VISION-GUIDED ROBOT FOR PLANETARY HABITAT ASSEMBLY

by

Kohl Alexander Whitlow

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Applied Energy and Electromechanical Systems

Charlotte

2019

ABSTRACT

KOHL A. WHITLOW. Vision-Guided Robot for Planetary Habitat Assembly (Under the direction of DR. AIDAN F. BROWNE)

The focus of this research is to develop a vision-guided autonomous system for structure or habitat construction on distant planets/moons in support of NASA's Space Technology Mission Directorate (STMD). The proposed approach would use building blocks produced via additive manufacturing from the in-situ environment and assembled using similar methodologies as used on Earth; the block creation and design are not the focus of this research. This is an alternative approach to current concepts being evaluated by NASA. A system prototype was created by mounting a robot arm with block manipulation capability atop a vector-drive robotic platform that enables the system to strafe omnidirectionally over the terrain. A vision system provides flexible peripheral input for object localization within the environment. The system carries blocks to the build location, then uses the vision system and auxiliary sensors to provide guided input for the automated piece-by-piece assembly. A prototype of a vision-guided robot for planetary habitat assembly successfully demonstrated autonomous erection of an emulated structure.

ACKNOWLEDGMENTS

I would like to acknowledge the mentorship and advising by Dr. Aidan Browne throughout my master's degree and research. Dr. Browne also provided much of the equipment and resources that were used in this concept development. I would further like to acknowledge Dr. Wesley Williams and Dr. Maciej Noras for their expertise and serving on my thesis committee. Lastly, it is important to acknowledge the financial support from the North Carolina Space Grant Graduate Research Fellowship Program for this project.

DEDICATION

I would like to dedicate this dissertation to my family and soon-to-be wife. Their support and encouragement provided me with the tools and work ethic to succeed. I am very thankful for their provision; especially from my parents for fueling my passion and educational drive throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES xii

LIST OF ABBREVIATIONS xiii

CHAPTER 1: INTRODUCTION 1

1.1 Background 1

1.2 Conceptual Overview 2

CHAPTER 2: OVERVIEW AND SYSTEMS DESIGN 4

2.1 Environment 4

2.1.1 Emulated Building Material 5

2.1.2 Loading Zone 7

2.1.3 Building Zone 7

2.2 Vehicle Overview 7

2.3 Vehicle Chassis 10

2.4 Robotic Manipulator 13

2.5 Vision System 20

CHAPTER 3: AUTONOMY AND METHODOLOGY 24

3.1 Vehicle Chassis Autonomy 26

3.2 Robot Arm Autonomy 32

3.2.1 Setting Coordinate Systems 33

3.2.2 End-effector Positioning 36

3.3 Vision System Autonomy 37

3.3.1 Machine Vision Fundamentals 38

3.3.2 Types of Vision Processing 40

3.3.3 Camera Calibration 42

3.4 Machine Vision Automation 44

3.4.1 Machine Vision Calibration 45

3.4.2 Finding Block Coordinates 46

3.4.3 Finding Block Placement Coordinates 50

3.5 Systematic Automation 54

CHAPTER 4: RESULTS AND OBSERVATIONS 56

4.1 Vector-drive Testing 56

4.2 Robotic Manipulator 57

4.2.1 Accuracy Testing 58

4.2.2 Tool Offset Mapping 61

4.3 Machine Vision Testing 63

4.3.1 Image Calibration 63

4.3.2 Edge Finding 66

4.3.3 Pattern Matching 70

4.4 Autonomous Structure Assembly 79

CHAPTER 5: CONCLUSIONS 85

5.1 Vehicle Chassis 85

5.2 Vision Guidance 85

5.3 Robotic Manipulator 86

5.4 Structure Assembly 87

5.5 Future Work and Other Applications 88

5.6 Summary 89

REFERENCES 90

LIST OF TABLES

TABLE 3-1: Edge finding settings used in the initial block finding function 48

TABLE 3-2: Edge finding settings used in the laser line finding function 51

TABLE 3-3: Edge finding parameters used in the corner finding function 52

TABLE 4-1: Robot arm positioning test data 61

TABLE 4-2: Tool offsets 62

TABLE 4-3: 500 mm calibration results 63

TABLE 4-4: 350 mm calibration results 65

TABLE 4-5: Laser line finding results 68

TABLE 4-6: Corner finding results 70

TABLE 4-7: Half-block pattern matching results 79

TABLE 4-8: Structure completion error 81

TABLE 4-9: Structure completion time 82

LIST OF FIGURES

FIGURE 2-1: Emulated building environments 5

FIGURE 2-2: Emulated building material representation 6

FIGURE 2-3: Complete robotic platform 8

FIGURE 2-4: System control strategy 9

FIGURE 2-5: Mecanum wheel diagram [8] 10

FIGURE 2-6: Vector-drive base configuration 11

FIGURE 2-7: Vector-drive base loaded with two blocks 12

FIGURE 2-8: DENSO available reaching area coplanar to the robot base [10] 14

FIGURE 2-9: DENSO available reaching area perpendicular to robot base [10] 15

FIGURE 2-10: DENSO axis labeling [9] 16

FIGURE 2-11: DENSO end-effector positioning time vs payload weight [9] 17

FIGURE 2-12: Gripper and finger assembly. 18

FIGURE 2-13: Gripper interfacing with emulated building material 19

FIGURE 2-14: Demonstration of the robotic arm in pick and place positions 20

FIGURE 2-15: 3D model of the camera mount 21

FIGURE 2-16: Ring lighting configuration for the camera system [12] 22

FIGURE 3-1: Automation overview 25

FIGURE 3-2: Vector-drive flowchart of operations 27

FIGURE 3-3: Ramp input (blue) vs step input (orange) 29

FIGURE 3-4: Shark Y0A21 analog response 31

FIGURE 3-5: VL6180X actual distance vs measured distance 32

FIGURE 3-6: DENSO manipulator coordinate reference [9] 34

FIGURE 3-7: Mechanical flange axes without TOOL axis transformation [9] 35

FIGURE 3-8: TOOL coordinate offset transformation [9]... 35

FIGURE 3-9: Pick/Place program operation 37

FIGURE 3-10: RGB (left), greyscale (middle), and binary (right) 39

FIGURE 3-11: Edge finding illustration [17].. 41

FIGURE 3-12: Pinhole camera technique representation.. 43

FIGURE 3-13: Dot-Grid image calibration distortion representation 46

FIGURE 3-14: Block finding flowchart of operations 47

FIGURE 3-15: Initial corner finding algorithm finding three corners 49

FIGURE 3-16: Half and full-block successful pattern match.. 50

FIGURE 3-17: Block placement flowchart of operation... 52

FIGURE 3-18: Block placement calculation methodology 53

FIGURE 3-19: Complete robotic system used in testing... 55

FIGURE 4-1: Motor throttle and RPM over time... 57

FIGURE 4-2: Gripper-mounted needle test apparatus... 59

FIGURE 4-3: Camera stand for measuring the hole diameter... 59

FIGURE 4-4: Position error measurements... 60

FIGURE 4-5: Robotic manipulator positioning test setup... 62

FIGURE 4-6: Error plot at 500 mm vertical offset.. 64

FIGURE 4-7: Error map with 350 mm vertical offset... 65

FIGURE 4-8: Initial block finding function success 67

FIGURE 4-9: 500 mm and 350 mm image perceptions 69

FIGURE 4-10: Normal distribution of block coordinates 71

FIGURE 4-11: Normal distribution of block rotation 72

FIGURE 4-12: Match score relative to the position, and angular rotation....................... 73

FIGURE 4-13: Processing time vs block color.. 74

FIGURE 4-14: Processing time vs block angle experiment 75

FIGURE 4-15: Processing time vs block position experiment.. 76

FIGURE 4-16: Reduced processing time comparison... 76

FIGURE 4-17: Scatter plot of located block positions 78

FIGURE 4-18: Structure start and end error diagram.. 80

FIGURE 4-19: Position one encoder count and object distance relative to set points 83

FIGURE 4-20: Final position encoder count and object distance relative to set points... 84

LIST OF ABBREVIATIONS

3D	Three-Dimensional
AM	Additive Manufacturing
cm	centimeter
CVS	Compact Vision System
dc	Duty Cycle
DC	Direct Current
DOF	Degrees of Freedom
FOV	Field of View
GHz	Gigahertz
HARP	Habitat Assembly Robot Prototype
IMAQ	Image Acquisition
IR	Infrared
I^2C	Inter-Integrated Circuit
kg	kilogram
m	Meter
mm	Millimeter
ms	Millisecond
NASA	National Aeronautical and Space Administration
NI	National Instruments
PDF	Probability Density Function
RAM	Random Access Memory
ROI	Region of Interest

ROM	Range of Motion
STMD	Space Technology Mission Directorate
TOF	Time of Flight
TRL	Technology Readiness Level
V	Volt

CHAPTER 1: INTRODUCTION

1.1 Background

Additive manufacturing (AM), also known as 3-dimensional (3D) printing, has become a proven and innovated field. AM has resulted in copious research related to printing construction materials from the in-situ environment [1]. Stemming from this concept, the National Aeronautical and Space Administration (NASA) presented the Centennial 3D-Printed Habitat Challenge. This challenge hosted contestants from around the world to develop a prototype system to 3D print habitats; specifically, space exploration habitats using emulated recycled building material [2].

NASA's mission to send humans back to the moon by the year 2024 for long term expeditions outside the Earth's atmosphere is ambitious and will require technological advancements. Most important of these advancements is preparing suitable living and working environments for humans on long term expeditions. Current concepts use mobile 3D printers and mixing containers to ultimately print these enclosures from the ground up [3]. However, "…it is impractical to have printers that are larger than actual buildings" [4], much less, transport this machine to another planet. This research explores the possibilities of a different method for habitat construction in extra-terrestrial environments by focusing on manipulating manageably-sized building blocks and assembling them into living quarters via an autonomous robotic system. Exploiting smaller-sized building materials rather than attempting to 3D-print entire structures, it is possible to produce a smaller, multipurpose platform that assembles habitats block-by-block. To accomplish this, blocks could be manufactured via 3D-printing materials supplied by the in-situ environment. The emulated building articles would be of block-like form and similar in shape and size to traditional blocks (bricks or cinder blocks) used on Earth. However, the design and manufacturing of these blocks are not a focus of this research. The overall objective of this research is to develop a vision guided autonomous system for structure assembly on distant planets/moons and demonstrate using emulated building blocks to erect a structure for a proof of concept system.

1.2 Conceptual Overview

It is assumed that upon arrival to the habitat construction zone, pre-determined building data will be available to the autonomous system such as “blueprints” or structure location. Functionally, the proposed system would start at a base station, or manufacturing hub, to be loaded with building blocks. From there, it would travel to a predetermined building zone where the structure is to be constructed. Here, it would use the “blueprint” to start the assembly process. Block-by-block, the system would assemble the structure, returning to the base station when it needs to be reloaded with building blocks. The overall process is conceptual and can be altered to suit the situational environment.

Radio-controlled robots on the Moon would be limited in their abilities due to latency issues caused by the long-distance from Earth. Attempting to remotely control a robotic system on other planets from the Earth would be even more difficult due to increased latency, greatly hindering the controllability of the vehicle. For this reason, such a system will be required to operate completely autonomously since its primary purpose is to assemble habitats for humans *before* they arrive on distant planets. This is a difficult challenge considering the ever-changing environments that will be forced upon the autonomous system. One of the best countermeasures for such a vehicle with these considerations is to incorporate machine, or computer vision into the autonomy [5]. Adding this degree of peripheral input can greatly impact the abilities of the system; the vehicle developed in this research will be largely dependent on machine vision input for object localization. The vision system could later be enhanced to incorporate obstacle avoidance to expand overall autonomous behavior.

Since the blocks proposed in this research are sized similar to those used on Earth, it is appropriate to consider the benefits of robotic arms that generally employ a high degree of freedom and work well in “pick-and-place” applications. When coupled with machine vision, such a system is advantageous and can serve a variety of different purposes. Mounting the machine vision camera on the end-effector of the robotic manipulator allows the camera to move in three dimensions to better view its environment and also reduces the number of cameras required for autonomy.

Complementing the vision-guided robotic manipulator, the system requires mobility for translating the surface so that the structure can be assembled from the ground up. Allowing the robot to roam freely on the surface will greatly increase its capabilities, such that it could assemble multiple structures and perhaps be repurposed for other tasks upon completion of its mission. This concept reduces the required overall size and complexity of a vehicle which is ideal for space applications due to reduced costs and decreased failure modes.

CHAPTER 2: OVERVIEW AND SYSTEMS DESIGN

A prototype system has been constructed as illustrated in this overview to complete the concept described in Chapter 1. The robotic system is broken down into three main subsystems: the vehicle chassis, robotic manipulator, and the vision system. Basic terminology used to describe components of the vehicle and are:

- Vehicle Chassis – vector-drive providing system mobility
- Robotic Manipulator – robotic arm used for block manipulation and camera positioning
- Vision System – camera and vision processing hardware providing machine vision feedback to the manipulator controller for object localization
- Material Bed – the surface on the rear of the vehicle that carries the material
- End-effector/Gripper – the tool used to interface the manipulator with objects
- HARP – Habitat Assembly Robot Prototype

All subsystems are assembled to create the vision-guided robot for planetary habitat assembly; each subsystem contains various supporting hardware and will be covered in the following overview. This simulated prototype for this system will be referred to as the habitat assembly robot prototype (HARP).

2.1 Environment

To test the HARP, a simulated building environment was created in a laboratory. The general layout of this environment is pictured in FIGURE 2-1. Terminology used to describe this work environment is as follows:

- Loading Zone – location at which vehicle is loaded with material
- Building Zone – location at which the robot will build the structure
- Simulated Building Blocks – blocks used to emulate AM building material
- Structure – arrangement of blocks to simulate habitat construction
- Laser Line – designation as to where the structure should be completed

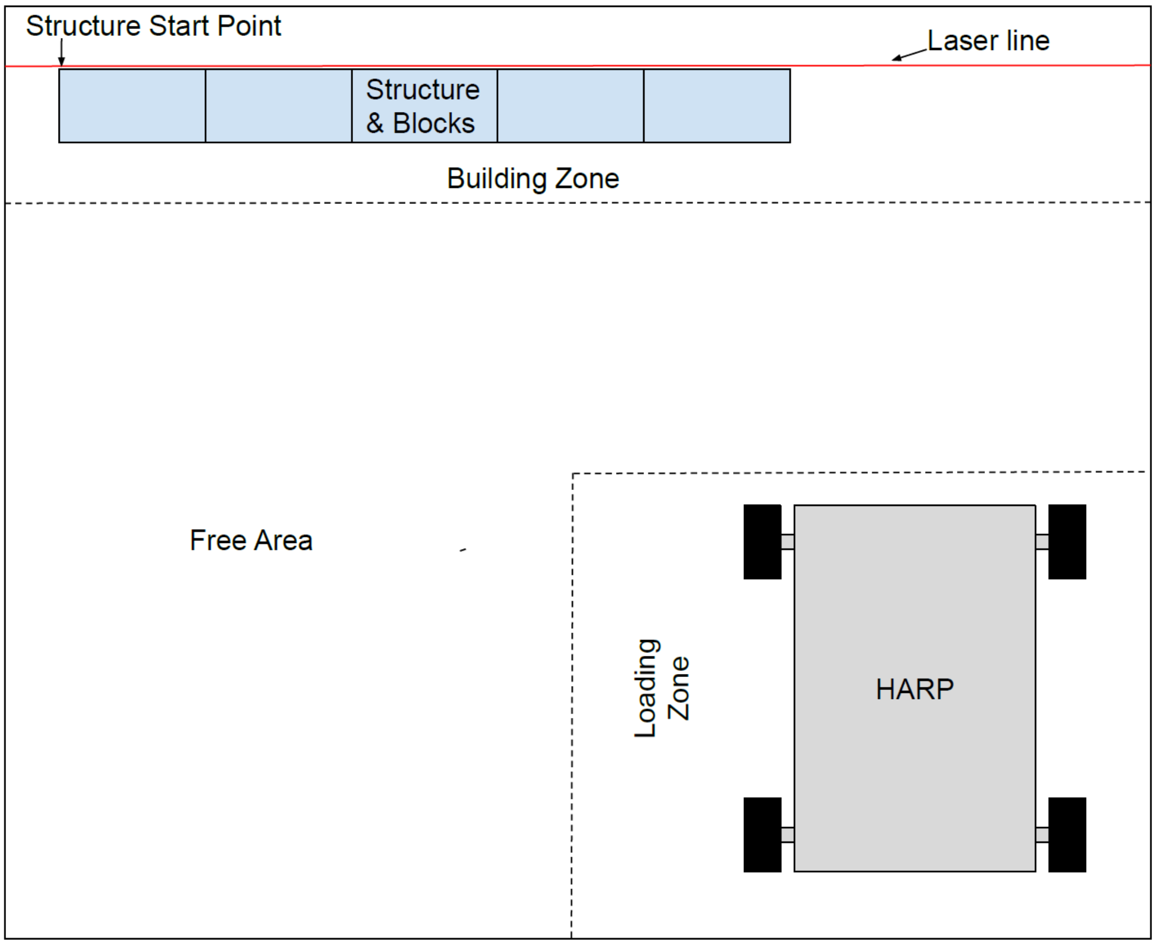


FIGURE 2-1: Emulated building environments

2.1.1 Emulated Building Material

The term "building block" will be used to describe the emulated building material throughout this document. The emulated building blocks used in this research are for proof of concept of physical 3D-printed block manipulation. On Earth, humans have used modular sized building blocks for centuries in structure assembly [6]. Continuing use of this technology from the construction industry, the building blocks can be assembled in a half-over bonding fashion, and corners can be built in overlapping uniformity for strength. The physical blocks used for the concept are represented in FIGURE 2-2 and merely represent the concept of the building material; whole blocks are 100 mm wide, 200 mm long, and 65 mm tall, and half-size blocks are 100 mm wide by 100 mm long by 65 mm tall. They closely resemble the size and shape of the proposed building materials, however, they do not represent the weight of the actual blocks that may be used. Also, the size and shape could vary dependent on the needs of the structure.

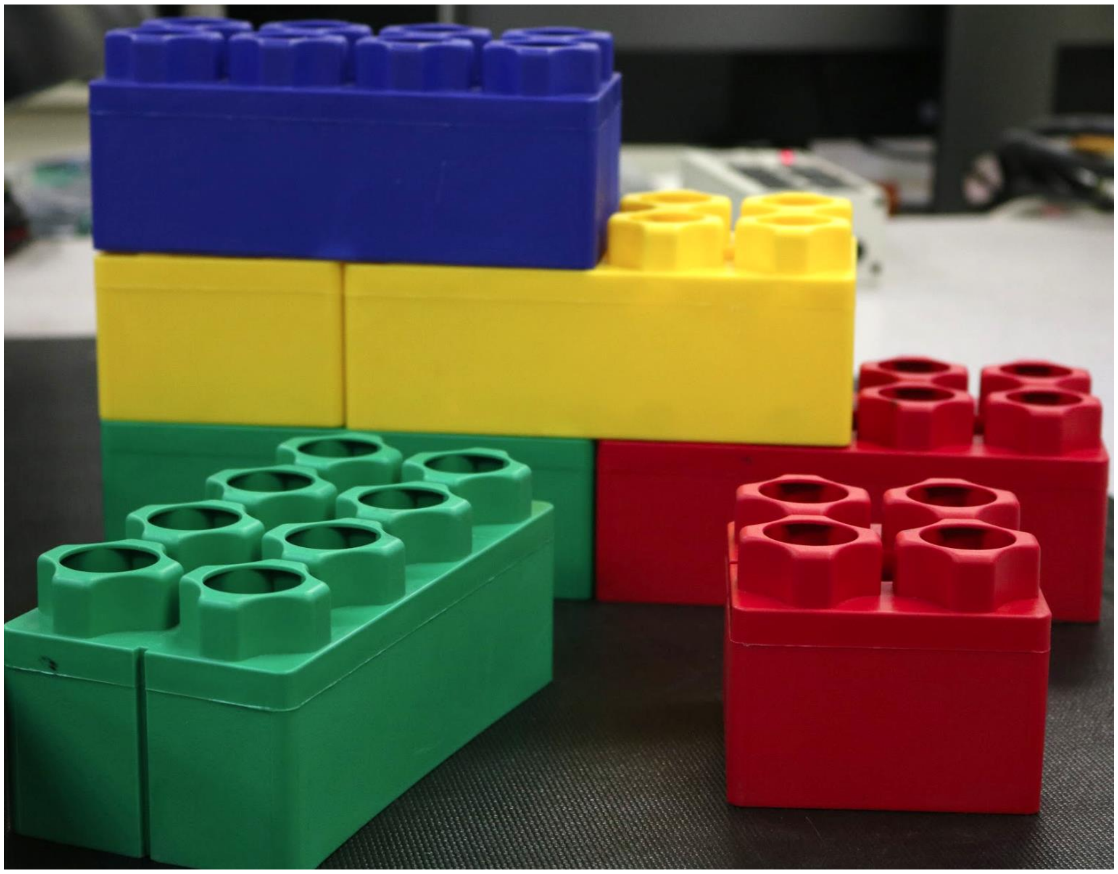


FIGURE 2-2: Emulated building material representation

2.1.2 Loading Zone

The Loading Zone is the location at which point HARP awaits to be loaded by an external device as labeled in FIGURE 2-1. This will serve as the theoretical location of the manufacturing area producing the blocks for the construction project. The overall concept would use an automated system to load pre-formed blocks onto the robot; however, for the purpose of this experiment, a human agent emulates the loading system by manually loading three blocks within $\pm 25°$ rotation on the material bed.

2.1.3 Building Zone

The “Building Zone” as presented in FIGURE 2-1 is where the actual structure is to be assembled. This is the location where HARP initially drives, and where it returns to after each material reload. At this location, a laser line is projected onto the surface and signifies where the blocks should be placed. It is assumed that the laser will be placed by another system in advance of commencing construction; it serves as a reference for the HARP.

2.2 Vehicle Overview

The vehicle is demonstrated in FIGURE 2-3 with all functional hardware loaded. The placement of these features allows the robot arm to rotate from the material bed to the build plane easily. Since the robot arm is mounted on the fore-end of the base, the heavy battery used for power was placed under the material bed on the opposite (aft) end as a counterweight to maintain stability and keep the center of gravity within the wheelbase of the vehicle. Also pictured, the camera is mounted to the end-effector alongside the manufactured gripper. The robot arm controller and vision controllers are mounted off of the vehicle and tethered for the proof of concept system due to laboratory configuration.

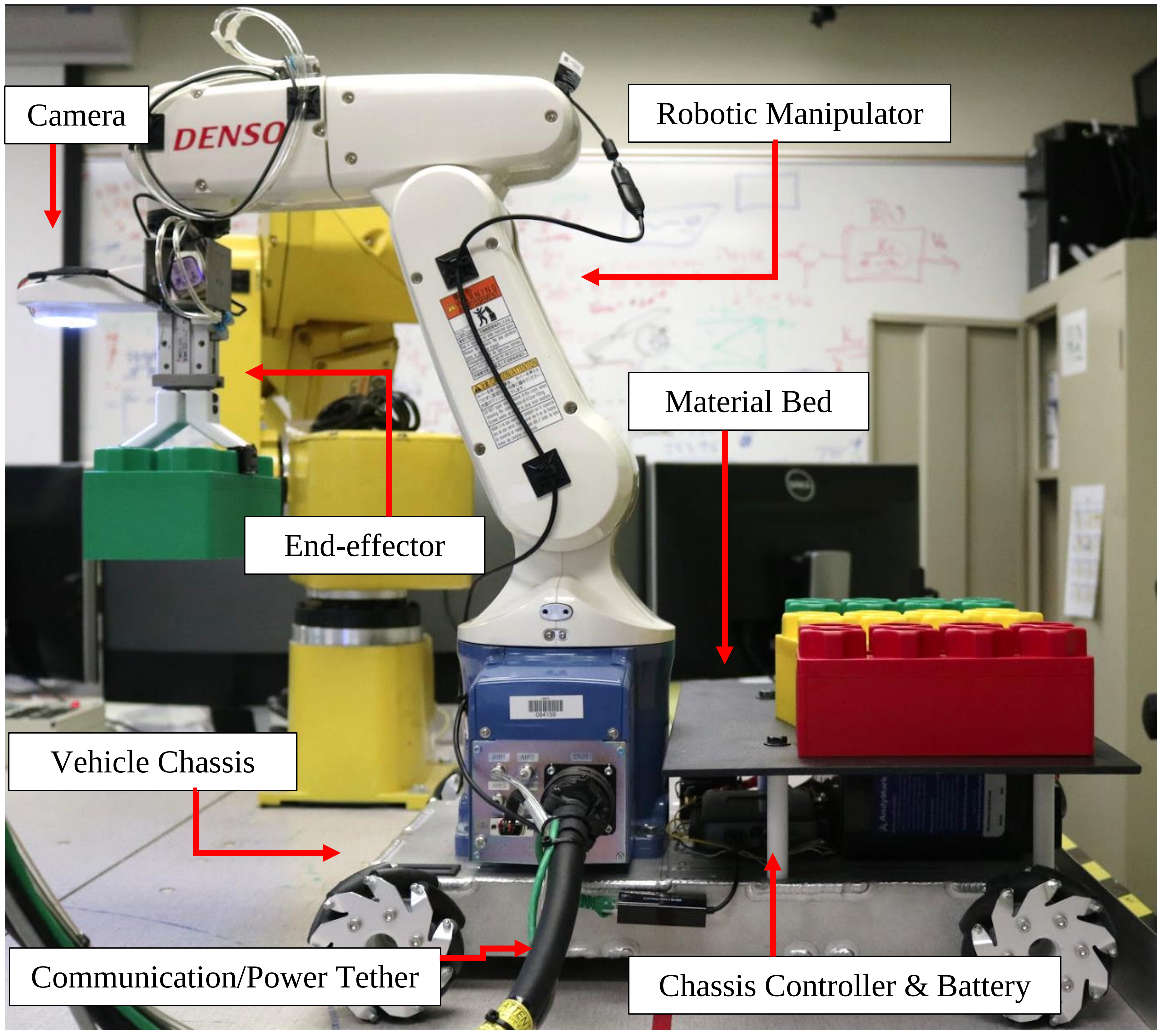


FIGURE 2-3: Complete robotic platform

An outline of the communications/control strategy is demonstrated in FIGURE 2-4. The vision system and chassis controllers are the main processing units. The vision system's main objective is image processing and manipulator positioning. However, it does not process robot arm path planning but passes positioning data to the robot arm controller for positioning computation. The chassis controller is employed purely for vehicle position control and sensory input. Each of the motors are equipped with quadrature encoders to provide dead reckoning feedback, while the short rangefinder and long rangefinder are fused for frontward object detecting. The vision system and chassis controller communicate bi-directionally via Ethernet communication relaying vital process status and commands.

