Conformal Additive Manufacturing and Cooperative Robotic Repair and Diagnosis

by

Joshua David Davis

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

August, 2018

Abstract

In the past several years exponential growth has occurred in many industries, including additive manufacturing (AM) and robotics, enabling fascinating new technologies and capabilities. As these technologies mature, the need for higher-level abilities becomes more apparent. For instance, even with current, commercial state-of-the-art technology in AM it is impossible to deposit material onto a nonplanar surface. This limitation prevents the ability to fully encase objects for packaging, to create objects with hollow features or voids, and even to retrofit or repair preexisting items. These limitations can be addressed by the introduction of a conformal AM (CAM) process or more concretely the process in which material is deposited normal to the surface of an object as opposed to solely planar layers.

Therefore, one of the main contributions of this work is the development of two novel methods to generate layers from an initial object to a desired object for use in two- and three-dimensional CAM processes. The first method is based on variable offset curves and subject to mild convexity conditions for both the initial and desired

object. The second method reparametrizes solutions to Laplace's equation and does not suffer from these limitations. A third method is then presented that alters solutions from the previous methods to incorporate hollow features or voids into the layer generation process. Although these hollow features must obey mild convexity conditions, the location and number of said features is not limited. Examples of all three layering methods are provided in both two- and three-dimensions. Interestingly, these same methods can also be applied to determine the collision-free configuration space in certain robot motion planning applications. However, ultimately, the most compelling application may be in the repair of damaged items. Given an accurate model of a damaged item, these techniques, in conjunction with fused deposition modeling devices embedded on robotic arms, can be leveraged to restore a damaged item to its original condition.

In a separate but similar vein, although robotic systems are becoming more capable each day, their designs still lack almost any semblance of a repair mechanism. This issue is increasingly important in situations where robotic systems are deployed to isolated or even hostile environments as human intervention is limited or impossible. The second half of this work focuses on solving this issue by introducing the Hexagonal Distributed Modular Robot (HexDMR) System which is capable of autonomous team repair and diagnosis. In particular, agents of the HexDMR system are composed of heterogeneous modules with different capabilities that may be re-

placed when damaged. The remainder of this work discusses the design of each of these modules in detail. Additionally, all possible non-isomorphic functional representations of a single agent are enumerated and a case study is provided to compare the performance between two possible iterations. Then, the repair procedures for an agent in the system are outlined and verified through experiments. Finally, a two-step diagnosis procedure based on both qualitative and quantitative measures is introduced. The particle filter based quantitative portion of this procedure is verified through simulation for two separate robot configurations, while the entire procedure is validated through experiments.

Primary Reader: Professor Gregory S. Chirikjian, The Johns Hopkins University

Secondary Readers:

Professor Marin Kobilarov, The Johns Hopkins University

Professor Noah Cowan, The Johns Hopkins University

Professor Michael D. M. Kutzer, The United States Naval Academy

Acknowledgments

Several years ago, I had a dream of working within the robotics industry. To begin that journey, I quit my full-time government job and enrolled in Johns Hopkins under the tutelage of Professor Gregory S. Chirikjian. Thanks to the wonderful opportunities at Johns Hopkins, over the past several years I have grown not only as student but also as an academic, a mentor, and a roboticist. I have gained a tremendous amount of knowledge and forged lasting relationships both personal and professional. For all of these fulfilling experiences, I am eternally grateful.

I must especially thank Professor Gregory S. Chirikjian for his guidance, knowledge, and flexibility over the last several years. He has a remarkable talent in applying concepts from robotics to seemingly unrelated fields and vice versa. His dedication to research and moreover his ability to distill complicated concepts into straightforward and unassuming language is truly inspiring. Professor Chirikjian pushed me to become better in every aspect of my life not just academia. I am thankful for all of his help; however, I am most thankful for his patience and flexibility. Two years

ago, Professor Chirikjian allowed me to pursue a summer internship with READY Robotics. He then encouraged me to take a full-time position as a Robotics Engineer while finishing my dissertation remotely. At the beginning of this year, I truly fulfilled my dream and became the Vice President of Platform at READY.

In addition to Professor Chirikjian, I would like to thank the other members of my dissertation committee, Professor Kobilarov, Professor Cowan, and Professor Kutzer, for their insights into this work as well as their dedication to teaching and the robotics community. I also want to thank my collaborators, specifically M. Kendal Ackerman, Qianli Ma, Kristopher Reynolds, Alexis Cheng, Jin Seob Kim, Yuttana Itsarachaiyot, Colin Lea, Haiyuan Li, Yunus Semivli, Baxter Eldridge, and Jack Holmes, for their thoughtful insight, hard work, and friendship. Additionally, I would like to thank the incredible support staff (Mike Bernard, Kevin Adams, and Alison Morrow) at John Hopkins that was always there to lend a helping hand or answer any questions. Furthermore, I need to thank the Metropolitan Washington Chapter of the ARCS Foundation for their financial support during several years of my dissertation as well as many of my coworkers at READY Robotics, especially Kel Guerin and Ben Gibbs, for their continued support.

Finally, I would like to thank my friends and family. The unconditional love and continual encouragement I have received from my parents and brother have kept me motivated and focused while I completed my dissertation. I also wanted to thank

ACKNOWLEDGMENTS

my newly extended family who always seems to be interested in my work no matter how much I ramble on. Most importantly, I want to thank the love of my life, my wife, Amanda. Through all of the sacrifices, the delays, and life-changing decisions you have been my biggest proponent. You have supported and encouraged me and sometimes you have even forced me to lock myself in a room to work. Thank you for it all.

Dedication

To my wife and family who have provided unending support and encouragement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation is divided into two main topic areas. The first, Algorithms for Conformal Additive Manufacturing, presents two methods for generating layers conformally about an object's boundary in both two and three dimensions. One promising application of these methods is in repair; that is, material can be deposited directly onto the surface of a preexisting item to restore it to its original state. The second topic, Cooperative Team Repair and Diagnosis in the Hexagonal Distributed Modular Robot (HexDMR) System, follows from investigating a similar application in robotics. Mainly, what are the necessary and sufficient design constraints, as well as algorithms, required to construct an autonomous robotic system capable of conducting team repair and diagnosis? Specifically, this work introduces the first-generation HexDMR system and then discusses two successive design iterations aimed at increasing au-

tonomy. Finally, algorithms for autonomously conducting team repair and diagnosis in the third-generation system are presented and experimentally validated. Further details into the specifics of these topics are encompassed in the following sections.

1.1 Algorithms for Conformal Additive Manufacturing

Until only recently, subtractive manufacturing, or the process of iteratively removing material from a piece of stock to create a new part, has been the predominant method used in prototyping and many other industrial manufacturing processes. However, with the advent and growth of additive manufacturing (AM), or the process of iteratively depositing material in successive layers to create a new part, this trend has begun to change.

Chapter 2 begins with a brief history of AM from its birth in several different research labs to its eventual commercialization and diversification while highlighting many of the use cases where AM is highly desirable. Conversely, in its current state, AM has several limitations including the need for sacrificial support material, the inability to create enclosed hollow features, and the inability to print onto non-planar surfaces. These limitations are addressed by the introduction of conformal additive manufacturing (CAM) or, in other words, the process of iteratively deposit-

ing material normal to preexisting surfaces. Several researchers have presented basic implementations of CAM processes, but none have developed methods for conformal layer generation onto arbitrary objects.

Chapter 3 begins by discussing desired properties of layers in CAM and then presents two separate methods for layer generation in two dimensions. The first method utilizes variable offset curves (VOC) to generate layers normal to the boundary of an initial object to generate a user-specified desired object. This method is limited to mild convexity conditions for both the initial and desired object. The second method, based on reparametrized solutions to Laplace's equation, does not suffer from these deficiencies. Graphical examples of the layers generated for each method are provided and compared. In addition, the orthogonality of the reparametrized solutions to Laplace's equation is examined.

Chapter 4 extends the work presented in Chapter 3 to three dimensions. Additionally, a method is presented that addresses one of the deficiencies in typical AM processes, the inability to create enclosed hollow features or voids. Given an initial object and a desired object with a set of hollow features that obey mild convexity conditions, this method first generates layers using one of the two previously described methods without the hollow features. Then, areas of affect are defined for each hollow feature and the previously generated layers are altered to accommodate the features. Furthermore, there are no limits to the location of the hollow features

(i.e., the features can be arbitrarily close together or even be adjacent to the initial or unadulterated desired object). Examples of this process in both two and three dimensions is then provided. Next, a theoretical physical implementation of a CAM system is described that utilizes a single six degree-of-freedom robotic manipulator (although only a minimum of five degrees-of-freedom are required) in conjunction with a fixturing system. Finally, although these methods were originally motivated by CAM, they are applicable to different topic areas as well. Specifically, this chapter concludes by investigating how these methods can be applied to robotic motion planning by identifying the entire collision-free configuration space for a mobile robot.

CAM and these new layering techniques enable a whole new realm of manufacturing processes that were previously unachievable using only single parts. In addition to prototyping highly complex mechanical parts, it is now possible to alter or augment the functionality of a part by printing directly onto its preexisting surfaces. CAM can also fully encase existing items for added protection during transport or even couple a mechanical interface to an electrical subsystem such as a computer mouse. Perhaps even more importantly, CAM may be used to repair preexisting items. Consider the trivial example where an appendage is broken off of an action figure and lost. As long as a computer model exists of the original item, CAM may be used to print onto the surface of the item and exactly replicate the lost piece (up to the resolution of the printer).

Exploring this idea of repair in manufacturing sparked another discussion in a separate industry where repair is all but absent. Although robotic systems become more capable each day, their designs still lack almost any semblance of a repair mechanism. It is along these lines that the ideation and development of the HexDMR system began.

1.2 Cooperative Team Repair and Diagnosis in the Hexagonal Distributed Modular Robotic System

Repair in robotic systems can more generally be thought of in the context of fault tolerance. Chapter 5 introduces this concept as well as the primary fault tolerant mechanism used by individual systems. That is, after a robot has identified an internal fault through some sort of diagnosis, an adaption algorithm is employed. For example, if a single wheel in a multi-wheeled mobile robot becomes damaged, the kinematics of the robot will adapt to account for the loss in mobility to maintain an expected locomotion behavior. Although adaptation is the primary mode of repair in single robot systems, the same is not true for cooperative multi-agent (or multi-robot) systems. For large numbers of extremely simplistic agents, repair is conducted

by simply discarding damaged components or robots; however, this method is not effective as it may lead to loss of capabilities over time. Instead, this work looks to the design of heterogeneous modular agents that can be repaired by team members to prolong operational lifetimes and maintain functionality.

Chapter 6 begins by examining common design features across the three generations of HexDMR systems. Then design improvements to the second-generation, HexDMR II, are presented that result in improved communication robustness and partially inform the design of the third-generation system. Next, the third-generation system, HexDMR III, is introduced and its design, including the design of each individual module, is discussed in detail. Unlike its predecessors, HexDMR III can hold up to twelve individual modules which allows storage of spare modules for repair at the cost of increased complexity. This change necessitated the enumeration of all possible functional non-isomorphic configurations (i.e., each functionally unique physical representation of a single agent in the system) to better understand the configuration space of the system. Finally, a case study is presented that compares the performance between a three- and six-wheeled agent.

Chapter 7 summarizes an attempt to conduct autonomous repair in the HexDMR II system and then outlines the design deficiencies that prohibited automated repair. Next, a repair procedure and a module identification protocol for the HexDMR III system are presented. The chapter concludes with separate experiments demonstrat-

ing the autonomous insertion and extraction procedures for a repair.

Once the repair process was validated, work continued to design a diagnosis procedure in Chapter 8. The chapter begins by introducing a new camera module that is capable of tracking fiducial markers, or augmented reality tags. When these tags are installed on a second robot, the camera module provides poses or rigid body transformations from the camera frame to the tag in the camera frame's coordinate system as elements of the three-dimensional Special Euclidean group. This new source of data enabled a holistic diagnosis procedure to be created. First, the failure modes of modules in the HexDMR III were analyzed. The results from the failure modes and effect analysis pointed to a two-step diagnosis procedure that relied on both qualitative and quantitative measures. In particular, the qualitative method is used to assess the health of camera, control, elevator, manipulator, and power modules on an agent while the quantitative method utilizes an unscented particle filter to diagnosis faults in any of the active drive modules irrespective of their configuration. The quantitative procedure was first verified in simulation for both three- and six-wheeled agents and then validated through experimentation with a three-wheeled agent.

1.3 Dissemination

The work included in this dissertation has been disseminated through several different avenues. In particular, the layer generation methods developed in Chapters 3 and 4 were first published in [1]. This work was then extended to include hollow features and subsequently published in [2]. Concurrently, a patent [3] was pursued with the United States Patent and Trademark Office. Separately, the work directed towards designing robotic systems with diagnosis and repair capabilities was also published. Specifically, the design of the HexDMR III system was first published in [4] and then published again in [5] with the addition of several experiments that highlighted the differences between base configurations in the system. The general methodology for repair, as well as specific repair experiments in the HexDMR III system, was published in [6]. Finally, the work related to diagnosis in robotic system has not yet been published; however, plans exist to do so in the future.

Part I

Conformal Additive Manufacturing

Chapter 2

A Review of Additive Manufacturing

Since the advent of the industrial revolution, traditional manufacturing has mostly relied on subtractive processes (i.e. the removal of excess material from some sort of stock) to form parts. In fact, subtractive processes are often used when creating master molds for processes such as forging and injection molding to produce products on a massive scale. Unfortunately, these subtractive methods have several drawbacks due to inherent constraints of the process. First, the process begins with stock of a suitable material (e.g. wood, plastic, metal, etc.) that is larger than the final product. This stock is then machined generating scrap which must be discarded or recycled. Second, parts that require complex internal features cannot be machined from a single

piece due to constraints on tooling. If these features are required, additional cost and effort must be expended to create multiple parts that must be fastened together to produce the desired product. This separation can lead to failures in parts that are exposed to high pressures or harsh environments. Finally, these parts often require costly, experienced machinists to meet high tolerances and short lead times.

In an effort to address many of these problems, research on additive manufacturing (AM) methods began in the early 1960s [7]. AM processes, in contrast to their subtractive counterparts, leverage a variety of processes to bind materials, creating solid structures. By the mid 1980s, several individuals began to commercialize early prototype systems [8, 9, 10, 11, 12]. Over the past several years, AM has exploded in popularity and usage as enterprising companies and hobbyists have released cost-effective personal solutions that have enabled AM at home. These additive solutions use three-dimensional (3D) models generated in software to autonomously deposit material in planar layers onto a build surface to generate exact facsimiles. As these objects are built from the ground up there are almost no constraints on the complexity of the interior of the part. Additionally, cost is typically related to the amount of build material expended as opposed to the complexity of the part or the required tolerances. Although these advances are impressive, the ability to fully enclose parts for packaging or to directly print onto nonplanar surfaces is still lacking. It is this research area, commonly referred to as conformal additive manufacturing (CAM),

that is still largely unexplored.

The remainder of this chapter is dedicated to providing proper context for the history of AM and the need for CAM. Section 2.1 details a brief history of AM including its origins, relevant patents related to commercialization, general descriptions of current AM processes, and a brief overview of its current utilization. Then the current limitations of AM are reviewed and the concept of CAM is fully introduced. Finally, current research thrusts in CAM are discussed.

2.1 A History of Additive Manufacturing Practices and Technology

AM is the process upon which a 3D object is constructed by depositing layer on top of layer of a specific (or multiple) material(s). Research into developing commercially realizable AM techniques or processes began over 60 years ago with an attempt by members of the Battelle Memorial Institute to create a solid object comprised of photopolymers hardened through the use of two laser beams with different wavelengths [7]. Then, in 1974, the Formigraphic Engine Co. demonstrated a similar commercial photochemical process [7]. In the following years, several patents were issued to different companies pursuing similar approaches; however, the first recognized real-world example of AM is attributed to Hideo Kodama who outlined

a process utilizing stereolithography to create 3D objects in [8]. Building on this approach, Charles W. Hull developed a stereolithography apparatus (SLA) that used computer inputs to selectively solidify material to build 3D objects. He patented his invention in 1986 [9] and launched the first commercial AM company, 3D Systems, soon after.

Over the next 15 years, the AM industry grew out of its infancy but was still not well known outside of industry. Commercial partners utilized SLA technology to rapidly iterate over product designs leading to the early adoption of the phrase "rapid prototyping." Additionally, during this time, researchers and commercial entities began developing and patenting other AM methods such as laminated objecting manufacturing or sheet lamination in 1988 [10], powder bed fusion or a version of selective laser sintering in 1989 [11], and binder jetting in 1993 [12]. Several other methods were developed, but perhaps the most influential and well known is fused deposition modeling (FDM) introduced by Scott Crump (a founder of Stratasys) in 1988. It is this method that most laypersons refer to today when they reference 3D printing. A summary of each of the current commercial AM technologies by process [13] is included in Table 2.1.

Around the year 2000, most AM technologies were reaching maturity and slowly becoming more available to the public. Over time, as more materials were introduced and patents began to expire, AM slowly became more accessible. With a better han-

Table 2.1: Summary of commercial AM technologies

Process	Description	Material(s)	Manufacturer(s)
Binder Jetting	A liquid bonding agent is selectively deposited to join powder materials	Polymers, Sand, Glass, Metals	3D Systems (3DS), ExOne
Direct Energy Deposition	Focused thermal energy is used to fuse materials by melting as they are deposited	Metals	Optomec, POM
Material Extrusion	Material is selectively dispensed through a nozzle or orifice	Polymers	Stratasys, Bits from Bytes, MakerBot, RepRap
Material Jetting	Droplets of build material are selectively deposited	Polymers, Waxes	Objet, 3DS
Powder Bed Fusion	Regions of material are selectively fused in a powder bed using sintering or melting	Metals, Polymers	3DS, EOS, Arcam
Sheet Lamination	Sheets of material are bonded to form an object	Paper, Metals	Fabrisonic, Mcor, Cubic Technologies
Vat Photopolymerization	Liquid photopolymer in a vat is selectively cured by light-activated polymer	Photopolymers	3DS, Envisiontec, OS-RC, Kudo3D, Formlabs

dle on material properties and availability, companies were able to rapidly transition rough prototypes into usable production models. Additionally, cost was no longer related to part complexity but rather to the amount of material consumed. Moreover, a direct correlation between AM fabrication-time and part complexity does not necessarily exist. In many cases, complex pieces which were previously composed of several different parts could now be seamlessly generated as a single part in a much shorter time frame. AM fabrication also offers relaxed design rules and simple part-

by-part customization. These properties coupled with the introduction of personal, FDM-based 3D printers at reasonable price points led to real exponential growth in the mid 2000s. Overnight, individuals with no prior design or manufacturing experience were able to produce physical hardware almost immediately, while experienced designers could create complex parts tailored for specific applications.

These advantages have transformed AM into a multi-billion dollar industry that only continues to grow and has led to rapid adoption by both the public and private sectors. In 2013, the Chief Naval Officer's Rapid Innovation Cell began the Print the Fleet (PTF) project aimed at leveraging AM technology on Naval vessels. The underlying motivation for the project was to ultimately enable rapid adaptation in the "changing landscape of warfare" [14]. Recent PTF initiatives include the evaluation of AM technology on an unarmed Joint High Speed Vessel and the sponsoring of a permanent installation of AM technology onto the USS Essex [14, 15]. In a parallel effort, NASA evaluated AM technology in zero gravity environments with a demonstration system recently deployed on the International Space Station [16]. The goal of this experiment was to demonstrate that a 3D printer functions nominally in a zero-gravity environment. The ultimate goal for projects like these is to enable rapid, on-site repair, replacement, and adaptation of mechanical (and potentially electrical) hardware.

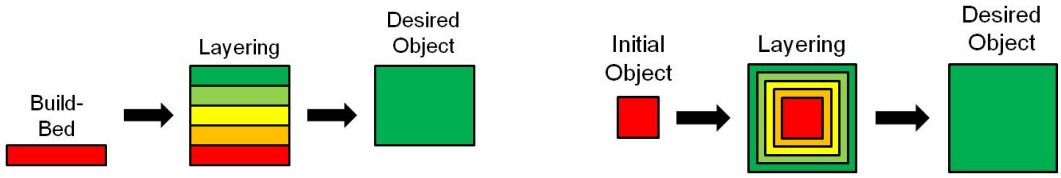
As adoption of AM increases, the limitations of the current technology will become

more distinct. Recent research in AM processes is focused on addressing limitations in properties of printed materials [17, 18, 19, 20]; however, existing methods have yet to be exploited to their full potential. While current AM enables tremendous innovation in part design, designers are still required to follow classical packaging and/or assembly rules. Furthermore, methods to fully encase objects with multilayer, thick features are still undeveloped. Examples of deficiencies and desired capabilities include retrofits (e.g. USS Whidbey Island prototype adapter brackets [14]), packaging (e.g. electronics and sensors), and repair. Many of these limitations can be overcome by the introduction of a CAM process.

2.2 Current Conformal Additive Manufacturing Technologies and Limitations

CAM is the process in which material is deposited in layers conformally about an object's natural boundary. In current AM, parts are manufactured by iteratively adding planar layers of material. Layers are defined by thin cross-sections of a part, and derived from an exported CAD (computer-aided design) model [21]. Commercial AM techniques generally use a "build-bed" that serves as the flat substrate for part

fabrication. The CAD model is imported into an AM software package, and positioned relative to the build-bed. Layers are then defined by equally spaced planar slices of the CAD model, parallel to the build-bed. This is effective for a wide variety of part geometries. Depending on the AM process, issues may arise with overhanging features, but this limitation is effectively solved by adding sacrificial support layers that are removed following the completion of the AM process [21]. Because of this required material, no AM technology is currently capable of manufacturing a closed, fully hollow feature. A graphical example of the differences between "traditional" planar printing and conformal printing is provided in Fig. 2.1, while Fig. 2.2 showcases a concrete theoretical example of CAM where material is iteratively deposited around the printed circuit board in Fig. 2.2a to create the mouse pictured in Fig. 2.2c.

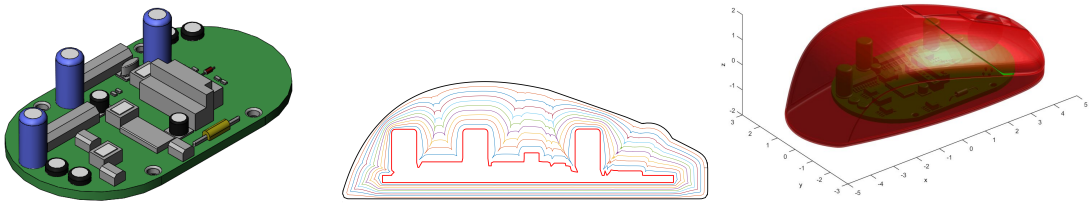


(a) A traditionally printed object

(b) A conformally printed object

Figure 2.1: Comparison of cross-sectional views for a printed object

Several researchers have developed modest implementations of conformal printing. For instance, Daniela Radtke and Uwe D. Zeitner of the Fraunhofer Institute for Applied Optics and Precision Engineering [22] and Yongjun Xie along with several collaborators from various Chinese institutions [23] separately developed direct-write laser-lithography methods to create diffractive optical elements. In these methods, a



(a) The initial printed circuit board

(b) Layers generated conformally about the printed circuit board

(c) A mouse generated through conformal additive manufacturing

Figure 2.2: An example of constructing a mouse from a printed circuit board through conformal additive manufacturing

laser is held stationary by a linear stage while a concave or convex lens is rotated along one of its major axes to write onto the surface of the lens. This setup enables uniform writing onto nonplanar surfaces while ensuring that the laser is always perpendicular to the surface of the lens. In a separate setting, another direct-write method utilizing aerosol jet technology was employed to print conductive metals onto differing surfaces [24]. This technology uses up to six degrees-of-freedom (DOF) to orient a stream of nanoparticles onto the surface of an object. In contrast to the previous methods, the object is held stationary while the jet is moved. Additionally, the jet is not required to be perpendicular to the work surface of the object. Although this method is effective at printing small electronic circuits, it is ineffective for larger jobs such as fully encasing an object. Furthermore, Paulsen et al. failed to provide a generalized method for generating layers conformally around an object.

Several other additive methods were developed to fabricate antennas and electronics onto/into mechanical components [25, 26]. But, in general, these AM techniques

only demonstrated the deposition of a single layer of material [25, 26]. Specifically, in [25] a highly-wetted conductive ink is deposited onto a spherical glass lens in a single layer. In this process, the lens is held stationary and a three DOF linear stage is used to print directly onto the surface. Due to the material properties of the ink, the print head is kept entirely vertical. These constraints render this method impracticable for printing onto more complex surfaces. This issue also arises for the method described in [26]; however, Vatani et al. employ a more intelligent method during printing to ensure consistent layer size. Similar to [25], printing is controlled by a 3 DOF linear stage with a fixed nonplanar object, but, in this case, care is taken to ensure that the vertical height from the print head is maintained while travelling either up or down a slanted surface.

Perhaps the most recent and complete work towards developing a truly conformal AM method was produced by Singamneni et al. [27, 28]. In their work, they propose a strategy for printing curved segments using currently existing FDM technology at the cost of requiring a large amount of sacrificial support material. First, sacrificial material is deposited according to traditional AM methods to generate a nonplanar scaffold. Then, material is deposited on top of the scaffold by smoothly adjusting the vertical offset of the print head while moving along the plane perpendicular to the print head. This motion generates continuous curved layers. Finally, the desired object is retrieved by removing the sacrificial support material much like current

processes. Although this method provides an algorithm for generating these layers from CAD models, it suffers from one additional drawback; curvatures over a certain threshold are unachievable.

In an effort to address the limitations in current CAM methods Chapters 3 and 4 develop two separate algorithms that construct multiple enveloping, conformal layers around an initial object to generate a desired final object, irrespective of the AM process being employed. Each method requires two inputs (the boundary of the initial object, and the boundary of the desired object) described in a common reference frame. The first method utilizes variable offset curves to generate layers, and is limited to pairs of initial and final desired objects that satisfy certain mild compatibility conditions. The second method leverages solutions to Laplace's equation, and is applicable to all pairs of geometric objects with differentiable boundaries. Next, a process is developed that alters the layers generated by these methods to incorporate 2D and 3D hollow features (or voids). For completeness, several 2D and 3D applications of each method (with and without voids) are presented. Results demonstrate successful layering for each method, and advantages and limitations of the presented methods are discussed. Finally, a case scenario is presented where these methods can be applied to robotic motion planning.

Ultimately with these new layering concepts, in conjunction with registration and manipulation methods commonly used in robotics, CAM can be extended to incor-

porate multilayer, thick features. These processes can be used produce seamless packaging, repair damaged hardware, incorporate heterogeneous materials into products to provide extra strength and/or durability, and provide retrofitting capabilities (for example, adding a handle or flange) to an existing piece of hardware ushering in a new wave of AM technologies.

2.3 Contributions

The main contributions of this work toward CAM are as follows.

- The definition of a layer in conformal additive manufacturing.
- The development of two separate, novel methods to generate layers conformally about an initial object to created a desired object in both two- and three-dimensions. Respectively, these methods are labeled the Variable Offset Curve Method and the Laplace's Equation Method.
- An additional method that creates local deformations to previous layering solutions to incorporate voids or hollow features into the volume between the initial and desired objects.
- Illustrative examples of the aforementioned methods in both two- and three-dimensions.

- The extension of these methods to robotic motion planning to identify the entire collision-free configuration space.

Chapter 3

Conformal Additive Manufacturing in $\mathbb{R}^2$

In Chapter 2 the current state of additive manufacturing (AM) technologies was reviewed and the concept of conformal additive manufacturing (CAM) was introduced. As noted in the previous chapter, the goal of CAM is to deposit material in a conformal manner around a given initial object to grow it into a desired object. In the planar case, CAM can be thought of as a new manner in which to specify layers for traditional additive manufacturing processes. However, in the more general three-dimensional (3D) case, CAM is an entirely new approach requiring completely different equipment and techniques. Currently, several rudimentary CAM methods have been demonstrated [25, 27, 28, 26, 23, 24, 22, 21]; however, these methods failed

to provide a general algorithm to generate conformal layers for arbitrary objects. Furthermore, these methods lacked an approach to incorporate hollow features within the generated layers or build volume.

