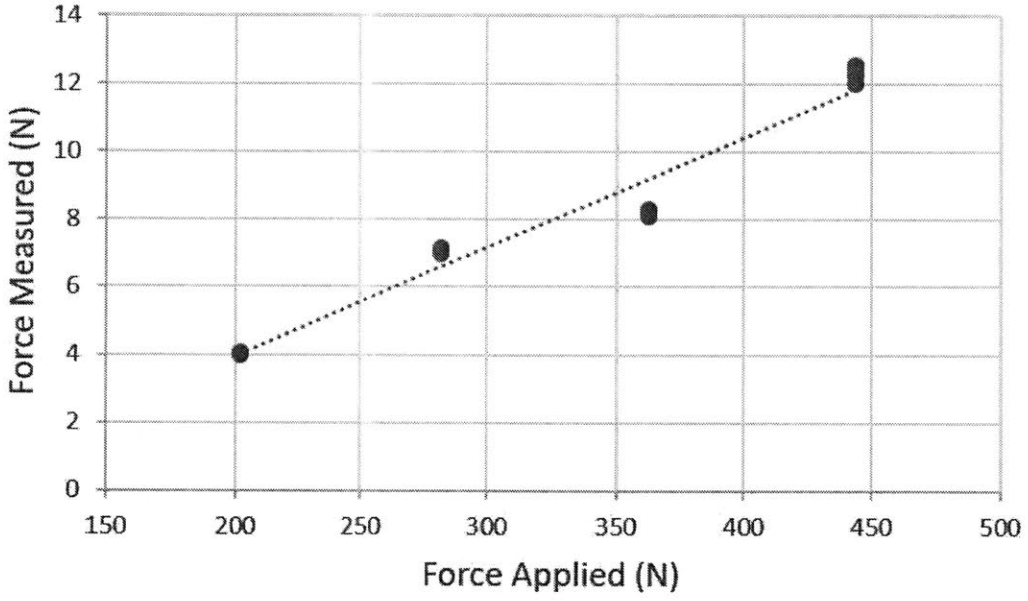

Figure 3-7: Actuated Force [N] vs. Measured Force [N]

Chapter 4

Conclusions

This paper outlines the design and accompanying navigation architecture for a new class of robotic arms capable of accessing small, confined spaces. With this updated motion planning system and updated mechanical design, TSERA is capable of achieving a greater workspace and allows the user to more intuitively control the robot through joystick control. This project brings TSERA a step closer to the manufacturing floor to solve the last-foot problem.

Future development on the TSERA will involve adding sensing capabilities as well as multi-segment navigation control in order to place the end effector at a desired position while also setting segment length and orientation for all other TSERA segments in order to optimize between different desired features, such as stiffness or compliance. This would develop the TSERA into an alpha prototype for delivery to industrial settings.

Bibliography

[1] A. Shikari and H., "Triple scissor extender robot arm: A solution to the last one foot problem of manipulation," *IEEE Robotics and Automation Letters*, vol. 3, pp. 3975–2982, July 2018.

[2] N. Donges, "Gradient descent in a nutshell." Towards Data Science, Mar. 2018. Available: https://towardsdatascience.com/gradient-descent-in-a-nutshell-eaf8c18212f0. [Accessed 15 January 2019].

[3] S. Briot and I. A. Bonev, "Singularity analysis of zero-torsion parallel mechanisms," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems Symposium on the Theory of Computing*, 2008.