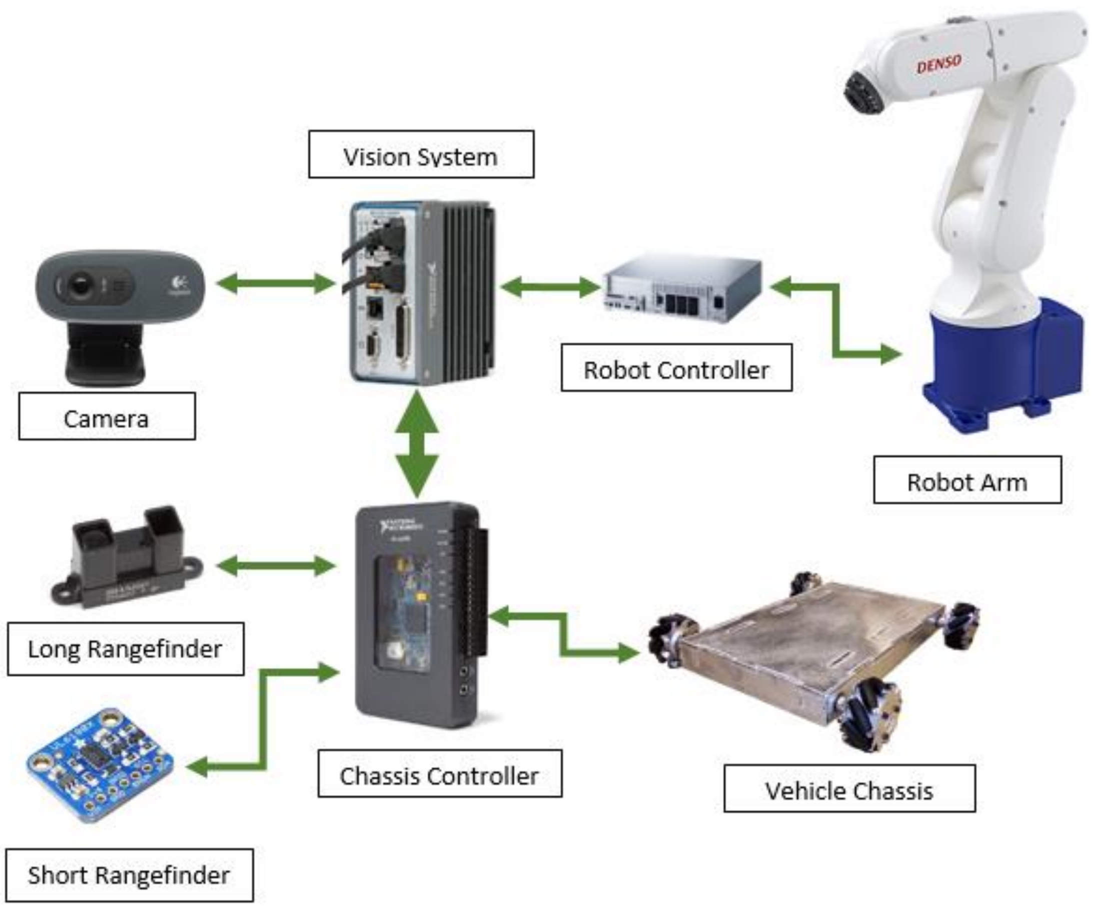


FIGURE 2-4: System control strategy

2.3 Vehicle Chassis

The objective of the vehicle chassis is to provide a means of transportation for building material, the manipulator, and the imaging hardware. It employs wheels from the Swedish drive family, also known as mecanum wheels. The mecanum wheel is based on the principle of a central hub with a number of rollers placed at an angle around the periphery of the hub” [7]. The wheel featured on this system is made up of one large outer wheel and nine rollers angled at 45° from the tangent of the outer wheel diameter; a similar model of the wheel used can be seen in FIGURE 2-5 from [8].

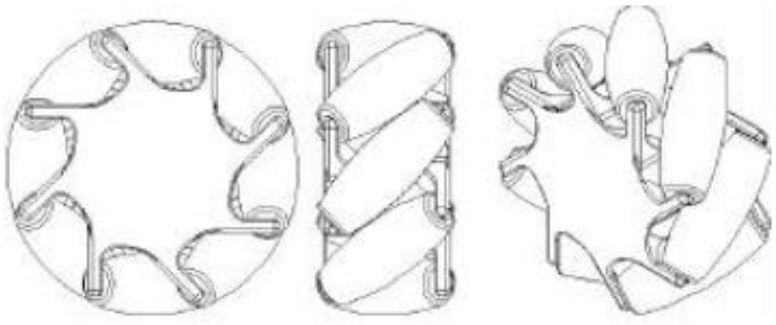


FIGURE 2-5: Mecanum wheel diagram [8]

The mecanum wheels allow the vehicle to move in an omnidirectional fashion, making it ideal for traveling in an efficient, vector-like movement. The chassis employs four mecanum wheels mounted in parallel sets on the rectangular base platform. A SuperDroid high payload platform and wheel system was purchased for this application, illustrated in FIGURE 2-6. The vehicle chassis has a 50.8 cm square-shaped footprint. This configuration was chosen because it provides a stable platform for the robotic manipulator while maintaining omnidirectional strafing abilities, and sufficient payload space.

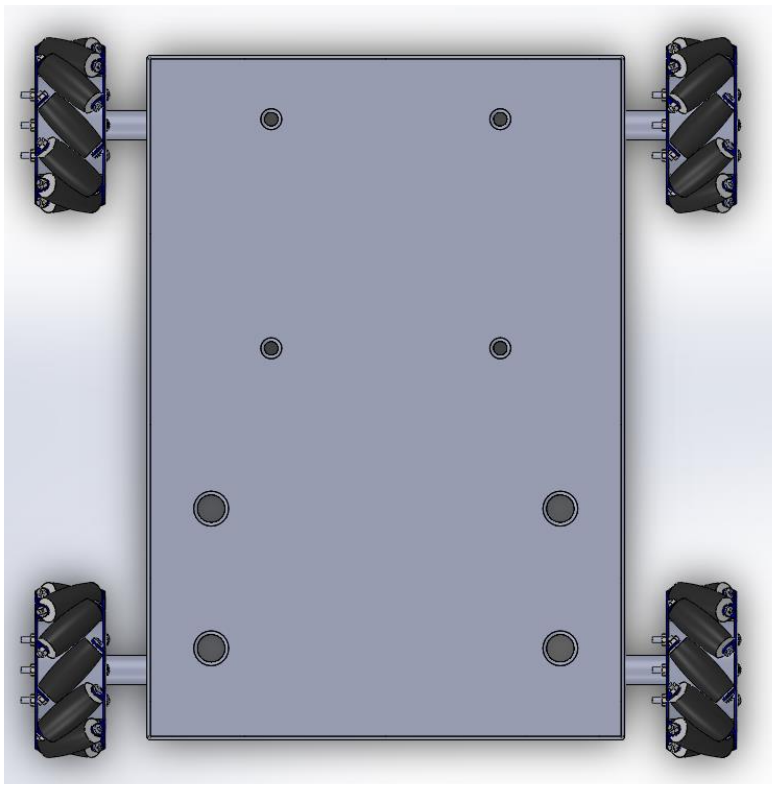


FIGURE 2-6: Vector-drive base configuration.

The material bed is mounted on the rear of the vehicle, as seen in FIGURE 2-7, to carry blocks and serve as a counterbalance; it also provides the vehicle with resources for consecutive block place/build operations. The material bed measures 300 mm deep and 500 mm wide to allow up to three blocks to be transported. The surface of the material bed was painted black to increase the contrast of the blocks with respect to the material bed to assist image processing.

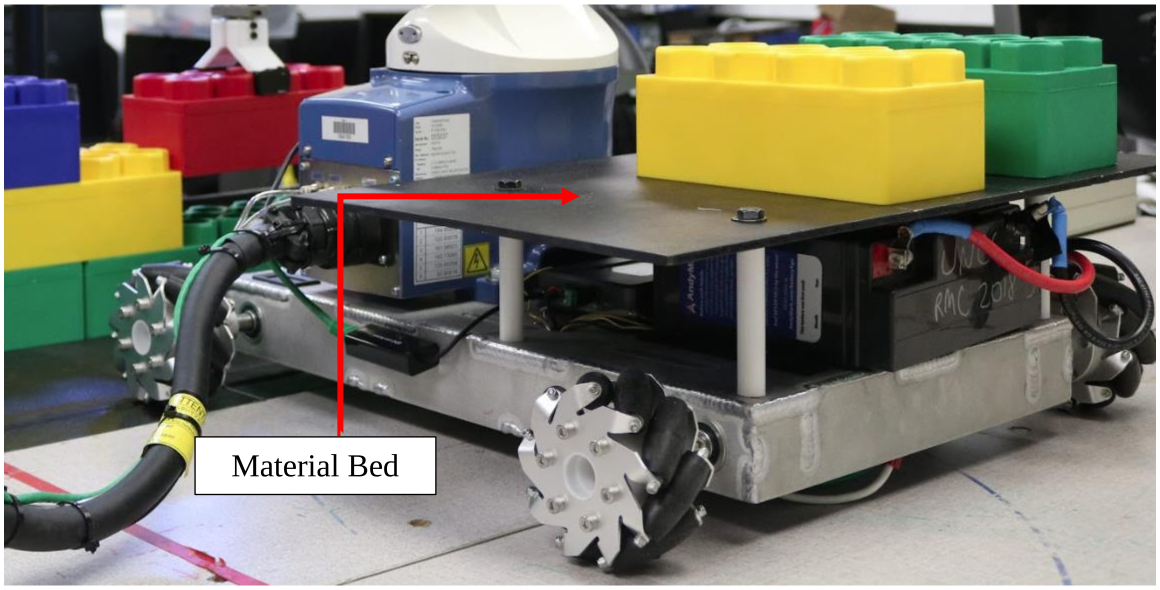


FIGURE 2-7: Vector-drive base loaded with two blocks

To accurately control system movement, a suite of electronics was chosen consisting of a National Instruments (NI) myRIO-1900, four Talon SRX motor controllers, four direct current (DC) motors with quadrature encoders, one Adafruit VL6180X digital time-of-flight rangefinder, and one Shark 2Y0A21 rangefinder. The myRIO controller provides up to 62 general-purpose input/out (GPIO) pins, a regulated +5V source, and $\pm$15V sources. Each motor provides a torque of 8.0 kgf-cm at 12 V, and a speed of 74 revolutions per minute (RPM) output which is sufficient for the payload (~17 kg) and desired controllability. The quadrature encoders are mounted on each motor providing feedback to the myRIO for position and speed feedback control. The Talon SRX speed controllers enable ramped control of the motors via independent pulse width modulation (PWM) signals. Lastly, the VL6180X and Sharp Rangefinders are mounted on the fore-end of the vehicle, opposite of the material bed. Each range finding sensor provides forward obstacle detection and ranging to assist in collision avoidance with the structure-under-construction or any other forward-facing objects. The analog output

Sharp IR sensor was chosen for its ability to sense distances up to 1.2 m. The VL6180X sensor was chosen based on its much finer resolution and accuracy at distances less than 200 mm.

2.4 Robotic Manipulator

The robotic manipulator used in this research is the DENSO VP-6242. Its main objectives are to provide camera positioning and manipulation of the building blocks in the assembly process. The DENSO module consists of a robotic arm with 6 degrees of freedom (DOF) and a 2.5 kg payload capacity, including any end-effector tool attachments. Also encompassed on the arm module are flexible end-effector mounting points and internal pneumatic piping for pneumatically actuated end-effectors. In its default configuration, the DENSO has a maximum reach of 432 mm from the center of the module base. Further defining the VP-6242 range of motion (ROM), the first rotational axis of the arm allows the unit to position the end effector $\pm 160°$ from the front axis of the base, opposite from the power tether as described in FIGURE 2-8. The remaining workable area of the system is demonstrated by the shading in FIGURE 2-9. These features define the flexibility of the system for the programming approach and how the vehicle will need to be positioned. All resources on the DENSO system are sourced from the DENSO VP-6242 Robot User Manual from [9] and [10].

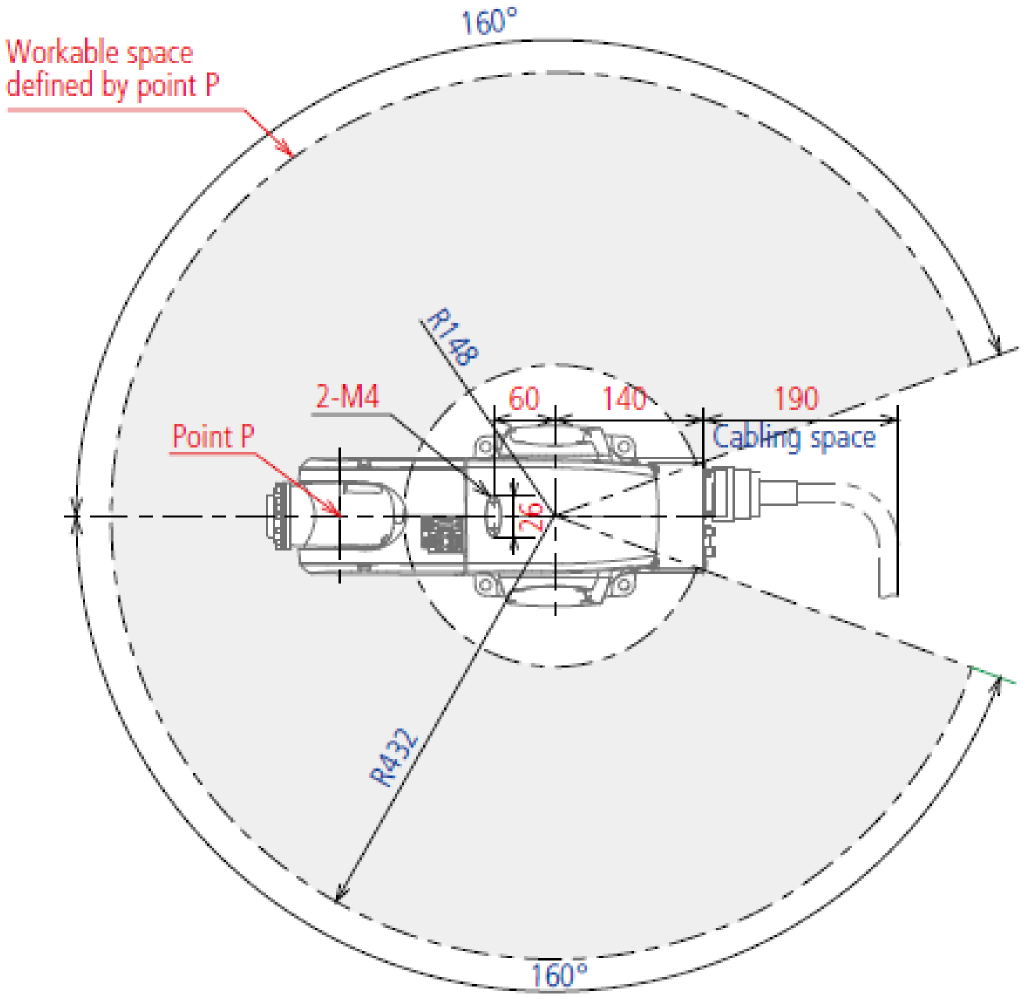


FIGURE 2-8: DENSO available reaching area coplanar to the robot base [9]

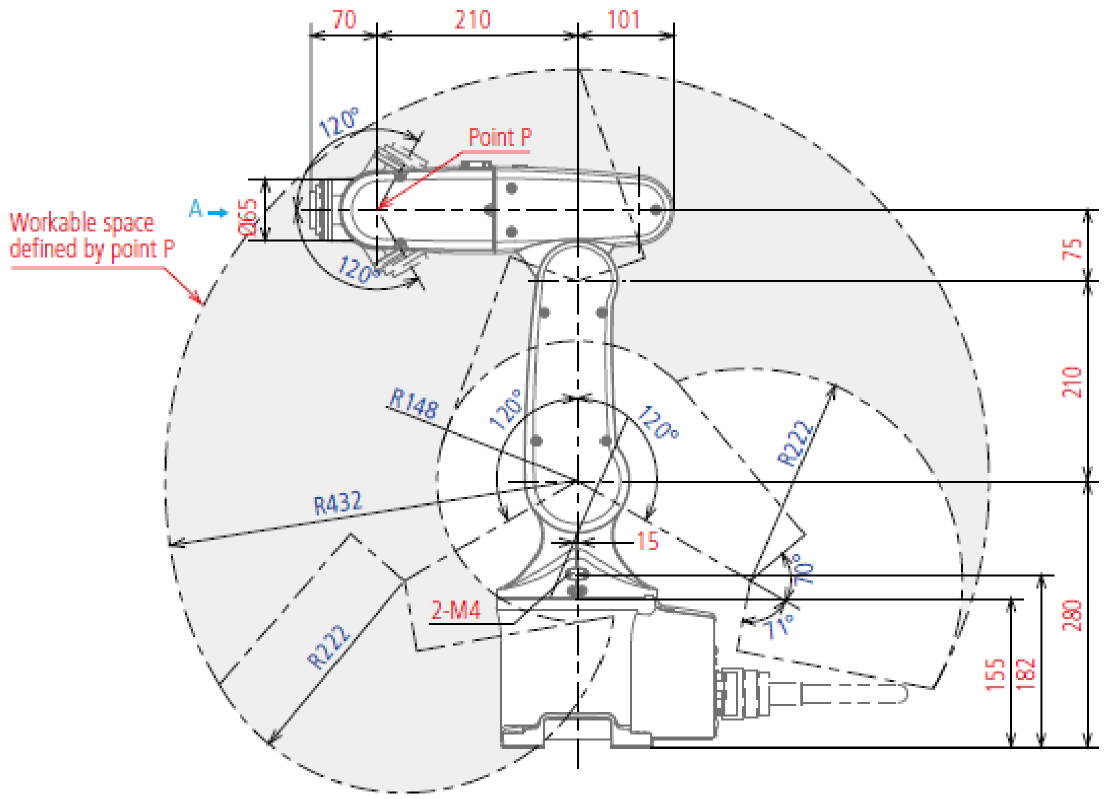


FIGURE 2-9: DENSO available reaching area perpendicular to robot base [9]

Joints on the VP-6242 are documented from the base up, starting at the first rotational axis as, $J_1, J_2, \dots J_6$ format. The end-effector mounting face enables the user to add a seventh DOF to the manipulator for axis J_7. DENSO documentation describes these joints with the graphic in FIGURE 2-10. FIGURE 2-11 represents the positioning time of each joint associated with the manipulator with respect to end-effector payload. Here, it can be seen that payload mass can affect the positioning time as much as 0.2 seconds. The characteristics presented in FIGURE 2-11 are important to consider when programming payload placement as it could affect the accuracy of placement by moving/departing the placement coordinates before it is acceptable. The total payload of the configuration used for both hardware and blocks is 0.95 kg and can affect positioning time up to 420 ms per 100 mm.

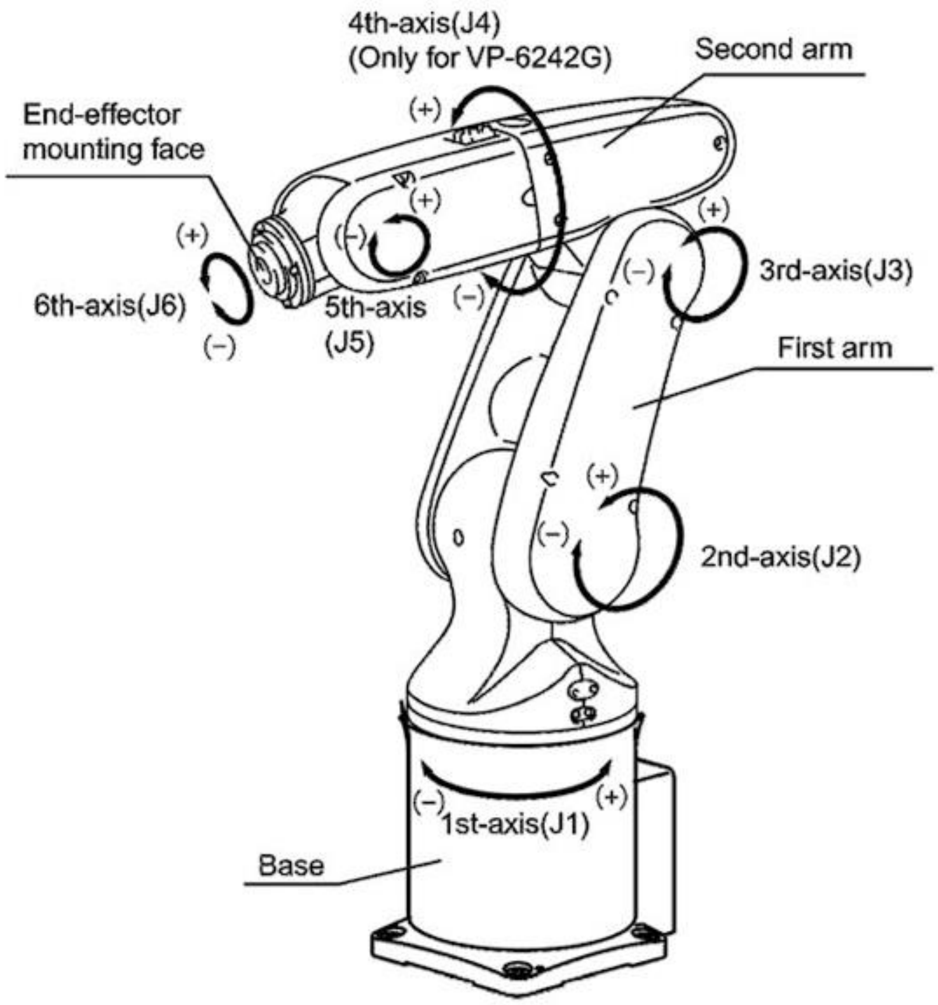


FIGURE 2-10: DENSO axis labeling [10]

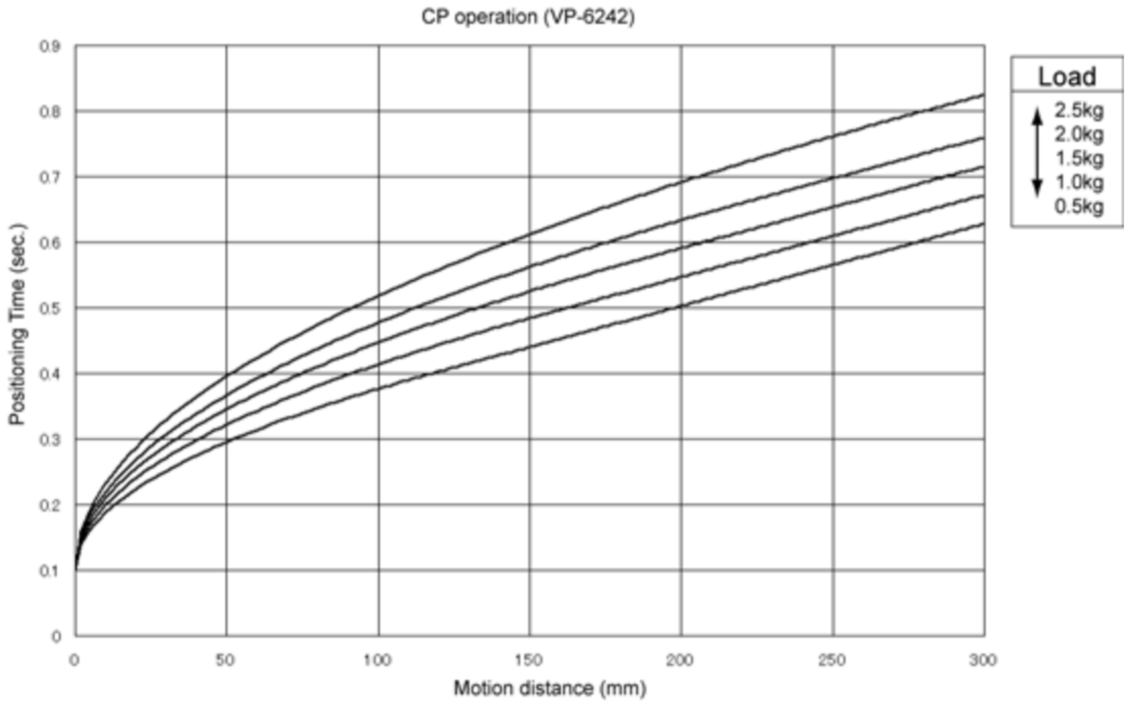


FIGURE 2-11: DENSO end-effector positioning time vs payload weight [10]

Per the manufacturer specifications, the manipulator is able to repeatedly position the end-effector in free space to ± 0.02 mm of accuracy. It's capable of accomplishing this by implementing a combination of precision stepper motors and encoders at each joint. Positioning controllability is made possible via the DENSO RC8 controller which handles the kinematics processing and control. Although the RC8 controller directly interfaces with the DENSO arm module, the Compact Vision System (CVS), which is outlined in section 2.5, is used with an application programming interface (API) to send movement commands to the RC8 controller.

To interface with the blocks, an SMC-MHZ2-20d pneumatically actuated gripper is fixed as the end-effector tool to axis J_6. The fingers were designed to mate specifically with the geometry of the block. The optimization of the gripper geometry is contingent on the corresponding block shape and size. To appropriately manipulate the material used in this research, two custom fingers were designed and 3D-printed to expand the open/close limits of the gripper from 15 mm – 25 mm, to 65 mm – 75 mm as can be seen in FIGURE 2-12. Pictured in FIGURE 2-13, the radius of the cylinder that contacts the building blocks is exactly the size of the circular protrusions of the building block. The cylindrical shape of the gripper attachment helps for misalignment ensuring the block is perpendicular to the gripper on every pick application as demonstrated in FIGURE 2-13. To further enhance successful material manipulation, the gripper was designed with 7 mm x 7mm chamfered tips. This was implemented to allow for passive correction of small positioning errors accumulated by the vision system. Considering the flexible gripper configuration, it is important to note that the gripper can be designed to accommodate varying block geometry. For this reason, it is desirable to accommodate a standard actuation end-effector tool, with detachable interfacing fingers. This flexibility can be very advantageous where varied designs of 3D-printed building material would require a special shape tool for compatibility with material manipulation.

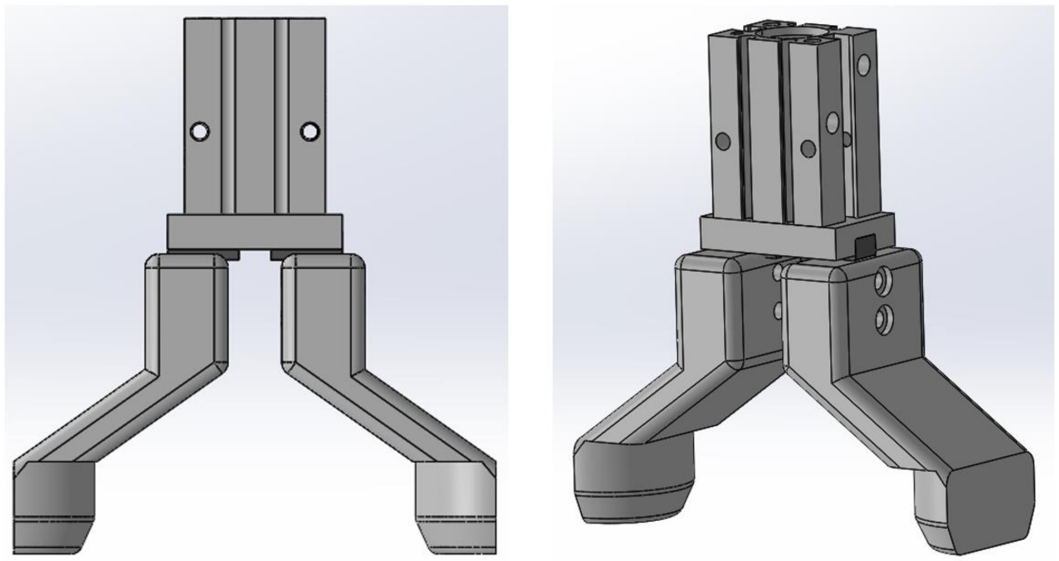


FIGURE 2-12: Gripper and finger assembly.