In Chapters 3 and 4 of this dissertation, two novel methods are presented that not only generate layers between an initial given object and a desired object, but also specify the conditions and methodology to incorporate an arbitrary number of hollow features within the desired object. Chapter 3 will focus on the formulation of both methods in two-dimensions (2D) in addition to providing a brief mathematical review on topics such as convexity, parametric cubic splines, and variable offset curves which are used extensively in the formulation of each method. In Chapter 4, each method is extended to 3D and augmented with the ability to incorporate hollow features. Throughout both chapters, the initial and desired object can be thought of as sets which includes both the interior of the object as well as its boundary. Additionally, depending on the context, curve and surface may be used interchangeably to refer to the boundary of the initial or desired object.

3.1 Mathematical Overview

Chapters 3 and 4 will rely heavily on the concepts of convexity, variable offset curves (or more generally variable offset features), and parametric cubic splines. This

section is meant to provide a brief overview on each of these topics.

3.1.1 Convexity

In mathematics, convexity arises in both the context of convex functions and sets. This work is mainly concerned with convex sets. A set $\mathbb{S}$ is convex if and only if

$$\alpha x + (1-\alpha)y \in \mathbb{S} \quad \forall \alpha \in [0,1] \tag{3.1}$$

for all $(x, y) \in \mathbb{S}$ or, in other words, the line segment connecting two points x and y is contained entirely in $\mathbb{S}$ for all points in the set.

3.1.2 Variable Offset Features

Given a closed parametric n-dimensional hyper-surface $\mathbf{x}(s)$, it is possible to generate a $(n-1)$-dimensional offset feature $\mathbf{o}(s; r)$ that is entirely normal to the surface of the form

$$\mathbf{o}(s; r) = \mathbf{x}(s) + r\mathbf{n}(s) \tag{3.2}$$

where r is a fixed scalar offset with value less than the maximal radius of curvature to prevent self-intersections of the offset feature and $\mathbf{n}(s)$ is the unit normal to the surface at s. For offset features, each point in the feature is the same distance away from

the initial surface along the normal. In $\mathbb{R}^2$ offset curves have been used extensively in milling applications to plan cutting paths and to estimate material loss. More recently, these offset features have been extended to variable offset features, in which the offset along the normal at each point on the hyper-surface can be arbitrarily set. Given a closed parametric hyper-surface $\mathbf{x}(s)$ a variable offset feature is constructed as

$$\mathbf{o}(s; r) = \mathbf{x}(s) + r(s)\mathbf{n}(s) \tag{3.3}$$

where now $r(s)$ is free.

In Chapters 3 and 4, variable offsets features or more specifically in $\mathbb{R}^2$ and $\mathbb{R}^3$ variable offset curves and surfaces, respectively, will be drawn upon extensively for generating layers in one of the methods due to the fact that CAM in mainly conducted in two- and three-dimensional Euclidean space. Both offset curves, and their more general counterparts variable offset curves, are well-established in the literature with several papers providing in-depth analysis of their analytical and algebraic properties [29, 30, 31, 32].

3.1.3 Parametric Cubic Splines

Many AM processes rely on objects modeled in computer-aided design software to generate layers for the manufacturing process. In some cases, these models only

provide a set of points or features that approximate the actual surfaces of the object to be created. Therefore, the methods in this chapter utilize parametric cubic splines to interpolate between these data sets in order to calculate normals and tangents to the curves or surfaces. Given a set of points approximating a curve in $\mathbb{R}^2$, a parametric cubic spline is formed by constructing piecewise parametric cubic polynomials. A single cubic polynomial is represented by

$$\mathbf{x}(s) = \mathbf{a}s^3 + \mathbf{b}s^2 + \mathbf{c}s + \mathbf{d} \tag{3.4}$$

where $\mathbf{x}(s) \in \mathbb{R}^2$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d} \in \mathbb{R}^2$ are coefficients that uniquely define the cubic polynomial, and $s \in [k, k+1)$ represents the interval on which the kth cubic polynomial is valid. The corresponding parametric cubic spline is then constructed from the cubic polynomials as

$$\mathbf{X}(s) = \begin{cases} \mathbf{x}_1(s) & s \in [0,1) \\ \mathbf{x}_2(s) & s \in [1,2) \\ \vdots & \\ \mathbf{x}_k(s) & s \in [k-1,k) \end{cases} \tag{3.5}$$

3.2 Desired Properties for Conformally Printed Layers

In contrast to a majority of current AM methods that deposit material in a fixed plane, CAM methods deposit layers along closed curves for 2D objects and along surfaces for 3D objects. This layering process enables the ability to fully encase objects for protection, to include hollow features in the build volume, and to reduce or eliminate the need for sacrificial support material as well as a myriad of other benefits. Since this layering process is drastically different than current AM methods, it lends itself to a new definition for layers in conformal AM processes.

A layer is defined as a bijective mapping that acts along the vector normal to the initial boundary and transforms points on the initial boundary to points on a separate unique boundary. Note that the desired boundary is also considered a layer and therefore a bijective mapping also exists between the initial and desired boundary. Intuitively, this mapping ensures that each point on the initial boundary maps uniquely to a point on each layer, and that each layer is also a closed boundary. The bijective mapping guarantees not only that layers will not intersect one another but that each individual layer does not self-intersect. Physically, this mapping ensures that material will never be deposited at the same point twice. Additionally, this method prevents arbitrary areas without material deposition (unless specified)

between subsequent layers provided proper layer thickness and continuity [33].

3.3 The Variable Offset Curve Method in $\mathbb{R}^2$

As defined in Section 3.2, layers are a bijective mapping between an initial curve $x_0(s)$ and a final or desired curve $x_f(s)$. One such method to generate these layers is to define variable offset curves (VOCs) relative to the initial curve. However, without additional assumptions, layers generated by VOCs are not necessarily guaranteed to be bijective.

3.3.1 Assumptions

The following assumptions will ensure that a bijective mapping exists between the initial and desired boundary as well as each subsequent layer.

Assumption 1: All layers, including the initial curve and the desired curve are closed and at least C^2 continuous.

That is, the function representing each curve is at least twice differentiable and the resulting function is continuous. More generally, a function $f(x)$ is said to

be continuous at a point x if

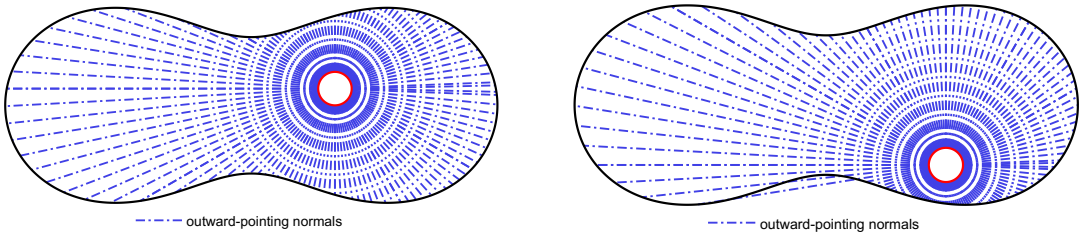
$$\lim_{x \to a} f(x) = f(a) \tag{3.6}$$

A curve is continuous if and only if every point in the curve is continuous. Furthermore, if the function representing the first derivative of the curve is also continuous at every point, then the curve is said to be continuously differentiable or C^1 continuous. The superscript n in C^n continues to increase for each additional derivative that yields a continuous function.

Assumption 2: The initial object is convex and fully contained inside of the curve representing boundary of the desired object.

Assumption 3: The boundary of the desired object is of a "compatible" nature with respect to the boundary of the initial object, meaning that each point on the boundary of the desired object intersects exactly one outward-pointing normal ray emanating from the initial object's boundary.

The first assumption prevents outward-pointing normal vectors from the boundary of the initial object from intersecting one another, while the second assumption ensures that every outward-pointing normal will intersect the boundary of the de-



(a) A compatible desired object **(b)** A non-compatible desired object

Figure 3.1: An example of the dependence of a compatible desired object on the position of the initial object

sired object at exactly one point. Moreover, taking the first and second assumption together, each point of intersection on the desired boundary is unique and the set of all points of intersection recover the boundary of the desired object. Finally, the last assumption ensures that the VOC method will fully reconstruct the boundary of the desired object for any given convex initial object (i.e. there will be no gaps on the boundary of the desired object that remove curvature, changes in convexity, or other features). Additionally, one may infer from the definition above that the compatibility of a desired object is highly dependent on the position and orientation of the initial object. Figure 3.1a highlights this linkage by providing two examples with the same initial and desired objects; in the first example the desired object is compatible and in the second it is not due to some outward-pointing normals intersecting the boundary of the desired object multiple times.

3.3.2 Formulation

The VOC method requires two inputs, the boundaries of the initial and desired objects, to generate conformal layers. Depending on the application or the industry, these boundaries may be specified in a variety of different manners; however, the VOC method requires that these boundaries are specified as two, C^2 functions. In practice, the curves that represent the boundary of each object are often approximated by a discrete number of points which may be supplied a priori or can be generated by sampling the boundary sufficiently well. Therefore, piecewise parametric cubic splines are used to generate closed curves γ_0 and γ_1, with $\gamma_0 \subsetneq \gamma_1$, from two ordered sets of points, $\mathbf{U} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\}$ and $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}$ that approximate the boundary of the initial and desired object.

As reviewed in Sec. 3.1.3, each parametric spline is described as a set of cubic polynomials of the form

$$\mathbf{X}(s) = \mathbf{a}s^3 + \mathbf{b}s^2 + \mathbf{c}s + \mathbf{d} \tag{3.7}$$

where $\mathbf{X}(s) = [x(s),\ y(s)]^T \in \mathbb{R}^2$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ are coefficients that uniquely describe each polynomial, and $s \in [k, k+1)$ represents the interval on which the kth polynomial is valid. For the remainder of this section, a single parametric cubic polynomial from each spline will be used to highlight the methodology. Furthermore, superscripts are appended to the polynomial coefficients (e.g. $\mathbf{a}^0$) to distinguish between

the splines representing γ_0 and γ_1.

Given one segment of these curves, tangent vectors are calculated by taking the derivative of the cubic polynomial with respect to the parametric variable s.

$$\mathbf{T}(s) = \frac{d\mathbf{X}}{ds} = 3\mathbf{a}s^2 + 2\mathbf{b}s + \mathbf{c} \tag{3.8}$$

Then, normal vectors are derived by appending a zero to the tangent vector and taking the cross product with the appropriate unit vector that completes a right-handed frame.

$$\mathbf{N} = \begin{bmatrix} T_1 \\ T_2 \\ 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} T_2 \\ -T_1 \\ 0 \end{bmatrix} \tag{3.9}$$

Remembering, from Assumption 2, that outward-pointing normal vectors projected from the boundary of convex objects do not intersect one another, unique parametric lines can be constructed that originate on the boundary of γ_0 and extend to γ_1. Each parametric line is of the form

$$\mathbf{X}(t) = (1-t)\,\mathbf{X}_i + t\mathbf{X}_f \tag{3.10}$$

where, again, $\mathbf{X} = [x,\, y]^T \in \mathbb{R}^2$, $t \in [0, 1)$ represents the interval on which the line is valid, $\mathbf{X}_i$ represents a point on γ_0, and $\mathbf{X}_f$ represents a point along the normal

projected from γ_0. To ensure that each parametric line is long enough to intersect γ_1, $\mathbf{X}_f$ is chosen such that

$$\mathbf{X}_f = \mathbf{X}_i + r\underline{\mathbf{N}} \tag{3.11}$$

where

$$r = \max_m \|\mathbf{v}_m - \mathbf{X}_c\| + \frac{1}{2}\left(\max_n \|\mathbf{u}_n - \mathbf{X}_c\| - \min_n \|\mathbf{u}_n - \mathbf{X}_c\|\right), \tag{3.12}$$

$\underline{\mathbf{N}}$ is the two-dimensional representation of N with the z-component removed, and $\mathbf{X}_c$ is the centroid of the region enclosed by γ_0.

With a suitable choice of r each parametric line is guaranteed to intersect γ_1. The point of intersection is determined by first equating the parametric line and the spline representing γ_1 and then solving for the parametric variables. Separating the point of intersection into its scalar components (where a subscript of 1 indicates the x-component and a subscript of 2 indicates the y-component), yields two equations in two independent variables

$$(X_{f1} - X_{i1})\, t + X_{i1} = a_1^1 s^3 + b_1^1 s^2 + c_1^1 s + d_1^1 \tag{3.13}$$

$$(X_{f2} - X_{i2})\, t + X_{i2} = a_2^1 s^3 + b_2^1 s^2 + c_2^1 s + d_2^1. \tag{3.14}$$

Solving for t in Eqn. (3.13)

$$t = \frac{a_1^1 s^3 + b_1^1 s^2 + c_1^1 s + d_1^1 - X_{i1}}{X_{f1} - X_{i1}} \tag{3.15}$$

and substituting t into Eqn. (3.14) results in the following cubic equation

$$\begin{aligned} 0 = \left(a_2^1 - \beta a_1^1\right) s^3 + \left(b_2^1 - \beta b_1^1\right) s^2 + \left(c_2^1 - \beta c_1^1\right) s \\ + \left(d_2^1 - \beta d_1^1\right) + (\beta X_{i1} - X_{i2}) \end{aligned} \tag{3.16}$$

where

$$\beta = \frac{X_{f2} - X_{i2}}{X_{f1} - X_{i1}} \tag{3.17}$$

The roots of Eqn. (3.16) correspond to the intersection of a cubic polynomial with a parametric line. In practice, the spline is comprised of $m - 1$ cubic polynomials and for a particular normal there are only two roots such that $s \in [0, 1)$. If t is further restricted such that $t \in [0, 1)$, then there is only one valid root and the intersection point, $\mathbf{X}$, can be obtained by substituting s into Eqn. (3.7) or t into Eqn. (3.10). Finally, the Euclidean distance between the point on the initial curve and the intersecting point on the desired curve is calculated.

This process is continued iteratively for each point in $\mathbf{U}$ and a single VOC is defined which is a bijective mapping of points on the initial curve to the desired curve.

Individual layers are generated by appropriate motion along normal vectors that originate on γ_0 and terminate on γ_1. For a given point t_{x_0} on the initial parametrized curve $\mathbf{x}_0(t)$ and a desired number of layers n_d each point t_{x_i} on a subsequent layer $\mathbf{x}_i(t)$ is defined as

$$\mathbf{x}_i(t_{x_i}) = \mathbf{x}_0(t_{x_0}) + i\frac{\|\mathbf{X}\left(\mathbf{x}_0(t_{x_0})\right) - \mathbf{x}_0(t_{x_0})\|}{n_d}\mathbf{n}(t_{x_0}) \tag{3.18}$$

where $i = \{1, 2, \ldots, n_d\}$ and $\mathbf{X}\left(\mathbf{x}_0(t_{x_0})\right)$ is the point of intersection between the normal line emanating from $\mathbf{x}_0(t_{x_0})$ and the desired curve. A layer is then defined as the set of all points for a particular i and, as desired, when $i = n_d$ the desired curve is recovered.

Interestingly, when defined in this manner, each layer is a VOC of the initial curve and the desired curve, but not a VOC of any of the intermediate layers. An exception occurs when the initial and desired curves form an annular region. This exception will be discussed further in the following chapter.

3.4 The Laplace's Equation Method

In this section a second method to generate layers conformally about an object's natural boundary in $\mathbb{R}^2$ is presented. In contrast to the VOC method presented in Section 3.3, the method introduced in this section does not impose restrictions on

the convexity of either the initial or desired object. For this method, layers are still defined as a bijective mapping between the initial boundary and each subsequent layer. However, layers are no longer VOCs of the initial boundary, but instead modified solutions to Laplace's equation existing between the equipotential boundaries of the initial and desired objects. Although solutions to Laplace's equation have many practical applications in physical systems such as electrostatics, fluid flow, and magnetostatics and even in the control of robotic systems [34, 35, 36, 37], they have not been applied to problems relating to AM processes.

3.4.1 Desirable Properties of Solutions to Laplace's Equation

Laplace's equation is a second-order partial differential equation (PDE) of the form

$$\nabla^2\varphi = \Delta\varphi = 0 \tag{3.19}$$

Any function, φ, that is at least twice continuously differentiable and satisfies Laplace's equation is called a harmonic function. Harmonic functions have several desirable properties, but two are of particular interest.

The first property is a corollary of the maximum principle, which states that if a function ψ is harmonic in a domain D and continuous in the closure of D, then

both the maximum and minimum values of the function in the closure of D are attained on the boundary [38]. Furthermore, one can show that a harmonic function, or solution to Laplace's equation, is completely determined by its boundary values. A direct consequence of this fact is that only two inputs (i.e. the initial and desired boundaries) are required to completely define and solve the problem of generating layers. Moreover, since the minimum and maximum values of a harmonic function must be attained on the boundary and the boundaries of the initial and desired objects can arbitrarily be assigned to have uniform, but different, potentials, then the solution can be completely constrained between the two boundaries. In addition, note that ψ can effectively be bound from above and below by choosing appropriate values for the boundaries of the initial and desired objects. If ψ if further assumed to be continuous throughout the domain, then there exists a continuum of closed equipotential boundaries between the boundaries of the initial and desired objects.

The second property defines the gradient at any point of an equipotential boundary as orthogonal to the boundary. Physically, the gradient of the scalar potential function results in a field, existing solely in the domain D, which is unique at every point. Given a point on the the boundary of the initial object and the field, field lines can be constructed that extend to the boundary of the desired object by integrating. More importantly, these field lines, originating from different points, do not intersect in the domain.

Proof. Assume that two arbitrary field lines originating from two different points on the boundary of the initial object intersect in the domain at some equipotential boundary. From the properties discussed in this section, it was affirmed that the gradient at a point on an equipotential boundary is always normal to the boundary. Therefore, after these two field lines intersect at an arbitrary equipotential boundary they will follow the same path until they terminate on the boundary of the desired object. Since the choice of the intersecting equipotential boundary was arbitrary, it must hold for all equipotential boundaries including the boundary of the initial object. Thus, the two field lines are the same and must have originated from the same point on the boundary of the initial object, which is a contradiction.

□

With this last property, unique, uniformly partitioned layers between an initial and desired object can now be constructed.

3.4.2 Formulation

On 2D Euclidean space, Laplace's equation is given by

$$\nabla^2 \varphi(x, y) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) \varphi(x, y) = 0 \tag{3.20}$$

where $\varphi(x, y)$ is a scalar harmonic function representing a field. Boundary conditions must be applied to solve this PDE for the potential. Therefore, the initial and desired potential curves or surfaces are treated as boundaries and the interior between the two boundaries as free space. Since potential flows from areas of high potential to areas of low potential and layers evolve from the initial boundary in CAM processes, the potential on the initial boundary is set to an arbitrary positive value and the potential on the desired boundary is set to zero. Essentially, these boundaries are treated as basic Dirichlet boundary conditions.

Solving Laplace's equation results in a harmonic function that describes the potential between the initial and desired boundaries. By taking the gradient of this harmonic function, the field between the two boundaries can be determined and, by integrating, field lines can be generated. Due to the nature of Laplace's equation, the equipotential surfaces are not uniformly spaced and do not lend themselves well to material deposition. This issue is resolved by reparametrizing the field lines extending between the initial and desired boundaries by arc length.

If each field line is only known for a discrete set of points (as is mainly true for numerical solutions), then a continuous curve can be formed by interpolating with a spline comprised of piecewise parametric cubic polynomials as in Eqn. (3.7). If the parametric variable in the spline is represented by t, then the arc length d between

two points $[\alpha, \beta]$ on one of the segments of the spline is calculated as

$$\begin{aligned} d &= \int_{\alpha}^{\beta} \sqrt{\left(\frac{dX_1}{dt}\right)^2 + \left(\frac{dX_2}{dt}\right)^2} dt \\ &= \int_{\alpha}^{\beta} \sqrt{\left(3a_1t^2 + 2b_1t + c_1\right)^2 + \left(3a_2t^2 + 2b_2t + c_2\right)^2} dt \end{aligned} \tag{3.21}$$

where a_i and b_i are the scalar components of the coefficients describing the cubic polynomial. The total arc length is then calculated as

$$D = \sum_{i=1}^{n} d_i \tag{3.22}$$

where n is the total number of cubic polynomials forming a spline that represents a single field line and d_i is the total arc length of each segment. In practice, the arc length can be computed numerically or approximated by summing the length of line segments that approximate the spline at a fine enough resolution.

These field lines can then be reparametrized with parametric piecewise cubic polynomials as

$$\mathbf{X}(s) = \mathbf{a}s^3 + \mathbf{b}s^2 + \mathbf{c}s + \mathbf{d} \tag{3.23}$$

to form a new spline where $\mathbf{X} \in \mathbb{R}^2$, s is the arc length at that specific point, and $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ are parameters that uniquely define each field line. In this case, when $s = 0$ the point on the spline is on the initial surface and when $s = D$, the point is on the desired surface.

The spacing between each layer can be calculated by dividing the total arc length of each field line by the desired number of deposited layers, n_d. Each layer $\mathbf{L}_j$ is then defined as the set points

$$\mathbf{L}_j = \left\{ \mathbf{X}_k \left(\frac{j s_k}{n_d} \right) \right\} \tag{3.24}$$

where $j = \{0, 1, \ldots, n_d\}$, $k = \{1, 2, \ldots, n_f\}$, n_f is the number of field lines, s_k is the total arc length of the kth field line, and $\mathbf{X}_k \left(\frac{j s_k}{n_d} \right)$ is the kth point of the set evaluated at a fractional portion of the arc length dependent on the current layer. Note that by using the above definition $\mathbf{L}_0$ is the initial boundary, $\mathbf{L}_{n_d}$ is the desired boundary, and each intermediate boundary $\mathbf{L}_j$ will be uniformly partitioned throughout the medium.

3.5 Results

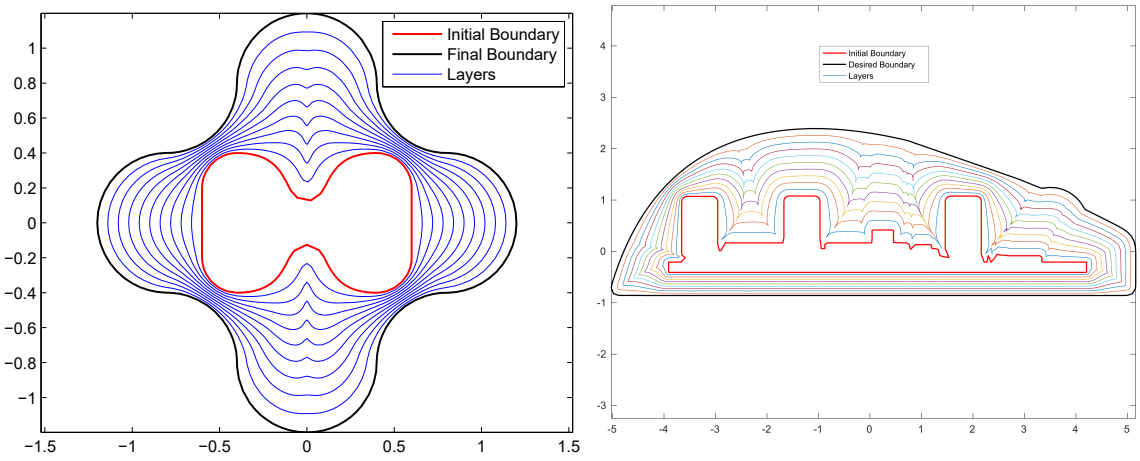
Both methods were simulated to verify their efficacy. The VOC method was solely implemented in MATLAB, while the Laplace's equation method was solved in COMSOL and solutions were manipulated in MATLAB to form uniformly partitioned layers. In the remainder of this section, examples of layer deposition for 2D objects are presented. Then the reasons for why reparametrized solutions to Laplace's equation do not produce orthogonal equipotential boundaries and field lines for intermediate layers is discussed and finally the results of each method are compared. In the following chapter, examples of layer deposition for both methods in 3D will

presented and compared. Additionally, examples of layer deposition in both 2D and 3D will be presented for objects with hollow features.

3.5.1 Examples of Layer Deposition

Layer deposition in 2D can range from simple examples, such as conformally generating layers around a circle to grow into a dumbbell (Fig. 3.2a), to complicated examples, such as encasing a circuit board for a computer mouse and growing into its plastic shell (Fig. 3.2b). In the first example, ten layers were generated using the VOC method and in the second example ten layers were generated using the Laplace's equation method. Although the examples are drastically different, it is clear that each layer is generated conformally about the boundary of the initial object. What may not be clear is the optimal number of layers to physically generate the desired object using an AM process.

In these examples, only ten layers were deposited which resulted in large geometric disparities between the layers. Most commercial AM processes can produce layer thicknesses of 0.150 mm or less [39, 40, 41, 42], while consumer grade AM processes can typically produce layer thicknesses of 0.2 mm or less [43, 44]. Minimum layer thickness varies by AM process and manufacturer; however, thicknesses of less than 0.125 mm are difficult or impossible to achieve. In practice, the number of layers can be chosen to match the layering resolution of the AM process or the layers can be

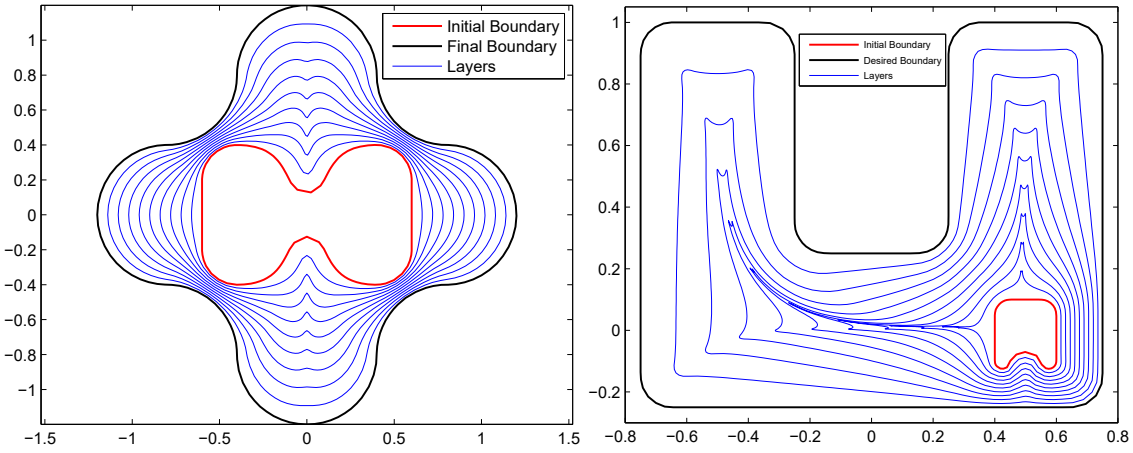


(a) Layers generated from a circle to a sphere **(b)** Layers generated for a computer mouse

Figure 3.2: Layers generated for arbitrary 2D objects

adjusted by the process presented in [45]. Alternatively, the velocity of the print head may be adjusted to print more material along certain portions of the layer or space-filling curves may be generated to fill in larger sections. Regardless of the method chosen, layers (as defined by this method) need to be selected to be greater than or equal to the minimum layer thickness of the AM process to create the desired object.

The previous examples highlighted layer deposition for both the VOC method and the Laplace's equation method; however, the main advantage of the Laplace's equation method (i.e. deposition between non-convex objects) is highlighted by depositing ten layers between arbitrary, planar, non-convex objects in Fig. 3.3. In this case, layering can be seen for complex geometries that are not limited by the placement of the initial object nor the convexity of either the initial or desired object.

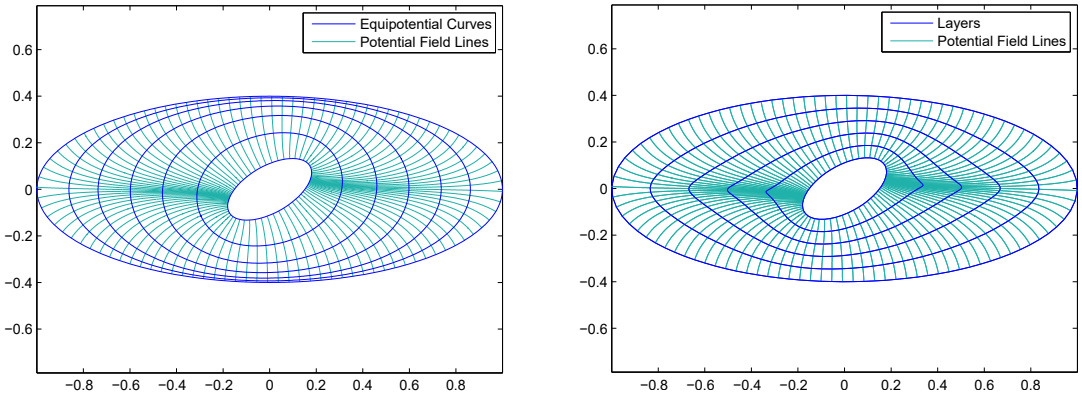


(a) Co-located non-convex objects **(b)** Off-center non-convex objects

Figure 3.3: Layers generated for arbitrary non-convex objects

3.5.2 The Nonorthogonality of Reparametrized Laplace's Equation Solutions

As discussed previously, field lines from Laplace's equation are orthogonal to each equipotential boundary. However, this property may have been altered when the solution was reparametrized to generate uniformly partitioned layers and by additional sources of error such as numerical roundoff. A 2D study was conducted to determine whether the field lines were still perpendicular to each layer. For this study, the initial and desired curves were selected as ellipses and five layers were generated. Figure 3.4a displays the initial equipotential curves before reparametrization and Fig. 3.4b displays the uniformly partitioned layers after reparametrization. Both figures have the same field lines (which are not altered) in the background.



(a) Original equipotential curves

(b) Uniformly distributed layers from reparametrization

Figure 3.4: Comparison of reparametrized layers for the Laplace's equation method

Table 3.1: The intersection angle between layers and field lines for 8 vertices per layer

Layer Number	Intersection Angle Between Layers and Field Lines (Degrees)							
1	44.54	133.99	97.41	85.51	60.48	40.81	99.85	91.25
2	44.15	135.68	97.51	83.99	58.75	42.08	101.53	91.39
3	52.46	128.26	95.52	84.76	64.85	50.80	99.53	91.04
4	67.33	114.03	92.90	87.00	75.80	65.67	95.34	90.53
5	90.02	89.61	90.03	89.96	89.97	90.39	89.96	90.00

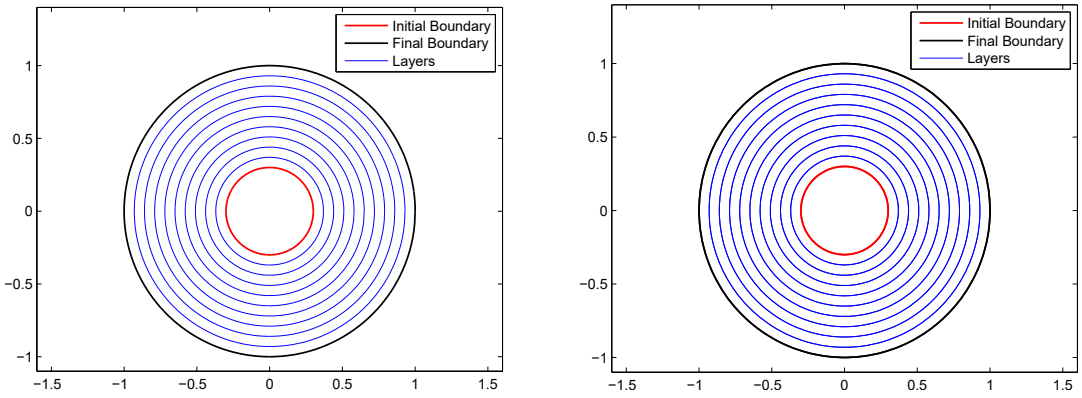
Visually, it may appear as if the intersections between the field lines and layers are orthogonal; however, a numerical study verified that this was not the case. The angle between the field lines and layers were calculated for a subset of the vertices via the dot product. Table 3.1 displays the intersection angle, in degrees, for 8 vertices in each layer.

From these results, it is clear that the newly parametrized layers are, in fact,

no longer equipotential curves. Reassuringly, the final layer, which was defined as a boundary condition for Laplace's equation, retains its orthogonality with the field lines. Although the field lines are no longer perpendicular to the intermediate layers, every layer is still unique. The points that define the layers are determined by moving along the unparameterized field lines (which were perpendicular to the equipotential boundaries) at different rates corresponding to the distance to the desired curve. As stated earlier, these field lines do not intersect one another between the initial and desired curves. Therefore, since each layer evolves outward from the previously defined layer, subsequent layers cannot intersect and are hence unique.

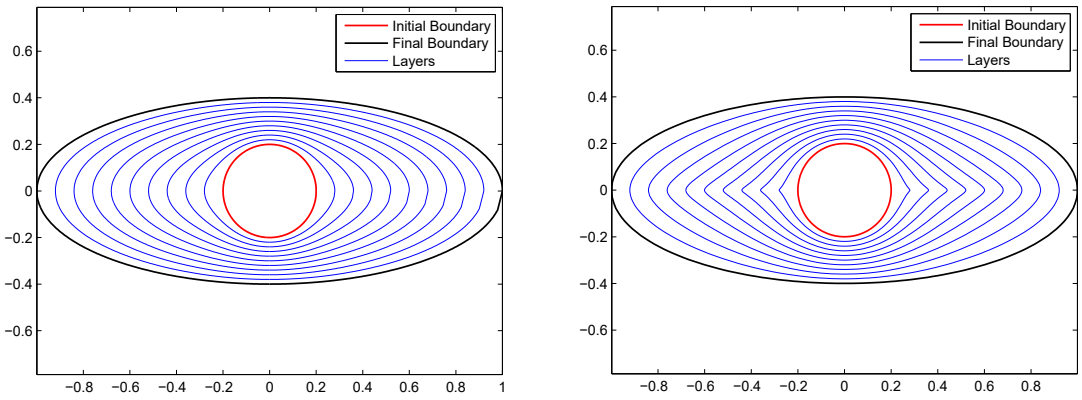
3.5.3 Comparison of the Methods

The Laplace's equation method was formulated for non-convex objects; however, it can also generate layers for convex and compatible objects. Both methods were compared by generating ten layers for planar convex and compatible objects. For the convex case, two geometric objects were tested. First, layers for an annulus were plotted in Fig. 3.5 and then layers for an ellipse with a circular cutout were generated in Fig. 3.6. The results clearly display that both methods produce the same results for the annulus, but not for the ellipse. Returning to an earlier discussion, solutions to Laplace's equation require that equipotential boundaries intersect the field perpendicularly. Since the initial curve and desired curve are still considered



(a) Layers generated by the VOC method **(b)** Layers generated by the Laplace's equation method

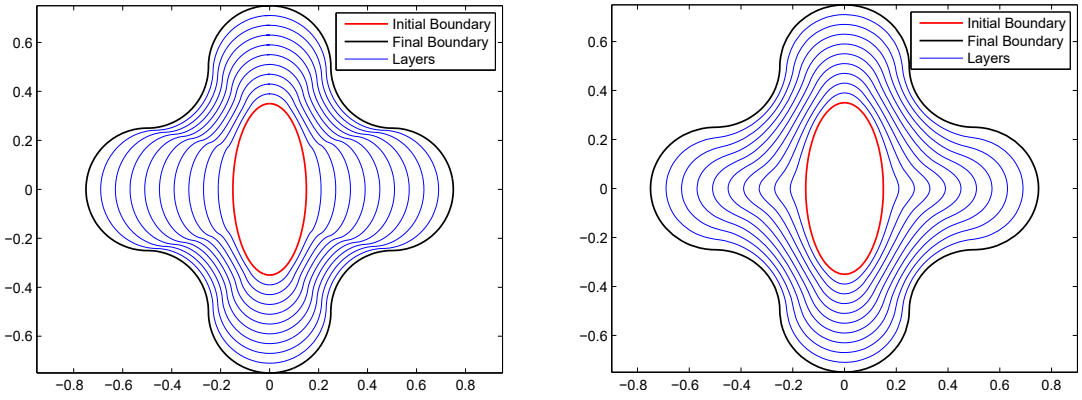
Figure 3.5: Layers generated for an annulus



(a) Layers generated by the VOC method **(b)** Layers generated by the Laplace's equation method

Figure 3.6: The general convexity case

equipotential curves, the field lines must intersect both curves perpendicularly. In the case of the annulus, normal lines from the initial curve are also normal lines of the desired curve. Therefore, the solutions to both methods are equivalent. The results from the second case confirm this notion, especially along the semi-major axis of the ellipse. Each layer generated by the VOC method is a minimum of C^1



(a) Layers generated by the VOC method **(b)** Layers generated by the Laplace's equation method

Figure 3.7: The compatible geometric object case

continuous, while only some of the layers from the Laplace's equation method are C^1 continuous. Interestingly, if the layers from both methods are superimposed over one another, the layers exactly overlap on the semi-major and semi-minor axes. As before, these locations are where the normal lines from both curves are the same.

For the compatible object case, an ellipse was selected as the initial curve and an adaptation of a "plus" sign was chosen as the desired curve. The layers for each method are presented in Fig. 3.7. As expected, the layers generated for this compatible geometry are not identical. Since the VOC method is limited to convex initial curves, there will be no instance where a compatible desired curve will share all its normal lines with the initial curve. Therefore, there is no compatible geometry (that is not convex) where both methods will produce the same results.

Although these comparisons were completed for 2D geometries, the same results

will apply in 3D due to the underlying properties of the algorithms.

3.6 Conclusions

In this chapter, two methods were presented to generate a cascade of enveloping layers, in two dimensions, between an initial and a desired object. The first method utilized VOCs relative to the boundary of the initial object and was constrained to convex initial objects and "compatible" desired objects. The second method manipulated solutions to Laplace's equation to generate layers and was not limited to objects of specific convexities. Notably, the layers that resulted from reparametrizing the solutions to Laplace's equation by arc length were no longer equipotential boundaries; however, each layer remained unique (i.e. did not intersect one another). Finally, the methods were compared to one another and the necessary conditions for equivalent solutions given a set of objects was outlined.

In the following chapter, these methods will be extended to 3D and additional examples and analysis will be provided. Furthermore, a strategy for incorporating hollow features into the build volume is also introduced.

Chapter 4

Hollow Features and Layers for Conformal Additive Manufacturing in $\mathbb{R}^3$

The previous chapter introduced two methods for generating layers in conformal additive manufacturing (CAM) processes in two dimensions (2D). This chapter will extend both methods to three dimensions (3D), introduce a secondary method that can incorporate hollow features into the build volume, and outline a system in which CAM may be possible. Examples of layer generation in 3D is presented for both the VOC method and the Laplace's equation method. Additionally, examples of layer generation in 2D and 3D with hollow features is also presented. Finally, a toy

example is provided where the methods developed in these last two chapter can be applied to robot motion planning.

4.1 Extending the Methods to $\mathbb{R}^3$

In Chapter 3, two methods were introduced to generate layers conformally from an initial object to a desired object in 2D. This section extends each method to 3D and provides examples of layer deposition for both.

4.1.1 The VOC Method

Four modifications are required to transform the VOC method in 2D to 3D. First, the set of points $\mathbf{U}$ and $\mathbf{V}$ describing the boundaries of the initial and desired objects must lie on a regular 3D grid. Second, the initial surface must be defined by twice differentiable parametric functions or be approximated by piecewise parametric bicubic patches. Additionally, the desired surface must have an implicit representation to be able to determine points of intersection on the surface, and, finally, normal vectors must be extended to the 3D case.

A parametric bicubic patch is described as the tensor product between two different parametric cubic splines, $\mathbf{P}$ and $\mathbf{Q}$. Given two parametric cubic polynomials $\mathbf{P}$

and $\mathbf{Q}$

$$\mathbf{P}(u) = \mathbf{a}_1 u^3 + \mathbf{b}_1 u^2 + \mathbf{c}_1 u + \mathbf{d}_1, \qquad u \in [s, s+1) \tag{4.1}$$

and

$$\mathbf{Q}(v) = \mathbf{a}_2 v^3 + \mathbf{b}_2 v^2 + \mathbf{c}_2 v + \mathbf{d}_2, \qquad v \in [t, t+1) \tag{4.2}$$

from two different splines then the bicubic patch over the rectangular region enclosed by coordinates $[(s,t),(s,t+1),(s+1,t),(s+1,t+1)]$ is given by

$$\mathbf{X}(u,v) = \mathbf{P}(u)\mathbf{Q}(v) = \sum_{i=0}^{3}\sum_{j=0}^{3} u^i v^j e_{ij} \tag{4.3}$$

where now $\mathbf{X} = [x,\ y,\ z]^T \in \mathbb{R}^3$, e_{ij} is the appropriate value for the multiplied spline coefficients from $\mathbf{P}$ and $\mathbf{Q}$.

Tangent vectors to the parametric surface are calculated by taking partial derivatives of the bicubic patches

$$\mathbf{T}_1 = \frac{\partial \mathbf{X}}{\partial u} = \left[\ \tfrac{\partial x}{\partial u}, \tfrac{\partial y}{\partial u}, \tfrac{\partial z}{\partial u}\ \right]^T, \tag{4.4}$$

and

$$\mathbf{T}_2 = \frac{\partial \mathbf{X}}{\partial v} = \left[\ \tfrac{\partial x}{\partial v}, \tfrac{\partial y}{\partial v}, \tfrac{\partial z}{\partial v}\ \right]^T. \tag{4.5}$$

Normal vectors are derived by taking the cross product of the tangent vectors in the order that preserves a right-handed frame, i.e. $\mathbf{N} = \mathbf{T}_1 \times \mathbf{T}_2$.

The intersection of the normal vector from the initial surface with the desired surface can be calculated by substituting the coordinates of the parametric line in 3D into the implicit equation and then solving the resulting polynomial for the parameter of interest.

For instance, consider an ellipsoid with principal semi-axes j, k, and l as the desired object. The implicit equation for this ellipsoid in $\mathbb{R}^3$ is

$$\frac{(x-x_c)^2}{j^2}+\frac{(y-y_c)^2}{k^2}+\frac{(z-z_c)^2}{l^2}=1 \tag{4.6}$$

where (x_c, y_c, z_c) is the point at the center. The equation for a parametric line that extends from the surface of the initial object $\mathbf{X}_i$ past the surface of the desired object $\mathbf{X}_f$ is given by

$$\mathbf{X}(t)=(1-t)\,\mathbf{X}_i+t\mathbf{X}_f \tag{4.7}$$

where $t \in [0,1]$. The point of intersection on the desired surface is then found by solving the following quadratic equation formed by substituting the scalar components of Eqn. (4.7) into Eqn. (4.6) for the positive root of t (the negative root corresponds to the intersection with the surface of the desired object along the inward-facing normal) and substituting back into Eqn. (4.7).

$$\left(\frac{q_1}{j^2}+\frac{q_2}{k^2}+\frac{q_3}{l^2}\right)t^2+\left(\frac{2q_1r_1}{j^2}+\frac{2q_2r_2}{k^2}+\frac{2q_3r_3}{l^2}\right)t+\left(\frac{r_1}{j^2}+\frac{r_2}{k^2}+\frac{r_3}{l^2}\right)-1=0 \tag{4.8}$$

where $q_1 = X_{f1} - X_{i1}$, $q_2 = X_{f2} - X_{i2}$, $q_3 = X_{f3} - X_{i3}$, $r_1 = X_{i1} - x_c$, $r_2 = X_{i2} - y_c$, and $r_3 = X_{i3} - z_c$

The corresponding layers are then formed according to the methodology specified in the 2D method. The layers resulting from the VOC method are presented in Fig. 4.1, where five layers are deposited onto a sphere to form a larger ellipsoid.

4.1.2 The Laplace's Equation Method

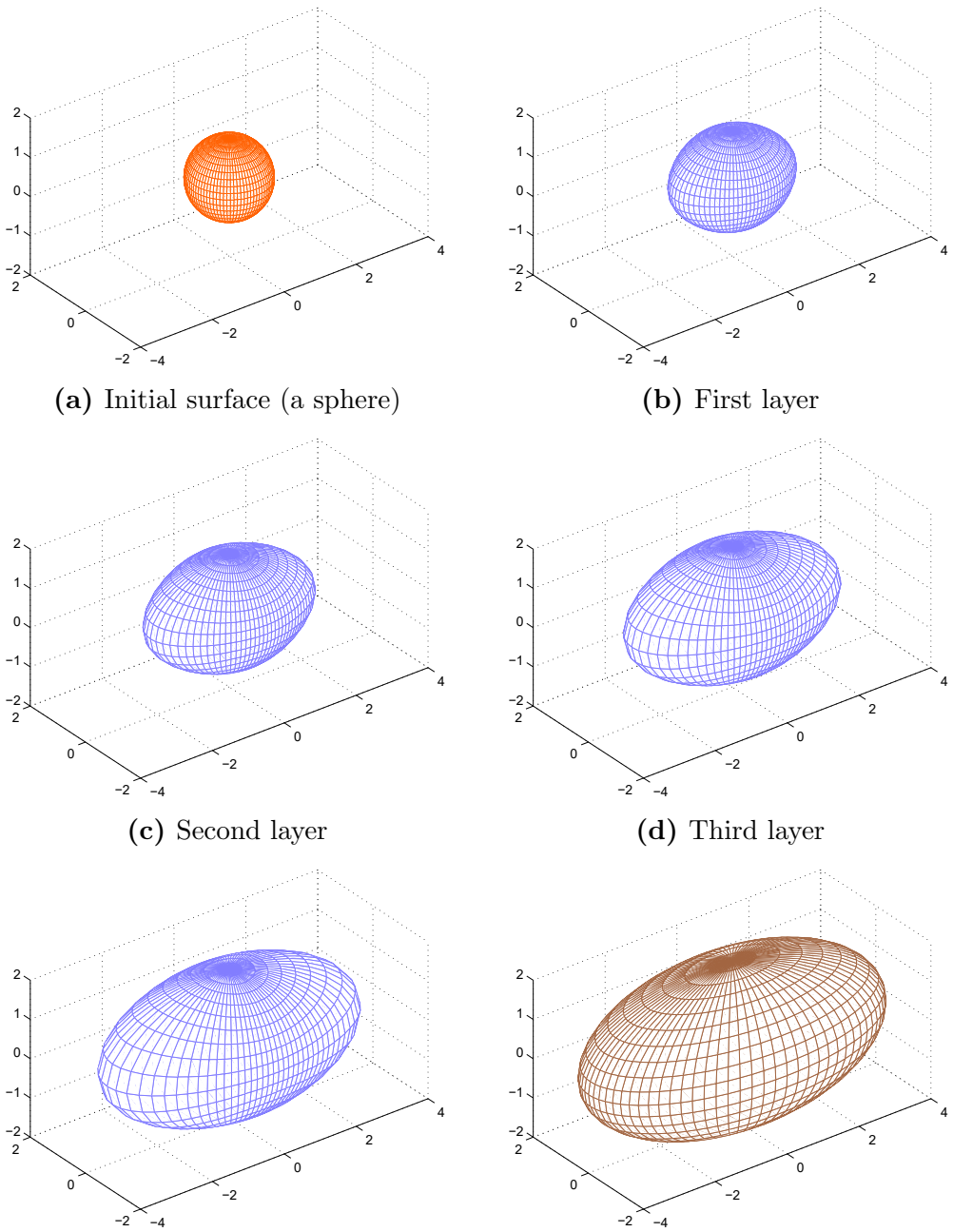
The two-dimensional Laplace's equation method has trivial modifications to extend it to 3D. First, Laplace's equation is now given by

$$\nabla^2 \varphi(x, y, z) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right) \varphi(x, y, z) = 0, \tag{4.9}$$

the equation of a single cubic polynomial comprising the spline that represents each field line is now $\mathbf{X}(s) = \mathbf{a}s^3 + \mathbf{b}s^2 + \mathbf{c}s + \mathbf{d} \in \mathbb{R}^3$, and the arc length of a single cubic polynomial between two points $[\alpha, \beta]$ is calculated by

$$\begin{aligned} d &= \int_\alpha^\beta \sqrt{\left(\frac{dX_1}{dt}\right)^2 + \left(\frac{dX_2}{dt}\right)^2 + \left(\frac{dX_3}{dt}\right)^2} dt \\ &= \int_\alpha^\beta \sqrt{(3a_1t^2 + 2b_1t + c_1)^2 + (3a_2t^2 + 2b_2t + c_2)^2 + (3a_3t^2 + 2b_3t + c_3)^2} dt \end{aligned} \tag{4.10}$$

With these modifications, it is now possible to generate layers conformally in 3D. Fig. 4.2 demonstrates the deposition of five layers onto an ellipsoid to form a larger



(a) Initial surface (a sphere)

(b) First layer

(c) Second layer

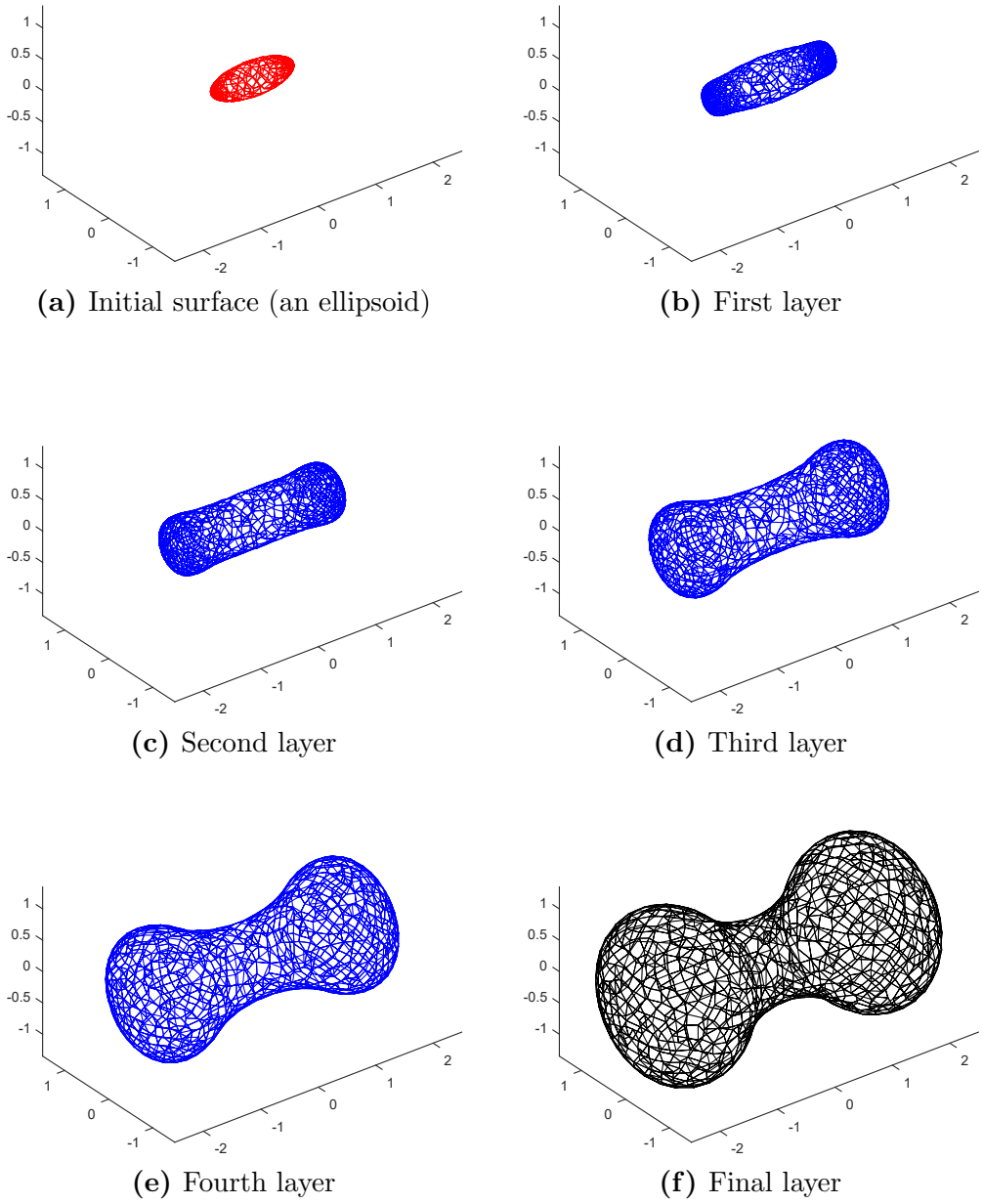
(d) Third layer

(e) Fourth layer

(f) Final layer

Figure 4.1: Surface evolution of an ellipsoid to a convex surface

non-convex "dumbbell-shaped" object using the Laplace's equation method.



(a) Initial surface (an ellipsoid)

(b) First layer

(c) Second layer

(d) Third layer

(e) Fourth layer

(f) Final layer

Figure 4.2: Surface evolution of an ellipsoid to a non-convex surface

4.2 Volumes with Hollow Features

As stated earlier, one of the possible benefits of a CAM process is the ability to create hollow features. The closest analogue in a traditional AM process is holes. Depending on the orientation of the hole relative to the deposition or print head, the hole is either filled with a secondary support material or the print head stops depositing material and is lifted at the edge of the hole and then continues printing on the opposite edge. Although this sort of process is certainly possible in the current framework, the resulting layers are no longer considered conformal as there is a break in the deposition process. Therefore, this section seeks a method that can effectively deposit layers around the desired hollow feature or void.

4.2.1 Limitations

First, this method is currently limited to hollow features such that the point at the geometric center (i.e., the centroid) is considered compatible (per the definition in Assumption 3 of Sec. 3.3.1). And secondly, this method treats hollow features as local deformations to the preexisting layers. That is, one of the previous methods (either VOC or Laplace's equation) is used to generate evenly partitioned layers for the volume without hollow features and then this method is applied to generate local deformations that force the layers around the hollow feature or void. The trade-off

for being able to conformally deposit material around these hollow features is that the layers are no longer equally partitioned in the local area of the deformations.

4.2.2 Formulation

This section will only present the formulation for incorporating hollow features in 2D; however, this method can easily be extended to 3D by applying the appropriate changes outlined in Sec. 4.1.1. Given an hollow feature in the build volume, the basic idea is to form an "area of effect" (AOE) around the hollow feature where local alterations to the layer are allowed. Specifically, it is desired to linearly transform points from within the hollow feature to the area between the boundary of the hollow feature and the boundary of the AOE. The implementation is as follows; first, given a set of ordered points $\mathbf{O}_i = \{\mathbf{o}_{i1}, \mathbf{o}_{i2}, \ldots, \mathbf{o}_{in}\}$ that represent the vertices of i hollow features, the centroid Oc_i of each feature is determined by

$$O_{c_i} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{o}_{ij} \tag{4.11}$$

where n is the number of points representing each feature. Then, the points representing each hollow feature are dilated or scaled through a linear transformation by a set factor, typically 1.5. This dilated feature is essentially the AOE, where points within the area are altered and those outside remain unchanged. Next, both the

hollow feature and its dilated representation are approximated as closed curves using piecewise parametric cubic polynomials to form splines following the process outlined in the beginning of Sec. 3.3.

If a point is determined to be within the AOE, the distances between the point of interest and the boundary of the hollow feature as well as the boundary of the dilated feature are calculated along the line that contains the point of interest and extends from the centroid of the hollow feature to its respective boundary. The point of intersection with these boundaries is determined following the procedure outlined in Sec. 3.3. Since these lines intersect with each boundary twice, the shortest distance is chosen and corresponds to the line from the centroid to the point of interest as opposed to the line from the point of interest to the centroid. These distances are then used in the equation of a parametric line to scale the original point to some location between the boundary of the hollow feature and the dilated boundary. The scaling factor is calculated as

$$s = d_{hf}(1 - \beta) + \beta d_{df} \tag{4.12}$$

where d_{hf} is the distance to the boundary of the hollow feature, d_{df} is the distance to the boundary of the dilated feature, and β is the ratio of the distance from the center of the hollow feature to the point of interest and the distance to the dilated

boundary. Each point p_i in the AOE is then transformed by

$$P_i = s\frac{p_i}{d} \tag{4.13}$$

where d is the distance from the centroid of the hollow feature to the point. Intuitively, this transformation moves a point located at the centroid to the boundary of the hollow feature and leaves points on the dilated boundary unchanged. Points in between these two boundaries are shifted along the line that originates at the centroid and contains the point of interest.