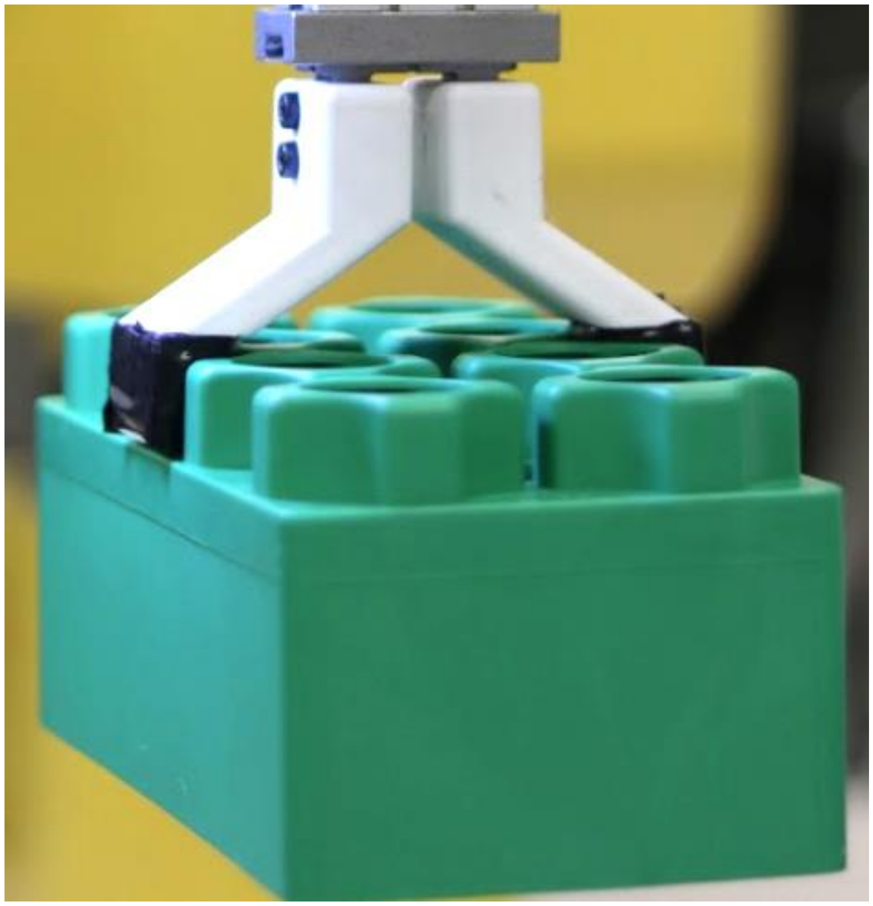


FIGURE 2-13: Gripper interfacing with emulated building material

The robotic manipulator is mounted on the fore-end of the vehicle chassis. Mounting the arm here enabled the system to both place blocks on the build plane and pick blocks from the material bed located on the rear of the vehicle. The assembly of the mounted robot arm is illustrated in FIGURE 2-14. Image “a”, taken from the rear of the vehicle, represents the arm picking material from the material bed, and image “b”, taken from the left side of the vehicle, illustrates the manipulator placing the object in line with the build plane.

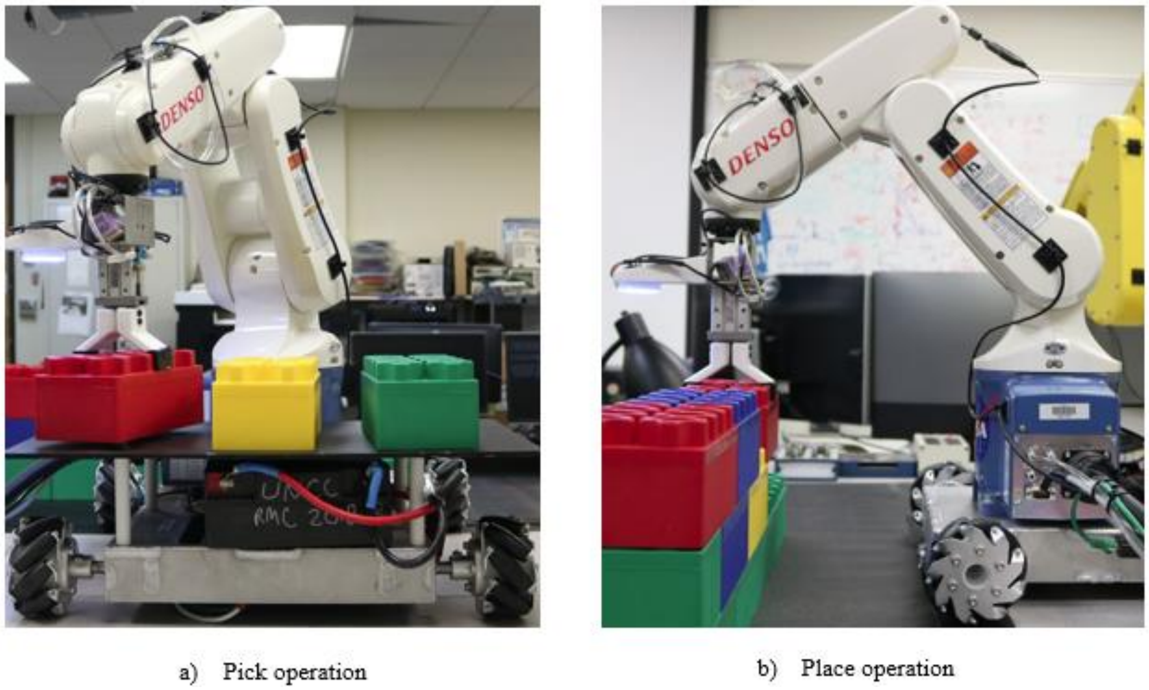


FIGURE 2-14: Demonstration of the robotic arm in pick and place positions

2.5 Vision System

To adapt to the changing environments the vehicle might encounter, a vision system for object targeting was implemented. The camera used in image acquisition for this research is a single Logitech C270 module. This device encompasses a 60° field of view (FOV) and can capture video up to 720p at 30 frames per second. This device was chosen for its USB compatibility with the vision controller and its range of resolution setpoints which is beneficial in lowering vision processing load.

The C270 camera is mounted on the end-effector tool of the robot arm. This enables the camera to move with the end-effector to achieve any viewing angle/position within the manipulators ROM. Using a single image acquisition device was possible by moving the end-effector to each image location rather than having multiple stationary imaging devices. A 3D-printed bracket was created in order to mount the camera to the end-effector. The camera mounting point relative to the camera lens center is projected 111 mm outward from the end-effector center to ensure the tool and building material would not obstruct the FOV. The camera center coordinates were offset from the center of the end-effector mount by 12 mm horizontally. A model of the mounting bracket can be seen in FIGURE 2-15.

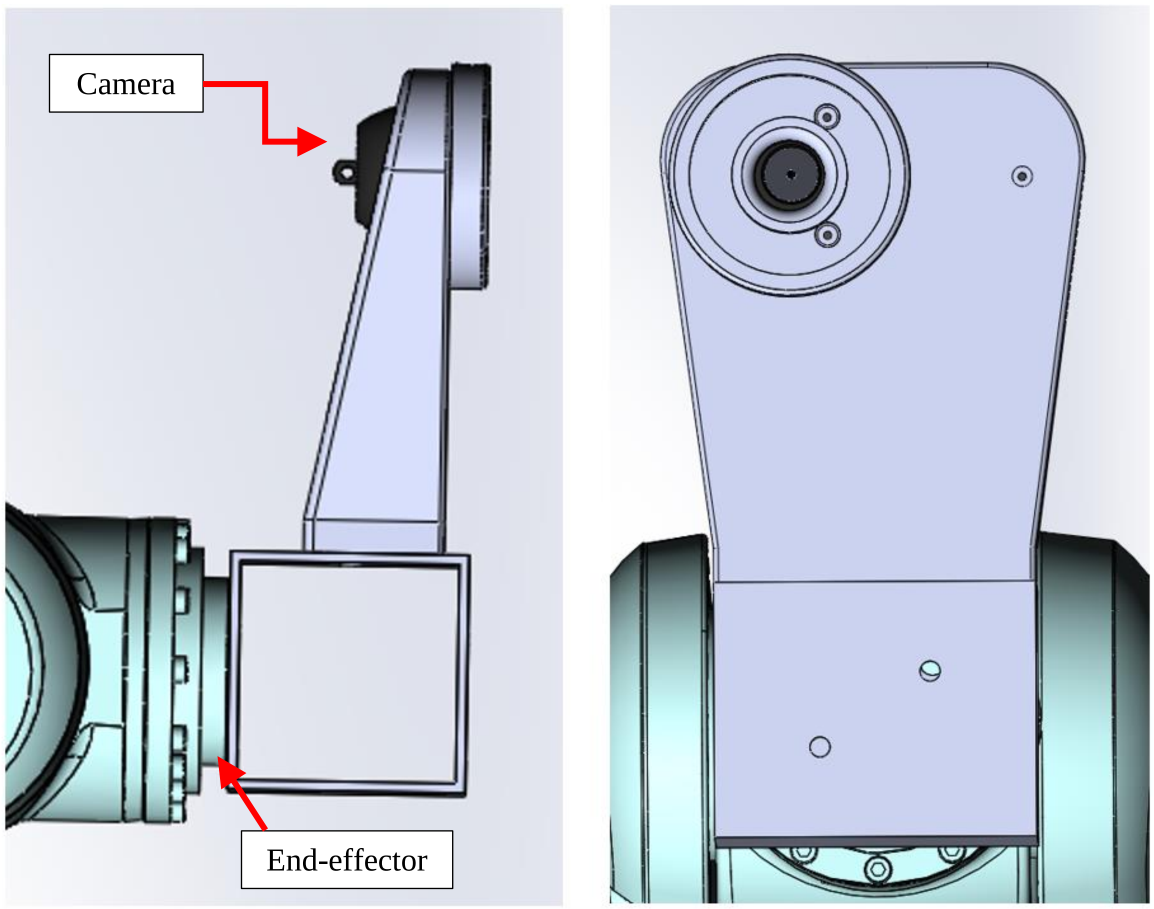


FIGURE 2-15: 3D model of the camera mount

Lighting was also determined to be a critical factor in image processing. Inadequate lighting can increase inconsistencies in many machine vision algorithms. A local, constant light source is critical for a mobile system since position based, natural lighting will change relative to vehicle location and time of day. Diffused lighting is an excellent way to scatter the light source such that the target object is illuminated from random angles thus increasing contrast to the material bed [11]. An on-axis ring light with diffuser material was chosen for this application. The ring light with diffuser was fixed directly below the camera with a constant radius around the lens of the camera. This helped the diffused light illuminate the working area from all directions. The lighting configuration implemented in this research is illustrated by the graphic in FIGURE 2-16 but is not an exact representation of the light propagation.

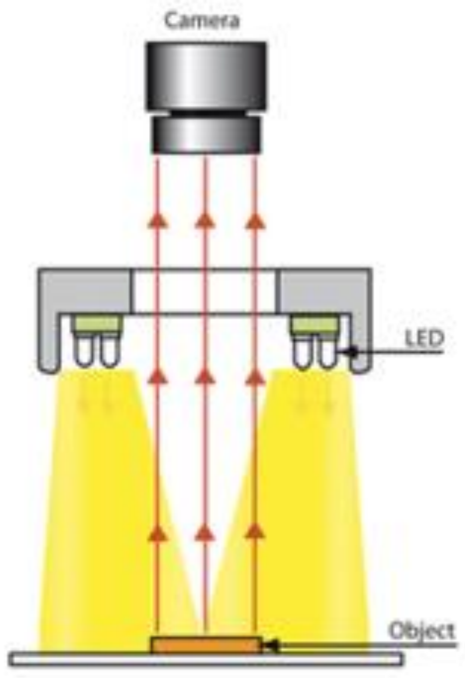


FIGURE 2-16: Ring lighting configuration for the camera system [12]

Completing the vision system, the controller used for the image processing and image acquisition is the NI CVS-1459RT. The CVS employs a 1.9 GHz quad-core Intel processor, with 4 GB of RAM, up to 32 GB of local storage, and a user-accessible field-programmable gate array (FPGA). The quad-core processor and accessible FPGA allow for fast, and efficient machine vision processing. For image acquisition, the CVS-1459 is outfitted with two USB 3.0 ports for high-speed camera interfacing which corresponds to the selected camera. Although this device is built for image processing, it also employs a 44-pin connector to interface general-purpose input/output (GPIO) connections. This enables the CVS to both process images and send commands to the robot controller. The Digimetrix DENSO library was also installed to interface with the VP-6242 system maintaining a NI development environment.

CHAPTER 3: AUTONOMY AND METHODOLOGY

Automation of this system is a critical aspect of the successful building procedures due to severe latency preventing manual control of the vehicle as described in the Introduction. A storyline of operation was created such that the vehicle should start at the loading zone awaiting to be loaded with material. A flowchart of the automation overview represented in FIGURE 3-1. When loaded, the vehicle will travel to the predetermined structure start position. Here, the vision system will look for a laser line that signifies where to place the first block. After the first block has been placed, the system will place the remaining blocks from the material bed. When the material bed is empty, the vehicle will travel back to the loading zone to be reloaded. When loaded, the vehicle shall again travel to the building zone to place the newly loaded material. The system repeats this process until the structure is completed. For the emulated demonstration, a straight wall is erected approximately 0.25 m high by 1 m long for the structure completion.

To approach the design process of this complex system, autonomy was broken up into three main tasks with further defined sub-tasks for individual processes. After each part was successfully completed and tested, each piece was assembled into the overall main program. Automation tasks are summarized by the following list:

- Vehicle Chassis Autonomy
  - Vector Movement
  - Sensor Data Acquisition
- Robot Arm Autonomy
  - Axis and TOOL identification

    - Pick Operations
    - Place Operations
- Vision System Autonomy
    - Image Acquisition
    - Edge Finding
    - Pattern Matching

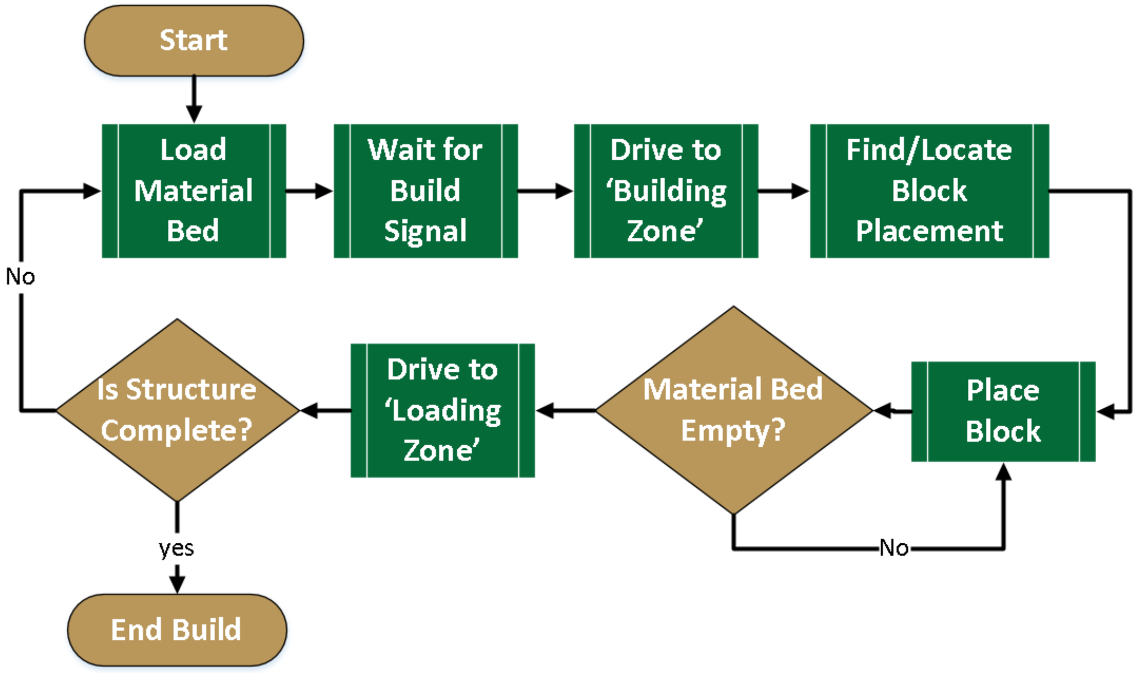


FIGURE 3-1: Automation overview

Programming these tasks yielded a naming convention as follows:

- *Drive – a* function to control the robot base
- *Move* – a function to move robotic manipulator directly to coordinates
- *Approach* – a function to approach a set of coordinates with the end effector with a vertical offset
- *Depart* – a function to depart from a set of coordinates with the end effector with a vertical offset
- *WORK* – used to assign a new work plane relative to the robot manipulator
- *TOOL* – used to assign end-effector offsets

- *Initial Edge Finding* – a function to locate edges of blocks on the material bed
- *Full Block Pattern Matching* – a function used to find full-size block matches
- *Half Block Pattern Matching* – a function used to find half-size block matches
- *Laser Line Finding* – a function to calculate the angle and position of the laser line
- *Corner Finding* – a function to find/calculate the corner of the simulated structure

3.1 Vehicle Chassis Autonomy

Automation of the vehicle chassis is handled by the myRIO controller. To control the vehicle, the myRIO acquires data from the encoders and rangefinders as outlined in Chapter 2. The overall task of the chassis is summarized in the flowchart in FIGURE 3-2; the vehicle must travel to the building zone after the material bed has been loaded, wait for the material bed to be unloaded, and travel back to the loading zone accurately and repetitively without external localization. This is accomplished by polling the rangefinders and encoders and fusing the data to calculate how far the vehicle has traveled and where it is relative to the structure.

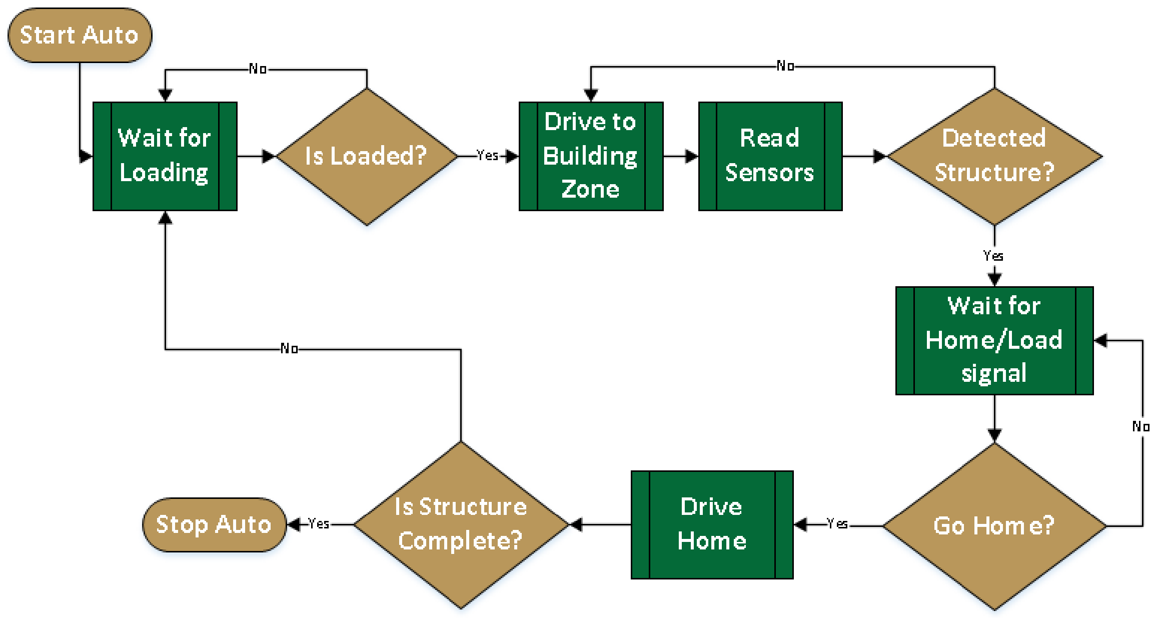


FIGURE 3-2: Vector-drive flowchart of operations

The myRIO pushes pulse width modulated (PWM) signals to each motor controller to govern the speed of each wheel independently. By varying the PWM signal, each motor throttle could be varied 0% – 100% in forward and reverse directions. Since PWM pulses are created with an 8-bit number, throttle control can assume integer values between 0 – 255. However, since each motor must be able to rotate clockwise and counterclockwise, the throttle increments are halved such that individual throttle steps measure 0 – 127 for each direction; essentially throttle increments of ~0.79%. The Talon speed controllers are expecting a PWM signal between 1 – 2 ms. The frequency of the PWM signal used was 333 Hz such that the period of the signal is 3 ms as obtained by equation (1). Therefore, the forward, reverse, and stop throttle values could be calculated from equations (2), (3), and (4) such that the maximum forward throttle duty cycle (dc) is 0.67, with a maximum reverse throttle dc of 0.33, and stopping throttle dc of 0.5. These throttle values are not arbitrary and are set based on the myRIO hardware limitation of the lowest PWM frequency of 333 Hz.

$$Period = \frac{1}{frequency} \tag{1}$$

$$\mathrm{T}_{reverse} = \frac{1_{ms}}{Period} \tag{2}$$

$$\mathrm{T}_{forward} = \frac{2_{ms}}{Period} \tag{3}$$

$$\mathrm{T}_{stop} = \frac{1.5_{ms}}{Period} \tag{4}$$

Adequate throttle control was still achievable so that vector-like movement as performed by Stephen Padgett in [13] could be described by equations (5) & (6). Where T_x and T_y are the resulting 2D throttle values, M is the throttle vector magnitude, and θ is the desired strafing angle.

$$T_x = M\sin(\theta) \tag{5}$$

$$T_y = M\cos(\theta) \tag{6}$$

When starting and stopping, it was important to utilize the throttle percentages to minimize wheel slip. A ramping function was created to ramp throttles up and down to the desired setpoint rather than setting the speed to the maximum at once. This reduced the moment of inertia on the vehicle-mounted hardware ultimately reducing wheel slip. Reducing wheel slip initially also helped increase dead reckoning accuracy. An example of a ramping input vs a step input is seen in FIGURE 3-3 where the throttle is linearly increased over time as opposed to setting the throttle to a maximum at once.

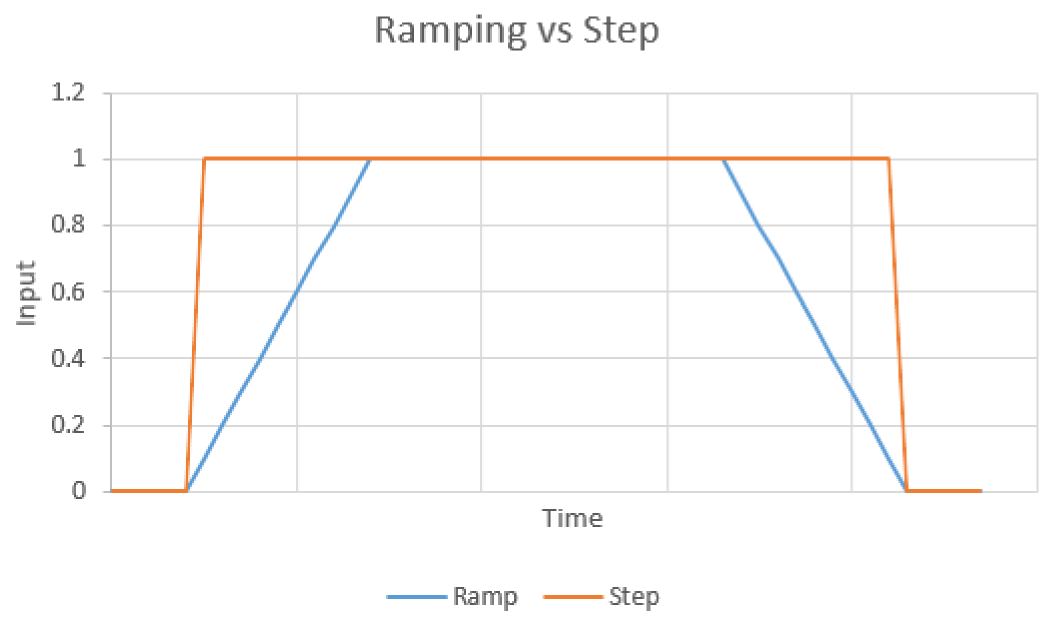


FIGURE 3-3: Ramp input (blue) vs step input (orange)

Displacement and velocity were calculated from the encoder output pulses. The encoders produce 1988 pulses per wheel revolution, which is determined by equation (7).

$$\frac{Pulses}{Wheel\ Revolution} = \left[\frac{Motor\ Rotations}{1\ Wheel\ Rotation}\right] \cdot \left[\frac{Encoder\ Pulses}{1\ Motor\ Rotation}\right] \tag{7}$$

Equation (7), was used to derive equation (8) calculating displacement (*d*). Likewise, equation (9) was derived to compute the rotational velocity of the wheel.

$$d = \frac{Measured\ Pulses}{\frac{Pulses}{Wheel\ Revolution} \cdot \frac{1\ Wheel\ Revolution}{Wheel\ Circumference}} \tag{8}$$

$$RPM = \frac{\left[\frac{Measured\ Pulses}{\frac{Pulses}{\text{Wheel Revolution}}}\right]\left[\frac{60\ (s)}{1\ (\text{min})}\right]}{Time\ Interval\ (s)} \tag{9}$$

Although the wheel encoders worked for dead reckoning, the ranging devices were implemented to ensure the robotic base did not collide with the structure or any other objects in its path. The Shark 2Y0A21 range finder was used for coarse distance measurements with a tested range of 10 cm to 100 cm, while the VL6180X sensor was used for distances less than 200 mm. Although both sensors use light for ranging data, each sensor works on different theories of operation. The Shark infrared (IR) sensor operates on the principle of measuring the reflected light intensity whereas the VL6180X functions on the time of flight (TOF) concept by emitting a light beam and measuring the time it takes for the beam to reflect to the receiver diode. Sensor fusion between the encoders and rangefinders allowed better control of the chassis for approaching the structure. This methodology allowed the control system to ramp down motor throttle based on how close the system was to its target destination whether by encoder count or rangefinder measured distance.

To calibrate the Shark sensor, a block was placed 100 mm in front of the sensor and then moved in increments of 50 mm up to 1000 mm. The analog voltage with respect to distance in (mm) from this test was captured and can be seen in and FIGURE 3-4. This data revealed a formula describing analog output voltage as equation (10) through empirical derivation. Where *V* is the analog output voltage of the device, and *d* is the object distance from the sensor. Solving for (d), resulted in equation (11) and was actively used to calculate forward distance from the object.

$$V = \frac{90.867}{d^{0.797}} \tag{10}$$

$$d = \sqrt[0.797]{\frac{90.867}{V}} = \frac{286.56}{V^{1.255}} \tag{11}$$

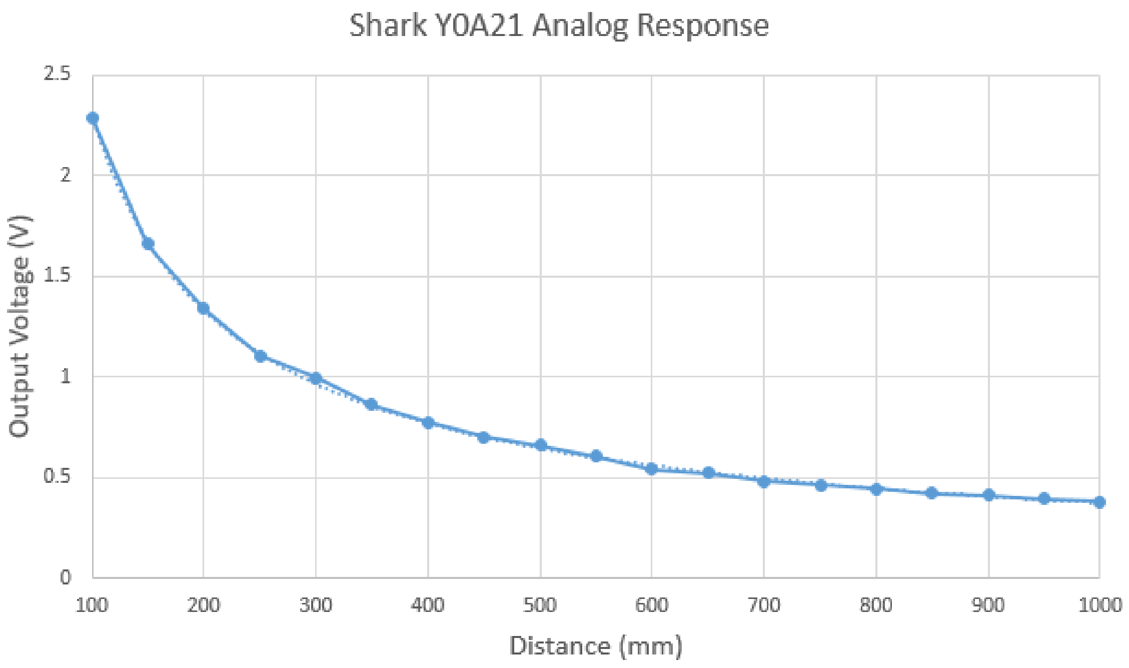


FIGURE 3-4: Shark Y0A21 analog response

The VL6180X sensor is a digital device communicating via inter-integrated circuit (I^2C) communication protocol. The output response is linear with all calibration/scoring set up by the manufacturer and processed internally on the chip. However, the measured distance was often recorded to be off by 6 – 10 mm. To counter this, the sensor was tested at distances between 10 – 200 mm in increments of 10 mm.