As currently formulated, the behavior for a point located at the exact geometric center is undefined. For this case, the fact that the point of interest is in fact chosen from the set of points representing an individual conformal layer is leveraged. First, the two adjacent points in the layer are transformed as prescribed above. Then, a line l_1 is created between the two points and, finally, the point located at the centroid is shifted to the boundary of the hollow feature along the line that bisects l_1. Another possible concern is when the hollow feature intersects or is tangent to the boundary of the initial object. For this case, any point that is transformed inside of the boundary of the initial object is discarded and no longer part of the layer. A final concern is when one point exists within multiple AOEs for different hollow features.

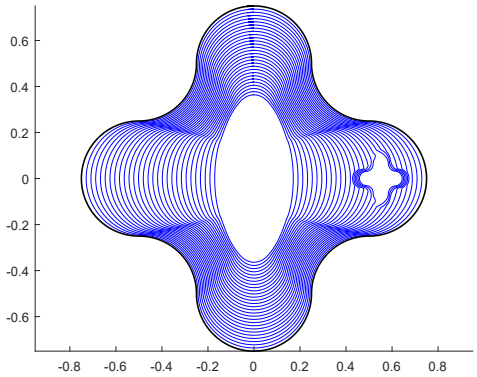
For this case, initial experimentation was conducted by virtually transforming the point for each individual AOE and then shifting the point to the average of the

virtual transformations. This method resulted in layers that oscillated repeatedly in the overlapping AOE due to the different strengths of the virtual transformations that depend solely on the distance of the point from the boundary of the AOE. This issue was no less pronounced when weighted averages for each virtually transformed point from each AOE were considered; therefore, an alternate method relying on interpolation was implemented. For each layer, the first and last transformed point that existed within overlapping AOEs was identified. These points and their adjacent points (that remained outside of the AOE) were then used as control points during the interpolation. The intermediate points between the first and last transformed points in the overlapping AOEs were then replaced with the interpolated segment to complete the layer. As a first pass, a linear interpolation method was considered but the resulting discrepancy in the smoothness at the end points was deemed to be too severe. Instead, a shape-preserving piecewise cubic interpolation method [46] was implemented that better preserved the smoothness along the layer. Furthermore, since this method is shape-preserving and it is effectively interpolating over a linear segment in between the control points, the original non-intersecting layers remained non-intersecting after alteration.

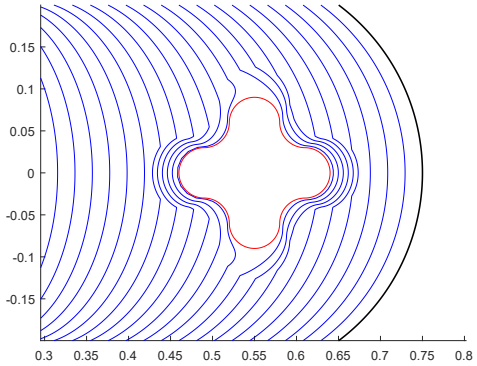
4.2.3 Examples

In the previous section, a method to generate layers for objects with multiple hollow features was presented. The geometric center or centroid of these hollow features was required to be compatible (per the definition given in Assumption 3 of Sec. 3.3.1). Hollow features that do not satisfy these convexity conditions can still be handled; however, a non-conformal process similar to methods currently utilized in traditional AM must be employed. Since the method to incorporate hollow features into the build volume is identical for both the VOC method and the Laplace's equation method, 2D examples will be provided utilizing the VOC method, while a 3D example will utilize the Laplace's equation method. It should be noted that the results for each method will be different, unless the initial layers generated for objects without hollow features are the same as in the case of the annular region in Fig. 3.5. Below, Fig. 4.3 provides three, 2D examples of layer generation using the VOC method from an initial ellipse to a desired star-shape with a different number of hollow features. From Fig. 4.3, it is evident that this method is able to conformally generate layers around various types of hollow features at the cost of locally disrupting the even partitioning and smoothness of the layers.

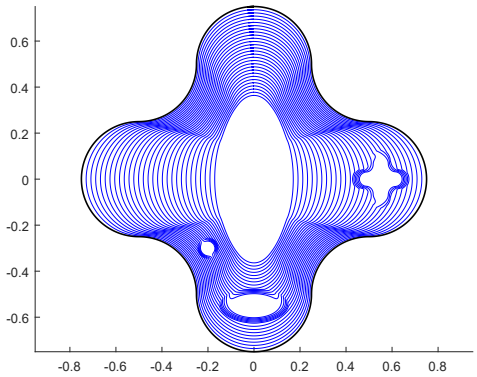
As previously mentioned, each of these 2D examples generalize to the 3D case and to the Laplace's equation method. Therefore, to minimize redundancy and to highlight the expanded capabilities of the Laplace's equation method, Fig. 4.4 displays



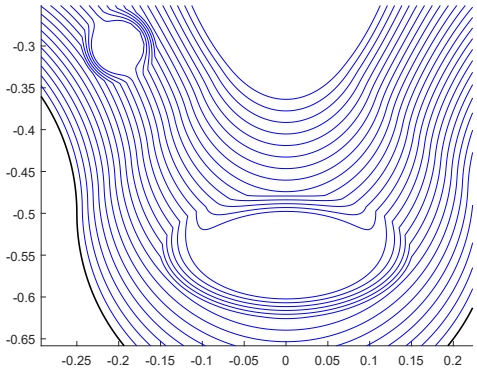
(a) Layer generation for a single hollow feature



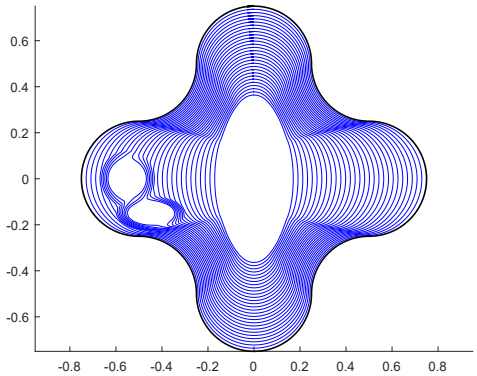
(b) Closeup of the layers around a single hollow feature



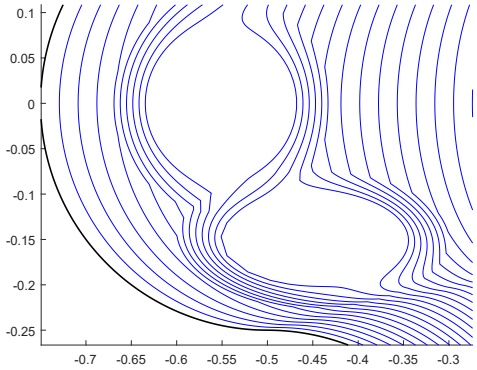
(c) Layer generation for multiple hollow features



(d) Closeup of the layers around multiple hollow features



(e) Layer generation for overlapping hollow features

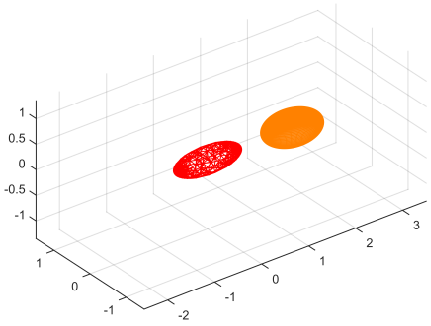


(f) Closeup of the layers around overlapping hollow features

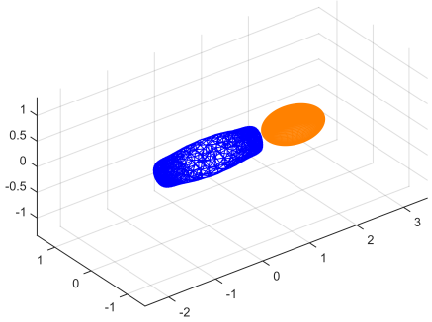
Figure 4.3: 2D layer generation using the VOC for single and multiple hollow features

the generated layers for the same initial and desired surfaces used in Fig. 4.2 with the addition of a single, hollow ellipsoidal feature. During this layer generation process the layers still evolve from the initial geometric object (an ellipsoid) into the desired object (a dumbbell), but beginning with the second layer the behavior is markedly different than the earlier example. In Fig. 4.4c, the layers initially contract away from the hollow feature. But as the untransformed layers transition past the centroid of the hollow ellipsoid as in Fig. 4.4e, the transformed layers envelop the remainder of the hollow feature. Once past the AOE of the hollow feature, as in Fig. 4.4f, the layer generation behavior again matches that of the previous example.

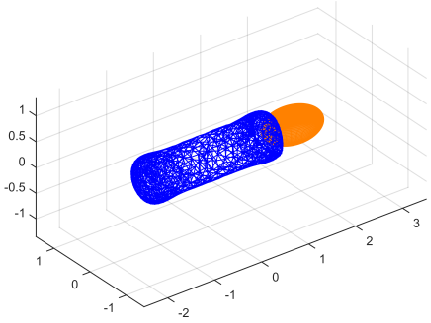
This specific implementation of incorporating hollow features or voids into the build volume was chosen due to its ability to alter previously generated layering solutions. In this manner, layers can rapidly be generated for different voids at a relatively low computational cost as long as the initial and desired objects remain the same. If instead a single method was desired to generate layers between an initial and desired object containing voids, one could repurpose navigation functions originally developed for robot motion planning to generate layers between the initial and desired object. Specifically, the method developed by Rimon and Koditschek in [47] applies artificial potential functions to a configuration space to generate equipotential curves that do not intersect obstacles. These curves, again, naturally extend to layering in CAM.



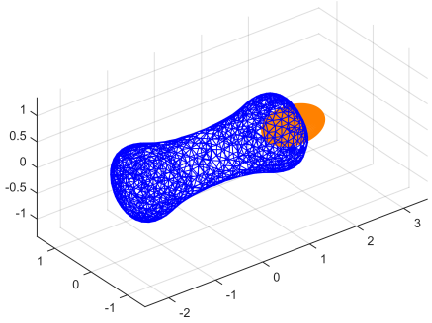
(a) Initial surface (left) with an ellipsoidal hollow feature (right)



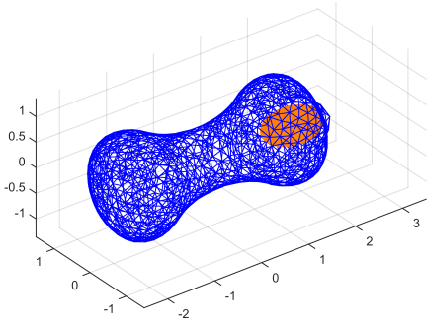
(b) First layer



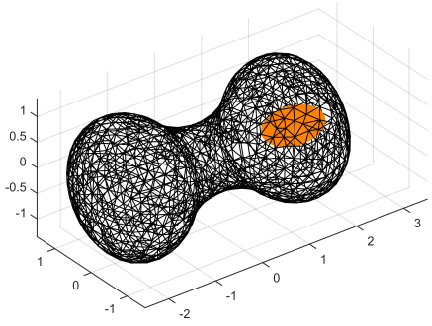
(c) Second layer



(d) Third layer



(e) Fourth layer



(f) Final layer

Figure 4.4: 3D layer generation from an ellipsoid to a non-convex surface with a single ellipsoidal hollow feature

4.3 Conformal Additive Manufacturing and Robotic Applications

In Chapters 3 and 4, two methods were introduced to generate layers conformally around an initial object to grow it into a desired object. This section presents a theoretical physical system that is capable of CAM as well as discusses how these methods could also be applied to robot motion planning.

4.3.1 A Conformal Additive Manufacturing Implementation

In traditional AM, material is deposited in a single plane, perpendicular to the direction of deposition. This plane remains constant while a secondary mechanism, parallel to the direction of deposition, alters the height at which a layer is deposited. It is for this reason that AM is frequently called "3D printing". In contrast, CAM necessitates a more flexible platform where material can be applied along the normal direction of an arbitrary surface. One such manner to provide such flexibility is to attach a deposition head to a six degree-of-freedom (DOF) robot arm such as Universal Robot's UR5 or KUKA's LBR iiwa. The initial object is then fixtured to a known location within the workspace of the robot arm. Layers are then be

deposited along the exposed surface of the initial object and then the robot arm flips the object over and deposits material on the other side to complete a layer. This process continues iteratively until the desired object is constructed. Fig. 4.5 presents

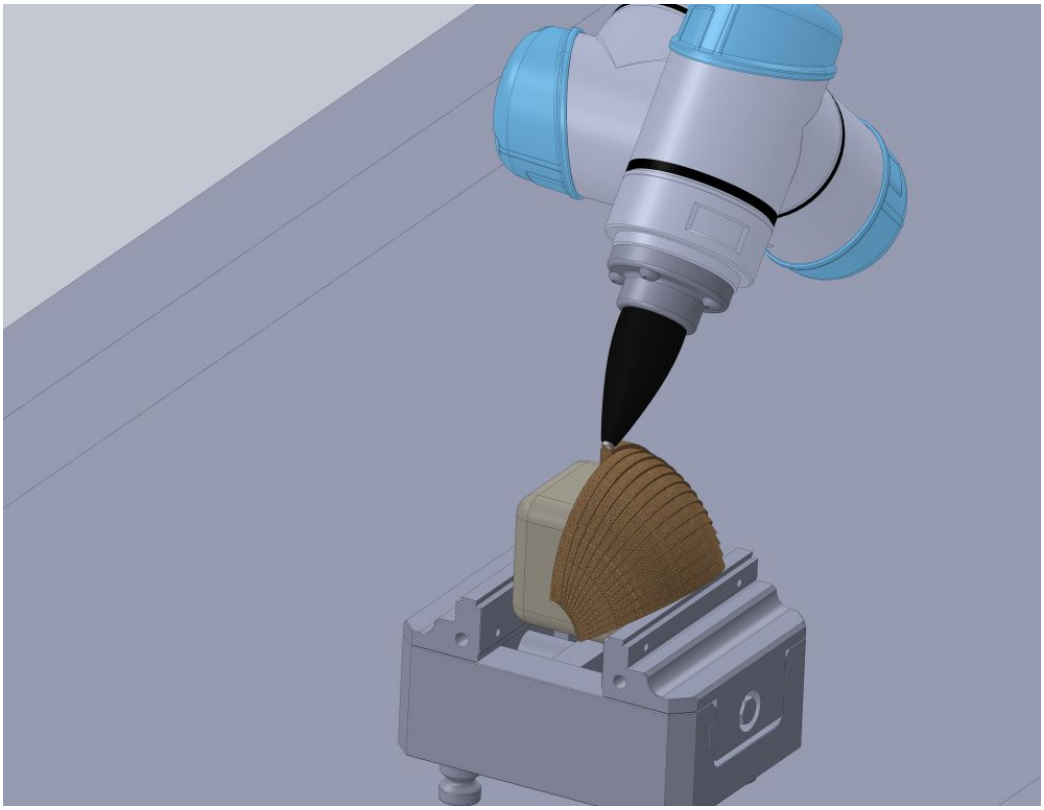


Figure 4.5: A possible physical system for implementation of CAM

one possible implementation of this system. In the presented system, a UR5 from Universal Robots is fixed to a rigid table in a known location. Additionally, a vise is also attached to the same table and the pose of the vise relative to the base of the UR5 is known precisely. Then, an initial object, in this case a cube with rounded sides, is secured in the vise, again in a known pose. The UR5 is equipped with a

heated nozzle that can deposit a material, such as ABS plastic, directly onto the surface of the cube. The layers generated from this work are then translated into a trajectory (by some other software) for the robot to execute and ultimately transform the initial cube into a sphere. Another possible implementation, which was also based on the work presented in this dissertation, was recently demonstrated by Kutzer and DeVries in [48] with two, separate robot arms.

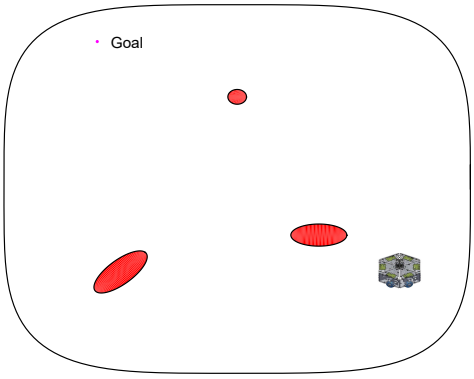
4.3.2 Robot Motion Planning

Although these methods were originally developed for generating layers in CAM, they can also be used in robot motion planning. Consider a robot operating within a bounded configuration space. In this context, the desired object represents the boundary of the configuration space of the robot and voids or hollow features represent obstacles. If the boundaries of these objects are dilated to account for collisions with the robot at certain configurations, then the robot can be represented by a single point. The motion planning process is then conducted as follows. First, the initial object is replaced with an infinitesimally small circle (or sphere) with its origin located at the centroid of the configuration space boundary. Next, one of the layer generation methods is executed to generate layers within the configuration space. By definition, these layers are closed and continuous and exist conformally around not only the initial object but also the configuration space obstacles. Therefore, the space between

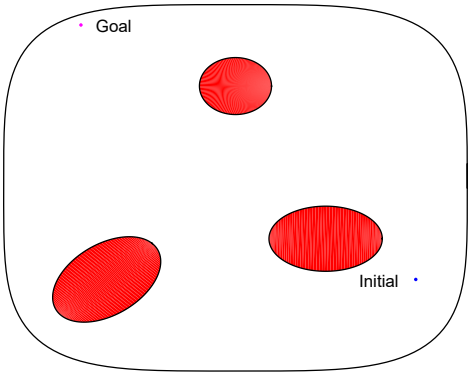
adjacent layers can be thought of as collision-free cells in the configuration space. And finally, a roadmap-based planner can be implemented to generate a trajectory from an initial configuration to a goal configuration within the configuration space.

Given an initial configuration g_0, a goal configuration g_d, and a roadmap (the collision-free cells generated from one of the layering algorithms), one implementation of the roadmap-based planner executes as follows. First, construct a line from g_0 to the next adjacent layer in the direction of g_d along the vector that stretches from the centroid of the virtual initial object to g_0. Then, construct a second line that extends from the centroid of the virtual initial object to the boundary of the configuration space along the line containing g_d. As long as an obstacle is not present, any two adjacent layers can be connected by drawing a line that originates on the first layer and terminates on the adjacent layer along the normal of the first layer. Now that both g_0 and g_d are connected to the roadmap and there is a method to connect adjacent layers (or collision-free cells), the next step is to determine the trajectory of the robot between both configurations.

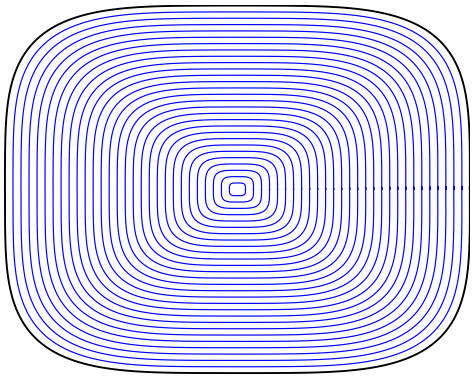
The trajectory is created by first traveling from g_0 to the adjacent layer in the direction of g_d. Once on the layer, the robot traverses until it intersects the second line defined above. The robot then successfully travels along this line to g_d unless an obstacle is encountered. In this case, the robot travels to the last possible layer before a collision occurs. Since layers were conformally generated around the obstacles, it



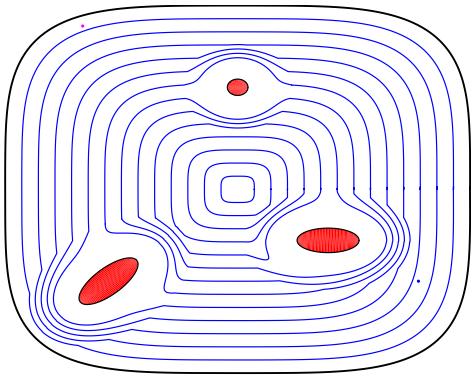
(a) HexDMR robot in a fixed workspace



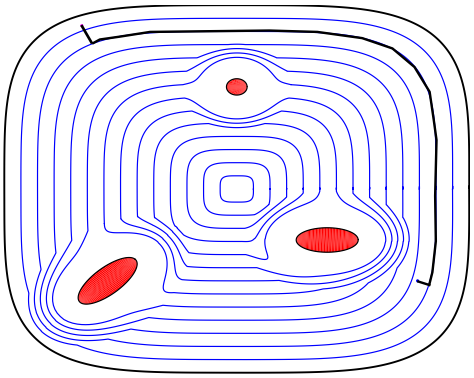
(b) The transformed configuration space by dilating obstacles



(c) Layers for the obstacle-free configuration space generated by the VOC method



(d) Layers, or collision-free cells generated by the VOC method



(e) A collision-free path for the robot between an initial and goal configuration

Figure 4.6: An example of layer generation for CAM used in robot motion planning

is guaranteed that the next adjacent layer will move the robot past the obstacle. Therefore, the robot travels along the layer until it is possible to construct a line that is tangent to the obstacle and normal to the current layer. Upon finding this line, the robot moves to the adjacent layer and backtracks until the line containing g_d is intersected. The process will continue until g_d is reached or a single layer is circumnavigated indicating that g_d is not reachable.

More concretely, consider a single robot from the HexDMR system (which will be discussed more thoroughly in the following chapters) and the workspace presented in Fig. 4.6a. Since robots in the HexDMR system are holonomic and operate in SE(2), the corresponding configuration space where the robot is shrunken to a point is presented in Fig. 4.6b. Then, Fig. 4.6d displays the layers generated for this configuration space using the VOC method. Finally, Fig. 4.6e shows the resulting path between an initial and goal configuration in this space that is collision-free.

One of the possible benefits of using these layer generation methods for robot motion planning is the fact that the layers are first generated for the configuration space without obstacles. Therefore, given a fixed configuration space boundary, the roadmap can be quickly regenerated for non-stationary obstacles. Additionally, for a given set of obstacles, the collision-free space is known in its entirety and probabilistic motion planning methods are not required. Finally, although computationally expensive and most likely impractical, the Laplace's equation method can be applied

to n-dimensional configuration spaces to generate similar results.

4.4 Conclusions

In this chapter, both the VOC method and the Laplace's equation method were extended to 3D. Examples were provided for layer deposition for both methods in 3D. Then, a method was introduced that manipulated solutions generated by both methods to incorporate hollow features or voids into the build volume. Multiple examples of incorporating hollow voids into 2D layer deposition were included for the VOC method and a single example of a hollow void in 3D was provided for the Laplace's equation method. This ability differentiates CAM from traditional AM and opens up a whole host of new opportunities from creating protective failure mechanisms in products to novel packaging solutions.

Next, one possible implementation of a CAM system was introduced. This hypothetical system included a single, six DOF robot arm and some additional tooling and discussed how material may be deposited conformally onto an object's surface. Finally, these same novel layer generation techniques were applied to robot motion planning. In this context, voids represent obstacles, the desired object represents the configuration space boundary, and the generated layers represent collision-free cells. A simple roadmap based planner was devised and an example was provided for a robot

from the HexDMR system. The generated solutions represent the entire collision-free space and non-stationary obstacles were accommodated through successive iterations of the void-handling method.

Part II

Cooperative Robotic Repair and Diagnosis

Chapter 5

Fault Tolerant Robotic Systems

Over the last several decades, the capabilities and presence of robotic systems in society has grown dramatically. Most notably, hundreds of thousands of industrial robotic arms have been deployed to factories to aid in assembling consumer goods; however, more recently robots have made strides into everyday products such as home assistants, drones, and medicine delivery aides in hospital settings. In all of these examples, these robotic systems are easily accessible and, for the most part, operate in moderately structured environments. The future of robotics however is in highly unstructured, remote locations and hazardous environments such as disasters areas and/or the exploration of foreign bodies in outer space. In these settings, there is no guarantee that a human operator may be available to retrieve and/or repair a robot in the field. Moreover, in some instances, it may even be dangerous for one

to do so. Therefore, for these systems to be practical and economical, they must be robust and adaptable to prevent or minimize possibly fatal system faults. To address this issue, there are two separate schools of thought. The first is to develop fault diagnostics and adaptive behaviors into a single complex system, while the second is to develop a cooperative team of low cost and low complexity robots such that the loss of a single entity does not necessarily mean the failure of the team. Both of these strategies, including examples and methodologies developed by other researchers, are discussed further in the following sections.

5.1 Individual Fault Tolerant Systems

A majority of mobile robotic systems in use today utilize a variety of high-powered computer processors as well as a multitude of sensors to perform tasks in semi-structured environments. These systems consolidate functionality into a single, highly capable agent. Typically, diagnostic checks are performed on different time scales over the duration of an agent's life. For instance, an agent may check its remaining power capacity every minute, while it may only check the health of larger subsystems after the agent fails to execute a command properly. After a fault has been detected, these systems often employ corrective or adaptive behaviors to extend lifetimes while possibly operating in non-optimal states. Section 5.1.1 summarizes some of the techniques

currently in use to diagnose faults and Sec. 5.1.2 outlines corrective behaviors for some of these systems.

5.1.1 Diagnosis Methods

Despite impressive advancements in robotics over the past several years, a majority of systems are still fairly fragile and encounter unexpected faults during operation. According to Parker in [49], some of the more common internal faults are individual robot (hardware) malfunctions, software errors or incompleteness, and communications failure. To address these concerns, several researchers have developed methods to diagnose various faults during runtime. For instance, [50] utilized a two-stage approach to first detect and then isolate a fault in a differential-drive mobile robot. Detection was accomplished by comparing the residual between a model-based state vector velocity estimate and the actual state vector velocity to a derived threshold. Once a fault was detected, three nonlinear observers were employed to isolate the fault to either the right wheel, left wheel, or a combination of unknown dynamic disturbances. [51] adopted a slightly different approach by utilizing a probabilistic hybrid automaton to diagnosis the severity of faults within the agent's omni-directional drive train.

While the previous two papers only diagnosed hardware faults, both Cai et al. and Goel et al. developed different algorithms to diagnose both hardware and sensor

faults. In [52], Cai presented a diagnosis framework utilizing several particle filters (PFs) for a four-wheeled mobile robot. By introducing several PFs, the observable state space was reduced in each individual filter and faults in each of the wheels as well as a gyroscope were able to be detected. Separately, Goel in [53], introduced several Kalman filters trained on individual failure mechanisms (i.e., encoders failures, gyroscope failures, or flat tires) of a Pioneer AT robot. A back-propagation neural network was then employed to determine which Kalman filter best matched the current state of the system. Finally, software related errors were diagnosed in [54, 55] by implementing model-based approaches to determine if software performance remained within known nominal ranges during operation. Deviations in communication services and execution time were tracked to determine if there were any irregularities.

5.1.2 Adaptation Strategies

In these particular systems, total system failure is rarely tolerated. Therefore, once a fault has occurred corrective behavior must be executed to prevent mission failure. One such method is presented by Brandstotter et al. in [51], where depending on the severity of a fault encountered in their three-wheeled omni-directional robot separate recovery mechanisms are executed. For instance, when a minor fault is detected in a single drive module, the gains in the redundant powertrain controller are adjusted to maintain the required control velocity. When a slightly more detrimental fault occurs

such as the complete failure of a single wheel in a drive module, the robot completes a controlled degradation and treats the omni-directional drive system as a differential drive system. This change also alters the control law and path planning of the robot. Finally, any fault that is deemed is too severe for recovery results in complete robot shutdown and notification of its operator.

Conversely, purely software related faults such as service call or module failures, [54, 55] present slightly different recovery behaviors. In [54], software methods or modules are deprecated upon failure, including dependent methods, and future requests are routed through equivalent or redundant functionalities. On the other hand, [55] presents a much more traditional manner of correcting software errors. In this method, modules containing failed services, as well as dependent software, are shutdown and restarted to recover the lost functionality. Even with these fault recovery behaviors, it is obvious that single failures can still be catastrophic in these single entity systems.

5.2 Cooperative Multi-Agent Systems

One such method to mitigate single-agent failures is to split functionality among several different agents (or modules) in the form of cooperative multi-agent systems (CMSs). For instance, agents in a CMS can be specialized for specific tasks pos-

sibly resulting in less complex and demanding designs. Additionally, work can be distributed to team members to reduce mission execution time. Moreover, if several agents were to enter fault states their responsibilities are reassigned to healthy agents, effectively increasing the robustness of the system. The use of CMSs has been proposed by several different researchers for data collection and exploration in hazardous environments [56, 57], including outer space [58]. Although the cost of fielding multiple agents or modules to form a robot may be more costly, this expense can be offset by their simplistic nature and increased efficiency/life [59]. In this context, there are two main categories currently being investigated by researchers, swarm and modular robotics.

In swarm robotics, hundreds to thousands of identical agents act together to accomplish tasks. To date, researchers have demonstrated group formation and self-organization [60, 61], leader-following [62], and several other behaviors [63]. Typically, the number of agents in these systems in chosen to strike a balance between reliability, task requirements, and cost. Since agents are relatively cheap, the preferred "repair" method is to simply discard agents from the swarm when they encounter a fault. In many cases, the health of an individual is not even tracked. Recently, this strategy has come under attack as work from Bjerknes and Winfield demonstrated that overall reliability falls with an increasing number of agents in the absence of corrective behavior [64]. Due to their simplistic nature, these robots are often not

designed to accommodate these behaviors. Therefore, the remainder of this section will address the diagnosis and repair behaviors present in modular robotic systems.

5.2.1 Modular Robotic Systems

Agents in modular robotic systems are comprised of elementary units (modules) connected together by docking mechanisms. In general, modular robots can be split into two distinct groups (i.e., homogeneous or heterogeneous) depending on their specific morphologies. Homogeneous modular robotic systems are comprised of identical modules normally arranged in lattice or chain-like structures. These systems have been used to demonstrate reconfiguration between locomotion modes [65, 66, 67], self-assembly into complex structures such as trusses or a six degree-of-freedom robot arm [68, 69, 70, 71, 72, 73], self-replication [74], and even transformations between one- and three-dimensional objects [75]. On the other hand, heterogeneous modular robots often employ homogeneous docking mechanisms and split capabilities across multiple modules [58, 76, 77]. In some of these systems, each module is itself an operational robot and the mating of several modules endows the system with additional capabilities or modes of locomotion [78, 79].

5.2.1.1 Diagnosis

Even with so much variety, the diagnosis procedures discussed in Sec. 5.1.1 can generally be applied to these systems depending on the system architecture. Moreover, new methods, such as [80], have been developed that exploit the underlying structure of homogeneous modular robots (i.e., the interconnections between modules whether mechanical, electrical, or communication-based) to represent the robot as an undirected graph. In [80], synchronized heartbeats are sent to adjacent modules to indicate fault-free behavior. If a heartbeat is not received within two cycles, the module is determined to be faulty and the location of this module is transmitted to a specified state monitor module through its healthy neighbors. As opposed to other diagnosis algorithms, this method solely depends on local, as opposed to global, information which is paramount in reducing computation in these already limited systems.

A separate algorithm was developed by Kutzer et al. in [81] to determine faults in individual heterogeneous modules of two-wheeled nonholonomic agents in a team setting. In this method, an external team member (in this case an overhead camera) observed a second team member while it executed a diagnostic maneuver. The discrete data captured for the maneuver is then fed into a PF that has been trained from both experimental and simulated data of the system when in known states. The state of the system is then diagnosed by selecting the system state or set of systems state with

the maximum joint probability. Just as both of these diagnosis methods are particular to the specific morphologies of modular robots, so are the repair procedures.

5.2.1.2 Repair

Repair in modular robotics typically falls into one of three categories. At the simplest level, repair can be accomplished by jettisoning damaged modules [65, 82, 83] and then reconfiguring. Similar to swarm robotics, modules in these systems are relatively cost effective and agents, especially in homogeneous systems, are comprised of several redundant modules. This strategy is not without limits as discarding additional modules can degrade the capabilities and functionality of individual agents after a critical threshold.

Much like the methods presented in Sec. 5.1.2, the second category of repair is adaption or fault-tolerant control. In these methods, agents adjust their control methodology to cope with fault related deficiencies. For instance, a model reference adaptive control law is used in [84] to update control gains and parameter estimates for individual modules. Each module is comprised of a single rotary actuator and the adaptation scheme and control law are derived in such a way that only data from the current module is required. In this fashion, the controller will continually update parameters to meet the commanded joint torques up to a certain threshold of actuator damage.

The third method follows the more traditional definition of repair (i.e., the removal and replacement of faulty modules with functional representatives obtained from storage or by scavenging another damaged agent) and applies solely to heterogeneous modular robots. Using reliability theory, Bereton and Kholsa proved that the aggregate mean time to failure for a team of repairable robots was significantly larger than that of a team of non-repairable robots [85]. They were also the first researchers to develop a repairable system based around seven desirable constraints for exhibiting autonomous module replacement [86]. Their system consisted of a forklift-like robot equipped with a black and white camera to execute the repair and a repairable robot containing modules with fork lift receptacles [87]. Although Bereton and Khosla presented a repair process, only certain subsystems were replaceable and important aspects, such as the locomotion system, were not addressed. Additionally, Bereton and Kholsa provided no means by which to diagnose when a module required replacement. Therefore, a need still existed for a system capable of autonomous diagnosis and repair to overcome unexpected faults and to extend functional lifetimes.

5.3 The Hexagonal Distributed Modular Robot System

The Hexagonal Distributed Modular Robot (HexDMR) system was designed to fit this need by delivering a system with easily interchangeable components and agents that are more robust against unexpected faults, adapt to environmental changes, and prolong system lifetimes. The HexDMR system was first conceptualized by Ackerman and Chirikjian in [88] as a proof-of-concept testbed to demonstrate the possibility of a more complex repair process. In this work, Ackerman and Chirikjian proposed necessary and sufficient design constraints for the development of a CMS capable of team repair by refining, combining, and adding to Bereton and Khosla's original constraints [86]. This first generation system (HexDMR I) included a single layer of six modules arranged in a hexagonal footprint. The modules were connected to a central hub that provided structural support for each module as well as 20 shared electrical connections. The manipulator module in this generation was composed of a fork-lift like mechanism with two separate actuators that interfaced with external modules to secure them to the second agent's central hub. The manipulator also contained a linear actuator to elevate the fork-lift which enabled the removal of a second agent's manipulator module. Even with this foresight, it was only possible to demonstrate a limited teleoperated repair with a nonfunctional mockup of a second

agent due to limitations in the design of agents in the system.

The second generation system (HexDMR II) sought to correct these deficiencies and allow for a completely autonomous repair process. Compared to the first generation system, the footprint of HexDMR II was approximately 66% smaller. Additionally, to increase the modularity of the system, the central hub was removed and the electrical connections were instead passed laterally around the modules. One side of a module contains and electrical circuit board that protrudes from the mating surface. The opposite side contains a recessed circuit board and two slats that prevent horizontal translation. This side of the module mates with the adjacent module's circuit board. Agents are then formed by interconnecting six modules into a hexagonal ring. Each module is secured by a sophisticated, passive two-part mechanism. The procedure for the insertion of a single module into an agent is as follows. First, the module to be inserted is lowered from above. As each side of the module mates with its conformally designed counterpart on each adjacent module, electrical connections are formed. Once the module is lowered far enough it rests on two latches that are extended from the adjacent modules. At this point the gripper on the manipulator module is retracted which simultaneously deploys the module's two internal latches. These latches mate with both adjacent modules and prevent linear motion. Rotational motion is then prevented by the conformally mated circuit boards on both sides of the module. This docking mechanism alteration introduced an additional degree-of-

freedom during the manipulation of modules as they now have to be lifted and lowered while being placed. Finally, an add-on camera attachment was constructed for the manipulator module to provide feedback during the autonomous repair process.

After constructing a prototype agent from the HexDMR II system, it became readily apparent that the electrical connections between the modules intermittently lost connectivity during motion which lead to erroneous trajectories at best and a completely unresponsive system at worst. This issue became markedly worse when trying to manipulate a module on another system as the loads were transferred to the drive modules. This connectivity issue was mainly due to small relative rotations in between the mating surfaces of adjacent modules and arose from generous tolerances that aided in removing and inserting modules in the robot during repair.

As a result, the second generation system underwent a significant design revision to incorporate more robust communication interfaces as well mechanical changes to aid in the repair process. This generation is delineated as HexDMR IIa and begins the true contribution of this dissertation. Ultimately, an autonomous docking procedure was demonstrated, however design flaws related to the mating system and the center of mass of the modules prevented removal or insertion. These design limitations necessitated another design revision and the creation of the third generation system (HexDMR III).

Agents in the HexDMR III system retained the new communication interfaces

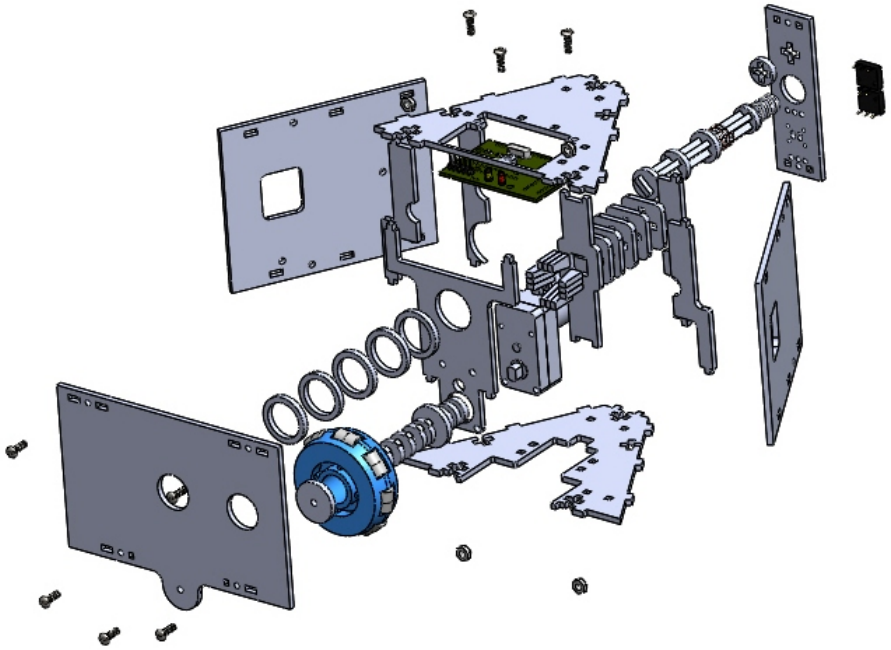


Figure 5.1: Exploded view of a module

introduced in HexDMR IIa and returned to a design with a central hub, albeit significantly different from the original HexDMR I system. In this design, only six electrical connection are shared between each module and six additional docking locations are included on a second level. This new layout enables the HexDMR III system to store additional modules to augment capabilities (e.g. provide further tractive force, extend power capacity, provide extra sensing) or to provide spare modules for use in the repair process. A computer-aided design representation of a fully deconstructed drive module of the HexDMR III system is presented in Fig. 5.1. This design revision also altered the method in which the manipulator interacted with modules. Instead of a fork-lift type design, the HexDMR III system utilizes a screw mechanism. Modules

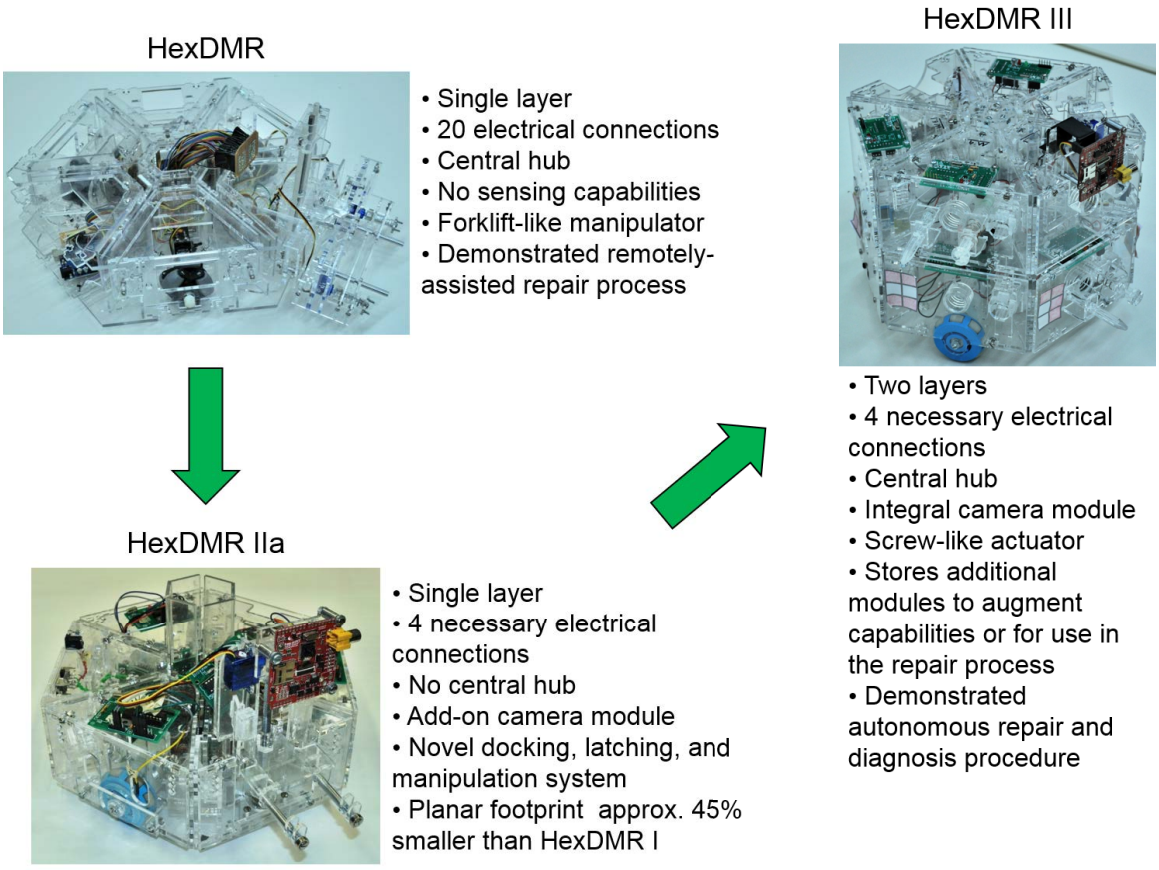


Figure 5.2: Comparison of the HexDMR systems

are now screwed into the central hub and secured by an alignment pin that fully constrains each module. Further improvements in this revision include a more powerful camera module and a new elevator module to transfer modules between levels. Finally, with these changes, a fully autonomous repair process [6] and fault diagnosis procedure were independently demonstrated.

A more general comparison of the three different generations of the HexDMR system is included in Fig. 5.2. The subsequent chapters discuss some aspects of the

second and third generation systems in greater detail. Specifically, the remainder of this work is organized as follows. In Chapter 6, the main design features of the HexDMR III system are summarized and the design of each individual module in the system is reviewed. Additionally, the non-isomorphic configurations of the HexDMR III system are enumerated and a case studying comparing the performance between three- and six-wheeled configurations is presented. In Chapter 7, a repair procedure for autonomously repairing a module in the HexDMR III system is outlined, and experiments for both inserting and extracting modules are presented. Finally, in Chapter 8, the failure modes of an agent in the HexDMR III system are reviewed. Then a two-part procedure for autonomously diagnosing faulty module(s) is introduced and validated in both simulation and experiments.

5.4 Contributions

The main contributions of this work toward cooperative robotic repair and diagnosis are as follows.

- The development of design methodology and capabilities relevant to creating repairable modular robots (with specific implementations in HexDMR IIa and HexDMR III).
- A method for enumerating functionally non-isomorphic configurations of robotic

systems.

- An autonomous repair procedure for modular robots.
- A two-part diagnosis procedure for determining underlying system faults in robotic systems consisting of qualitative measures to determine binary faults and an unscented particle filter based quantitative procedure to identify continuous failure modes.

Chapter 6

HexDMR: Design

The design process of any robotic system is typically guided by a set of requirements or constraints directly related to the purpose or functionality of that robot. The primary purpose of the HexDMR system is to demonstrate the possibility of autonomous team repair and diagnostic procedures. In fact, the first generation system (HexDMR I) was designed relative to six constraints for repair. The first three constraints (i.e., homogeneity and robustness of repair, completeness of repair, and resolution of repair) were proposed by Bererton and Kholsa in [86] and were extended by Ackerman et al. in [88] to include independence of repair, ubiquity of repair capabilities, and versatility of agents when designing the first generation of the HexDMR system. These constraints remained a driving factor during the redesigns of the second and third generation systems and resulted in common design elements, albeit

with sometimes different implementations, across all generations.

The remainder of this chapter summarizes these common design features and specifies necessary design improvements to the HexDMR II system. Additionally, the design of the HexDMR III system including all of its core modules is discussed. Then, the four base configurations of the HexDMR III system are identified and methodology is presented to generate configurational trees, based on non-isomorphic, functional agents, to determine all possible configurations of the system. Finally, an informative case study is presented that compares the kinematic performance and power consumption of two base configurations of the HexDMR III system.

6.1 Common Design Features Across Generations

The HexDMR system was designed with two goals in mind; that is the autonomous diagnosis and repair of individual agents or robots within the team. Each generation of the HexDMR system has strove to further this goal by providing incremental improvements. For instance, HexDMR I was designed as a proof-of-concept for demonstrating how modularity can lead to repairability. HexDMR II and its slightly modified counterpart HexDMR IIa were intended to demonstrate an autonomous repair process and introduce sensing capabilities. Finally, HexDMR III was designed to demon-

strate fully autonomous diagnostic and repair procedures. As this system evolved several core design elements remained constant along generations even though individual implementations may have been different. These core elements were chosen to best comply with the design constraints of repair mentioned above and to build upon previous successes. These elements can be split into two distinct groups, common features among robots and common features among modules.

6.1.1 Common Robot Features

Several features that greatly facilitated the success of the repair process were implemented across all generations of the HexDMR systems. Briefly, these features are a modular configuration for agents, a holonomic drive implemented through omnidirectionals wheels, and an overall hexagonal geometry. Each of these features is visited in more depth in the proceeding subsections.

6.1.1.1 Modular Configuration

The modular configuration adopted by all generations of the HexDMR system enabled a homogeneous repair process across all agents. That is, within each generation, each module was designed to be removed by the same procedure. In the first two generations this process included unlatching and transporting modules by a forklift-like manipulator, while in the current generation modules are screwed and unscrewed by

the manipulator. Additionally, the design of the manipulator in HexDMR III enabled manipulator modules to remove one another without additional steps, whereas in the previous generations the end-effector of the manipulator had to be removed before proceeding. Although this change slightly decreased the resolution of repair for the manipulator module in HexDMR III, it also reduced the cost and complexity of the manipulator while increasing the robustness of repair.

6.1.1.2 Holonomic Drive

To manipulate modules, a docking procedure between agents is necessary. Depending on the design of the docking system, the agents may have to perform fine-tuned maneuvers to successfully dock. Nonholonomic approaches to this issue result in a large number of corrective actions to generate small motions in constrained directions and, in general, cause difficulties [89]. Conversely, holonomic approaches allow instantaneous acceleration in any direction and orientation enabling more efficient docking procedures.

One goal in all HexDMR systems was to simultaneously reduce the number of actuators present in each module to limit complexity while maintaining the overall holonomic drive. Several drive systems satisfy these constraints; however, only two options minimized the required number of drive modules. The first consisted of two drive modules containing steerable omni-directional wheels and a third module con-

taining a passive castor for support. In this configuration, three modules and four total actuators are required. The second option, consisted of three, non-steerable, omni-directional wheels evenly spaced about the centroid of the robot. This configuration resulted in three identical modules with three total actuators radially spaced 120° apart. Clearly, the second option is preferable and is employed in all HexDMR systems as it limits complexity in the drive modules, increases overall homogeneity of the system, and improves maneuverability.

6.1.1.3 Hexagonal Geometry

When designing the HexDMR systems, great thought was spent ensuring that each module was easily accessible and replaceable in order to address the completeness of repair constraint. Convex geometries, such as rectangles or hexagons, provide collision-free paths to each module for external actuation. However, by increasing the number of modules in a set footprint, the area available for external manipulation is effectively reduced if overall scale is maintained. For example, consider a 1- by 1 unit square; if four identical modules are placed in the square and each edge is reserved for a single module then each face has unit length. If instead, a hexagon is inscribed within the same unit square then the face of each module is 0.5176 units, and furthermore, if an octagon is considered then each face is 0.4142 units. Therefore, a balance must be struck between the resolution and completeness of repair.

As stated earlier, three drive modules arranged in a radially symmetric fashion, in addition to power and control modules, are required to implement a holonomic drive system for the robot. A square footprint does not contain enough modules for simple operation, while a pentagon does not maintain proper wheel spacing. The hexagonal footprint properly arranges the drive modules for the holonomic drive and also provides more space over an octagonal footprint to perform docking procedures.

6.1.2 Common Modular Features

The HexDMR systems were primarily designed to demonstrate diagnosis and repair processes through the use of heterogeneous modules. Although these module types were diversified to achieve specific capabilities, overarching hardware and mechanical structures were maintained to reduce overall cost and increase homogeneity. Moreover, core functions such as manipulation, processing, locomotion, etc. were split across several different module types to limit the mechanical and electrical complexity of any given failure. Two features extend to every module in the HexDMR system. The first is the trapezoidal footprint that defines the perimeter of each module and the second is the docking mechanism which is essential for reconfiguring agents.

6.1.2.1 Trapezoidal Footprint

Within each generation, each module in the HexDMR system (with the exception of the manipulator module in HexDMR I and II) has the same planar trapezoidal footprint. The individual height or scale of each module may vary depending on installed components or function, but this distinction only affects possible agent configurations as opposed to overall geometry. This footprint is mainly a consequence of the hexagonal geometry of individual agents which was thoroughly discussed in [6], but was also chosen due to three additional benefits.

First, during docking, the two outer faces of adjacent modules help guide the manipulated module toward the preferred docking position. Second, the outer faces of docked modules are designed to be flush and coincident with one another which theoretically should prevent the introduction of electrical noise through small, extraneous movements of the electrical connectors during agent motion. Lastly, the outer shell of each module, which forms the trapezoidal shape, is designed to be easily removable to enable quick access and replacement of internal components with minimal work. The shell, and a majority of other mechanical components in the modules, are constructed from laser-cut acrylic to increase modularity and ease of repair.

6.1.2.2 Docking Mechanism

Repair in a heterogeneous modular system is conducted by replacing faulty modules with identical copies. Therefore, a docking mechanism is required to effectively mate a manipulator to any module within a second agent. In this manner, the manipulator can either actively or passively release the faulty module from its parent while fully supporting the module's weight. Although the docking mechanism for each generation of HexDMR system has evolved with each iteration, one feature has been maintained; every module is designed to contain only passive mating interfaces. In other words, faulty modules do not actively interact with the manipulator to extract or insert themselves.

In HexDMR I, the docking mechanism consisted of a support structure provided by the central hub and two internal latches within each module. When rotated by the external manipulator these latches locked the module in place. In this system, the manipulator was required to insert modules into the central hub within the plane. HexDMR II differed drastically from HexDMR I as it employed a passive latching system, but also required modules to be inserted perpendicular to the plane. For this generation, the central hub was removed and as a result modules within an agent were self-supporting and latched into one another. When the manipulator mated with a module it would retract the latches, then the module would be lowered from above such that a board containing electrical connections slid into a corresponding slot on

the adjacent module. Simultaneously, the corresponding slot on the inserted module slid over the other adjacent module's electrical board. The extended latches from the adjacent modules supported the inserted module and as the manipulator pulled away the internal latches deployed to secure the module. Although the docking procedure for this generation was more complicated, it had the added benefits of reducing the number of actuators in the manipulator module and removing the central hub from agents which reduced cost and increased overall homogeneity.

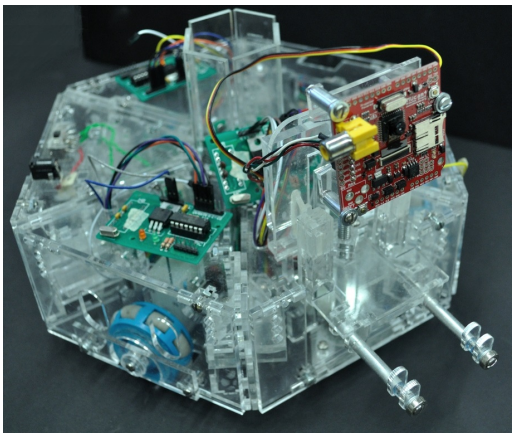
HexDMR III drew on lessons learned from both of its predecessors adopting the planar insertion process from HexDMR I and implementing a single actuator in the manipulator module similar to HexDMR II. Additionally, a minimal central hub was reintroduced. The docking mechanism for modules in the HexDMR III system was based on a screw-like mechanism. The central hub contains threaded inserts while every other module contains a central acrylic shaft with a conformally threaded "screw" which extends the length of each module. The manipulator then externally actuates the screw to mate modules to the central hub. Each module also contains a passive friction mechanism that mates with a pin on the manipulator. This pin ensures that modules are fully supported and can be removed from the central hub.

6.2 Improvements to the HexDMR II System

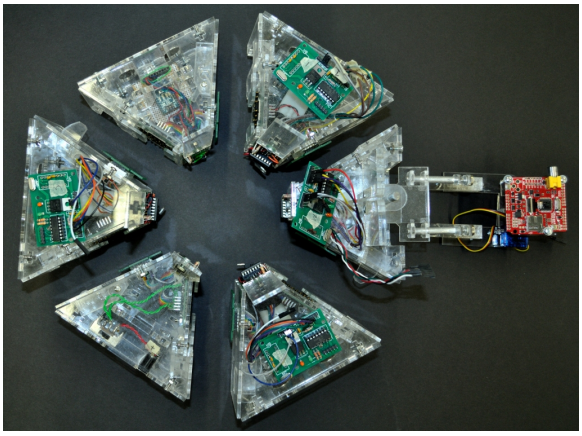
The original implementation of the HexDMR II system drew heavily on inspiration from HexDMR I while also introducing several improvements. The central hub was removed to further increase homogeneity of the system. The planar footprint of agents was reduced by 43% which reduced cost and increased maneuverability. The internal latching mechanism inside of each module was altered to be more passive enabling the removal of an actuator from the manipulator module and, most importantly, a new sensing (camera) module was added to provide feedback to the system. These improvements were supposed to enable an autonomous repair process, but unfortunately the new design also introduced undesirable motion between the electrical connections of adjacent modules. These minute movements were enough to prevent reliable connections and in some rare cases resulted in mismatched connections that led to unexpected behavior. In some cases, these motions also caused latches within the modules to unexpectedly retract resulting in reduced stability of the entire agent. Both issues made reliable use of HexDMR II unsustainable.

These issues were addressed in HexDMR IIa, a minor design revision of the originally envisioned HexDMR II. HexDMR IIa introduced a new electrical interface as well as secondary microcontrollers for all actuated modules to minimize the num-

ber of required electrical connections. Additionally, the internal latching mechanisms inside of each module were retrofit with passive components to ensure continual engagement. Figure 6.1 provides a representative example of an agent in the HexDMR IIa system, while the specifics of these improvements are discussed in detail in the following subsections.