The measured distance was recorded and compared to the actual distance to determine the offset. Here it was discovered that the sensor actually operated between 10 – 180 mm. A plot of measured vs actual distance can be seen in FIGURE 3-5 where the calibration offset was determined to be 6.86 mm as described by equation (12). This data was used to accurately report ranging measurements in the automation process.

$$d = 0.993 \cdot d_{measured} + 6.862 \tag{12}$$

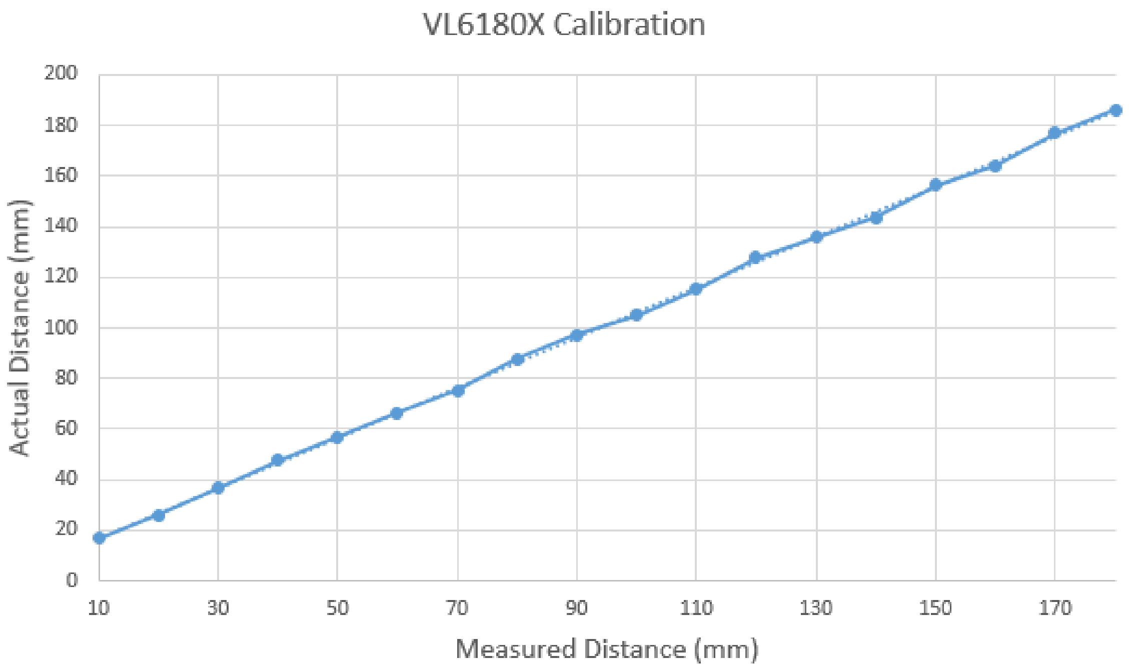


FIGURE 3-5: VL6180X actual distance vs measured distance

3.2 Robot Arm Autonomy

The robot arm serves two primary purposes: position the camera at the desired viewing locations, and physically manipulate the building material. Mounting the camera as an end-effector tool proved to be beneficial as the camera could be positioned nearly anywhere in the robot's ROM. This allowed one camera to successfully acquire all desired images in the robot's working area. Furthermore, the 6 DOF arm provided the required dexterity to accurately manipulate the material in 3D space.

3.2.1 Setting Coordinate Systems

The robotic manipulator is capable of functioning in three modes of operation. "X-Y Mode" is used for positioning the payload at a coordinate on a 3-dimensional Cartesian plane with respect to axis J_1 center, "Joint Mode" is used for controlling each individual joint, and "TOOL Mode" is used to define new reference coordinates with respect to the end-effector tool. For the purpose of this research, the "TOOL" mode of operation was chosen as the primary means for end-effector positioning. This technique was used to identify a coordinate system for both the gripper and camera devices. Similarly, a working plane for both the material bed and build plane was assigned using the "WORK" feature of the robot controller. This created a 3-dimensional reference point for the end-effector with respect to the working surface and base coordinates as presented in FIGURE 3-6. Identifying new TOOLs allowed the controller to achieve accurate and intuitive 3D coordinate control. This is necessary because both the camera and the gripper are mounted as the end-effector tools.

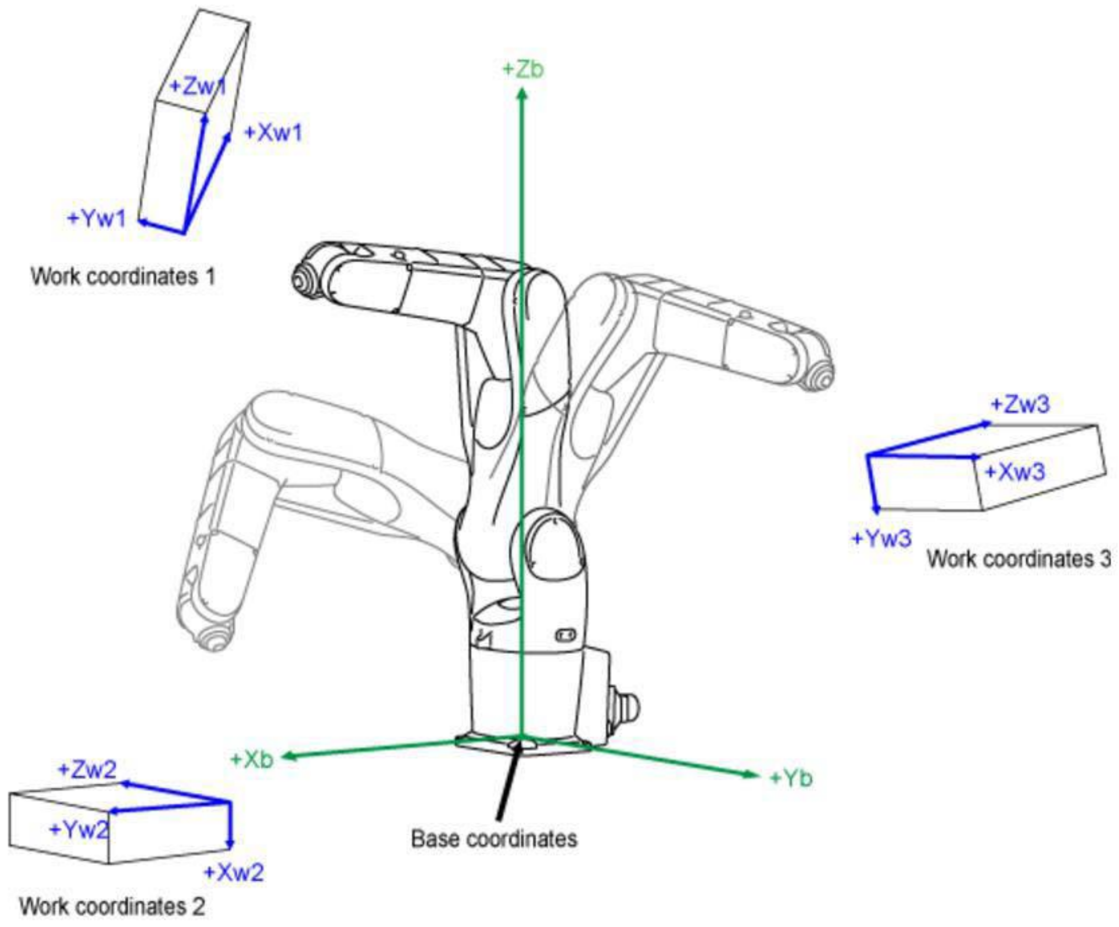


FIGURE 3-6: DENSO manipulator coordinate reference [10]

The mechanical flange interface is used to describe the default TOOL and is presented in FIGURE 3-7 such that the *z-axis* (Z_m) is projecting outward from the flange surface and the *y-axis* (Y_m) is pointing to the orientation keyhole. These axes are re-oriented as seen in FIGURE 3-8 such that Z_t replaces X_m, X_t replaces Z_m, and Y_t replaces $-Y_m$. Knowing this transformation, tool coordinates were created by measuring the total 3D offset of the tool center in millimeters for both tools with respect to axis J_6 center. These offsets not only allow for accurate 3D representation of the tool, but it also prevent the robot from moving the tool into a colliding path with any part of the DENSO system.

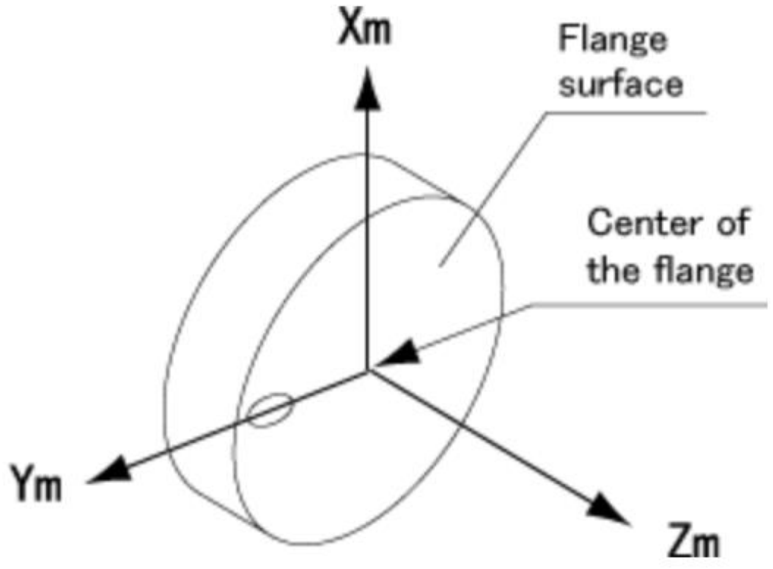


FIGURE 3-7: Mechanical flange axes without TOOL axis transformation [10]

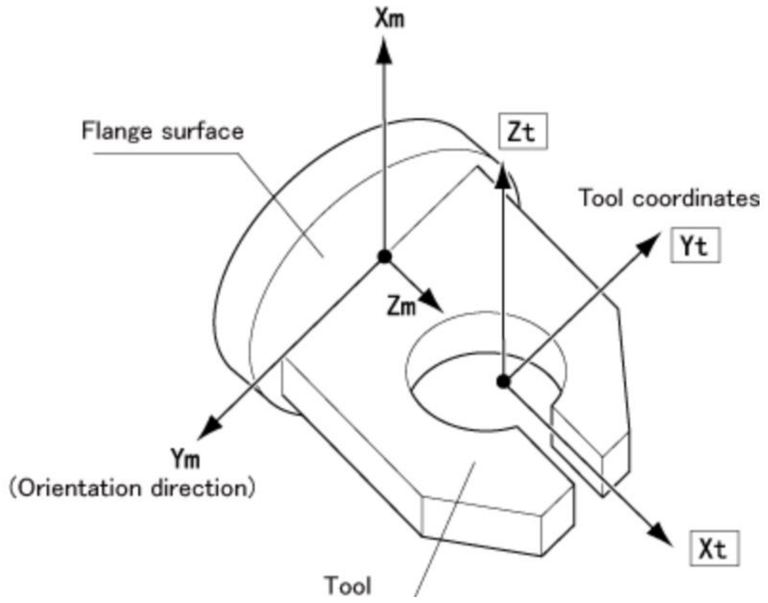


FIGURE 3-8: TOOL coordinate offset transformation [10]

3.2.2 End-effector Positioning

For the movement of the robotic arm, the manipulator was programmed with the LabVIEW DigiMetrix toolkit. This toolkit introduced the commands for interfacing with the DENSO RC8 controller. The primary commands used in arm positioning are “approach”, “depart”, and “Cartesian/move”. The approach command was used to approach the desired position with a predetermined vertical offset. Oppositely the depart command was used to depart from the position with a predetermined vertical offset. Both of these commands are especially useful when the object needs to be picked or placed precisely in a vertical fashion. However, the Cartesian/move control method was used to move to the exact desired position by the quickest possible route. This was especially useful immediately following the approach or depart commands. Each movement was tested with different methods of interpolation and movement characteristics to optimize behavior. Ultimately, a simple pick operation was created using the approach command to move directly above the desired location with the gripper in the open state. This was immediately followed by the Cartesian function to move directly to the absolute coordinate. At this point the gripper was be actuated to close, followed by a depart command. The place command uses essentially the same procedure but swaps the open/closed states of the gripper to release the object rather than capture it. A flowchart of the pick and place methods summarize this procedure in FIGURE 3-9.

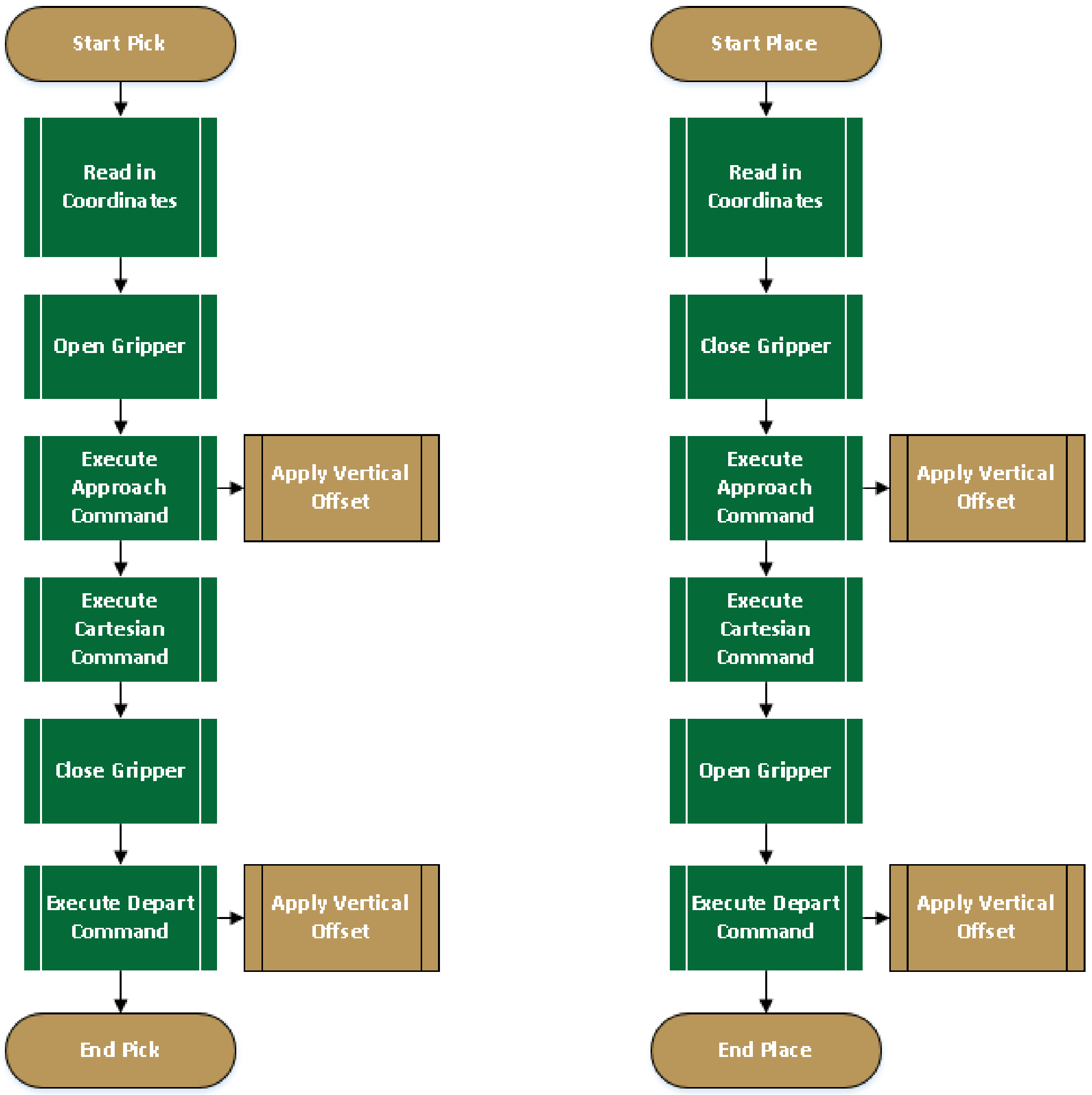


FIGURE 3-9: Pick/Place program operation

3.3 Vision System Autonomy

Machine vision is the most important task of this research for reaching a state of autonomy due to the near-endless possibilities that the vehicle could encounter when deployed. To handle this, a vision system was used to locate blocks and the structure for object manipulation. This is a driving factor in the successful automation of HARP.

3.3.1 Machine Vision Fundamentals

To understand how this vision processing system works, it is important to understand the individual devices and how they operate. The camera is described by its functional abilities such as FOV, resolution, and image capture format (32-bit Red/Blue/Green, greyscale, or binary). The FOV is the maximum viewing angle projecting outward from the camera lens. Resolution is measured by the maximum number of picture elements, or pixels, in the *y-axis* by the maximum number of pixels in the *x-axis*. Combining these functions, cameras work on the principle of recording varying light intensity from an array of imaging sensors in the n by m pixel array [11], [14].

Image processing is handled by using a variety of techniques. The first step in processing and image is choosing what format the image will be processed in. This will influence the speed and methodology of processing an image. There are three basic image formats for image processing; a greyscale image is represented as an 8-bit number, with individual pixel values ranging 0 – 255 in single integer increments. Binary images are often derived from greyscale images, where pixel intensity is represented as an 8-bit number and every pixel assumes two states; 0 or 255. Red/Blue/Green (RBG) images are recorded as 32-bit images where 8 bits are used to make up each R, B, and G parts of the pixel. The remaining 8 bits of the 32-bit number are often not used. Many computer vision algorithms are performed on greyscale images that are derived from the 32-bit color image by extracting a certain color plane from the image [15]. Kanan and Cottrell [16] state that "The main reason why grayscale representations are often used for extracting descriptors instead of operating on color images directly is that grayscale simplifies the algorithm and reduces computational requirements" meaning a grayscale image processing will be less intensive on the vision processing hardware and can often be processed faster. Binary images can be processed even faster but this format often filters out or omits important characteristics or features of the image. For these reasons, greyscale image format was chosen for image processing in this research. A visual representation RGB, Greyscale, and binary image formats can be seen in FIGURE 3-10.

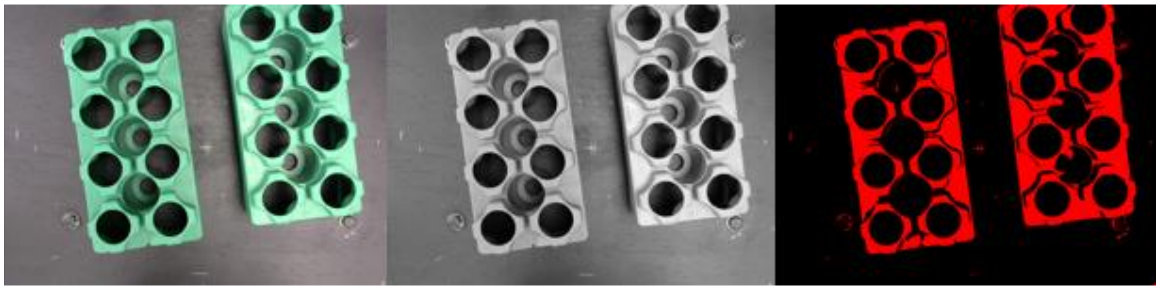


FIGURE 3-10: RGB (left), greyscale (middle), and binary (right)

Further reduction of the computational strain on the vision processing hardware can be accomplished by identifying a region of interest (ROI). Hornberg states "The ROI identifies the areas of interest for machine vision application and removes the uninteresting image data that is beyond the ROI" [11]. Establishing an ROI focuses processing power on one designated area and omits all other regions that do not need to be processed. This further reduces the computational requirements by the vision processing hardware if the object of interest falls in the ROI. It is crucial to verify that the selected region of interest encompasses the object or feature of interest. ROIs were extensively used in many algorithms for this system.

It is common to have a certain level of unwanted noise present in the image. Filtering becomes an advantageous option in this scenario with many different options to work with. One of the most common, and also used in this research, is the Gaussian or smoothing Gaussian filter. The Gaussian filter describes the probability density function of a random variable such that it filters out high frequency by averaging adjacent pixels creating a low pass filter [14]. In equations (13) and (14), σ is the standard deviation for the distribution, x & r are horizontal pixels, and c represents vertical pixels where equation (13) is a one-dimensional filter and equation (14) is a two-dimensional filter [11].

$$g_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-x^2/(2\sigma^2)} \tag{13}$$

$$g_\sigma(r,c) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(r^2+c^2)/(2\sigma^2)} \tag{14}$$

3.3.2 Types of Vision Processing

Edge finding techniques are a basic form of image processing, as well as the foundation of other machine vision algorithms. The edge finding technique searches for an abrupt change in the consecutive pixel intensities. Often, a significant rate of change in pixel intensity signifies the edge of an object. “Edge detection methods usually rely on calculations of the first or second derivative along the intensity profile” which better characterize the edge data [17]. Burger and Burge describe general edge finding techniques by equation (15) [18]. Where the function $f(u)$ presents a rapid change in pixel intensity (u) in the x or y axes. To visualize this concept, refer to FIGURE 3-11 [18].

$$\frac{df}{du}(u) \approx \frac{f(u+1)-f(u-1)}{(u+1)-(u-1)} = \frac{f(u+1)-f(u-1)}{2} \tag{15}$$

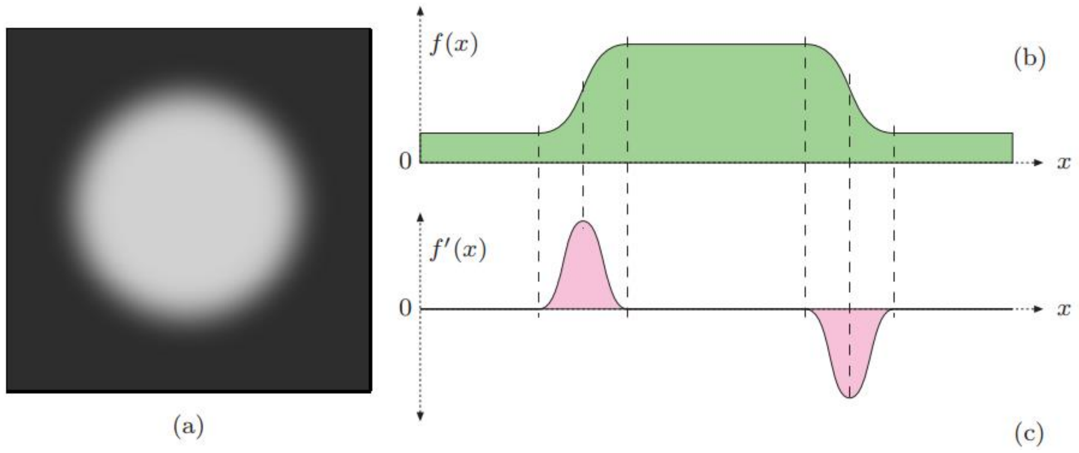


FIGURE 3-11: Edge finding illustration [17]

Pattern matching is an advanced technique that starts with a characteristic image, or template, and searches a given region of interest of a new image for a match. This is advantageous when the desired match closely resembles the pattern. As mentioned, edge finding is the basis of many machine vision algorithms and is often applied in pattern matching by examining the relevant edges in an image and comparing them to a base template. Pattern matching was used as the secondary check of object locating in this research. Though pattern matching is extremely useful and accurate, often, undesired information is captured in images. To eliminate this unnecessary load on the vision processing hardware, the pattern, or template is modified to ignore certain aspects of the object that may not always be present in an image depending on the object position relative to the camera. This, in turn, increases the likelihood of finding a match and decreases processing time.

According to [19], pattern matching in its simplest form can be represented by equation (16), and (17). Where *x* and *y* represent pixel intensity values and a “small value of *d(x, y)* or a high value of *s(x, y)* is indicative of pattern similarity”.

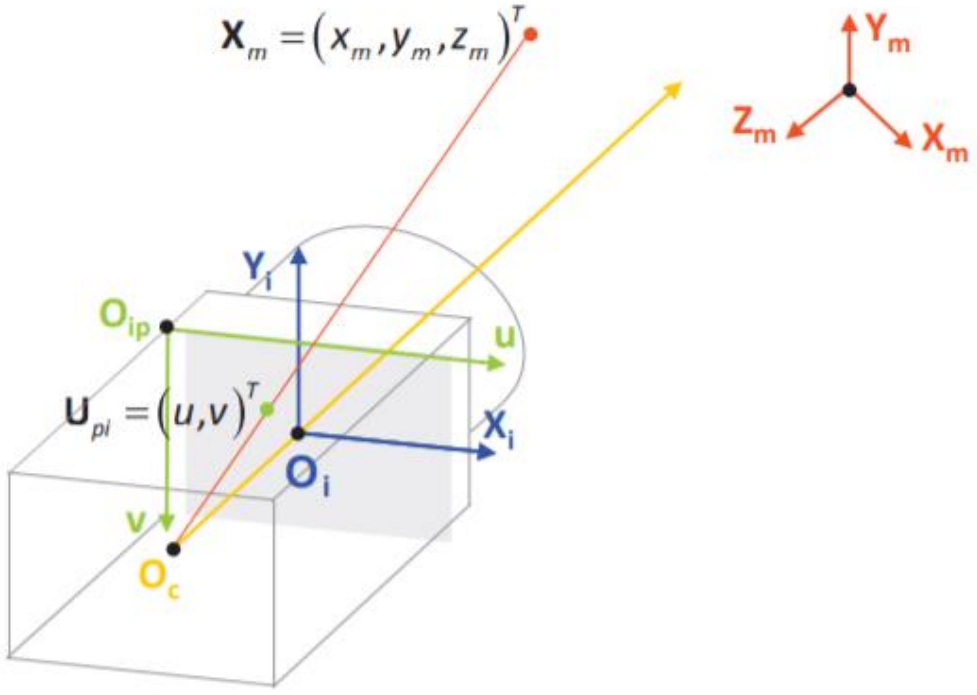
$$d(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 = \frac{1}{N}\|x - y\|_2^2 \tag{16}$$

$$s(x,y) = \frac{1}{1 + d(x,y)} \tag{17}$$

3.3.3 Camera Calibration

Any vision system that interacts with the outside world likely will report processing data in real-world units. This is characterized as a spatial resolution in millimeters per pixel by Hornberg [11]. To transpose pixel coordinates into real-world units, it is necessary to perform an image calibration. This will allow the vision system to relay calculated image data relative to the camera. However, the camera has many characteristics that prevent direct pixel to unit mapping that must be taken into consideration.

In this research specifically, since the distance from the camera to the object of interest can always be known, the Pinhole Camera Model calibration technique is sufficient. Caja et al from [20] present this method as, “a camera model transforms the coordinates of a point in space (3D) to the coordinates of a point in an image (2D), i.e., explains the process of forming an image with a camera”. This is generally summarized by the graphic in FIGURE 3-12 and the matrix expression in (18).