(a) Isometric view of an agent of HexDMR IIa



(b) Exploded view of an agent of HexDMR IIa

Figure 6.1: Views of an agent of the HexDMR IIa System

6.2.1 Electrical Bus Improvements

In HexDMR II, the required electrical connections for each module were separated into separate conduits. Each agent in the system contained 20 individual electrical connections including but not limited to two connections for power, two control lines for each actuated module (8 total connections), and two data lines for the camera module. As agents in the HexDMR II system drove, it became clear that the electrical

connections were not persistent. That is, in at least one of the twelve interfaces between individual modules, connections were severed and reconnected as the robot moved. This specific issue presented itself in different manners such as loss of power, loss of control of a particular actuated module, and loss of sensing data which rendered the robot unusable.

Several solutions were considered, but ultimately a design that minimized the number of required connections for each module was selected. At a high level, a low-level PIC microcontroller, in the form of an add-on printed circuit board (PCB), was added to drive and manipulator modules to control their internal actuator. The main control module then used asynchronous serial communication (which requires only four lines) to communicate with the low-level controllers. Although this design change added additional complexity to the actuated modules, it also reduced the overall complexity of the electrical system, increased stability, and maintained the resolution of repair for individual modules. After introducing the new microcontrollers, the four remaining unused electrical connections were reassigned to provide redundancy for the serial communication. The specific implementation of this new PIC board and the corresponding communication interface changes are highlighted in the following subsections.

6.2.1.1 Implementation

As stated above, a PIC microcontroller was added to each actuator module to reduce the total number of electrical connections between modules. Several criteria were considered during the search for an appropriate microcontroller. Specifically, since the microcontroller must have bi-directional speed control over its actuator, at least two pulse-width modulation (PWM) channels were required. Additionally, the microcontroller must have the ability to communicate with asynchronous serial communication devices (typically through a universal asynchronous receiver-transmitter or UART) over baud rates compatible with the main controller (an ATMEGA168) which necessitated an external clock. Originally, using the interconnection of modules, a single crystal oscillator was shared between each actuated module as the clock. Unfortunately, due to noise and differing crystal excitations from each module this solution did not work. Instead, an add-on module was added to each PIC board. Finally, in keeping with the design constraints of repair summarized above, the microcontroller must minimize cost and complexity to limit the homogeneity and resolution of repair. A more complex chip adds unnecessary additional capabilities and is larger which further differentiates actuated modules from control or power modules and increases the resolution of repair as entire modules are replaced during faults which include the entire PIC board.

The selected microcontroller, a PIC16F1825 [90], struck a balance between these

criteria. The PIC16F1825 is an enhanced midrange 8-bit microcontroller with 14 pins enabling four channel PWM generation, external clock operation, analog-to-digital conversion (ADC), serial asynchronous communication through a UART, and in-circuit serial programming (ICSP) for rapid prototyping and code updates. The PIC was mounted on a custom PCB that included connectors for providing power, communication, ICSP programming to the PIC, and a separate connector to control the module actuator. Additionally, the internal clock on the PIC tended to be fairly inaccurate and was subject to electrical noise; therefore, a clock signal from an external 16MHz crystal oscillator was provided. Two indicator light-emitting diodes (LEDs) were also added to the board with their required passive components (capacitors and resistors). Finally, space was left on the PCB to include a quadruple half-H driver (H-bridge) for bi-directional motor control. In many cases, the existing electrical infrastructure of actuated modules already included an H-bridge and in this case these signals were treated as passthroughs on the board.

In practice, the board is supplied 7.4V through the electrical bus and the microcontroller is supplied 5V through a voltage regulator. The H-bridge is supplied regulated 5V for logic comparisons and either regulated 5V (for the manipulator module) or unregulated 7.4V (the other actuated modules) for driving the motor. Depending on the application, the H-bridge inputs are sent either a steady-state or PWM signal from the PIC to control actuator speed. The indicator LEDs are also

provided 5V. The first LED is green and has two specific functions. First, the LED indicates if the PIC board has power and second, the LED blinks if the PIC board receives an invalid serial command which is helpful when diagnosing connection or baud rate issues. The second LED is red and blinks every time a byte is received by the PIC over serial communication. An electrical schematic of the board and all of its components is provided in Fig. 6.2 while an actual populated board is presented in Fig. 6.3.

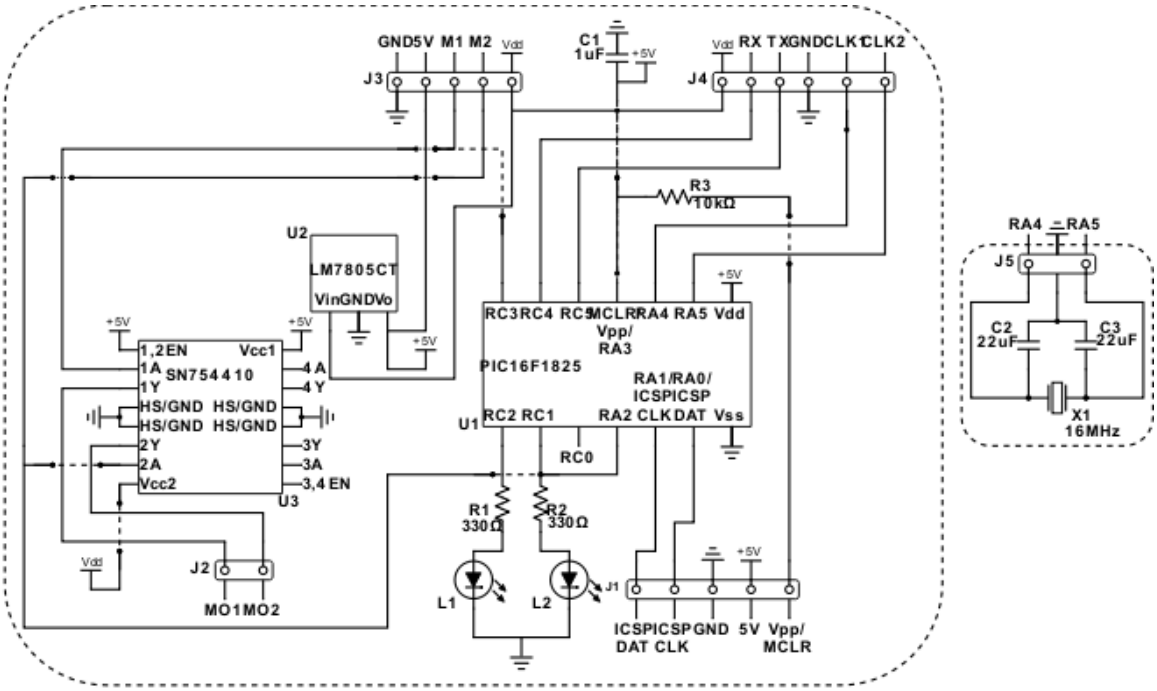
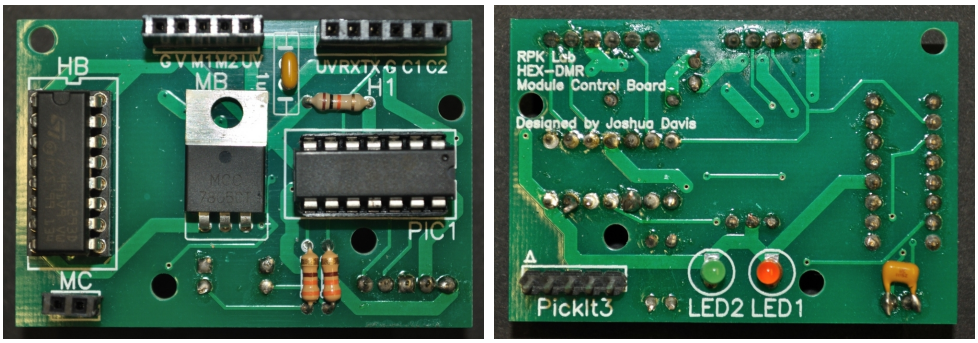
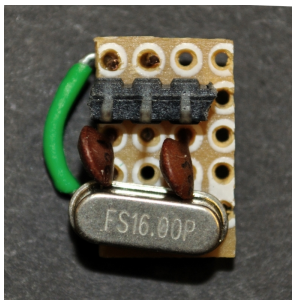


Figure 6.2: Electrical schematic for a PIC board in HexDMR IIa



(a) Top of board

(b) Bottom of board



(c) Add-on external crystal oscillator

Figure 6.3: A PIC board for an actuated module in HexDMR IIa

6.2.1.2 Communication Interface

With the addition of four new microcontrollers, the communication interface must evolve from a simple centralized control scheme to a master-slave paradigm. In other words, the control module must have full authority over the actions of the actuated and camera modules. In practice, this paradigm is achieved by connecting the transmission line of the control module to the receive line of the other modules and the receive line of the control module to the transmission line of the other modules. The other modules are programmed to only send data when they have been addressed and

are otherwise quiet. Additionally, the microcontrollers are assigned unique identifiers and only act on commands that are specifically addressed to them. The specific implementation of the data packets sent between the control module and the actuated modules is reviewed more thoroughly in Sec. 7.2.

6.2.2 Latching Improvements

In addition to the new PIC boards, a solution was sought to keep the internal latches within modules fully engaged unless acted upon by a manipulator. The simplest solution added torsion springs to keep the latches normally engaged. Care was taken when choosing the spring force to ensure that the manipulator module could exert enough force on the latches without slipping or pushing the second agent. Rectangular sheet-metal strips were also added to the latches to provided an enlarged surface area for manipulator interaction and to ensure that the manipulator remained fully engaged while lifting modules. An example of the augmented latching system on a manipulator module in the HexDMR IIa system is provided in Fig. 6.4.

6.3 Design of the HexDMR III System

Learning from the shortcomings of the previous generations, HexDMR III was designed to finally achieve fully autonomous repair and diagnosis. One of the main

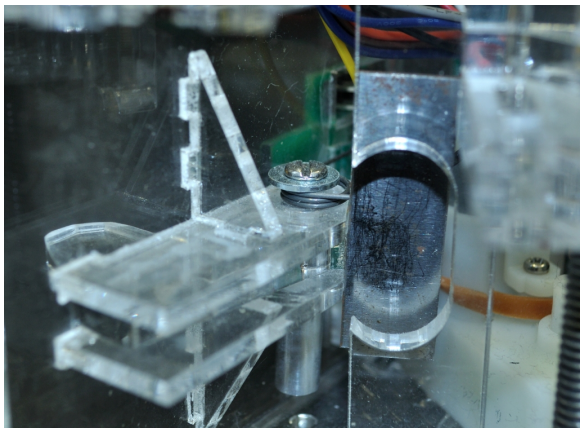


Figure 6.4: A manipulator module in the HexDMR IIa system with an augmented latching system

improvements in HexDMR III was the introduction of the two-tiered central hub. Agents are now capable of carrying twelve modules organized in two hexagonal layers of six modules. This new layout provided several benefits including attaching modules in parallel such that removing a single module did not impact the connectivity of any other module. It also enabled storage of additional, noncritical modules to provide further tractive force, extend power capacity, augment sensing, and/or improve stability. Finally, the additional docking locations in the HexDMR III system enabled the camera module to be transformed from a custom add-on installed on the manipulator module in HexDMR II to a standard module designed in the same manner as its counterparts. This change increases the homogeneity of the system and robustness of repair as now a single process is used to interact with all modules in the system.

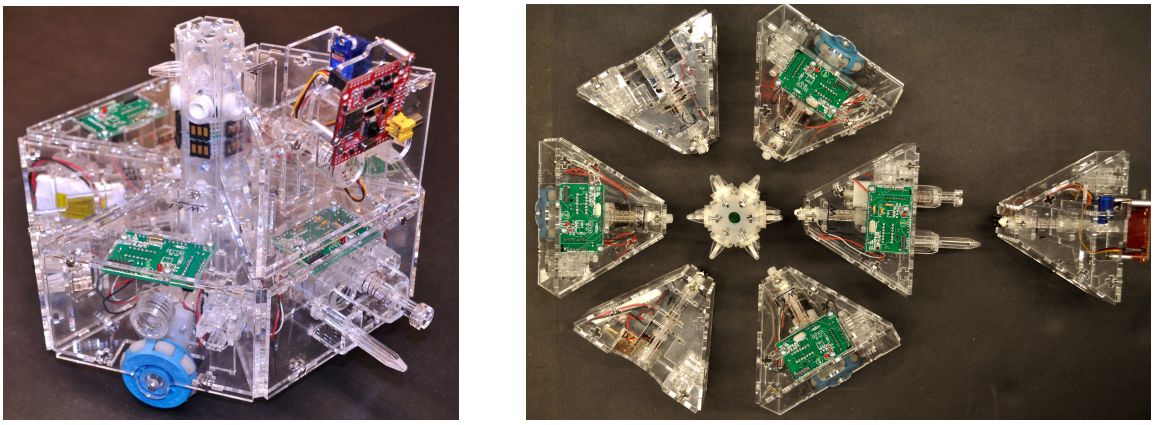
These design changes also necessitated the creation of an elevator module to trans-

Table 6.1: Overview of different module types

Module Type	**Function**	**Important Components**
Central Hub	A passive structure housing mechanical and electrical connections for modules	Electrical bus
Elevator	Enables moving modules between layers	PIC board, geared DC motor, QR reflectance sensors
Manipulator	Enables manipulation (attachment and detachment) of modules from the central hub	Current sensor, PIC board, geared DC motor for module manipulation
Drive	Provides holonomic motion for the agent	Geared DC motor, omni-directional wheel, PIC board
Camera	Provides a sensing modality for each agent	CMUcam4
Control	Handles external communication, decision-making, and sends commands to actuated modules to perform kinematic and manipulation procedures	Atmega168-20PU microcontroller, Xbee wireless radio
Power	Provides power to all other modules	Polymer Li-ion battery

fer modules between tiers during the repair process. While developing both the manipulator and elevator modules, it was determined that the PIC board needed to include an additional sensor to prevent the actuators from stalling and needlessly draining an agent's power supply. These changes and other improvements to the PIC board are outlined in Sec. 6.3.1. Additionally, a general overview of each module in HexDMR III is provided in Table 6.1 and physical characteristics of each module

are summarized in Table 6.2. Finally, Fig. 6.5 provides a representative example of an agent in the HexDMR III system. The remainder of this section is dedicated to discussing the design and features of each individual module in detail.



(a) Isometric view of an agent of HexDMR III

(b) Exploded view of an agent of HexDMR III

Figure 6.5: Views of an agent of the HexDMR III System

Table 6.2: Module types and physical characteristics

Module Type	**Maximum Dimensions**			**Mass (g)**
	L (mm)	**W (mm)**	**H (mm)**	
Central Hub	75.90	65.74	171.5	163
Elevator	116.1	132.0	185.9	537
Manipulator	179.0	132.0	82.55	257
Drive	116.1	132.0	117.8	262
Camera	125.3	132.0	116.4	234
Control	116.6	132.0	82.55	215
Power	116.1	132.0	82.55	225

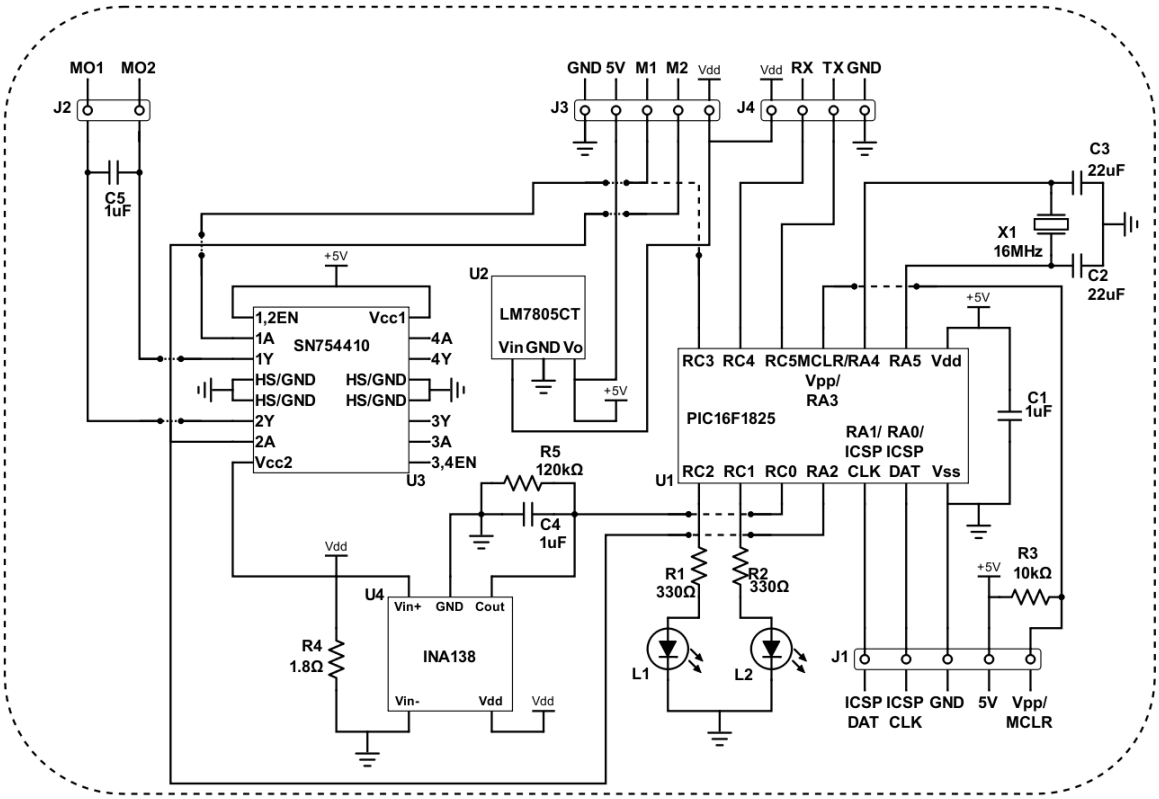
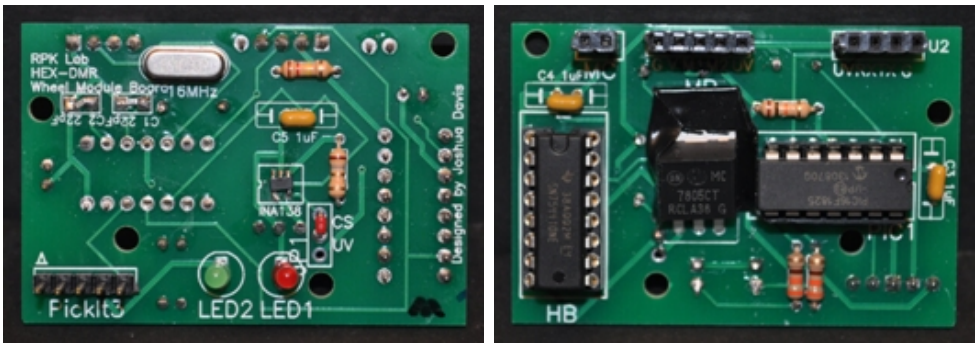


Figure 6.6: Electrical schematic for the second generation PIC board

6.3.1 Second Generation PIC Board

As discussed in Sec. 6.2.1, PIC boards capable of low-level motor control were added to actuated modules to reduced the required number of electrical connections for the system. Actuated modules in the HexDMR III system adopted this original design albeit with some slight improvements. The second generation PIC board retained the PIC16F1825 microcontroller, H-bridge, and two indicator LEDs; however, it also introduced a current sensor and a single-stage low-pass filter (i.e. ceramic capacitor) that was placed in parallel with the motor leads (the H-bridge outputs) to



(a) Top of board **(b)** Bottom of board

Figure 6.7: The second generation PIC board

reduce electrical noise in the PIC's serial communication lines. The PIC boards were also directly installed into the top of actuated modules so that the indicator LEDs were easily visible and that the leads for the ICSP were accessible. The updated electrical schematic is provided in Fig. 6.6 and the physical hardware is presented in Fig. 6.7.

6.3.2 HexDMR III Modules Types

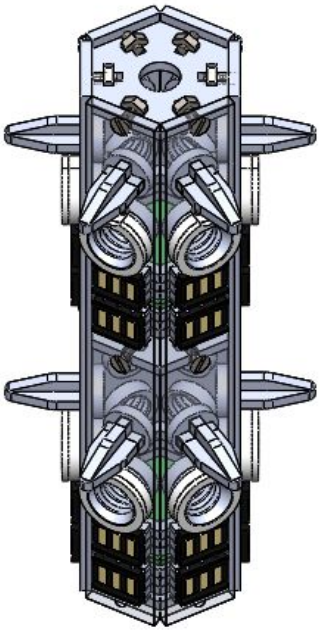
In the current iteration of the HexDMR system, there are seven different types of modules that can be assembled to form an agent. Of these seven types of modules, four are required to field a minimally functional agent (i.e., one that can drive blindly). The remaining three types of modules provide additional capabilities such as sensing and manipulation to further augment the system. Figures 6.8 and 6.9 present the computer-aided design implementations of each module, while Figs. 6.10 and 6.11

provide a physical representative of each completed module. The specific design considerations for each of the implemented modules are discussed in the following subsections.

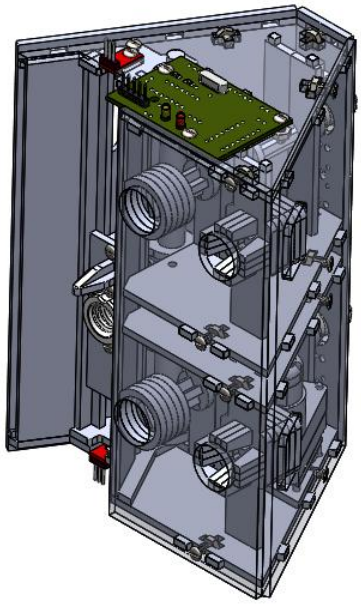
6.3.2.1 Central Hub

The shape of the central hub is a byproduct of the choice of docking mechanism and the number of modules chosen to maximize the available surface area during docking given a minimum number of required modules on the base layer. As a result, the central hub took the form of a two-layer, hexagonal column (Figs. 6.8a, 6.10a). The main function of the central hub is to provide mechanical and electrical connections to the modules. Mechanical connections are achieved by screwing the central shaft of modules into a conformally threaded insert located on the face of the central hub. The screw and inserts are coarsely threaded (3.15 threads/cm) to reduce actuation times and to conserve energy. A dual-purpose alignment pin is also present on each face of the central hub. During docking, the alignment pin helps correct small positional errors (i.e., theoretically, misalignments of up to 0.293 radians) between the screw on the end of the docking mechanism and the insert on the central hub. Upon securing the module to the hub, the alignment pin constrains one of the module's degrees of freedom and effectively prevents arbitrary rotations about the screw.

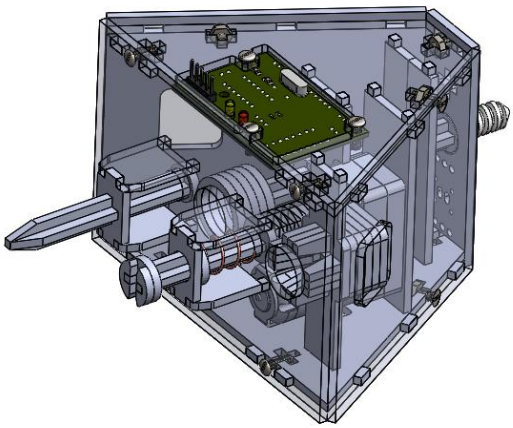
As modules are docked with the central hub, electrical connections are simulta-



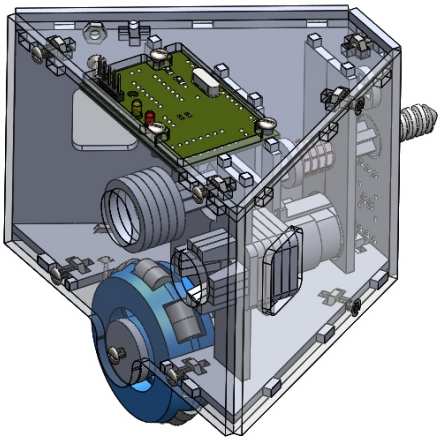
(a) CAD Central Hub



(b) CAD Elevator Module (E)

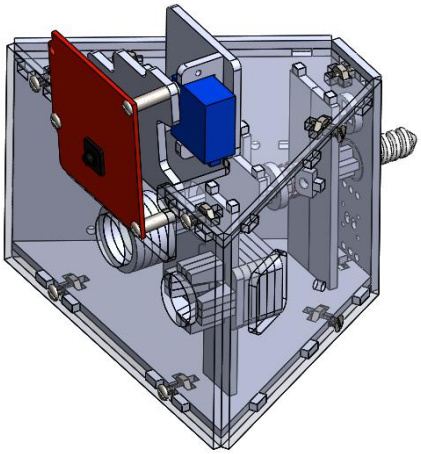


(c) CAD Manipulator Module (M)

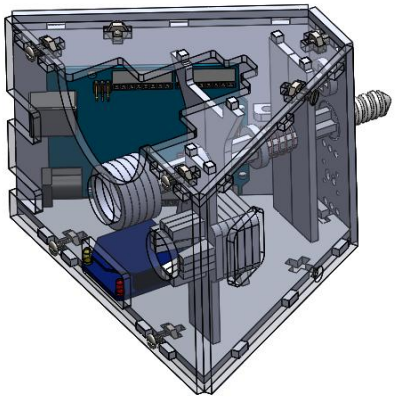


(d) CAD Drive Module (D)

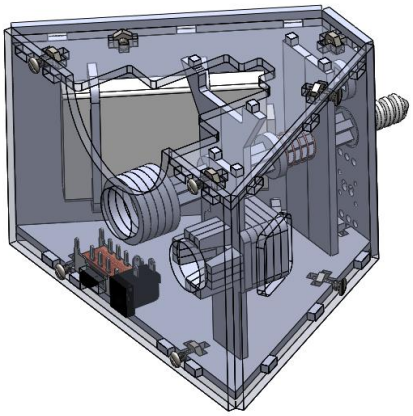
Figure 6.8: CAD representations of the types of modules for the HexDMR II system (1/2)



(a) CAD Camera Module (C_a)



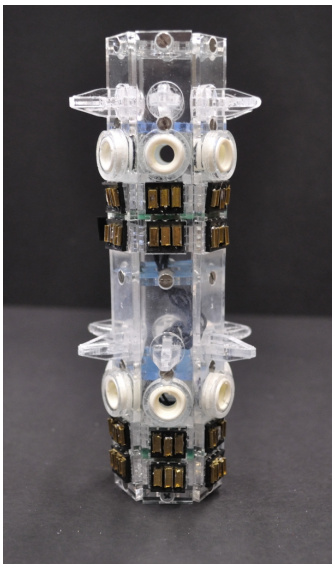
(b) CAD Control Module (C)



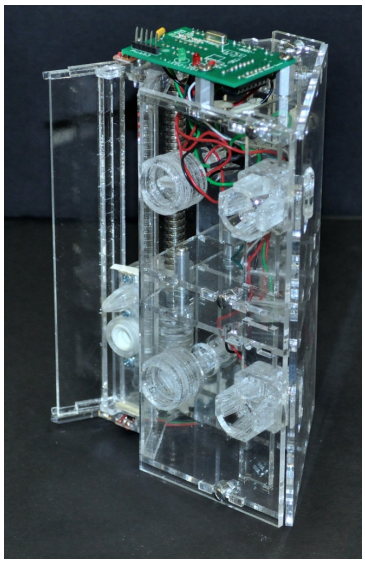
(c) CAD Power Module (P)

Figure 6.9: CAD representations of the types of modules for the HexDMR II system (2/2)

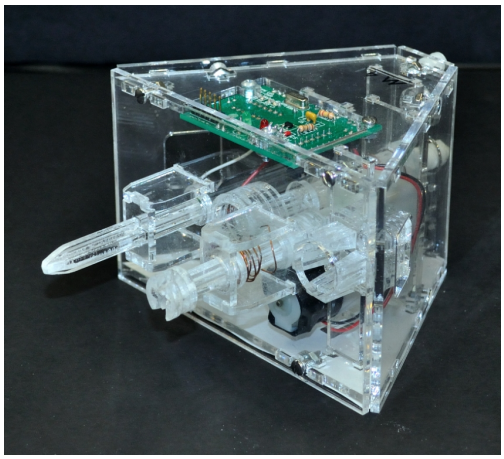
neously established. Each docking location on the central hub is outfitted with two, three-pin female connectors. As the module is screwed into the hub, the spring-loaded male connectors, positioned on the rear face of the module, are pressed into the corresponding female connector establishing an electrical connection. Each pin on the male connector has 1.5 mm of travel resulting in semi-compliant connections that