$\mathbf{X}_m = (x_m, y_m, z_m)^T$
Y_m
Z_m
X_m
Y_i
O_{ip}
u
$\mathbf{U}_{pi} = (u,v)^T$
X_i
O_i
v
O_c

FIGURE 3-12: Pinhole camera technique representation

$$\begin{pmatrix} w\cdot u \\ w\cdot v \\ w \end{pmatrix} = \begin{pmatrix} a_x & s & x_0 \\ \cdot & a_y & y_0 \\ \cdot & \cdot & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & -t_x \\ r_{21} & r_{22} & r_{23} & -t_y \\ r_{31} & r_{32} & r_{33} & -t_z \end{pmatrix} \begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix} \quad \begin{Bmatrix} \boldsymbol{U_{hpi}} = \boldsymbol{K}[\boldsymbol{R}| - \boldsymbol{t}] \cdot \boldsymbol{X_{hm}} \\ \boldsymbol{U_{hpi}} = \boldsymbol{P} \cdot \boldsymbol{X_{hm}} \end{Bmatrix} \tag{18}$$

$\boldsymbol{U_{hpi}} = (w\cdot u, w\cdot v, w)^T$ *Represent pixel coordiantes*

$\boldsymbol{X_{hm}} = (x_m, y_m, z_m, 1)^T$ *Represent real − world coordinates*

$a_x =$ *Horizonatl pixel size*

$a_y =$ *Vertical pixel size*

$s =$ *Skew paranter*

$(x_0, y_0) =$ *Principal point coordinates*

$\boldsymbol{R}$ & $\boldsymbol{T}$ *Represent camera position and orientation*

Lastly, the geometric distortion is represented by the change in location of the image points that are attributed to the lens characteristics in equations (19) and (20). Where $\dot{u}$ and $\dot{v}$ are the distorted coordinates, δ_u and δ_v are the geometric distortion properties of $\dot{u}$ and $\dot{v}$. δ_u And δ_v are modeled by equations (21) and (22). Where $\Delta u = u - x_0$ and $\Delta v = v - y_0$ and $k_1, k_2, p_1, p_2, s_1, s_2$ are the distortion characteristics [20].

$$\dot{u} = u + \delta_u \tag{19}$$

$$\dot{v} = v + \delta_v \tag{20}$$

$$\begin{gathered}\delta_u = k_1\Delta u(\Delta u^2 + \Delta v^2) + k_2\Delta u(\Delta u^2 + \Delta v^2)^2 + p_1(3\Delta u^2 + \Delta v^2) \\ + 2p_2\Delta u\Delta v + s_1(\Delta u^2 + \Delta v^2)\end{gathered} \tag{21}$$

$$\begin{gathered}\delta_v = k_1\Delta v(\Delta u^2 + \Delta v^2) + k_2\Delta v(\Delta u^2 + \Delta v^2)^2 + 2p_1\Delta u\Delta v \\ + p_2(\Delta u^2 + 3\Delta v^2) + s_2(\Delta u^2 + \Delta v^2)\end{gathered} \tag{22}$$

3.4 Machine Vision Automation

Overall, the prototype implements many machine vision algorithms. Of these, pattern matching and edge finding were used most often. These algorithms were configured in a two-step process to verify block pick and placement applications. Edge finding was used as the primary tool for locating objects in the *initial block detection*, *laser line finding*, and *corner finding* functions. This served as an accurate, reliable, and time-efficient means of object localization relative to the camera's current position, especially when only fragments of the object are within camera FOV. Pattern matching was used as the secondary check for block localization via the *pattern matching* function. However, prior to executing these processes, an image calibration was completed.

3.4.1 Machine Vision Calibration

Image calibration is of the most importance when implementing machine vision to interact with the outside world. This is a result of requiring accurate measurements in real-world coordinates. Using the Vision Assistant application from LabVIEW, a calibration was successfully completed to examine overall distortion percentage, mean and max error map, standard deviation, as well as a pixel to real-word unit mapping. The Dot Grid method was used in this process with the Distortion Model calibration and implemented in every machine vision application. Since the vision system acquires images at multiple different *z-axis* elevations, a calibration was required at each to eliminate inaccurate measurements. Mapping the error relative to the camera lens after image correction revealed the plot in FIGURE 3-13 to demonstrate an example of distortion attributed to the camera and imaging hardware.

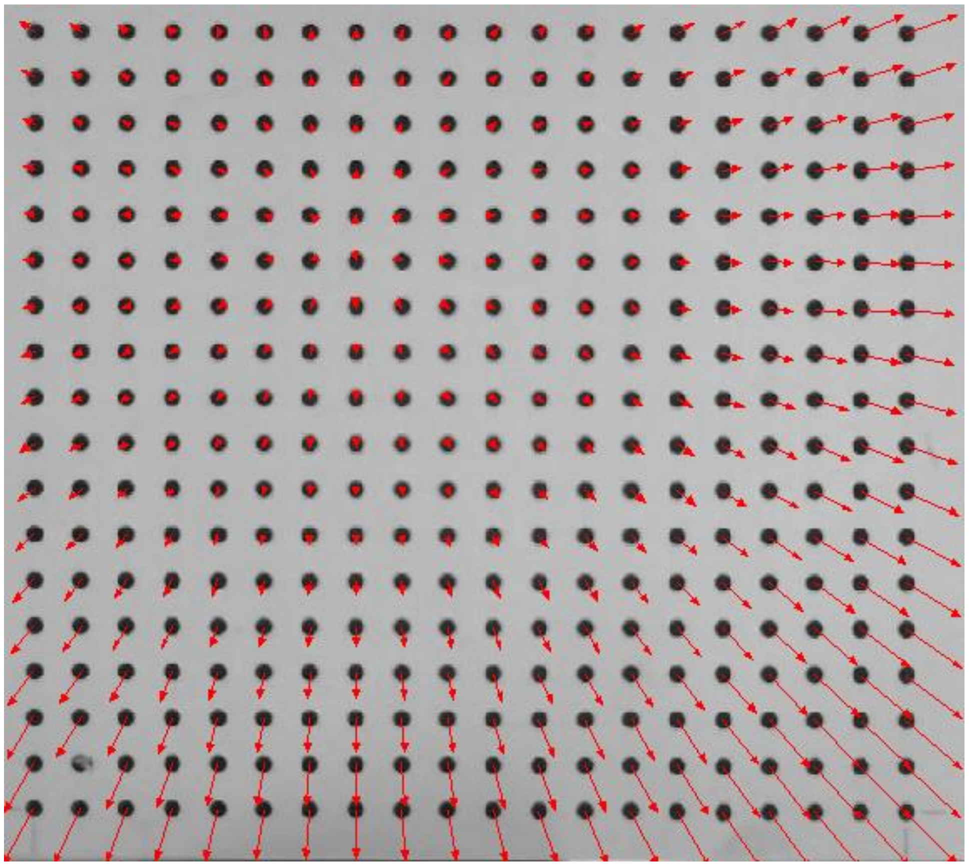


FIGURE 3-13: Dot-Grid image calibration distortion representation

3.4.2 Finding Block Coordinates

A summary of the vision tools/process used in locating block coordinates is presented in FIGURE 3-14. Where edge finding is used to find the approximate center of each block for the *initial block finding* function, to then reposition the camera above these coordinates acquiring an image at each location. Finally, these images are then compared to a template searching for the actual block location in the image relative to real-world coordinates to complete the *pattern matching* function.

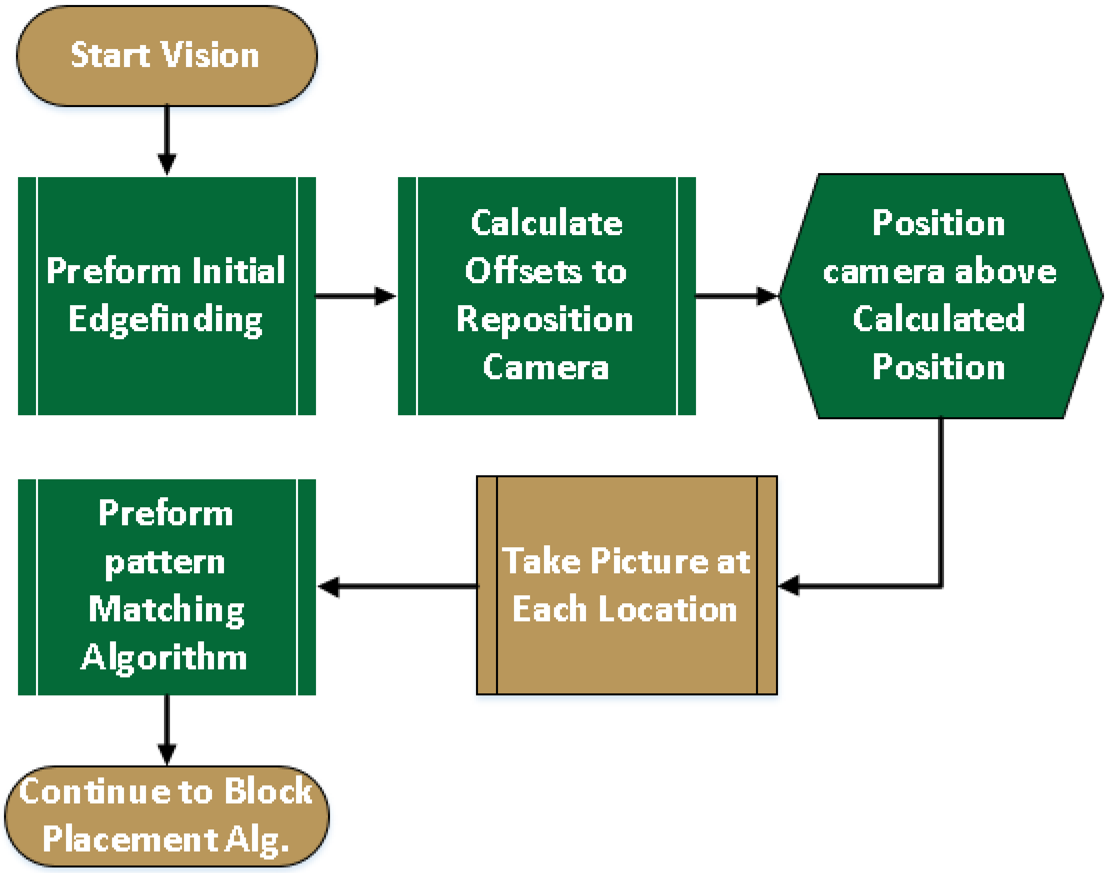

Start Vision
Preform Initial Edgefinding
Calculate Offsets to Reposition Camera
Position camera above Calculated Position
Take Picture at Each Location
Preform pattern Matching Algorithm
Continue to Block Placement Alg.

FIGURE 3-14: Block finding flowchart of operations

The FOV for the camera limited the ability to see all blocks loaded on the material bed at once. To counter this, approximate block locations are computed by implementing a simple edge finding algorithm which is known as the *initial block finding* function. Here, the manipulator positions the camera at the highest possible point above the material bed center. An image is then acquired and saved in memory. After an image is captured, the edge fining algorithm is applied. The edge finding algorithm is tailored to search specific ROI's for edges within $\pm 25°$, of the vertical for rotation depending on the edge parameters. Using the NI Vision Assistant 'Caliper' function, the algorithm can then calculate where the edges intersect to characterize the block corner, as well as compute the angle of each edge with respect to the vertical/horizontal. With this information, the approximate center can be calculated. The settings used in the *initial block finding* algorithm are presented in TABLE 3-1.

TABLE 3-1: Edge finding settings used in the initial block finding function

Edge Polarity	Dark to Bright
Edge Type	Best Edge
Minimum Edge Strength	25
Kernel Size	13
Projection Width	13
Pixel Gap	10

Calculation of the approximate center coordinates is illustrated by equation (23) and (24) using *sin* and *cos* trigonometry properties. Where *h* is the known diagonal length to the center of the block from the corner, θ is the angle from the camera vertical to the edge of the block, *y* is the magnitude to move vertically, and *x* is the magnitude to move horizontally with respect to located block corner. FIGURE 3-15 illustrates the edge finding algorithm locating up to three block corners. The algorithm can use the *x* and *y* coordinates to command the arm to move the camera to each calculated center point to acquire an image at each location.

$$|y| = |\mathrm{h} \cdot \cos(26.57° \pm \theta)| \tag{23}$$

$$|x| = |\mathrm{h} \cdot \sin(26.57° \pm \theta)| \tag{24}$$

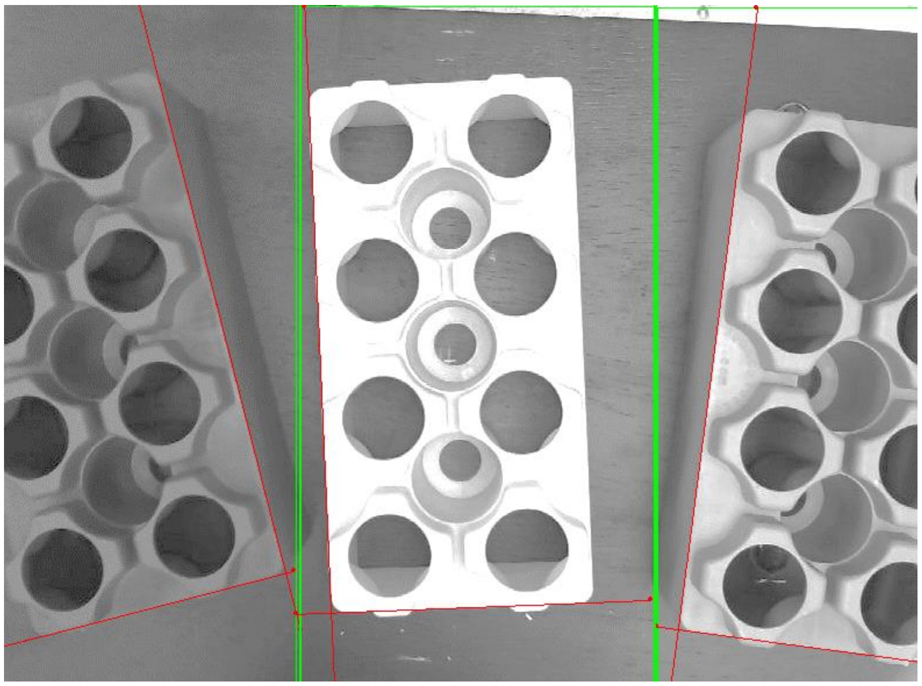


FIGURE 3-15: Initial corner finding algorithm finding three corners

After an image has been acquired at each of the approximate block centers, the developed algorithm examines them individually with the *pattern matching* function. Each image is compared to a template using the LabVIEW Pattern Matching algorithm. Here, the pattern matching template is customized to ignore certain areas and focus on others. Using correlation scores, the image pixel intensity is analyzed to see if there is a match. From this match, the new center coordinate are calculated by combining the results from the edge finding algorithm and pattern matching algorithm. This information is used as the final coordinate of the block center. The pattern matching script outputs a non-destructive overlay illustrating where the match is located on the image as demonstrated in FIGURE 3-16.

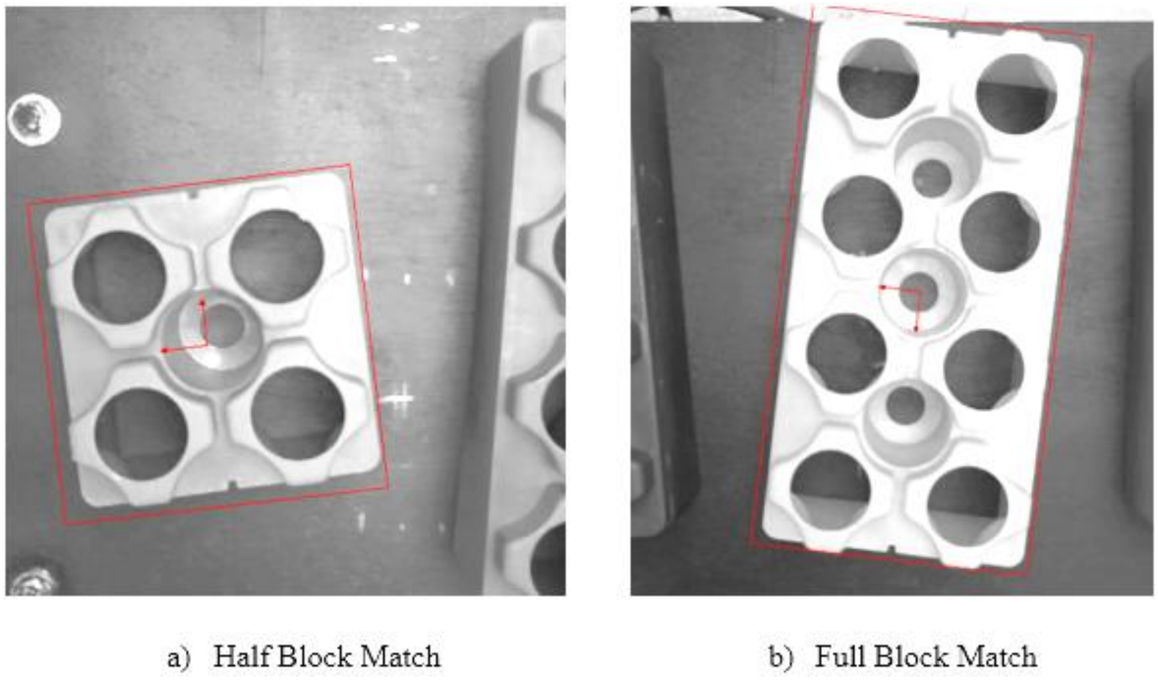


FIGURE 3-16: Half and full-block successful pattern match

3.4.3 Finding Block Placement Coordinates

In addition to finding the blocks, the vision algorithm must be robust enough to determine where to place the blocks. If the structure has not been started, the vision system executes the *laser line finding* function examining the build plane for a laser line as a reference point to place the first block. This provides both a starting position, as well as angle so the rangefinder sensors can later be used for vehicle positioning. The “Advanced Edge Finding” option in the Vision Assistant application was used since this environment could present much background noise. The parameters used in this process are presented in TABLE 3-2.

TABLE 3-2: Edge finding settings used in the laser line finding function

Detection Method	First Edge Projection
Search Direction	Bottom to Top
Edge Polarity	Any Edge
Minimum Edge Strength	20
Minimum Edge SNR	6.09
Kernel Size	5
Search Gap	15
Edge Angle Range	$\pm 25°$

After the first block has been placed, the corner of the structure is used to identify where to place the next block. To do this, the manipulator positions the camera at 500 mm above the middle of the working surface and acquires an image. Then, the *corner finding* function is used to locate the corner of the partially assembled structure. After this is accomplished, the algorithm positions the camera at these coordinates with a vertical offset of 350 mm to acquire yet another image. The *corner finding* function is executed once more to recalculate the coordinates of the corner with respect to the new camera position. This two-step process was used to ensure accurate placement of the block since the visual perception of the corner can vary depending on the angle of which it's viewed. Further definition of this process is illustrated by the flowchart in FIGURE 3-17.

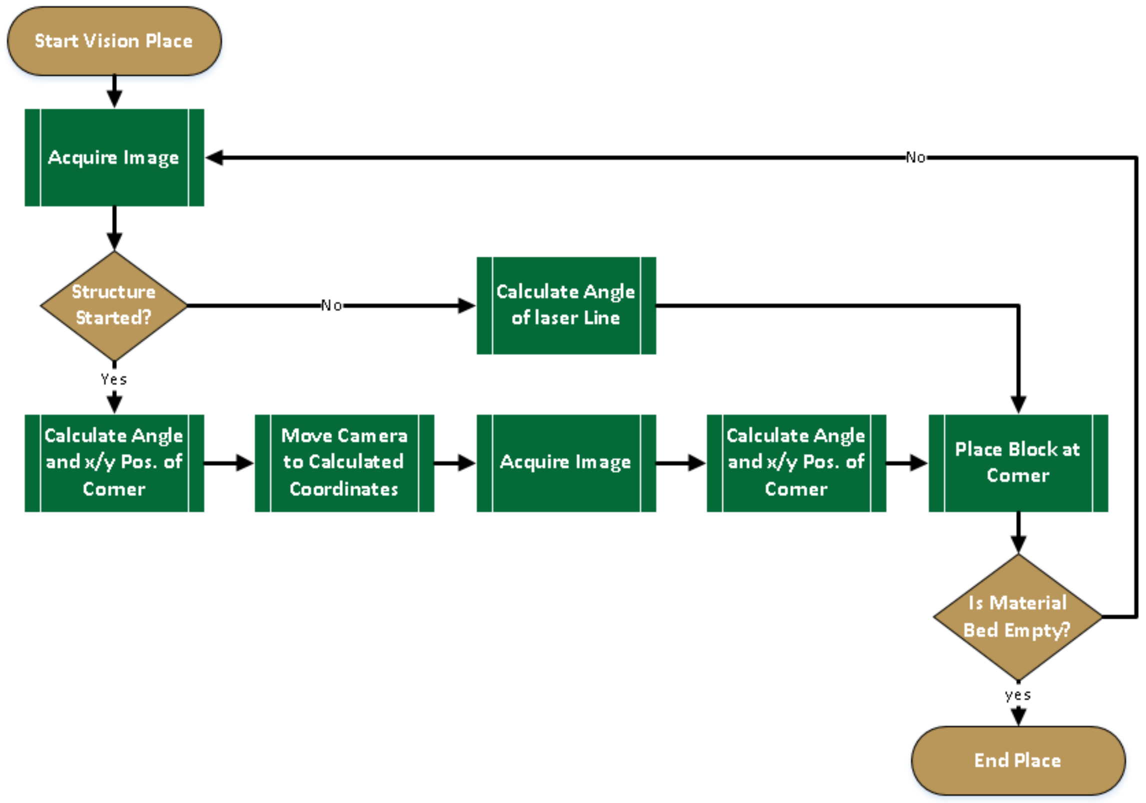

Start Vision Place
Acquire Image
No
Structure Started?
No
Calculate Angle of laser Line
Yes
Calculate Angle and x/y Pos. of Corner
Move Camera to Calculated Coordinates
Acquire Image
Calculate Angle and x/y Pos. of Corner
Place Block at Corner
Is Material Bed Empty?
yes
End Place

FIGURE 3-17: Block placement flowchart of operation

The edge finding algorithm used in the *corner finding* function was tuned with the "Advanced Edge Finding" option in the Vision Assistant application as done with the *laser line finding* function to decrease the effects of excessive noise. This was tested with all available block colors with the parameters listed in TABLE 3-3.

TABLE 3-3: Edge finding parameters used in the corner finding function

Detection Method	First Edge Projection
Search Direction	Bottom to Top
Edge Polarity	Any Edge
Minimum Edge Strength	20
Minimum Edge SNR	6.09
Kernel Size	5
Search Gap	15
Edge Angle Range	$\pm 20°$

Processing both images, the angle of the structure with respect to the camera is used to calculate the magnitude of the *x* and *y* offsets using equations (25) and (26) as demonstrated in FIGURE 3-18. These equations are very similar to those used in Section 3.4.2 but *x* and *y* are interchanged because the blocks are approximately horizontal with respect to the camera/vehicle. FIGURE 3-18 graphically represents calculated offsets from equations (25) and (26) of which the gripper should position the block. As noted in the graphic, β is the angle between the structure and the camera horizontal axis, α is the constant angle from the edge of the block to the diagonal (h). Moreover, θ is the algebraic sum of these angles used to calculate *x* and *y* displacement vectors.

$$|y| = |\mathrm{h} \cdot \sin(\theta)| \tag{25}$$

$$|x| = |\mathrm{h} \cdot \cos(\theta)| \tag{26}$$

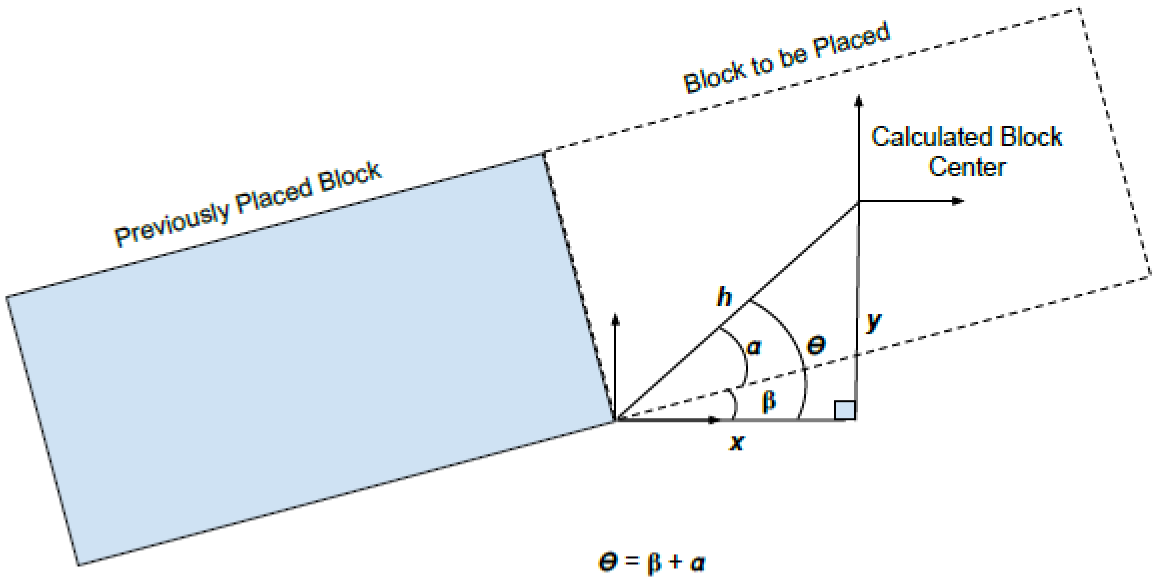


FIGURE 3-18: Block placement calculation methodology

3.5 Systematic Automation

Each of the subsystems previously mentioned must work together to complete the overall automated vehicle. Providing feedback between systems improves successful process completion. Since the CVS and myRIO are two separate controllers, they must communicate during the process for system verification. To do this, a shared variable node was generated on the myRIO device. This allows variables to be monitored by both devices in order to reveal what state the system is currently operating in.

Overall, the sub-processes specified above are condensed into two main programs; the *vehicle positioning* and *structure assembly* programs. The *vehicle positioning* program runs on the myRIO, and the *structure assembly* program runs on the CVS. The vehicle movement commands are specified for four building zone locations categorized as positions 1-4 with the loading zone remaining stationary. The vehicle is positioned at any of the positions based on the status of the structure by strafing across the surface always returning to the loading zone to be reloaded. The structure assembly program, however, is only used when it is in a position to build the structure. The *block finding* function is repeatedly called to locate blocks while the *build* functions are utilized to manipulate these blocks into the structure. An image of HARP building the structure is seen in FIGURE 3-19.

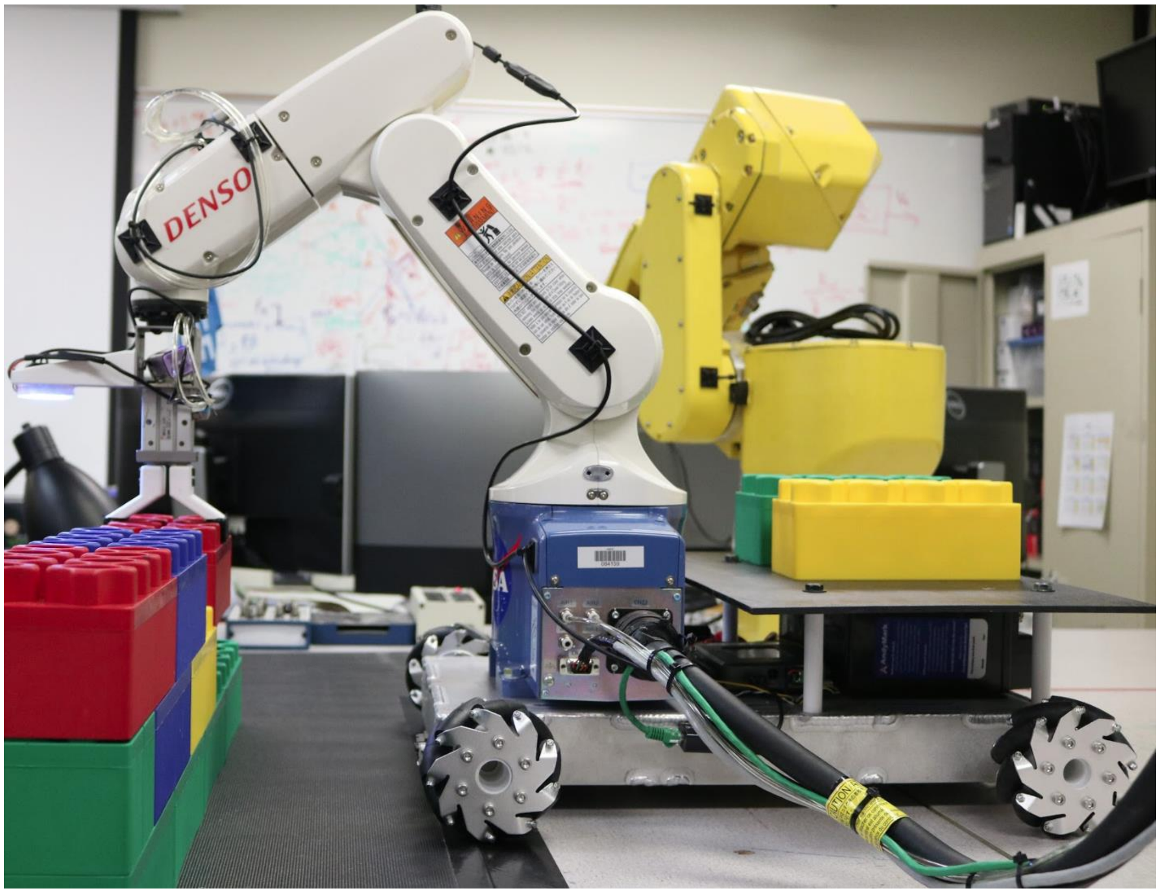


FIGURE 3-19: Complete robotic system used in testing

CHAPTER 4: RESULTS AND OBSERVATIONS

The results from the data collection methodology are presented in the following sections. In this chapter, each system is tested individually and later tested as a combined system to ensure correct operation and functionality.