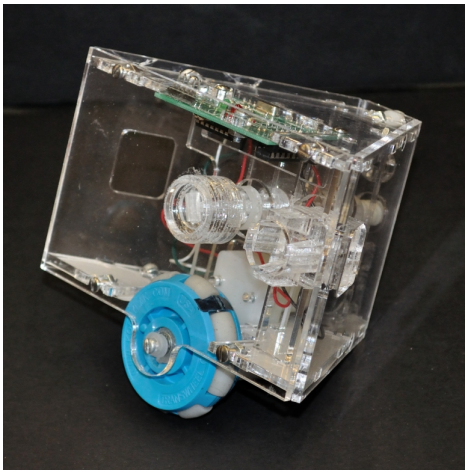
(a) Central Hub



(b) Elevator Module (E)



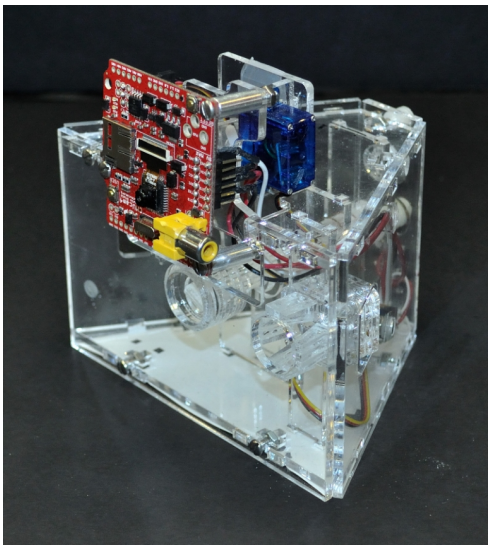
(c) Manipulator Module (M)



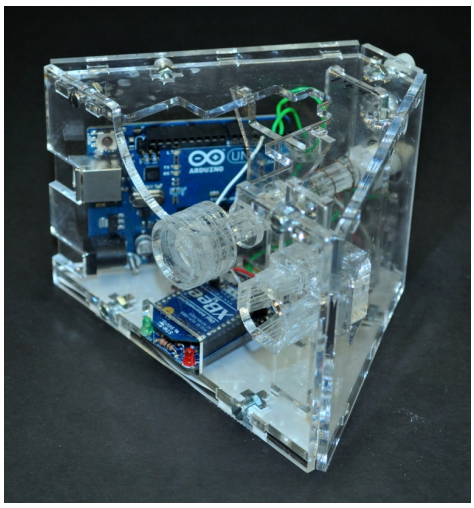
(d) Drive Module (D)

Figure 6.10: Hardware prototypes of modules for HexDMR II (1/2)

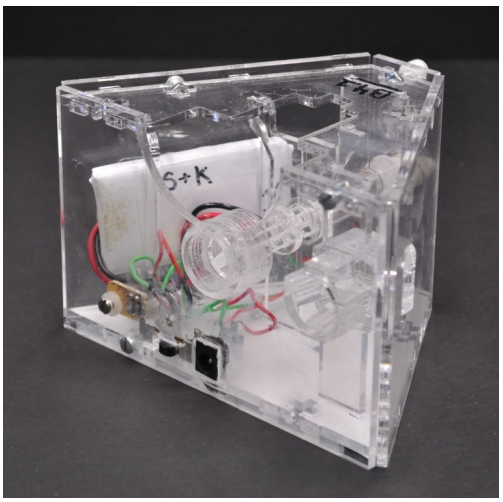
increases robustness. The female connectors on the hub are augmented with brass c-channels to further increase robustness. Connections between modules are enabled by a custom-designed PCB (Fig. 6.12) which lies in the interior of the hub and is



(a) Camera Module (C_a)



(b) Control Module (C)

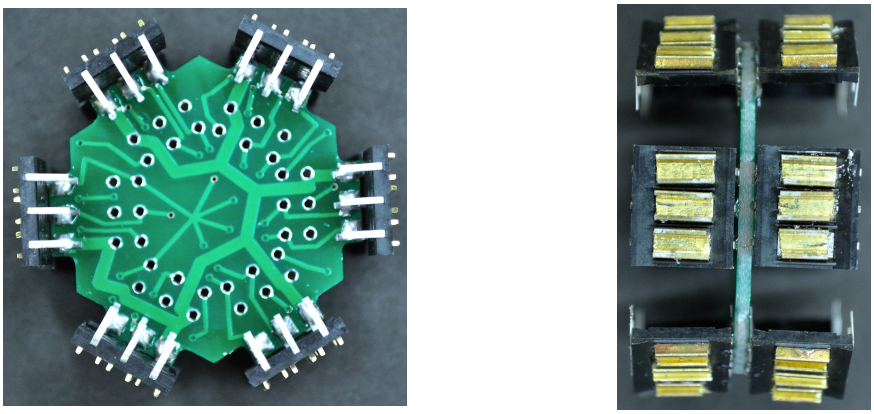


(c) Power Module (P)

Figure 6.11: Hardware prototypes of modules for HexDMR II (2/2)

soldered to each female connector. Each layer on the central hub requires its own PCB and the PCBs are connected by ribbon cable.

Although the HexDMR III system is equipped with six separate electrical con-



(a) Top View **(b)** Side View

Figure 6.12: Central hub PCB with connections

nections, only four are currently in use. Two connections are reserved for power, or the positive and negative terminals of the power module, two connections are used as transmission (TX) and reception (RX) lines for asynchronous serial communication, and the final two spare connections are for future development. These four main lines form the electrical bus for the HexDMR III system and typical connections to the bus for a seven module agent are depicted in Fig. 6.13.

6.3.2.2 Elevator Module (E)

In [4, 5], the advantages of a multi-layer system were highlighted and several theoretical options to transfer modules between levels were briefly mentioned. After careful consideration, it was determined that an elevator module best preserved the homogeneity of the HexDMR III system as well as the completeness of repair. The

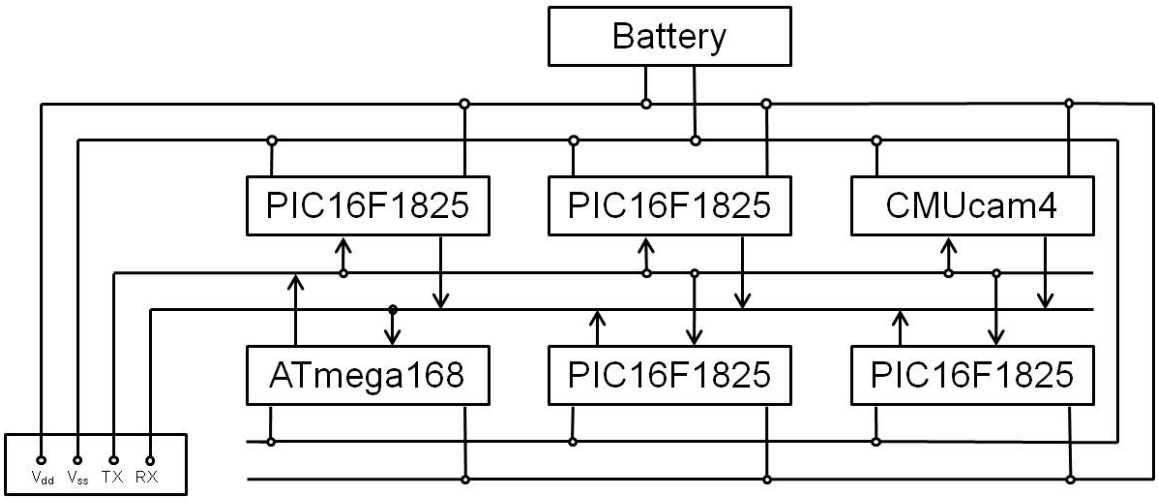


Figure 6.13: Electrical bus for a HexDMR III agent

resulting module is displayed in Figs. 6.8b, 6.10b.

Unlike the other modules, the elevator module contains two docking mechanisms and occupies both layers on the central hub. Modules are manipulated between layers by first docking a module into a carriage on the left side of the elevator that contains a threaded insert. The carriage is placed in a vertical slotted track and holds a nut that travels along a threaded Acme rod when the geared direct current (DC) motor is actuated. For homogeneity and to lower cost, the motor was chosen to be the same as those used in the drive and manipulator modules. Due to limited spacing in the module, the DC motor was connected to the Acme rod, located on the opposite side of the module, by a gear train that maintained the original torque and motor speed. The thread spacing on the rod was selected to strike a balance between the torque required to lift a module and the speed at which the module would be lifted.

Electronically, the elevator is wired to and controlled by a PIC board that is identical to those installed in the other actuated modules. In addition, the elevator module is equipped with two infrared reflectance sensors that are positioned at the distal ends of the vertical slot for the carriage. The analog outputs of the sensors are connected in parallel such that if the carriage sufficiently approaches either sensor, the signal dramatically drops and triggers the PIC board to stop the carriage.

6.3.2.3 Manipulator Module (M)

The manipulator module (Figs. 6.8c, 6.10c) is the only module capable of removing or adding modules to either the central hub or elevator. During a module extraction procedure, the manipulator module is aligned with and driven towards the second agent. As the manipulator is driven forward, a 39 mm alignment pin engages a conformal friction mechanism on the interior of the other module. The front of the alignment pin is tapered to enable successful mating with misalignment errors of up to 2.54 mm and 0.33 radians. The alignment pin also positions the end-effector in-line with the central shaft on the second module. This central shaft is supported by two vertical pieces of acrylic that extend throughout the interior of the module and contains a spring fashioned from 25 AWG copper wire that extends between a collar on the shaft and one of the vertical supports to ensure that contact is maintained with the end-effector during actuation.

The front of the end-effector is outfitted with a recessed slot that is conformal to the rectangular extrusion on the central shaft of the module when properly aligned. The end-effector also contains a copper spring along part of its length. This spring provides a normal force such that when the two shafts align they snap together. It also enables the end-effector to retreat into the manipulator as the shaft of the docking mechanism is unscrewed from the central hub. Upon successful mating, the alignment pin provides sufficient friction to ensure that the other module is removed from the central hub when the manipulator drives away. The end-effector and alignment pin are positioned in such a manner that one manipulator module can remove another.

The manipulator is controlled by a PIC board that is equipped with an additional current sensor to terminate actuation. During docking, the end-effector is stopped when the current reaches a steady plateau, above the nominal free rotation voltage, which indicates stalling or a completely docked module. During extraction, the PIC board instead waits for the current to drop to a steady-state, below a certain threshold, indicating free-rotation.

6.3.2.4 Drive Module (D)

The drive module's (Figs. 6.8d, 6.10d) main purpose is to provide locomotion for agents in the HexDMR III system. Each drive module is equipped with a 49.2 mm diameter omni-directional wheel. These wheels have eight cylindrical rubber rollers

which allow sliding along the axial direction and still maintain the "no-slip" condition in the tangential direction. Although there are 7.5 mm gaps between each roller, each wheel maintains at least one point of contact with the ground at all times. The wheels are centered and located 12.9 mm from the front of the module. When drive modules are installed on the base layer of an agent, they provide an average of 30 mm of ground clearance. Moreover, when the drive modules are docked and spaced evenly apart, they distribute the center of mass towards the geometric center of the central hub yielding higher stability. The omni-directional wheels are attached to a geared DC motor through a keyed acrylic shaft. The motors are driven through a PWM signal, specified by the control module, and generated by the microcontroller on the PIC board. The rotational speed of each motor is then adjusted by altering the duty cycle of the PWM signal.

6.3.2.5 Camera Module (C_a)

Currently, the camera module (Figs. 6.9a, 6.11a) provides the only sensing modality for the HexDMR III system. Since the main purpose of the HexDMR III system is to demonstrate a low-cost/low complexity, robust autonomous repair and diagnosis process, a CMUcam4 camera was selected. This camera provides simple color tracking and readily interfaces with the microcontroller in the control module, over serial communication, keeping control and initial development simple. The camera

is mounted to a servo motor to increase the functional field of view by tilting. The camera is located on the top of the module and extends above and forward of the normal footprint. Therefore, the camera module can only be installed on the lower layer (if a module is not present above it) or on the upper layer.

6.3.2.6 Control Module (C)

The control module (Fig. 6.9b, 6.11b) handles all decision-making for each agent in the HexDMR III system. The module is equipped with an ATmega168-20PU microcontroller for processing data and serial communication. The control module sends commands to the camera and actuated modules through the TX line on the electrical bus and receives information back on the RX line. In addition, the microcontroller is connected to an Xbee wireless radio through a software serial protocol to communicate with other agents. The Xbee radio is centered, below the central shaft, on the front of the module for easy access and removal. The top of the control module has a cutout to enable the storage of a drive module in the upper layer. Once the drive module is docked, the control module can only be removed after the drive module is removed.

6.3.2.7 Power Module (P)

The power module (Figs. 6.9c, 6.11c) is effectively the second passive element in the HexDMR III system. At least one power module is required to field a functional agent; however, additional power modules may be added to achieve longer runtimes. Power modules have two separate modes of operation which are adjustable through a switch installed on the front of the module. The switch is partially recessed such that another agent cannot activate it. The first mode of operation supplies power to the agent, while the second mode isolates the battery and enables charging. The module also contains a DC power jack on the front of the module allowing easy access for charging. Each power module is equipped with one 800mAh, 7.4V lithium-ion polymer battery. The battery is directly connected to the male electrical connections on the rear of the module. For extra precaution, and to prevent back-charging in the case of multiple batteries on an agent, a diode is placed in-line with the positive terminal. Similar to the control module, the power module has a cutout to enable storage of drive modules above it.

6.4 Non-Isomorphic Configurations of the HexDMR III System

As mentioned previously, agents in the HexDMR III system can be comprised of up to twelve modules, arranged in two vertically stacked rings of six modules. Depending on the location, number, placement, and type of module agents can be formed with vastly different capabilities. If there were no constraints on the placement of modules and it was assumed that each module only occupied one layer then there are 6^{12} or 2.18×10^{9} possible configurations. Luckily, this is not case.

In the remainder of this section, the underlying constraints for assembling functional agents from the current set of modules is discussed. Then, a method for enumerating possible agent configurations for certain base configurations is outlined and finally a case study comparing two base configurations for a specific task is presented.

6.4.1 Module Placement Constraints

There are two types of constraints that guide the placement of modules. The first is a functional constraint and relates to the number of a certain type of module required for a minimally functional agent. The second is an interference constraint and relates to the geometry of surrounding modules which encroach into separate locations. The functional constraints for the HexDMR III system are as follows:

- Control module; Qty (1)
- Power module; Qty (1)
- Drive modules placed evenly apart on the base layer in a radially symmetric fashion; Qty (3)

and the interference constraints are:

- Drive modules on the upper layer may only be placed above control or power modules.
- Camera modules may be placed on the lower layer only if another module is not present above it.
- Elevator modules must occupy both layers.

6.4.2 Base Configurations

Identifying all possible agent configurations can be a tedious and convoluted task. Separate studies, such as [91], provided a mathematical construct to enumerate all unique, non-isomorphic configurations of modular robotic systems. However, due to constraints on the placement of modules and the heterogeneity of this system a slightly different approach must be followed. Therefore, the task of identifying all non-isomorphic functional configurations was broken down into more manageable base configurations. A base configuration is defined as a set of drive modules positioned on the bottom layer of an agent. In a sense, drive modules can be thought of as a limiting case; at least three modules are necessary to achieve a holonomic drive and a maximum of six modules greatly constrains the remaining configurations.

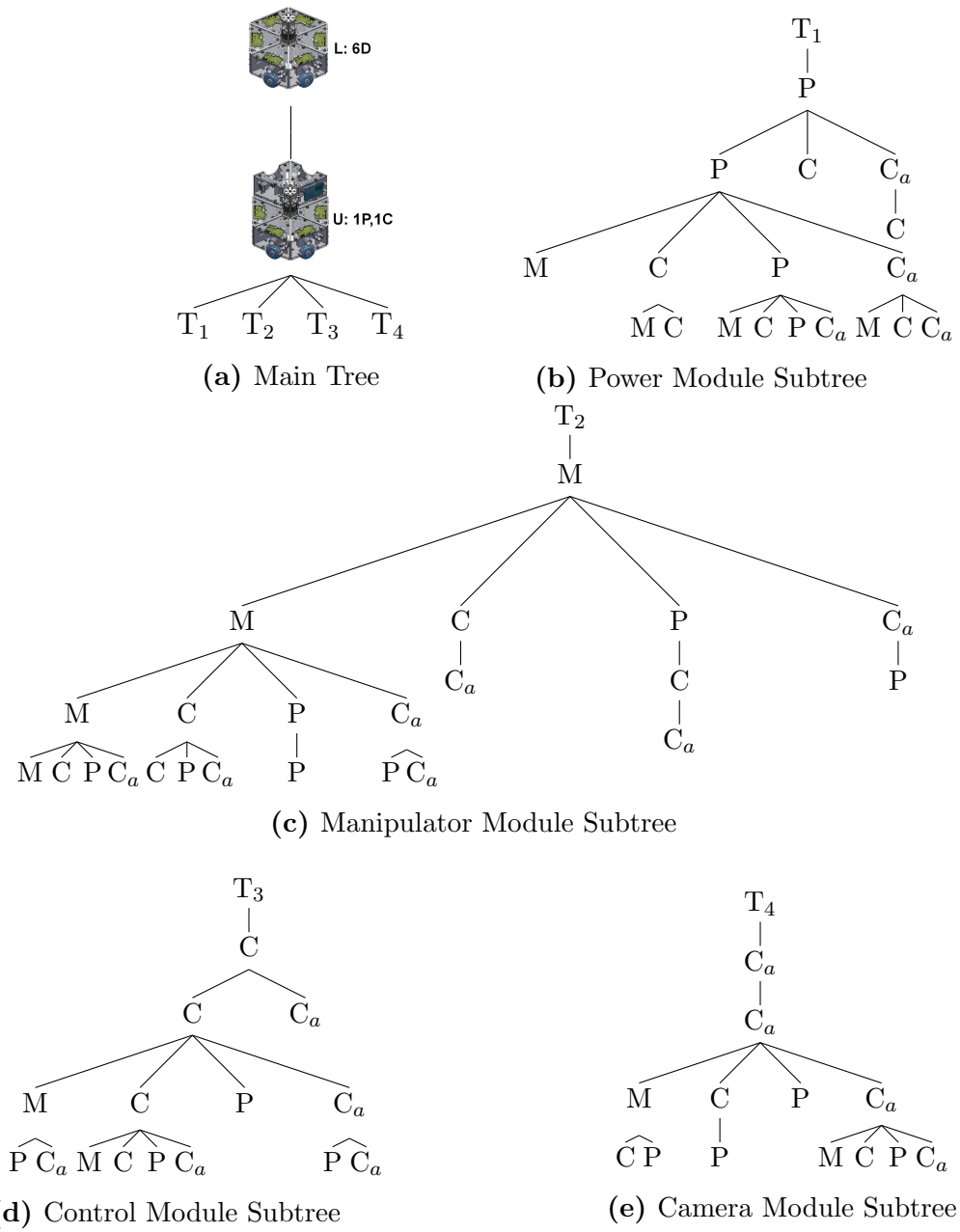
For each base configuration, a tree was constructed that details the type and location of subsequently added modules. The process begins by adding either control and/or power modules to the root of the tree (the base configuration). This branch represents the first minimally functional agent and then subsequent configurations can be enumerated. Due to the radially symmetric nature of the HexDMR III system, module permutations on single layers were excluded from the count of unique agent configurations. The collection of trees for each base configuration forms a forest and clearly establishes all possible agent configurations. For this particular system, the six drive module base configuration yields the smallest number of overall agent configurations and the corresponding configuration tree is presented in Fig. 6.14. In this figure, T_i represents a placeholder between the main tree and subtrees and M, P, C, and C_a represent modules types as described in the previous section. The first functional configuration is located at the first branch in Fig. 6.17a and each subsequent branch provides a new, unique configuration. The remaining configurational trees are located in Appendix A. In total, the HexDMR III system has 10,503 possible unique configurations. The specific breakdown per base configuration is provided in Tab. 6.3.

Table 6.3: Number of agent configurations per base configuration (or the number of drive modules on the lower level)

Base Configuration	Number of Configurations
3	7374
4	2467
5	592
6	70

6.5 HexDMR III Configurational Case Study

With 10,503 possible agent configurations, choosing the proper configuration for a specific mission can be difficult. To give some insight into this process, a case study is presented comparing a three-wheeled agent to a six-wheeled agent for a long-duration mapping task using a camera module. In this particular case, it is assumed that only one agent is necessary and the primary failure mode is loss of power. Therefore, each agent is comprised of one control module, one camera module, and the proper number of drive modules. The remaining locations are filled with power modules. In this case study, the theoretical kinematic performance and power consumption of each agent will be examined as they execute a given trajectory.



(a) Main Tree

(b) Power Module Subtree

(c) Manipulator Module Subtree

(d) Control Module Subtree

(e) Camera Module Subtree

Figure 6.14: Configurational tree for the six-wheeled base configuration

6.5.1 Configuration-based Kinematics

One of the key design elements of the HexDMR III system is that each agent can achieve holonomic motion in each of the base configurations. In fact, Indiveri presented a generalized method in [92] to derive the kinematics of a N-wheeled robot with omni-directional wheels arranged in an arbitrary configuration assuming perfect rolling. Following this procedure, the kinematic equations of motion for the three-wheeled configuration (W1, W3, W5) are derived according to the module orientations pictured in Fig. 6.15.

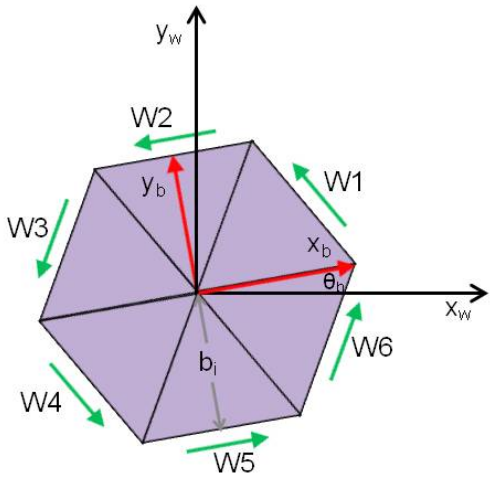


Figure 6.15: Coordinate reference frame for kinematics

$$\begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{\theta}_w \end{bmatrix} = \begin{bmatrix} \cos(\theta_w) & -\sin(\theta_w) & 0 \\ \sin(\theta_w) & \cos(\theta_w) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} \tag{6.1a}$$

$$\begin{bmatrix} r_1\dot{W}_1 \\ r_3\dot{W}_3 \\ r_5\dot{W}_5 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} & \frac{\sqrt{3}}{2} & b_1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & b_3 \\ 1 & 0 & b_5 \end{bmatrix} \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} = T_3 \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} \tag{6.1b}$$

where Eqn. (6.1a) transforms body velocities into world-frame velocities via a rotation matrix with an axis of rotation out-of-the-plane, and r_i, b_i, and $\dot{W}_i$ from Eqn. (6.1b) are respectively the radius, distance from the wheel to the center of rotation, and velocity of the ith wheel. The kinematic equations of motions for the six-wheeled agent are

$$\begin{bmatrix} r_1\dot{W}_1 \\ r_2\dot{W}_2 \\ r_3\dot{W}_3 \\ r_4\dot{W}_4 \\ r_5\dot{W}_5 \\ r_6\dot{W}_6 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} & \frac{\sqrt{3}}{2} & b_1 \\ -1 & 0 & b_2 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & b_3 \\ \frac{1}{2} & -\frac{\sqrt{3}}{2} & b_4 \\ 1 & 0 & b_5 \\ \frac{1}{2} & \frac{\sqrt{3}}{2} & b_6 \end{bmatrix} \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} = T_6 \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} \tag{6.2}$$

Note that $T_6 \in \mathbb{R}^{6\times 3}$ is rank 3 (except for degenerate choices of $b_1, b_2 \ldots, b_6$), and thus the wheel velocities must live in the range of T_6 to avoid slipping, i.e. the wheel

velocities are over-constrained.

Simulation can provide some insight into the kinematic performance of each base configuration by using Eqns. (6.1) and (6.2) to produce the proper wheel velocities to track an arbitrary trajectory (Fig. 6.16) on flat, ideal terrain. The associated wheel velocities or inputs for each wheel are presented in Fig. 6.17.

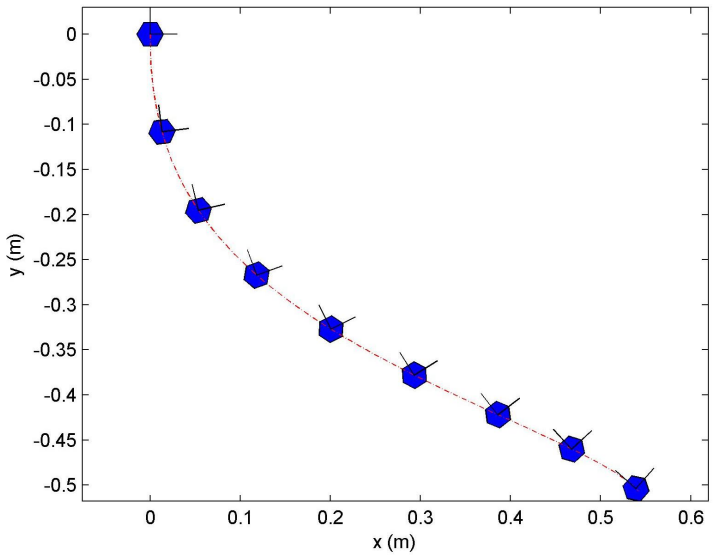
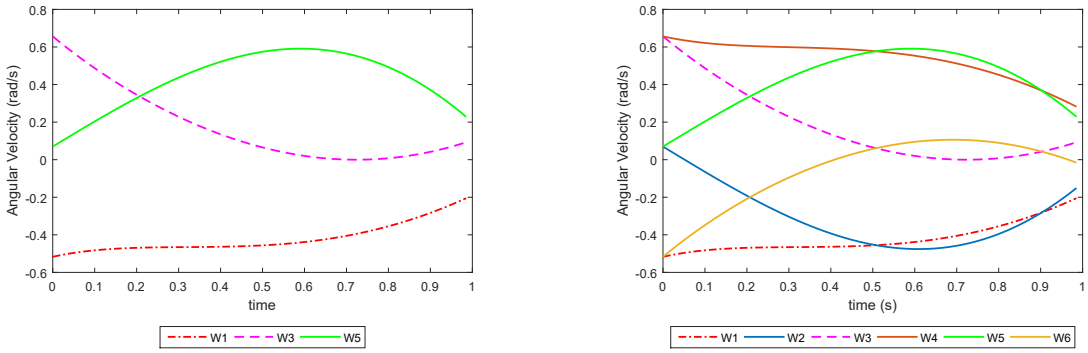


Figure 6.16: Trajectory for a HexDMR III agent

From the results, it is clear that the wheel velocities in the three-wheeled and six-wheeled case are identical. Moreover, this trajectory was generated using the same agent velocity which indicates that the number of drive modules does not contribute to agent speed. Therefore, the benefit of a six-wheeled configuration is mainly due to the traction force provided by each additional wheel. Again, assuming ideal terrain and perfect conditions, a six-wheeled agent can drive up a hill that is twice as steep



(a) Angular velocities for a three-wheeled agent

(b) Angular velocities for a six-wheeled agent

Figure 6.17: Wheel angular velocity inputs for the given trajectory

compared to its three-wheeled counterpart. Furthermore, additional drive modules increase the number of contact points with the ground simultaneously improving the stability of the agent at the cost of increasing overall friction and power consumption. Assuming more realistic conditions, the six-wheeled configuration offers one final advantage. Consider a situation when the agent loses traction (gets stuck) in a particular location. Although the three-wheeled and six-wheeled configurations can perform the same holonomic maneuvers, the six-wheeled configuration can realize approximately octuple the amount of random wheel motions (including paired wheel motions) to try to free itself, effectively improving its probability of mission success.