4.1 Vector-drive Testing

Control and maneuverability was a vital aspect of the testing procedure. The first approach to vector-drive testing was analyzing throttle control. This is essential in order to accurately control the movement and direction of the vehicle. To do this, motors were tested from 0 to 100% in both forward and reverse. The velocity of each motor was captured in revolutions per minute (RPM) by recording encoder counts over a set period of time. These counts were then transposed into RPMs by equation (9). A plot comparing motor throttle in percent and RPMs over time can be seen in FIGURE 4-1. This test also analyzed the controllability of the motor by ramping the throttle in increments of 1.48%. Both forward and reverse responses were recorded to verify that each direction operated appropriately in a linear fashion to the maximum of 74 RPMs.

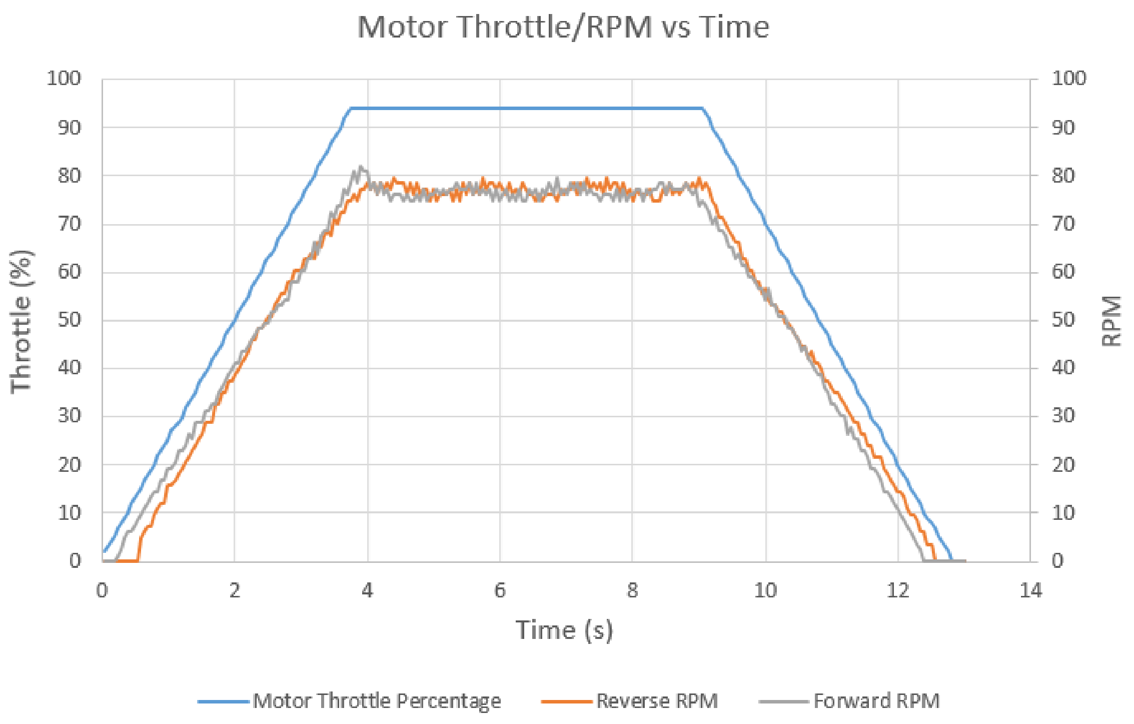


FIGURE 4-1: Motor throttle and RPM over time

Concluding this test, it was observed that better motor control for the robotic base would provide more accurate vehicle movement. It was recorded at low throttle values, that the motors were jumpy and did not provide fine throttle resolution which hindered dead reckoning and velocity control. Incorporating a speed control system with at least 16 bits of resolution could significantly impact the controllability of the base unit.

4.2 Robotic Manipulator

Accuracy testing and tool offsets were tested in this section. Accuracy testing was performed before tool offsets were finalized to verify that the positioning error of the robotic manipulator would not affect tool offsets.

4.2.1 Accuracy Testing

David Vutetakis performed a calibration for his robotic system using the exact robotic manipulator that was used in this research. This test was used to validate the manipulators' performance and benchmark the accuracy and precision thereof. A methodology similar to Mr. Vutetakis' was used to verify the manipulator's performance did not change since the arm was mounted on the vehicle chassis and mobile relative to the build plane. To test this, a foam board with a soft yet rigid material was precisely placed on the build plane. All supporting hardware including the gripper, ring light with power source, and camera were mounted on the end-effector for accurate tool weight representation. A special gripper attachment was 3D printed to project a needlepoint with a diameter of 0.40 mm as shown in FIGURE 4-2.

Using the robot arm, the needle was then inserted into the foam board ten times with 20 mm of space between each puncture. A second series of ten punctures was created with the same 20 mm spacing. On the second series, the needle was positioned and then re-inserted into each of the holes a total of ten times. In summary, twenty holes were created; the needle was re-inserted into half of these holes an additional nine times each. Once the process was complete, a Basler ACA2500 machine vision camera was fixed on a camera stand 40 mm off the surface of the board, as seen in FIGURE 4-3, to capture an image of each hole. From here, Vision Assistant was used to calibrate the camera and measure the mean diameter both sets of ten holes.

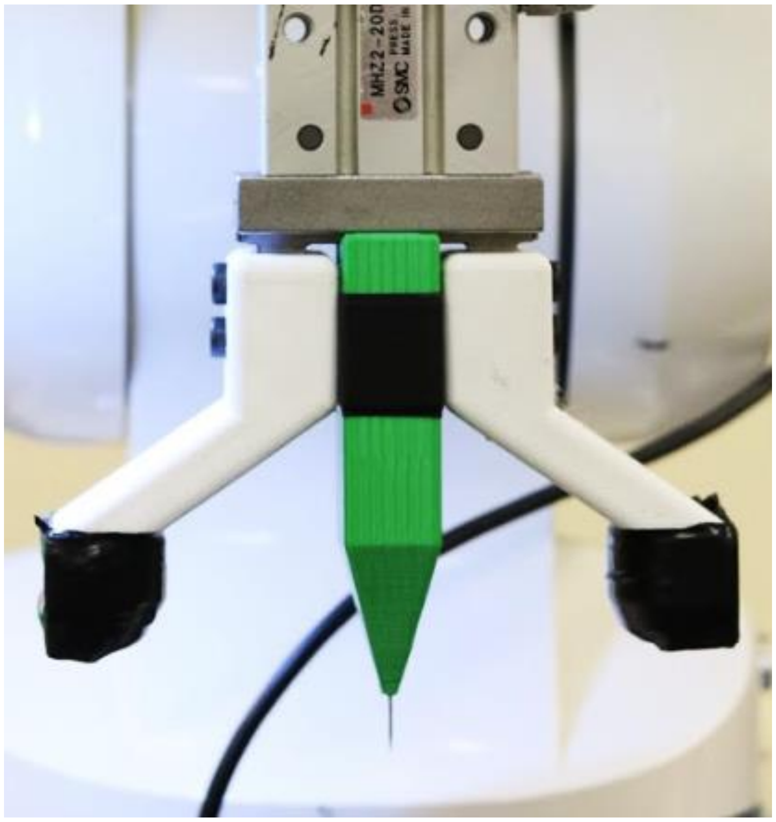


FIGURE 4-2: Gripper-mounted needle test apparatus

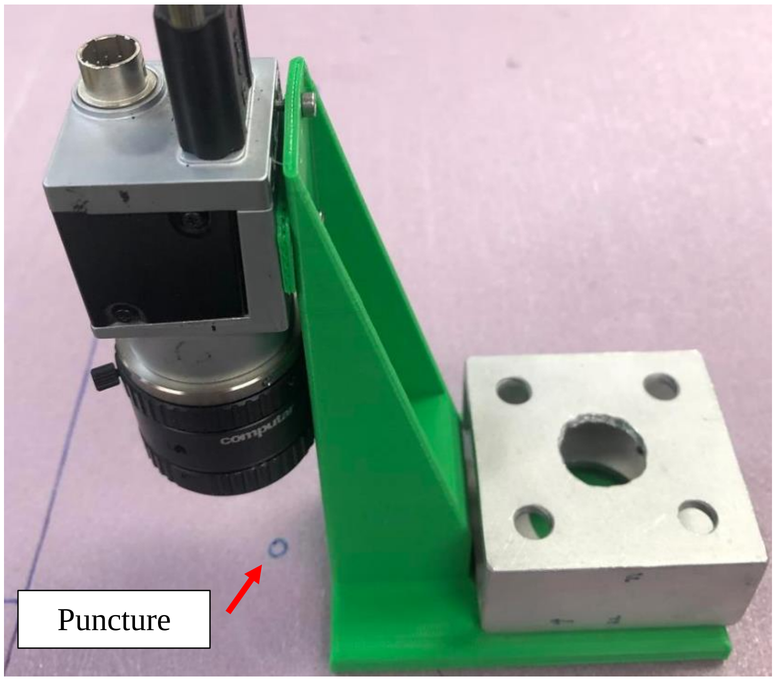


FIGURE 4-3: Camera stand for measuring the hole diameter

FIGURE 4-4 demonstrates the error measurement methodology where ε_{arm} is the manufacturer specified maximum positioning error ($\pm$0.02 mm), $r_1$is the radius of single punched hole, and r_{10} is the radius of the hole punched ten times. This illustrates the error relative to the manufacture specified error and relative to the single punched holes.

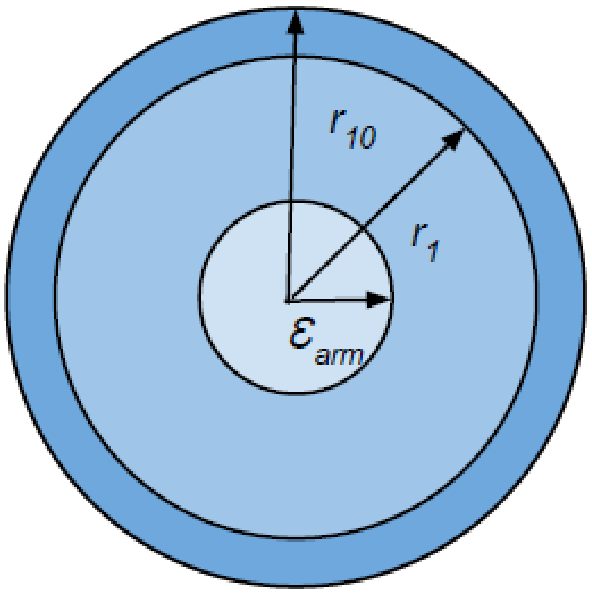


FIGURE 4-4: Position error measurements

After this experiment was completed, the mean diameter of the single punched holes was compared to the mean diameter of the 10x punched holes. Here it was determined the mean diameter of the single punched hole was 0.357 mm with the mean diameter of the 10x punched holes measured at 0.422 mm. The difference in means is 0.065 mm which can be interpreted as a positioning precision of $\pm$0.0325 mm. This test confirms that while mounted on the mobile base, the robotic manipulator positioning error is 0.0125 mm greater than the manufacture specification. This precision error was determined to be negligible due to the very small effect it would have on gripper positioning with respect to the size of the blocks/structure. The complete results can be viewed in TABLE 4-1

TABLE 4-1: Robot arm positioning test data

Hole Number	10X Holes Diameter (mm)	1X Holes Diameter (mm)
1	0.411	0.328
2	0.415	0.395
3	0.419	0.400
4	0.420	0.369
5	0.414	0.3782
6	0.414	0.356
7	0.409	0.322
8	0.402	0.354
9	0.426	0.311
10	0.490	0.357
Mean	0.422	0.357
Standard Deviation	0.025	0.030

4.2.2 Tool Offset Mapping

To verify the gripper coordinates had been successfully mapped, a series of three targets were clearly marked on the same material used in the previous test. This was done to outline where the manipulator was commanded to position the end-effector tool in real-world coordinates as seen in FIGURE 4-5. Each position was set relative to the true base coordinates of the manipulator as previously outlined in FIGURE 3-6. Position 1 was located at coordinates [-200 mm, 300 mm], position 2 was located at [0 mm, 300 mm], and position 3 was located at [200 mm, 300 mm]. Upon arrival to the position, the needle mounted on the end-effector modification was then inserted into the material leaving behind a mark on the surface. This process was repeated ten times at movement speeds of 25%, 50%, and 75% to rule out speed-related positioning errors. This test revealed that the gripper coordinates were off by negative 1 mm in the *x-axis* and positive 1 mm in the *y-axis* and are recorded in TABLE 4-2.

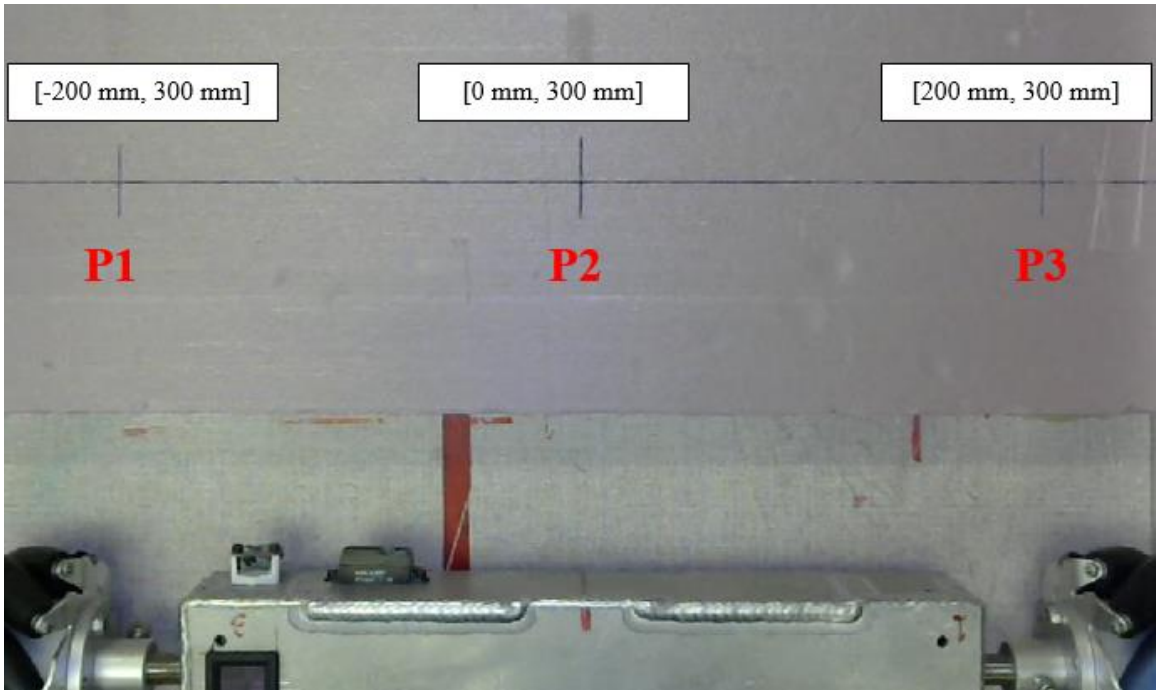


FIGURE 4-5: Robotic manipulator positioning test setup

To verify the camera offset, position P2 of FIGURE 4-5 was used for the reference center point as to where the camera should be positioned. The camera was repositioned until the intersection of P2 was measured at the center pixels of the pixel array using the Vision Assistant application. The offsets recorded in these procedures for both the camera and gripper are recorded in TABLE 4-2

TABLE 4-2: Tool offsets

TOOL Offset (mm)	X	Y	Z	Rx	Ry	Rz
Gripper	-1	1	181.31	0	0	-90
Camera	-18.5	107	50	0	0	0

4.3 Machine Vision Testing

Several aspects of machine vision are to be tested as this greatly impacts the functionality of the assembly process. Of these, calibration, edge finding, and pattern matching were of the greatest interest.

4.3.1 Image Calibration

The results from the calibration performed by the vision system are recorded at both 500 mm elevation and 350 mm elevation. The results are represented in tabular form as seen in TABLE 4-3 and TABLE 4-4. Furthermore, a plot of the error after image correction is seen in FIGURE 4-6 and FIGURE 4-7 with respect to image pixels in the *x* and *y-axis*.

TABLE 4-3: 500 mm calibration results

Mean Error	0.06220 mm
Max Error	0.07025 mm
Standard Deviation	0.00148 mm
Percent Distortion	0.10563 %
Average Calibration Time	1.19 ms

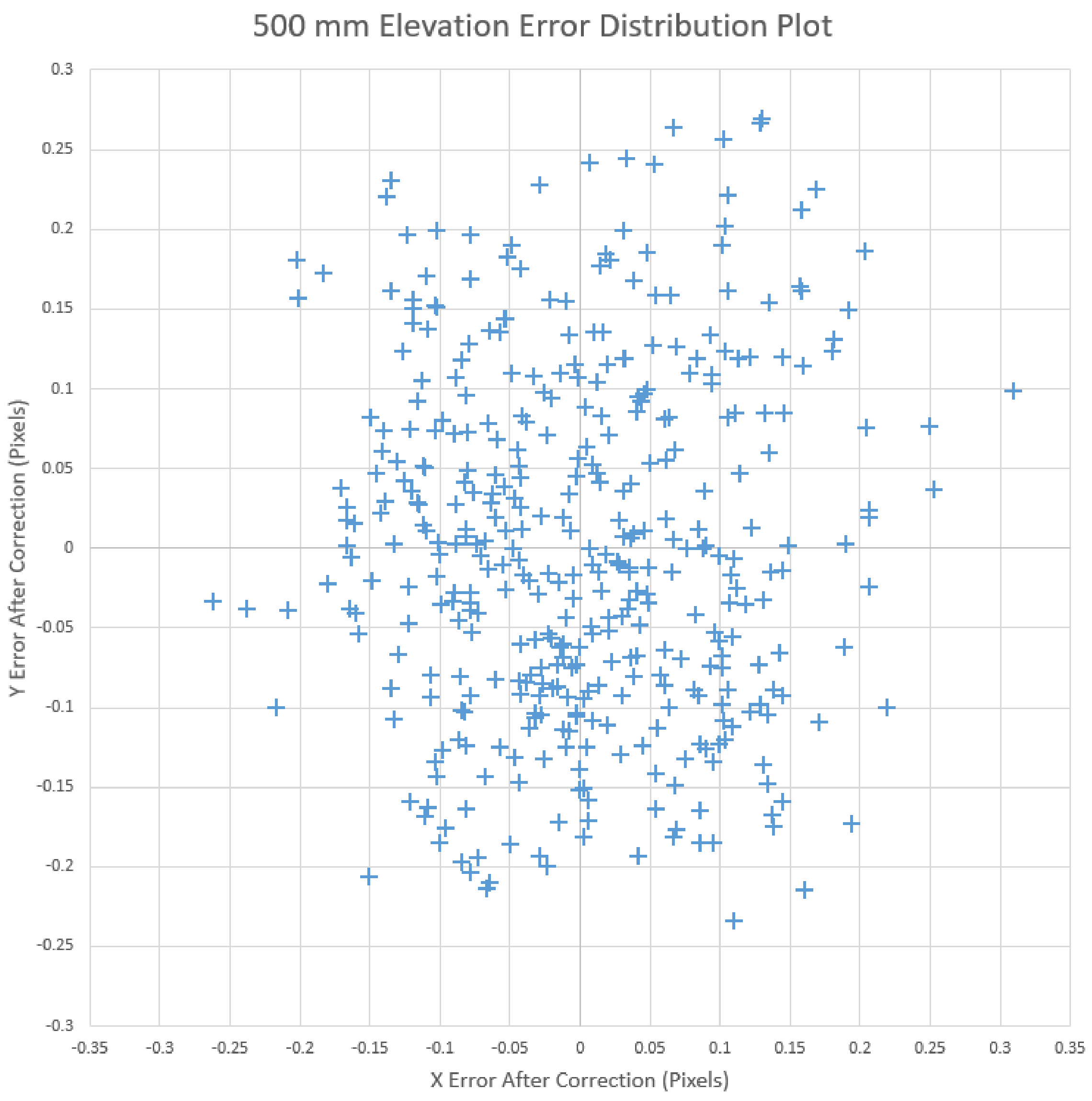


FIGURE 4-6: Error plot at 500 mm vertical offset

TABLE 4-4: 350 mm calibration results

Mean Error	0.05091 mm
Max Error	0.05285 mm
Standard Deviation	0.00037 mm
Percent Distortion	0.08810 %
Average Calibration Time	1.20 ms

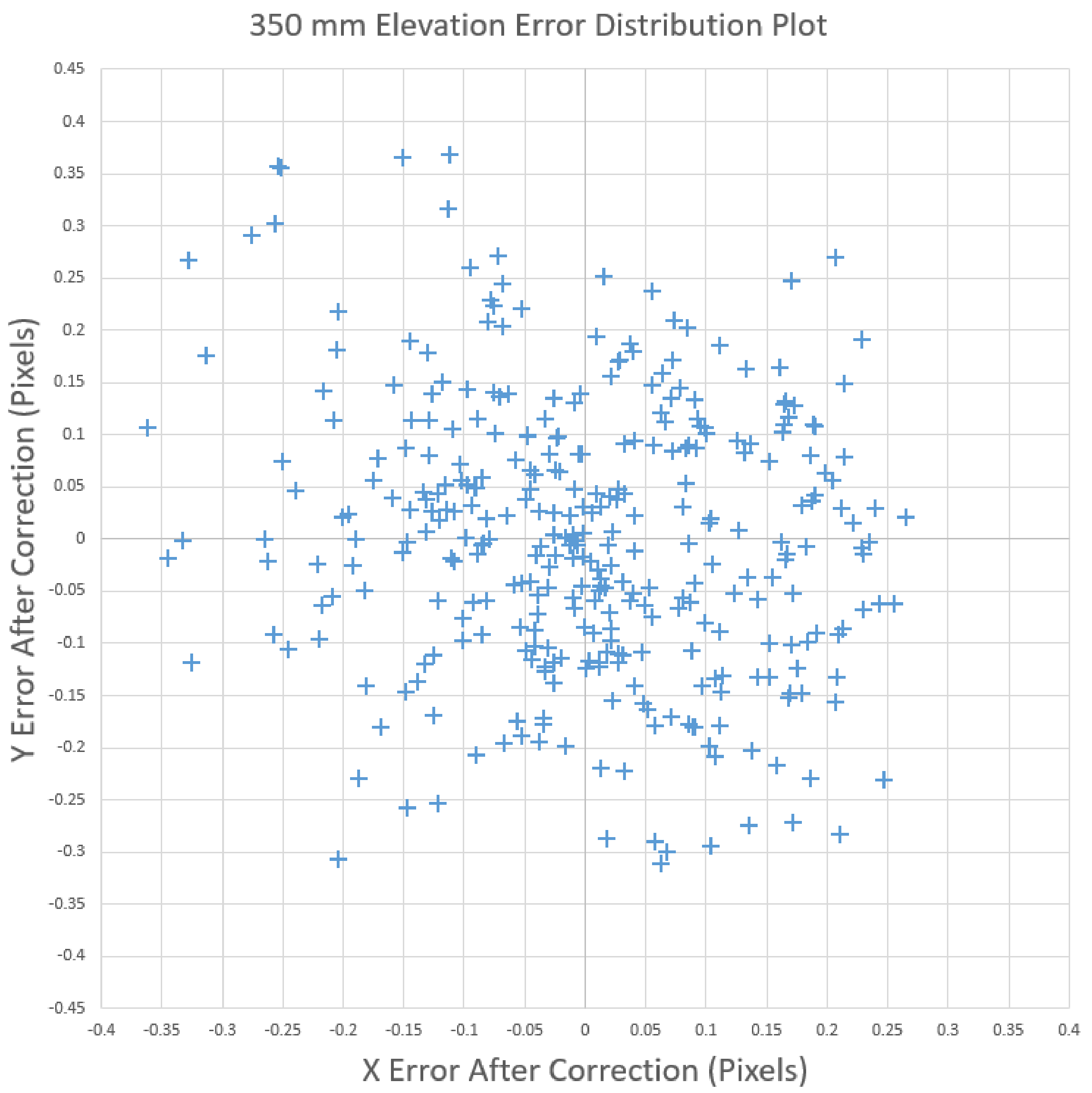


FIGURE 4-7: Error map with 350 mm vertical offset

In comparison, the maximum difference in error and distortion between the two calibrations is ~0.1 pixels, however, different calibrations are essential due to the change in elevation which would results in inaccurate measurements and calculations if not considered.

4.3.2 Edge Finding

4.3.2.1 Initial Block Location

This *initial block finding* function searched the entire camera FOV for at least three block corners on the material bed. The parameters used in this process were fine-tuned for the most accurate results and are recorded in TABLE 3-1. With these parameters set, the function was tested by placing blocks on the material bed and positioning the camera over its center. In this test, it is assumed that three blocks will be loaded on the material bed by another machine with some degree of accuracy. This was simulated by manually positioning the blocks on the material bed with at most, $\pm 25°$ rotation, separated by at least 10 mm. This test was performed a total of 25 times to prove the functionality of the algorithm. FIGURE 4-8 represents a common block configuration and successful *initial block finding.* Here, the green boxes represent the ROIs and the red lines are the calculated edges per the ROI. Of the 25 tests, three block corners were successfully located in every experiment.

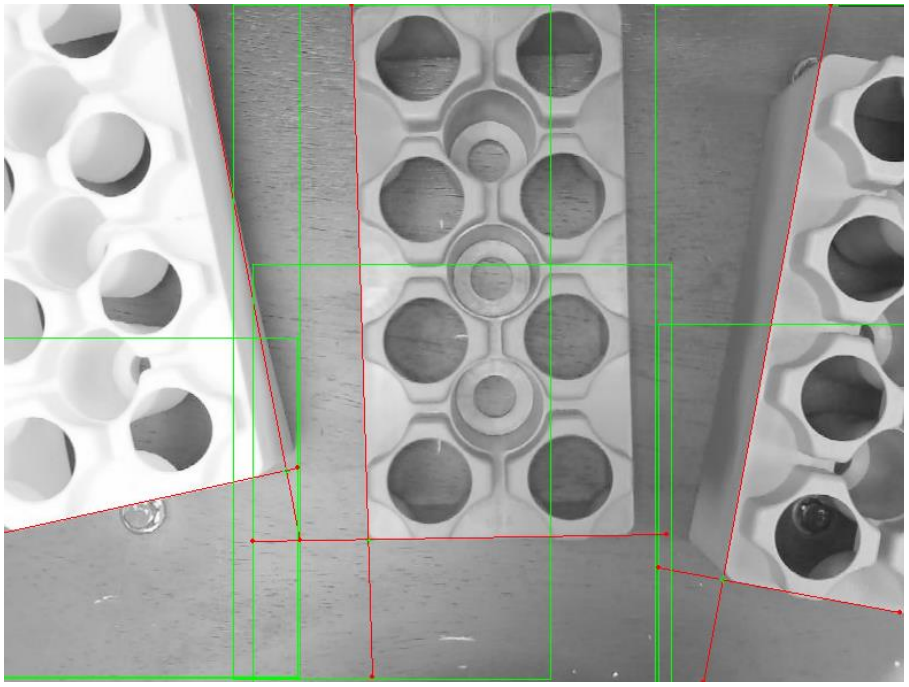


FIGURE 4-8: Initial block finding function success

After the test was completed with three blocks, it was again performed with only two blocks to examine if the algorithm reads false positive. Here it was concluded that the algorithm continued to work accurately with only two blocks if they were placed with the original parameters in the previous paragraph. It was later discovered that there was the potential for a false reading if the block merged into the opposing ROI. As a solution, the loading device would be capable of communicating how many blocks were placed on the material bed.

4.3.2.2 Laser Line Finding

Testing the *laser line finding* function, HARP was positioned at random with the laser line in the camera FOV. The function was the executed recording the results and summarizing them in TABLE 4-5. These findings revealed that if the laser line was within the camera FOV, the line was located. However, it was also discovered that as the intensity of the laser line faded, false positives could be recorded due to the decreased signal to noise ratio (SNR) in the background environment. Appropriate measures were taken to ensure the laser line intensity did not fade during operation by implementing a regulated power source. Although the laser line was located every time it was within the camera FOV at full intensity, the actual measurements of the laser's position and orientation varied slightly. This error was used to characterize the accuracy of the *laser line finding* function with an average *y-axis* misalignment of 5.04 mm and a standard deviation of 3.28 mm. The angle was recorded to have an average error of 0.88° with a variance of 0.87°. It is important to note that the “Actual Laser Position (mm)” is measured from the center of the manipulator base to the laser line whereas “Measured Position (mm)” is computed from a combination of the vision and manipulator data.

TABLE 4-5: Laser line finding results

Actual Laser Angle (deg.)	Measured Laser Angle (deg.)	Actual Laser Position (mm)	Measured Position (mm)	Processing Time (ms)
~0	-0.73	366.27	366.46	96
6.43	7.75	356.33	352.03	96
10.64	12.78	330.63	323.50	94
-5.37	-5.27	346.52	355.38	94
-9.4	-9.31	384.77	389.45	95
Mean	-	-	-	95
Error Standard Deviation	-	-	-	0.82

4.3.2.3 Corner Finding

Testing the *corner finding* function was completed similar to the laser line finding test by positioning HARP at random with the partially assembled structure in the camera FOV. The full *corner finding* function was executed to record the results through five different tests as summarized in TABLE 4-6. Here it was determined that the two-step corner process was beneficial by examining the position error between the 500 mm and the 350 mm results. This error is a result because the 500 mm image is taken in line with the center of the vehicle, and the 350 mm results are taken in closer alignment to the corner of the structure with the provided offset coordinates of the 500 mm calculation. The visual perception of this is represented in FIGURE 4-9 where image "a" is taken at 500 mm above the center of the build plane, and image "b" is taken 350 mm above the corner of the structure at the build plane, as previously explained in Section 3.4.3.

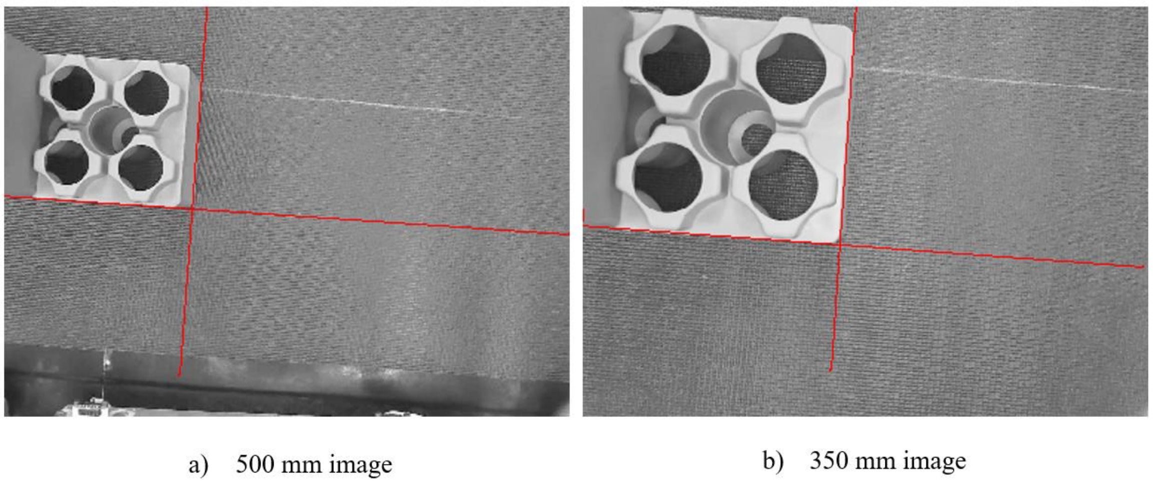


FIGURE 4-9: 500 mm and 350 mm image perceptions

The results in TABLE 4-6 are indicative that the average error in the *x-axis was* 9.99 mm and 17.78 mm in the *y-axis* with an error variance of 4.04 mm in the *x-axis* and 3.36 mm in the *y-axis*. This low variance and high mean error indicate that the two-step process is beneficial to block placement. The coordinates for these results were then verified to be true by placing a block at each position in-line with the partially completed structure. Further testing of this function was completed in section 4.4 where structure completion error was recorded.

TABLE 4-6: *Corner finding results*

Test Number	500 mm	350 mm	$\lvert Error\ X \rvert$	$\lvert Error\ Y \rvert$	Processing Time
1	(-59.50,-8.47)	(-72.07,-22.95)	12.57	14.47	202
2	(-9.98,45.35)	(-14.86,27.26)	4.87	18.09	207
3	(-8.94,7.11)	(-15.30,-11.35)	6.36	18.46	200
4	(-51.30,-31.47)	(-63.94,-46.49)	12.64	15.02	203
5	(64.94,-14.48)	(51.45,-37.37)	13.49	22.89	209
Mean	-	-	9.99	17.78	204.20
Standard Deviation	-	-	4.04	3.36	3.70

4.3.3 Pattern Matching

Concluding the edge finding algorithms, pattern matching was completed. The *pattern matching* function was challenged to locate both full blocks and half blocks, however, testing of each size was completed separately.

4.3.3.1 Pattern Matching: Full-Size Block

The first test performed by this algorithm was to experiment with the script's ability to find a block in random positions and orientations within the camera's FOV. To complete this test, a full-size block was placed randomly on the material bed with at least 95% of surface area within the cameras FOV. This was performed 40 different times to validate the pattern matching abilities, by successfully finding 40 of 40 matches. Each located block coordinate and angular rotation is compiled in FIGURE 4-10 and FIGURE 4-11. Where the red line in FIGURE 4-10 represents the *x-coordinate* distribution with a standard deviation of 42.69 mm, and the blue line represents the *y-coordinate* distribution with a standard deviation of 27.55 mm. The viewable area in this camera position is 300 mm x 200 mm, calculating that the variance of block position was 14.2% in the *x*-axis, and 13.78% in the *y-axis*. The black line in FIGURE 4-11 represents the angular rotation distribution with a standard deviation of 116.61 degrees. The large variance with respect to FOV indicates that the block was placed randomly with respect to the camera module FOV.

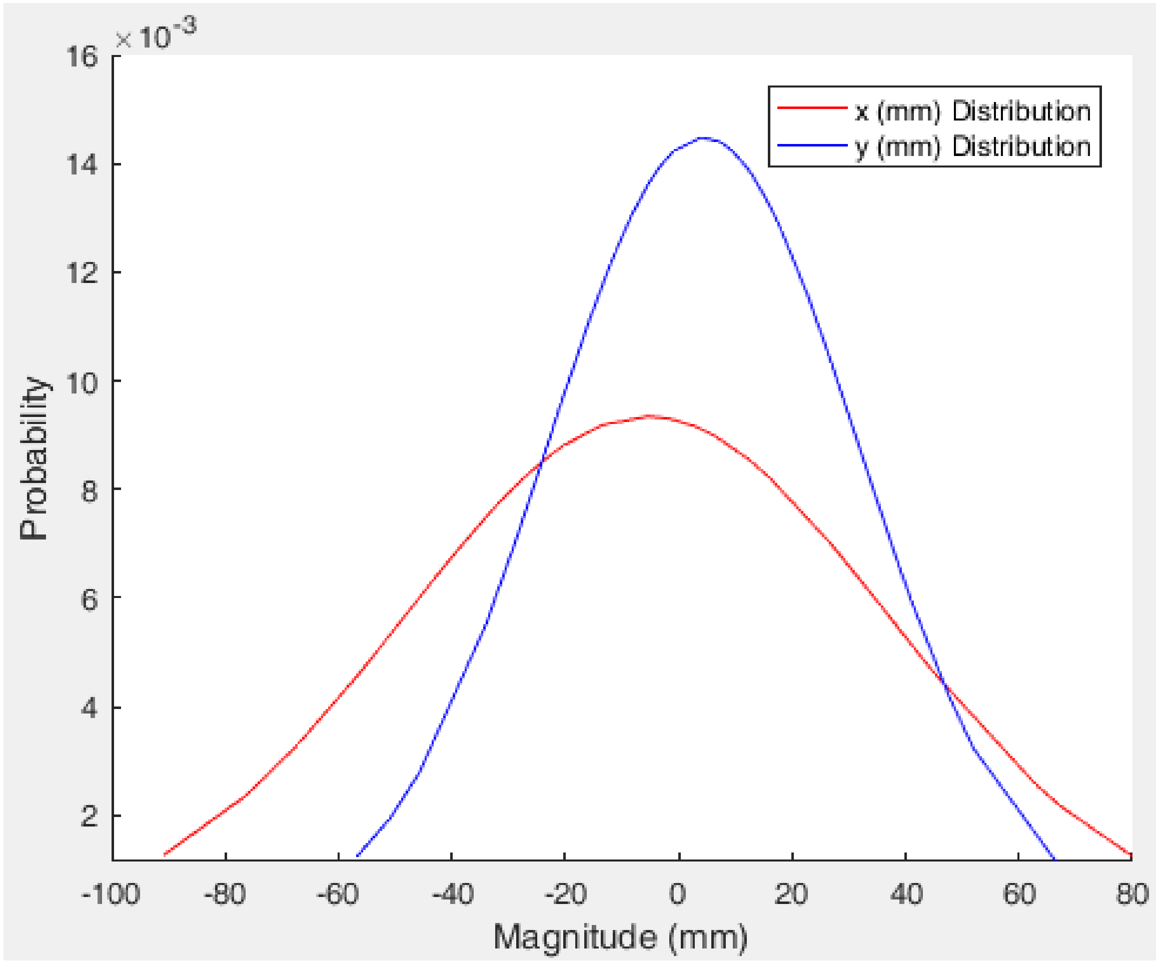


FIGURE 4-10: Normal distribution of block coordinates

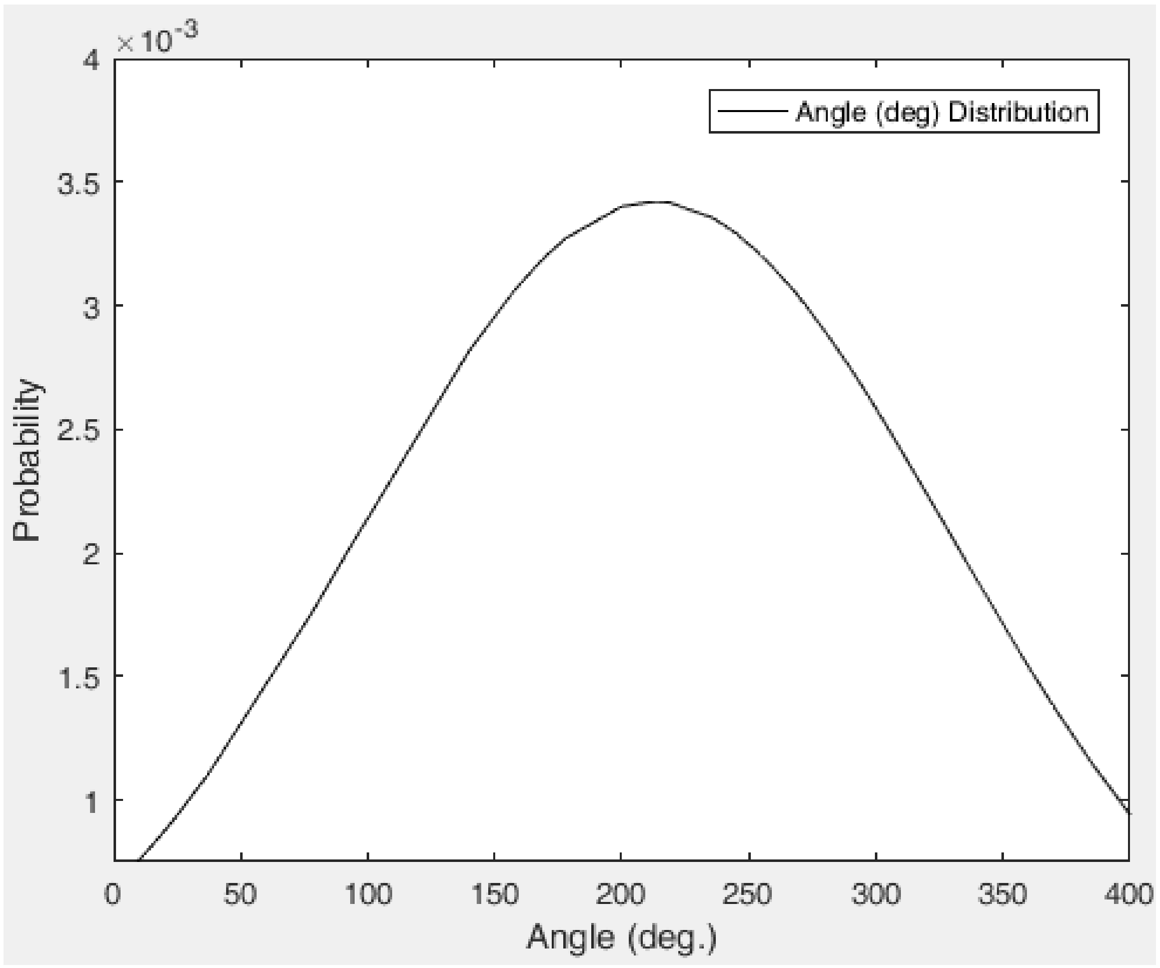


FIGURE 4-11: Normal distribution of block rotation

The processing performance of each image varied slightly. Further testing of the vision script was performed to analyze if position, rotation, or color affected the processing time. The first examination of the pattern matching test was to compare the match score vs both position, and angular rotation. This was done to record if the position or angular rotation affected the results. The angular rotation test was accomplished by placing a full block from $0° - 360°$ degrees in increments of $45°$ directly under the center of the camera lens. For the position test, the block was placed with vertical alignment in random positions away from the camera center with zero rotation. It was shown that block position relative to the camera negatively affected the match score more than rotation. However, it was observed with angular movement, processing time increased by 289.28 ms on average. The average score for the angled block was 963.22/1000 with a standard deviation of 10.41. For the random position, the average score was 938.61/1000, with a standard deviation of 18.72. FIGURE 4-12 displays the data captured in this experiment with the blue bars as match score vs position, and the orange bars as match score vs angular movement. This test demonstrated that with better initial alignment to the block, the results are calculated faster and with more accuracy.

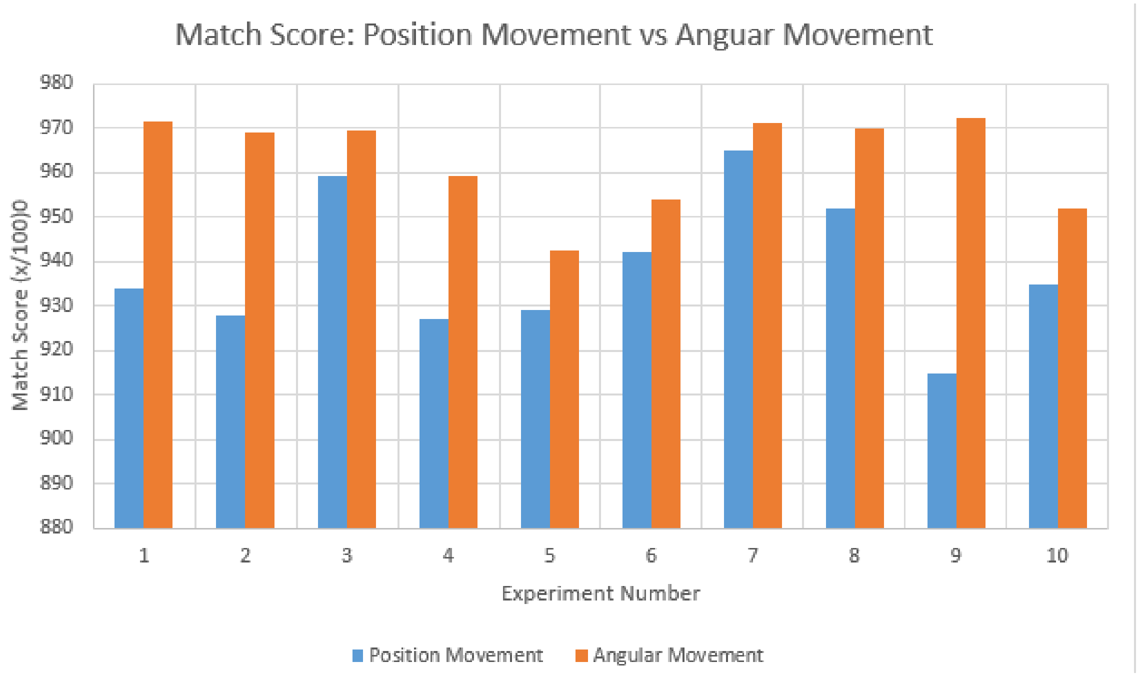


FIGURE 4-12: Match score relative to the position, and angular rotation

When testing the effects of color, it was discovered that each image resulted in a mean processing time of 1235.42 ms with a standard deviation of 133.88 ms. The data represented in FIGURE 4-13 implies that color had little correlation on processing time except for the yellow block. On average, the *pattern matching* function processed the yellow block images 247.23 ms faster than the average processing time of all other colors.

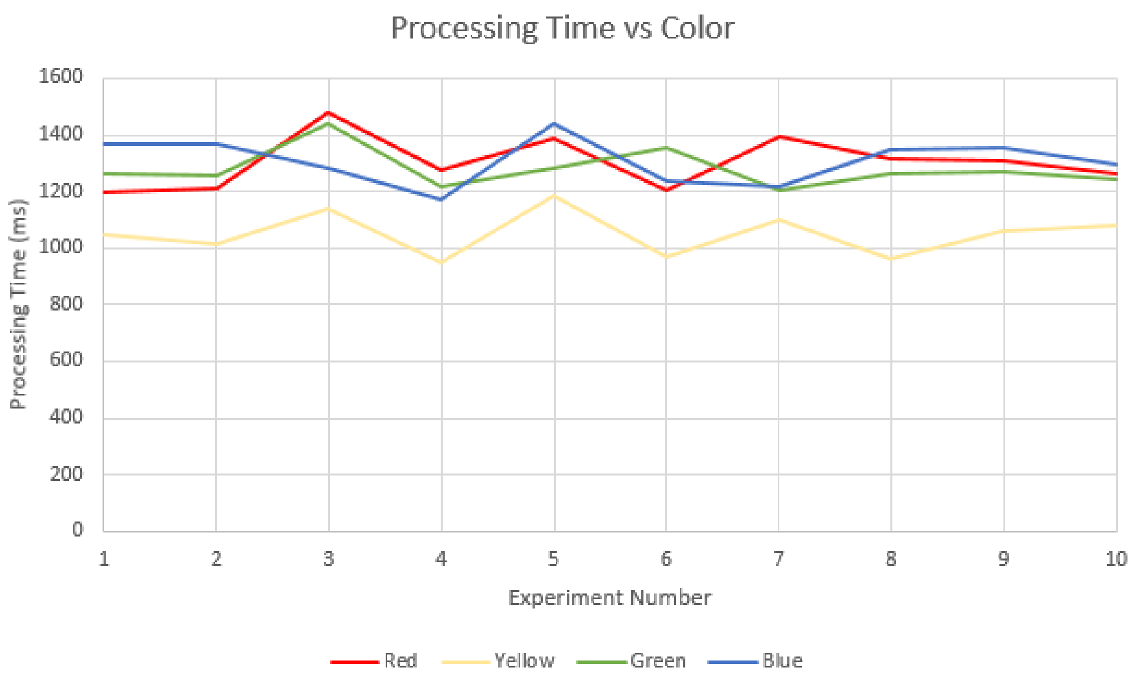


FIGURE 4-13: Processing time vs block color

FIGURE 4-14 illustrates processing time versus block rotation. Here, the plot indicates that there is little correlation between processing time and magnitude of angular rotation.

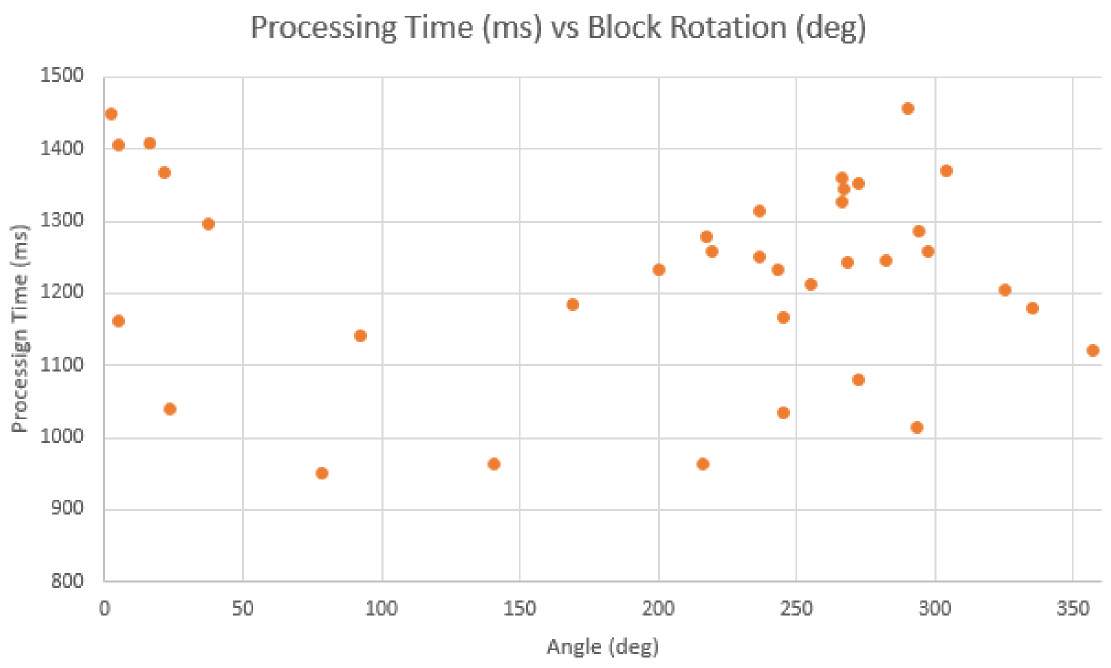


FIGURE 4-14: Processing time vs block angle experiment

FIGURE 4-15 presents processing time vs block position. As with block rotation, there appears to be no correlation with processing time versus position.

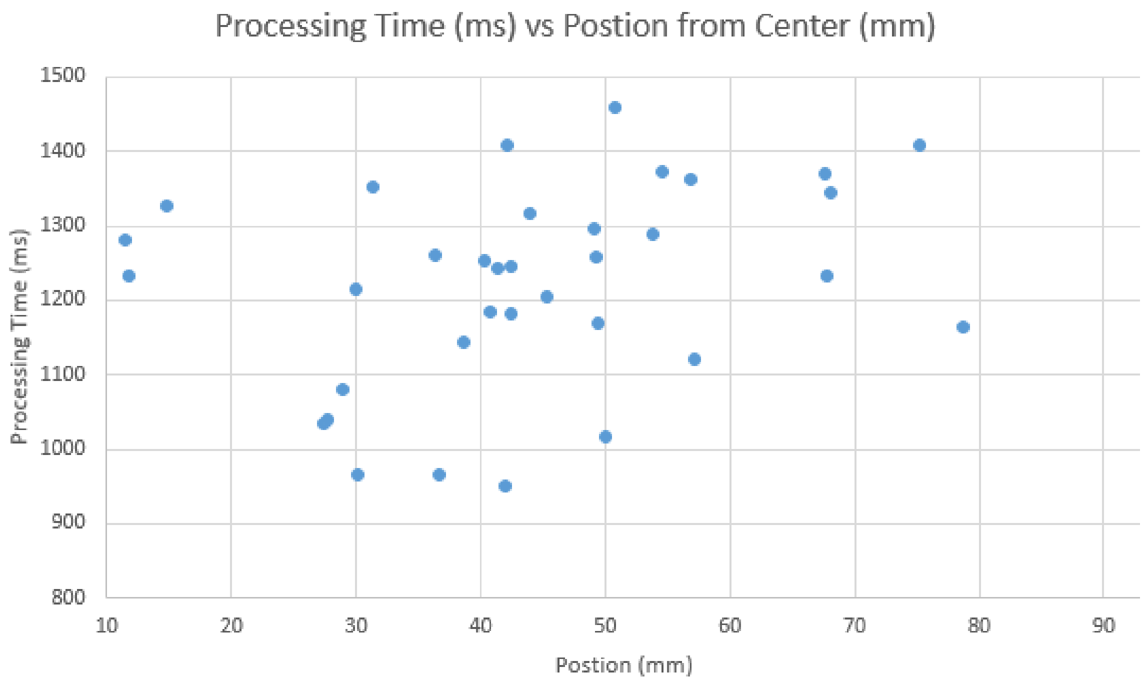


FIGURE 4-15: Processing time vs block position experiment

Further testing of the pattern matching algorithm revealed that it could search the image for the presence of a template in one-degree increments approximately every 3-5 ms. This was noted such that the easiest way to reduce processing time was to narrow the search angle. Since the *initial block finding* function was used to re-orient the camera over the approximate block location and angle, the *pattern matching* function was tuned to search the image within $\pm 25°$ for the template rather than $0° - 360°$. This resulted in processing the image 5.27 times faster; or processing an image every 245.5 ms on average. A chart comparing the processing times with $360°$ (orange) and $\pm 25°$ (blue) is illustrated in FIGURE 4-16.

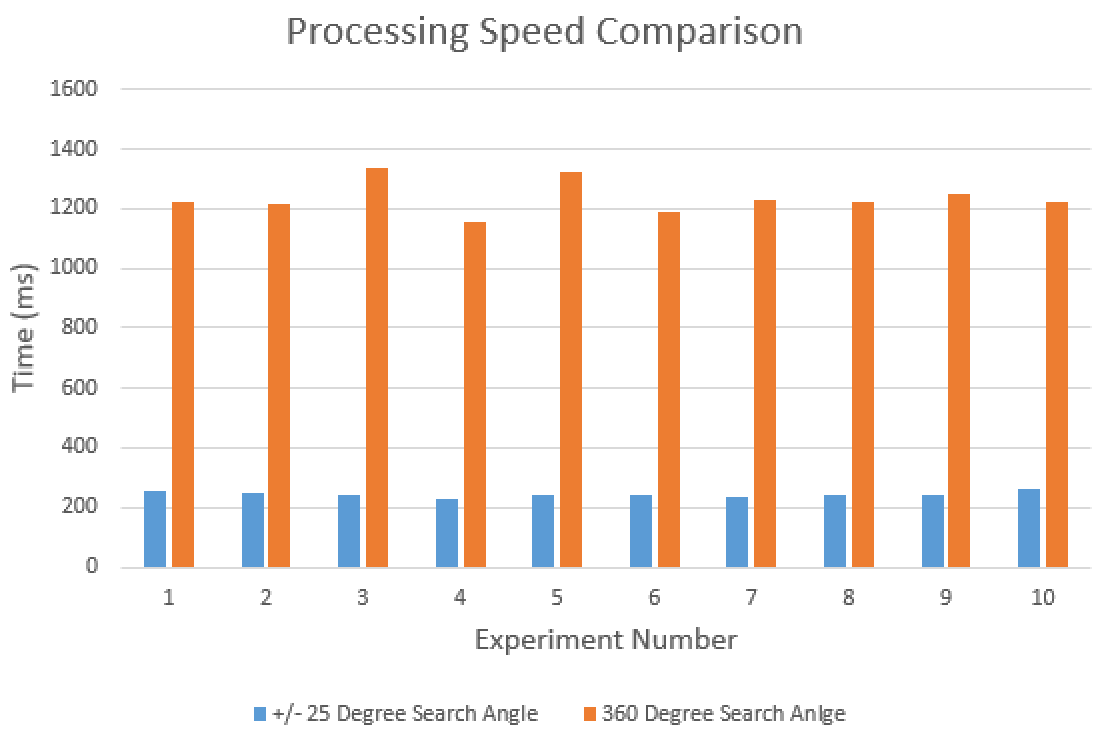


FIGURE 4-16: Reduced processing time comparison

4.3.3.2 Pattern Matching: Half Size Block

Repeating the steps in Section 4.3.3.1, the half-block pattern matching routine was tested 40 times with the object placed randomly in the camera FOV. The results from this test are summarized by the plot in FIGURE 4-17 where the red line represents the *x-coordinate* distribution with a standard deviation of 55.24 mm, and the blue line represents the *y-coordinate* distribution with a standard deviation of 38.76 mm. Again, the spread of this dataset indicates that the block was placed randomly within the camera FOV. Of these, the block was found a total of 40 out of 40 times with an average matching score of 831.4/1000, and matching score standard deviation of 37.85.

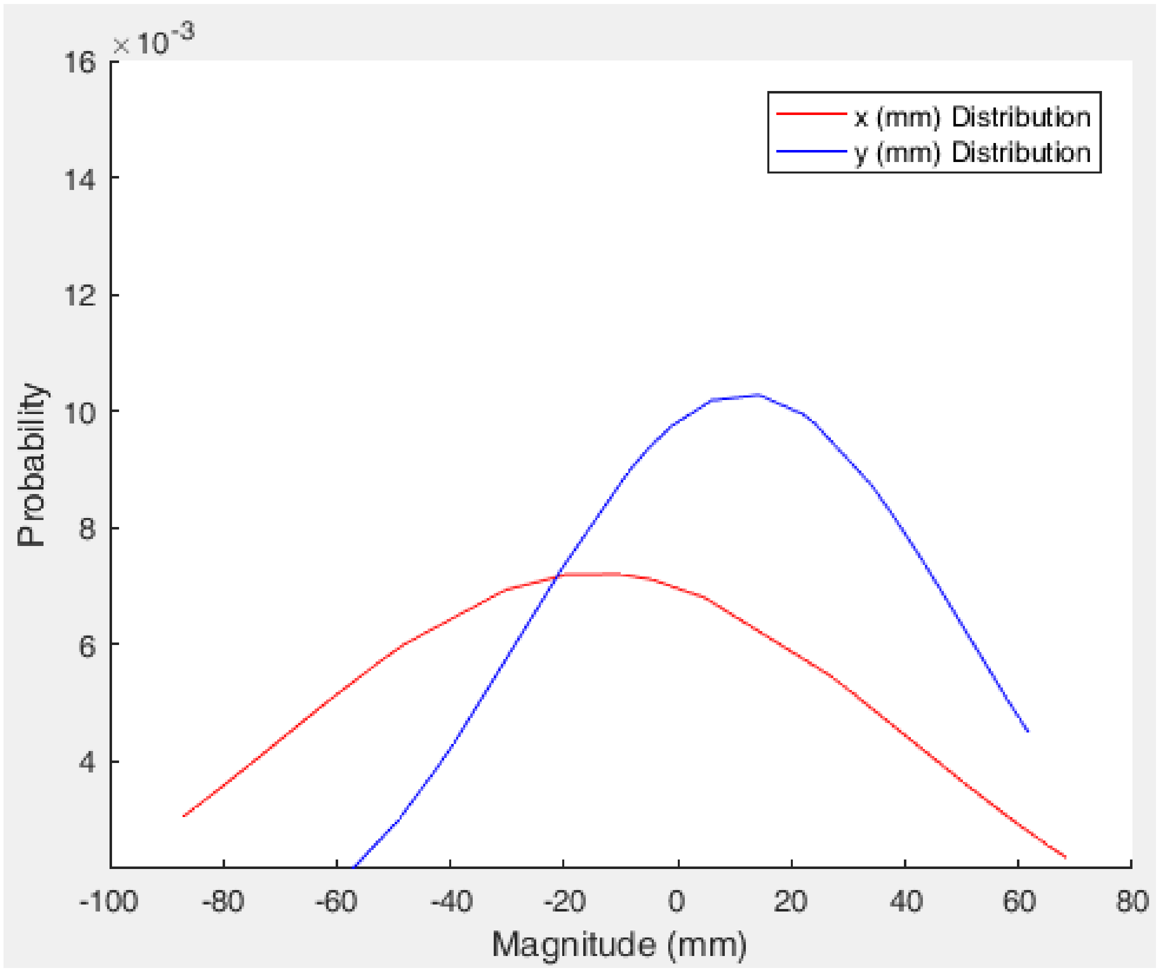


FIGURE 4-17: Scatter plot of located block positions

As done with the full block test, processing time was analyzed examining the effects of position, rotation, and color. It was discovered that the processing results were similar to that of the full-block tests such that the test had little correlation to processing time. The results from the half-block pattern matching tests are summarized in TABLE 4-7. Here, processing time has a mean of 493.75 ms and a standard deviation of 40.94 ms. As with the full-block pattern matching, the yellow block processing time was faster; on average, 52.5 ms. However, after lowering the search region to ±25°, the mean processing time of all colors was reduced to 131.80 ms with a standard deviation of 6.11 ms as presented in the following table.

TABLE 4-7: Half-block pattern matching results

Block Type	Average Processing Time 360° (ms)	Average Processing Time $\pm 25°$	Standard Deviation 360° (ms)
Red	515.00	177.20	40.94
Blue	527.90	196.10	
Green	514.10	184.72	Standard Deviation $\pm 25°$ (ms)
Yellow	466.51	167.33	
Average	493.75	181.35	14.92

4.4 Autonomous Structure Assembly

Several aspects of the overall autonomy were analyzed. The primary focus of scoring was to analyze the placement of the blocks by recording the first block placement relative to the laser line compared to the position of the final block placed relative to the laser line. Performing this test presents a model for accumulated error over a one-meter span. In a separate test, the assembly process was timed to test consistency and determine if any process variation resulted in increased build times. Furthermore, to verify that the vehicle positioning devices worked within their programmed tolerances, rangefinder distance and encoder counts were recorded throughout the process and analyzed.

During the automated building process, the imaging system searches for a laser line to place the first block. When the line is located, the system then places the block coincident to the laser line. At this point, the placement error relative to the laser line is recorded. From here, the program adds onto the structure three blocks at a time until the structure is completed. Since the blocks are not permanently cemented in place, each consecutive build operation could potentially perturb previously set blocks resulting in a drifting error if the placement is not properly aligned. The error of the structure relative to the laser line is recorded throughout the process. This is used to measure where, if any, error accumulates during structure assembly. An illustration describing potential error measurements is described in FIGURE 4-18; this graphic is exaggerated and not to scale for illustration purposes.

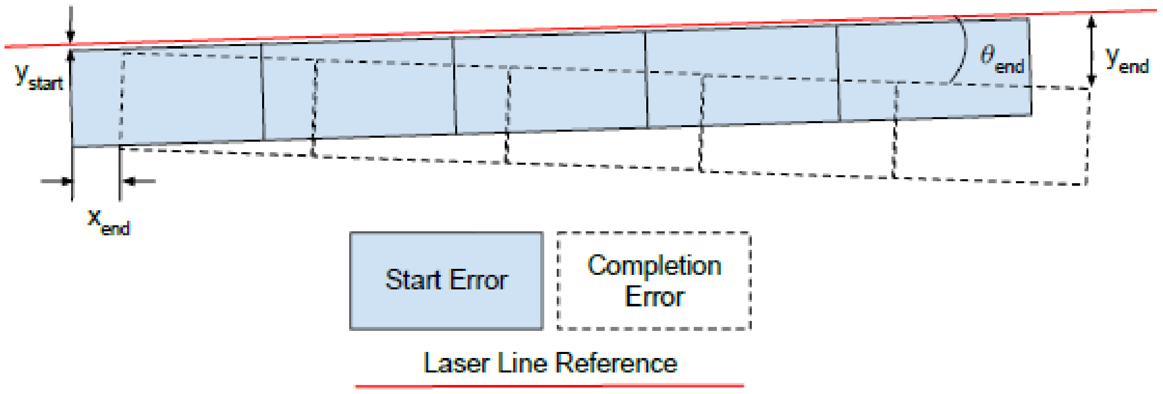


FIGURE 4-18: Structure start and end error diagram

TABLE 4-8 summarizes the error over five different tests. It is important to note that the error in the *x-axis* will always be zero for the first block placement since it sets the starting point of that axis. The completion error of the *x-axis* for the final block placement is measured with respect to where the first block was placed with a mean of 4.02 mm. The average structure start error was off by 2.71 mm in the *y-axis* with an angular misalignment of 0.113°. To complete the structure, the average *y-axis* error relative to the laser line was 2.78 mm with angle misalignment was off by 1.307°.

TABLE 4-8: Structure completion error

Test Number	Build Start			Build Finish		
	$\varepsilon_x\ (mm)$	$\varepsilon_y\ (mm)$	$\varepsilon_{angle}\ (deg.)$	$\varepsilon_x\ (mm)$	$\varepsilon_y\ (mm)$	$\varepsilon_{angle}\ (deg.)$
1	0	0.55	0.161	3.28	3.46	1.376
2	0	4.73	0.129	4.36	6.93	0.129
3	0	3.25	0.070	4.4	2.06	3.008
4	0	3.03	0.166	4.65	0.2	0.166
5	0	1.97	0.041	3.4	1.27	1.856
Mean	-	2.71	0.113	4.02	2.78	1.307
Variance	-	1.56	0.056	0.63	2.60	1.214

Next, the overall completion time of the assembly process was evaluated. Recorded in TABLE 4-9, the autonomous assembly process was completed a total of five times. The length of time was measured and any visually observed errors that occurred were recorded. These five tests revealed that the average structure completion time was 6 minutes 20.9 seconds with a standard deviation of 7.5 seconds. Overall there was one major error in test 1, which occurred when placing the last block at build position 3 was not properly aligned. This resulted in having to manually fix the block to continue the process. Test 2 and 5 had minor errors at which the block slipped out of the gripper. The block was then re-inserted into the gripper and the process was continued. This was not considered a critical error because the gripper was at fault and could be improved for a better mate with block geometry.

TABLE 4-9: Structure completion time

Test Number	Minutes	Seconds	Comments
1	6	17.13	Place top block incorrectly
2	6	20.804	1 block slipped out of the gripper
3	6	17.695	Success
4	6	13.608	Success
5	6	35.138	1 block slipped out of the gripper
Mean	6	20.875	-
Standard Deviation	0	7.488	-

Completing the full-system tests, the Shark IR sensor data was recorded with respect to encoder counts over time as illustrated in FIGURE 4-19 and FIGURE 4-20. Also pictured in this graph are the setpoint values to signal the vehicle to stop when it has reached its destination. The purpose of this comparison is to analyze the difference in encoder counts and rangefinder data relative to their setpoints when the vehicle has arrived at a build location. FIGURE 4-19 represents this data for traveling to the first position. Here it can be seen that both the encoder count and rangefinder setpoint values have been reached upon arrival at build position one as highlighted by the red oval. FIGURE 4-20 presents this data relative to position 4. This figure demonstrates that the range finder stopped the vehicle before the encoder set point was reached. This is because the vehicle would have collided with the structure if only encoder counts were used for positioning. Using encoder counts for positioning accumulates error over the course of the assembly process resulting in positioning drift. This pattern was noticed throughout the assembly process proving that dead reckoning alone may not be sufficient in positioning schemes due to wheel or roller slip in acceleration/deceleration. Sensor fusion between the encoders and range finders was essential in successfully traveling to and from the structure for the assembly process.

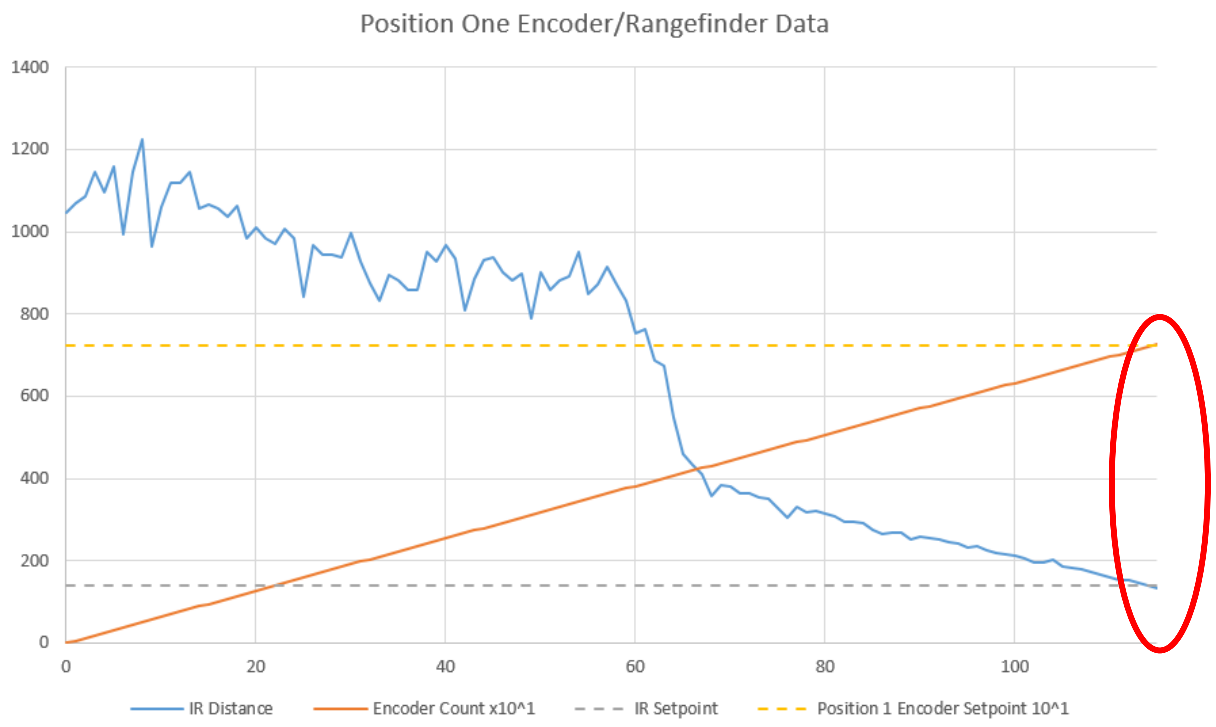


FIGURE 4-19: Position one encoder count and object distance relative to set points

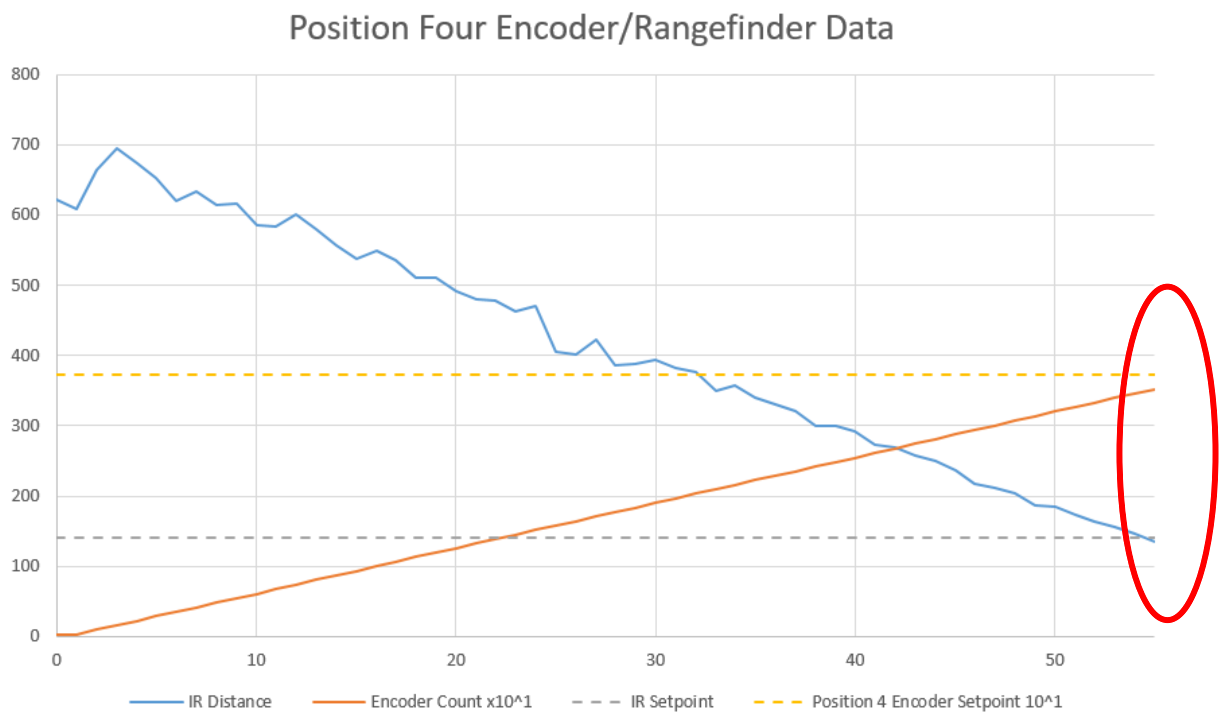


FIGURE 4-20: Final position encoder count and object distance relative to set points

CHAPTER 5: CONCLUSIONS

5.1 Vehicle Chassis

Overall, mobilization of the robotic system was considered a success and a critical factor of the simulated concept. All hardware with the exception of the RC8 controller and CVS was fitted on the vehicle chassis. This system also was capable of carrying three blocks to the building zone. The vector-drive enabled the HARP to strafe the surface to the desired build position while maintaining a perpendicular heading with respect to the building zone. This provided an efficient means for transportation of the assembly hardware. It was noted after preliminary testing that dead reckoning was not sufficient in vehicle positioning, thus the ranging devices were implemented to ensure the correct spacing from the structure and that the vehicle did not collide with an object in its path. The preliminary results indicate that these positioning schemes still may not be sufficient in vehicle positioning. Another means of vehicle localization should be considered in future applications so more accurate traveling can be accomplished in its environment.

5.2 Vision Guidance

The vision system was successful in locating building material of two different sizes for the assembly process. Locating where to place the material was also successful via vision guidance and was confirmed in vigorous testing as outlined in Chapter 4. Since accuracy was the primary concern for the vision system, all block/structure locating based algorithms work on a two-step process. An image was taken to calculate the approximate location of the material first since the object of interest was often not completely in the camera FOV. Using these results, a second image was captured at these locations to calculate the exact position of the material/structure. This two-step process enabled a self-checking algorithm so that the vision system could accurately locate the blocks and/or structure.

The block finding algorithm was improved to locate the building material with a 96.32% match, every 245.5 ms. The overall image processing time was reduced by narrowing the angular search region to $\pm 25°$ which was 5.27 times faster than searching $360°$ for the object. Testing revealed that the block position relative to the camera had no correlation with processing time but decreased the overall match score by 2.46 % on average. Interestingly, processing images with yellow blocks were observed to be computed 247 ms faster than that of the red, blue, or green average block processing times. Calculating where to place the blocks was accomplished using advanced edge finding techniques to locate the corner of the structure. Testing proved this to be an appropriate means of structure assembly by recording 4.02 mm of error in the *x-axis* and 2.78 mm of error in the *y-axis* on average resulting in a 5.82% structure completion error.

5.3 Robotic Manipulator

It was also confirmed that the DENSO robot arm was sufficient in end-effector positioning whilst mounted on the mobile base. The needle puncture test as performed in section 4.2.1 proved that the precision of the end-effector positioning could be repeated to $\pm$0.0325 mm while mounted of the base unit. This variance is much smaller than the smallest block size (100 mm x 100 mm) and was considered to be negligible. Through the use of a 3D-printed gripper, the end-effector was successful in interfacing with the emulated building blocks. The 6 DOF supplied by the DENSO system proved to be sufficient in picking and placing the material in any 3-dimensional position within its ROM.

The gripper was designed such that it directly mated with features on the emulated building blocks used in this research. An important detail in the gripper finger design significantly increased the reliability of the pick operations. A 7 mm by 7 mm chamfer on the tooltip passively allowed for slight misalignments of up to ~3.5 mm that could have conspired from the coordinate transposition of the vision system or positioning error. Also, it is important to note that the gripper was geometrically optimized to interface with the blocks used in this research, and could be remodeled to mate with a wide range of block geometries.

5.4 Structure Assembly

A simulated structure 0.25 m tall by 1 m long was successfully assembled via the *build straight wall* function. It is presumed that the *build straight wall* function is one of the many structure assembly functions that could be programmed into the machine i.e. *build arc, build roof*, etc. This structure was successfully assembled multiple times throughout the testing process. Complete testing revealed that structure completion was assembled with an average position error of 4.02 mm in the *x-axis* and 2.78 mm in the *y-axis.* Furthermore, it was concluded that the average structure assembly time was measured to be 6 minutes and 20 seconds placing a total of 16 blocks per build. Overall, this vehicle served as an initial research platform such that a proof of concept model was successfully developed to autonomously erect a simulated structure. This was accomplished by assembling manageably-sized building blocks rather than attempting to 3D-print entire habitats in real-time/space.

5.5 Future Work and Other Applications

When considering improvements to the overall system operation, many conclusions can be made. The most significant is likely upgrading the vision system to incorporate stereovision such that better depth perception can be accomplished. Although this was not a large factor in a controlled environment, it will likely be difficult to place blocks more accurately without improving the depth perception of the vision system to accommodate for un-parallel working planes. Since the camera is mounted on the gripper of the robot arm, it could be positioned in different locations for image acquisition such that simulated stereovision could be accomplished with only one camera. The assembly process could further be improved by using the vision system to inspect the assembly progress intermittently and correct misplacements in real-time to ensure that the structure is completed as expected. This improvement would solve the critical error that occurred in testing in section 4.4. The final recommendation for improvement is to incorporate more accurate vehicle positioning hardware. This would greatly enhance the accuracy of vehicle localization relative to the building zone. In order to be implemented in a planetary environment, vehicle positioning should be improved.

The HARP described in this research could be used as a multipurpose device once its primary task is completed. Additionally, the vehicle could be used in applications on Earth. Such a system could significantly impact on the construction industry. Also, the system could be used to build structures in an environment that is unsafe for humans.

This concept stems from the Chernobyl disaster containment effort in which the entire site was sealed off by a permanent structure. With the proper design, a fleet of assembly vehicles could be deployed in a hazardous environment to build a containment structure around a dangerous site such as the Chernobyl Nuclear Reactor. Overall, this simulated vehicle is proposed to prepare structures for humans when it is unsafe or unavailable for them to do so by traditional means of labor. This, in turn, will lower the risk of human life by implementing robotic vehicles in hazardous environments in place of humans.

5.6 Summary

This research concluded that the concept of autonomously assembling structures via manageably-sized building blocks can be accomplished with a mobile robotic system. Each sub-system worked together to ultimately complete the goal of assembling a simulated structure 0.25 m high by 1 m long. Overall, the vision system positively impacted autonomy by providing a means of locating where to pick and place building material. In addition, the robotic arm provided an accurate and reliable mode of manipulating the building material. Furthermore, the mecanum-drive vehicle chassis worked well to move the system hardware in omnidirectional, vector-like strafing to the target location. To quantify the implementation status of the HARP, NASA's Technology Readiness Level (TRL) was used [21]; as the HARP has been successfully tested in a laboratory environment, it is concluded that the HARP is currently at TRL 4. The concept of autonomously assembling preformed modular blocks into a structure was proven feasible using the habitat assembly prototype and should be continued in future research.

REFERENCES

[1] B. Kading and J. Straub, “Utilizing in-situ resources and 3D printing structures for a manned Mars mission,” *Acta Astronaut.*, vol. 107, pp. 317–326, Feb. 2015.

[2] “NASA’S 3D-Printed Habitat Challenge a NASA Centennial Challenge.” National Aeronautics and Space Administration, 2015.

[3] A. Rajan, V. Akre, N. Nassiri, A. H. AlAli, and Z. B. Sabt, “3D Printing of Buildings in UAE: Success and Failure factors,” in *2018 Fifth HCT Information Technology Trends (ITT),* Dubai, United Arab Emirates, 2018, pp. 368–372.

[4] X. Zhang *et al.*, “Large-scale 3D printing by a team of mobile robots,” *Autom. Constr.*, vol. 95, pp. 98–106, Nov. 2018.

[5] R. Bischoff and V. Graefe, “Machine Vision for Intelligent Robots,” p. 10, 1998.

[6] “Reflections on the Origins of Architectural Science and a Glimpse to its future,” *Archit. Sci. Rev.*, vol. 47, no. 1, pp. 1–8, Mar. 2004.

[7] J. Salih, S. Yaacob, and R. Mohd, “Omni-Directional Mobile Robot with Mecanum Wheel,” *1st Int. Workshop Artif. Life Robot.*, pp. 101–106.

[8] O. Diegel, A. Badve, G. Bright, J. Potgieter, and S. Tlale, “Improved Mecanum Wheel Design for Omni-directional Robots,” p. 5.

[9] “Technical Specifications VP-5243/ VP-6242.” [Online]. Available: https://www.densorobotics-europe.com/fileadmin/Products/VP-5243_and_VP-6242_technical_Data_Sheet/VP-5243_and_VP-6242_technical_Data_Sheet.pdf. [Accessed: 06-Dec-2019].

[10] “VP-5243/6242 | 5-and 6-Axis Robots | products | industrial robots | DENSO WAVE,” *denso-wave.com*. [Online]. Available: https://www.denso-wave.com/en/robot/product/five-six/vp.html. [Accessed: 03-Jun-2019].

[11] A. Hornberg, *Handbook of Machine and Computer Vision: The Guide for Developers and Users*. Weinheim, Germany: Wiley-VCH Verlag GmbH & Co. KGaA, 2017.

[12] “Direct Ring Light,” *Vital Vision Technology Pte Ltd*. .

[13] S. T. Padgett and A. F. Browne, “Vector-based robot obstacle avoidance using LIDAR and mecanum drive,” in *SoutheastCon 2017*, Concord, NC, USA, 2017, pp. 1–5.

[14] D. Vutetakis, “Replacement of Human Operator with Vision-Guided Robot for Electronic Component Pick-and-Place Work Cell,” University of North Carolina at Charlotte, Charlotte, 2016.

[15] K.-S. Kwon and S. Ready, “Practical Guide to Machine Vision Software,” p. 287.

[16] C. Kanan and G. W. Cottrell, “Color-to-Grayscale: Does the Method Matter in Image Recognition?,” *PLoS ONE*, vol. 7, no. 1, p. e29740, Jan. 2012.

[17] “Edge Detection,” in *Practical Image and Video Processing Using MATLAB®*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2011, pp. 335–363.

[18] W. Burger and M. Burge, *Principles of digital image processing: core algorithms*. London: Springer, 2009.

[19] R. Brunelli, *Template Matching Techniques in Computer Vision*. Chichester, UK: John Wiley & Sons, Ltd, 2009.

[20] J. Caja, E. Gómez, P. Maresca, and M. Berzal, “Development of a Calibration Model for Optical Measuring Machines,” *Procedia Eng.*, vol. 63, pp. 225–233, 2013.

[21] T. Mai, “Technology Readiness Level,” *NASA*, 06-May-2015. [Online]. Available: http://www.nasa.gov/directorates/heo/scan/engineering/technology/txt_accordion1.html. [Accessed: 14-Nov-2019].