Table 6.4: Maximum tractive force of a three-wheeled and six-wheeled agent

	Tractive Force (N)		
Type of Surface	**Three-Wheel Configuration**	**Six-Wheel Configuration**	Ratio
Concrete	6.72	11.04	1.64
Tile	7.20	11.68	1.62
Wood	7.20	11.60	1.61

6.5.2 Experimental Validation

To test the claims related to the maximum tractive force and the angle of incline that each agent can climb two separate experiments were conducted. The first measured the maximum tractive force of each agent by attaching a linear, spring-loaded force gauge to each central hub and then commanding each agent to drive forward (at the same speed) on a flat, level surface. This process was repeated five times, on three separate surfaces, and the average force generated by each agent, as well as the ratio of forces between the two agents, were recorded in Table. 6.4. Additionally, a picture of the experimental setup is included in Fig. 6.18.

Averaging these three ratio data points, it is clear that the six-wheeled agent has approximately 1.63 times more tractive force than the three-wheeled agent. There are two possible factors that contributed to this value being smaller than the expected value of 2. First, during testing, the six-wheeled configuration slipped on occasion which limited the maximum tractive force of the agent. This slipping was most likely caused by poor contact between the ground and the wheels. Since the six-wheeled

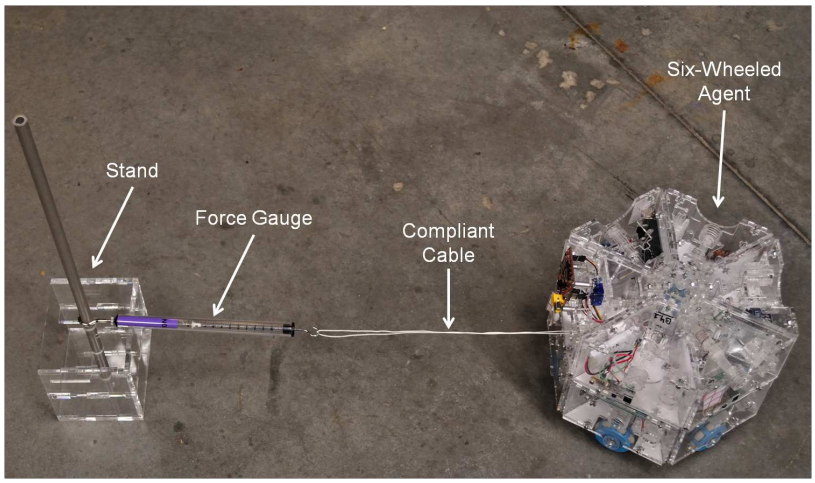


Figure 6.18: Experimental setup for determining tractive force

agent has six points of contact and only three points are necessary to define a plane, if one of the wheels is slightly misaligned it will not contact the ground. Also, assuming that every wheel is in contact with the ground then each wheel is subject to a different normal force due to the non-symmetric mass distribution of the robot. The second cause can be attributed to internal friction in the drive modules which is mainly due to the connection between the drive shaft and the front acrylic panel of the module.

The second experiment sought to quantify and validate the claim that a six-wheeled agent can climb steeper terrain than a three-wheeled agent. To do so, both agents were commanded to climb an inclined platform multiple times. A successful trial, at a specific incline, was marked by the agent being able to move forward after starting at a complete standstill. After each trial, the incline was increased and the experiment was repeated until the agent could no longer move forward. As expected the results indicated that the six-wheeled agent could climb an incline 6° or 1.67 times

steeper than the three-wheeled agent. Specifically, the six-wheeled agent climbed a slope of 14° 19' 12" and the three-wheeled agent climbed a slope of 8° 31' 48".

Due to a lack of completed battery modules, the three-wheeled agent was configured slightly differently than described in Sec. 6.5 (i.e., the agent contained three battery modules, two manipulator modules, and three control modules). In fact, the configured agent was 44 grams more massive than the described agent and this additional weight implies that the three-wheeled agent can climb a slightly steeper slope (possibly up to a half a degree more). Additionally, although this test was executed on a single surface, the basic claim that the six-wheeled agent can climb steeper terrain than the three-wheeled agent should generalize to other solid surfaces with similar coefficients of friction.

6.5.3 Power Consumption

Although the six-wheeled configuration appears to achieve better kinematic performance, the resulting power consumption must still be taken into account. Power consumption, or more accurately current draw (assuming constant voltage), of each module type was experimentally measured for an operational agent. The power module was assumed to provide no contribution, while the camera and control modules were assumed to have a constant draw. The current draw of the drive modules was measured at eight different angular velocities as well as in an idle state. A second-

order polynomial was fit to the the measurements to generate current draw across all possible angular velocities. Using these data points, the current draw over the trajectory was simulated for both of the base configurations.

As expected, the simulation revealed that the six-wheeled agent drew twice the amount of current. Arbitrarily assuming that this trajectory took one minute to complete, the three-wheeled agent could complete approximately 375 iterations and the six-wheeled configuration could only complete approximately 121 iterations.

Thus, for this particular mission there is a very obvious trade-off between the two configurations. The three-wheeled configuration is capable of operating approximately 3.1 times as long as the six-wheeled configuration. However, the six-wheeled configuration can traverse steeper terrain, achieve better stability, and possibly overcome partial loss of traction more easily. Ideally, each agent would be reconfigured to maximize efficiency and minimize power consumption over the course of the mission.

6.6 Conclusions

Several generations of agents from the HexDMR system have been developed, each stemming from the first archetypal example HexDMR I. In each generation, slight design improvements and experimental features have been cultivated to finally reach a system capable of autonomous team repair and diagnosis. HexDMR II was introduced

to provide sensing capabilities to enable the autonomous repair process; however, unforeseen consequences of the new module design led to unreliable agents. Many of these issues were rectified in a minor design revision which introduced HexDMR IIa. In this system, the electrical interface was reduced from twenty to four active connections and the communication interface, comprised of binary electrical signals, was replaced by a simple serial protocol by the introduction of an 8-bit microcontroller on a custom developed PCB called the PIC board. Additionally, the latching system on HexDMR II was improved in HexDMR IIa by retrofitting springs and by adding larger contact points onto the existing latching system. These additions ensured that the latches were always engaged unless acted upon by the manipulator.

Even with these improvements, HexDMR IIa failed to demonstrate an autonomous repair process due to other design deficiencies that will be discussed in Sec. 7.1.1. As a result, the docking mechanism as well as several modules were completely redesigned in HexDMR III to provide a more robust and reliable experience. Specifically, a two-tiered central hub capable of hosting six modules per layer was introduced. Each docking location in the central hub contains six shared electrical connections and a threaded insert where a conformally threaded screw on each module can be secured. This new layout provides additional locations for spare modules to be stored or to customize agents to improve stability, augment sensing, and increase power capacity. These extra slots also allowed the add-on camera module in HexDMR IIa to

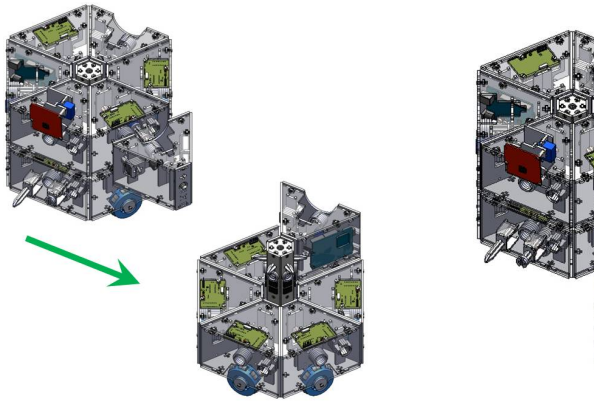
be transitioned to a full standalone module in HexDMR III which can be mounted modularly throughout the structure. Finally, a new elevator module was created to transfer modules between layers. This new module and in fact all actuated modules were designed with a slightly improved, second generation, PIC board.

With these improvements, the HexDMR system was finally able fulfill its purpose by demonstrating autonomous team repair (Chapter. 7) and diagnosis (Chapter 8) procedures. Moreover, this new design greatly expanded the possible configuration of an individual agent. In particular, a methodology was presented to enumerate all functional, non-isomorphic configurations of the HexDMR III system. In total, there are four base configurations that support 10,503 unique configurations. To gain some insight into these configurations a case study was provided that discussed the trade-offs between two such base configurations for a given mission. Ultimately, the reconfigurable and repairable nature of the system should be utilized to best satisfy mission requirements and environmental demands.

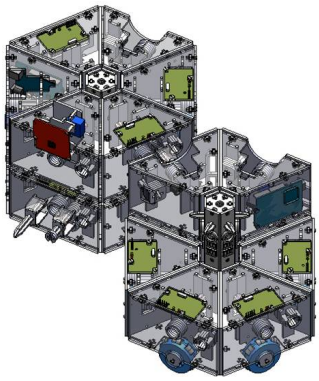
Chapter 7

HexDMR:Repair

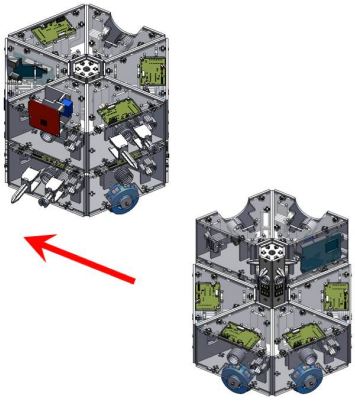
Currently, repair in robotic systems is most often conducted by human intermediaries. In rare cases, autonomous repair procedures have been demonstrated by robotic systems in very limited capacities. Commonly, these procedures involve building levels of redundancy into the robotic system and discarding damaged modules when they cease to function. Over time this strategy can result in a loss of responsiveness or even actionable capabilities. This problem is further compounded when these systems are deployed to remote or dangerous work environments where outside intervention is all but impossible or prohibitively expensive. In order to address this issue, a team of heterogeneous modular robotic agents was designed according to the constraints presented in [88]. Repair, in this system, is achieved by removing damaged modules and replacing them with healthy modules much like the process presented in Fig. 7.1.



(a) Driving forward to dock a module



(b) Attaching a module to the central hub



(c) Driving backward to continue previous task

Figure 7.1: Graphical representation of the module insertion process

This chapter applies the designs developed in Chapter 6 to an autonomous repair process. First, the evolution of repair in the HexDMR systems is outlined and deficiencies among individual generations are discussed. Then a strategy to identify actuated modules as well as their location in an agent is described. This strategy is paramount in maintaining the functionality of any individual agent. Finally, repair

procedures for both inserting and extracting modules in the HexDMR system are introduced and experiments are presented to validate the entire process.

7.1 Teleoperated Repair in the HexDMR IIa System

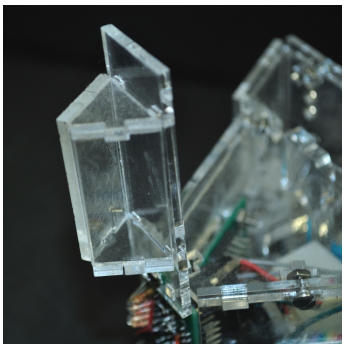
Teleoperated repair procedures are typical attempted before their autonomous counterparts to verify that repair is in fact possible. In certain cases, for instance the HexDMR I system, the design was focused on a proof-of-concept and there was no intention to exhibit autonomous repair procedures. Therefore, the first generation truly intended to demonstrate an autonomous repair process was HexDMR II. Regrettably, it became evident during initial teleoperated testing that improvements to the electrical connections between modules were necessary before the system could even be deemed usable. After the changes outlined in Sec. 6.2, teleoperated testing began on the improved HexDMR IIa system. The robots now drove reliably and fully utilized the holonomic drive capability of the design; however, new problems arose upon testing the extraction and insertion procedures.

7.1.1 Repair Related Design Deficiencies

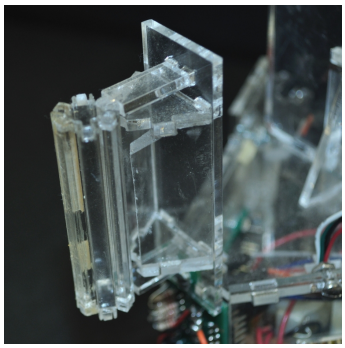
Although the robot-conducting-repair (RCR) successfully docked with the robot-in-fault (RIF) without issue, the RCR could neither place a new module or remove an existing module due to one major design deficiency. Specifically, the center of mass in a majority of the modules is not collocated with the lifting point but rather offset further towards the centroid of the robot. This offset, coupled with design of the end-effector on the manipulator module, causes the lifted module to rotate about its lift point which in turn creates contact between the mating interfaces of the adjacent modules. In particular, the corners of the silicon boards that contain the electrical connections between modules dig into the corresponding acrylic guides on the opposing mating surface. In fact, this contact effectively binds the module to the RIF during extraction and as the module is lifted it exerts enough force to lift the opposite side of the RCR off of the ground. During insertion, the friction from binding is large enough to unset the module from the end-effector of the manipulator. The manipulator is unable to detect this change and continues descending until it bottoms out on the module. As the end-effector continues to move even further downward, it begins to lift the RIF off of the ground. The end result of both repair procedures is failure.

Several design modifications, in different combinations, were employed to try to the alleviate the issue. First, a teflon-based grease was applied to the faces of the

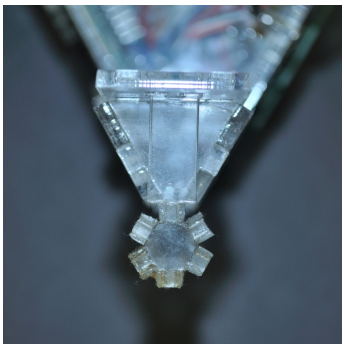
acrylic guides to reduce friction between the silicon board and acrylic. Then, the edges of the silicon boards were sanded to round the corners. Finally, a trapezoidal acrylic structure pictured in Fig. 7.2a was added to the rear of the back panel of each module to generate a normalization force when inserting or extracting modules. For instance, when a module is being inserted and begins pivoting about the lift point, the normal force from contact provides a corrective moment to keep the module almost completely vertical. This structure, by itself, relies on the force generated from adjacent faces which in some cases may not be enough; therefore, each manipulator module contains additional structure as well as the trapezoidal feature to provide a normal force directly opposing the rotation generated when inserting or extracting a module. Three views of this structure are provided in Fig. 7.2.



(a) The generic normal force generating structure



(b) Isometric view of the manipulator module normal force generating structure



(c) Top view of the manipulator module normal force generating structure

Figure 7.2: Normal force generating structures installed at the top of modules in the HexDMR IIa system to aid in the repair process

Ultimately, although these design modifications achieved better results during

teleoperated insertion and extraction, the process was still unsuccessful. The knowledge gained from these failures directly influenced the design of the robots in the third generation system.

7.2 Autonomous Module Identification with HexDMR III

One of the major design constraints presented in [88] was homogeneity or the process in which similar or preferably identical interfaces are maintained across agents and even modules. Although the previous chapter spoke to homogeneity in the sense of hardware, the same design concept applies to software as well. With the introduction of the PIC board, there are now two separate microcontrollers present in robots in the HexDMR system. Therefore, in order to obey the homogeneity constraint, the software installed on each type of controller should be as identical as possible. In the case of the control modules, identical software was employed to demonstrate the repair procedure that is described in detail in Sec. 7.3; however, the software on the microcontrollers of the PIC boards was slightly different. Specifically, the only difference in software between each PIC board is a unique 8-bit identifier or address which enables the control module to send commands to actuate specific PIC boards. These commands results in activation of the end-effector in the manipulator, motion

of the carriage on the elevator module, or the rotation of a wheel. This difference is actually crucial in order to realize a holonomic drive for robots in the HexDMR system.

As stated before, each PIC board requires four connections for functionality (two for power and two for communication). Since there is only a single channel for communication, a quasi master-slave paradigm was employed where the control module sends commands to the PIC boards as well as the camera and only the camera is allowed to stream data back to the control module. In limited situations, the other actuated modules may send data back after a specific request was executed by the control module. In this manner, no two communication devices are competing over the same line which would result in nonsensical data being received. Commands sent to the PIC boards are packaged in simple packets. Each packet contains four bytes; the first byte specifies which recipient should act on the packet, the second byte is a character representing a command, the third byte is a parameter related to the command (if the command does not require a parameter this value is set to 0), and the final byte is an 8-bit cyclic-redundancy-check to ensure that the received packet was not corrupted. A graphical representation of a single packet can be seen in Figure 7.3.

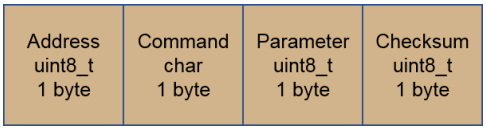


Figure 7.3: PIC Command Packet Structure

As a result of these design decisions and the fact that drive modules can be indiscriminately replaced a method had to be developed to correctly identify each drive module with respect to a certain frame in order to achieve a holonomic drive system. Additionally, this method may be used to identify the new address of any actuated module that has been replaced. To begin, this method assumes that the locations and addresses of each actuated module are known on creation or construction of the agent. Once a module has been removed, but not yet replaced during the repair process, the control module sends a universal command to the PIC modules that requests their address. To avoid over utilization of the communication line, the response time from each agent is staggered. Specifically, each module delays its response by an integer multiple of five milliseconds multiplied by an integer representation of its own address. That is, if one module's integer-valued address is 10, then the module waits 50 milliseconds before responding. This 5 millisecond window per module ensures that only one module is sending data over the line at a time. The response itself contains an address and is formatted according to the packet described in Fig. 7.3. If the set of addresses has changed from the last query, the control module removes the missing address from its internal list of modules. The same process is repeated once a new module has been inserted and the control module again updates its internal list. This enables the module to remap the input commands it sends to maintain the original functionality of the agent. One drawback related to this method is that it assumes

that the sole control module in an agent has not been replaced. To overcome this issue a more advanced method is briefly outlined in the Chapter 8 that can relink the addresses and locations of drive modules on the base layer of the agent as long as camera data is present.

7.3 Autonomous Repair in the HexDMR III System

One of the main considerations in the design of the HexDMR system was autonomous repair. As discussed earlier, HexDMR I and IIa were able to separately demonstrate limited teleoperated capabilities, but it was HexDMR III where a fully autonomous repair procedure was first demonstrated.

7.3.1 System Configuration

Agents in the HexDMR III system are controlled using an using open-loop kinematic model (i.e., commands are sent to each drive module and it is expected that these modules behave properly). This design consideration resulted from two specific constraints: minimizing complexity of individual modules and minimizing the cost of the system. However, with the addition of the camera module, feedback can be

injected into the open-loop control kinematics to provide better performance. This feedback is especially important when trying to dock with a module to conduct a repair. As stated in Sec. 6.3.2.5, the camera module contains a CMUcam4 camera which can track single colored objects in its field-of-view. The tracking information is limited to the pixel coordinates of two corners of a rectangular bounding box surrounding the tracked color, the percentage of pixels inside the bounding box that are the tracked color, and the centroid, or the average of pixel coordinates of the tracked pixels in the bounding box. Therefore, to aid in providing feedback during the repair process the top of the central hub of each agent was equipped with alternating rectangles of distinctly colored paper that can easily be tracked by the CMUcam4.

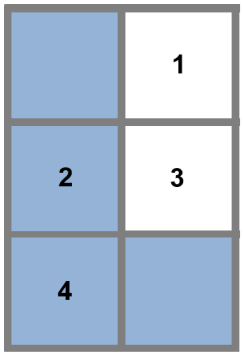


Figure 7.4: An example barcode with a value of 5

In addition to installing markers on the central hub, 4-bit barcodes (Fig. 7.4) were added to the face of each of the modules. These barcodes were used as stand-ins for identifying faulty modules without the of use of a yet-to-be implemented diagnosis procedure (which will be discussed and implemented in Chapter 8). Each barcode is

constructed from six squares, each representing a single bit, organized in three rows of two columns. The top-leftmost and bottom-rightmost squares are both colored and reserved for identification of the extent of the barcode. Of the remaining four squares, a colored square represents a value of 1 while an uncolored square represents a value of 0. Once aligned with a module using the colored rectangles on the central hub, the camera module can individually window each square in the barcode in a specific order to determine the address of the module in front of it. For example, the camera reads each square according to the order in Fig. 7.4 and if only the leftmost square in each row was colored then the resulting binary sequence is 0b0101 or as an integer 5. For the toy examples presented in the rest of this dissertation a 4-bit barcode or16 total possible addresses is more than sufficient; however, if the system was larger a more complex barcode could be implemented. For instance, a square barcode of three row and three columns with two reserved squares for identifying the barcode results in seven remaining squares or 2^7 (128) addresses.

7.3.2 Repair Procedure

Given two agents in the HexDMR III system configured according to the processes outlined in Secs. 7.2 and 7.3.1, repair is now possible. In a majority of repair scenarios, one robot has been diagnosed with at least one faulty module while a second robot is present to complete the repair. The repair procedure outlined in Fig. 7.5 assumes that

only these two robots, the RIF and the RCR, are necessary, that the faulty module has already been located, and that a spare module is readily available to conduct the repair. Each step in the procedure is summarized in more detail below.

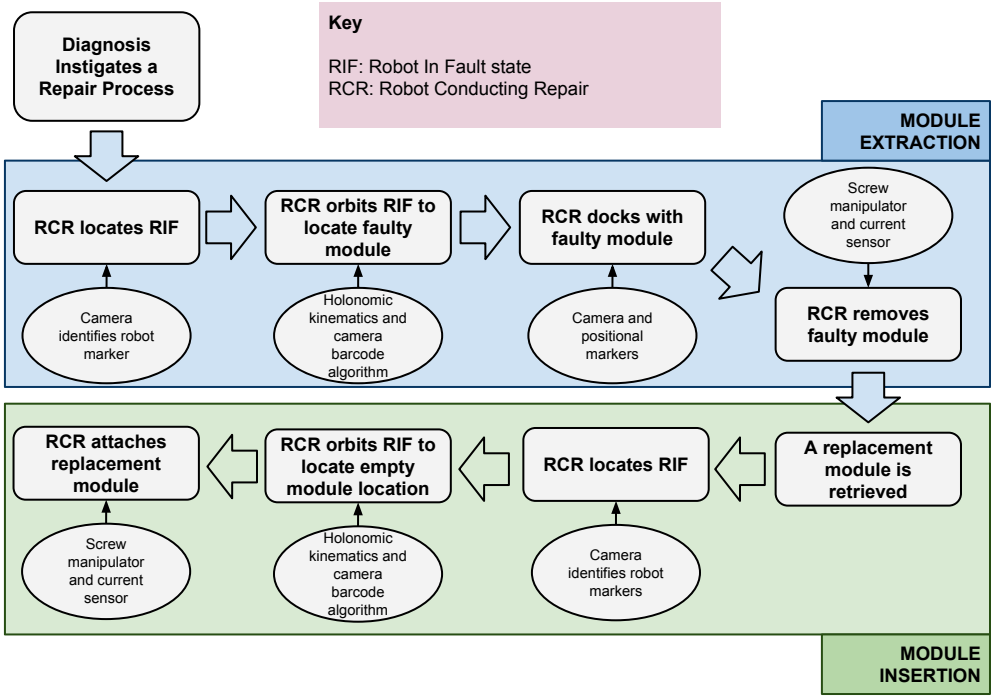


Figure 7.5: Overview of the repair process

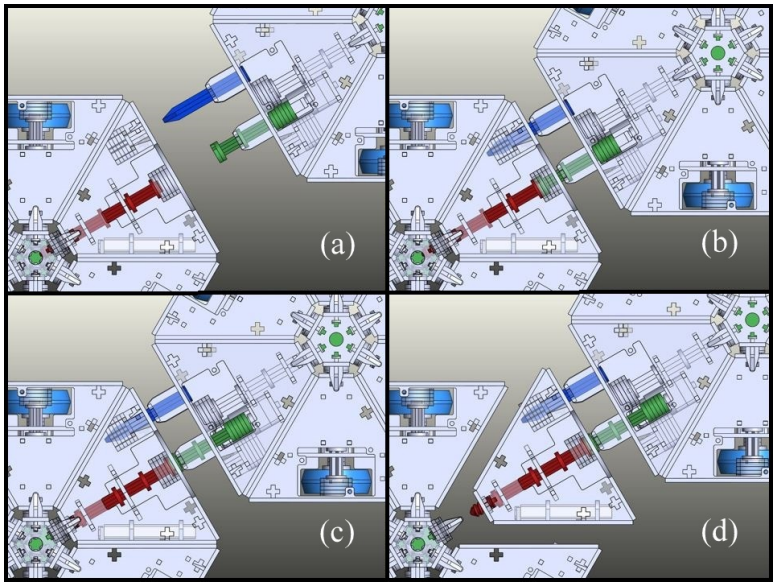
Assuming the RIF has been identified, another robot is assigned the role of RCR to conduct the repair. The RCR begins by rotating about its center until it observes the alternating colored markers on top of the RIF's central hub with its camera module. Once the RIF is located, the RCR centers itself on the colored marker located in the middle of its FOV and approaches the RIF until it reaches a specified distance, while simultaneously ensuring that the centroid of the marker remains in a specified

range relative to the FOV. The RCR must then determine if the module it is facing is the "faulty" module by comparing a 4-bit barcode on the face of the module to the barcode of the "faulty" module. If the barcode does not match the designated module, the RCR orbits the RIF in the counterclockwise (CCW) direction until it is aligned with the next alternatively colored marker, and hence the next module, is centered. This process repeats until the proper module is identified.

Once the "faulty" module has been centered, a docking procedure is initiated. The RCR approaches the RIF, while correcting for errors, until it reaches a certain distance away. Then the RCR strafes to properly align its docking pin with the friction mechanism on the "faulty" module of the RIF. Once aligned, the RCR drives forward effectively mating both the RCR and the"faulty" module. The screw mechanism is activated and continues rotating until the screw has been released from the hub (indicated by a current sensor on the manipulator). Finally, the RCR backs away from the RIF completing the module extraction procedure (Fig. 7.6a).

Before the module insertion procedure begins, the RCR must drop off the "faulty" module and retrieve a "healthy" module from another agent (either by scavenging or using a spare module). Once the RCR is equipped with a "healthy" module it returns to the RIF and follows a similar searching procedure used during extraction; however, since a module has been removed, the absence of a barcode is used to determine the location for insertion. The search continues until the insertion location is discovered

(a) Module extraction from the central hub



(b) Module insertion into the central hub

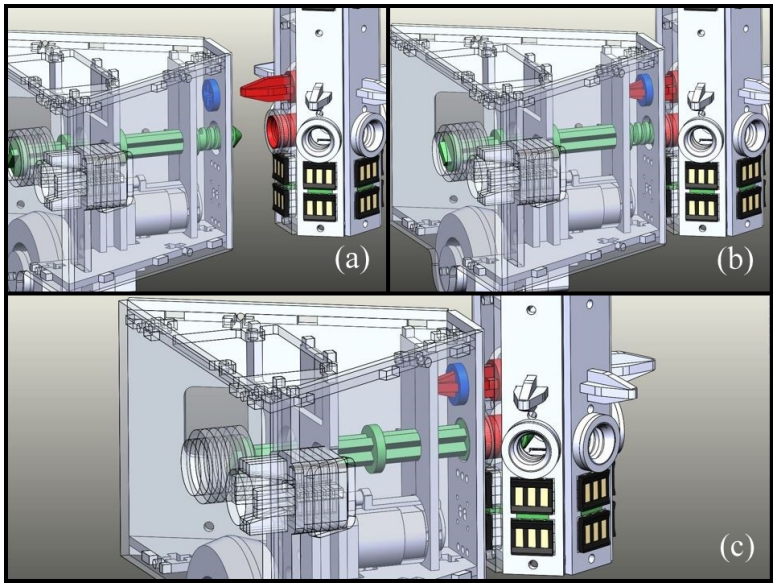


Figure 7.6: CAD Representation of Repair Procedures

and then the RCR centers itself and drives forward to dock the "healthy" module with the central hub. Upon docking, the screw mechanism rotates clockwise (CW) until

the current sensor indicates motor torque saturation or stalling (Fig. 7.6b). Finally, the RCR backs away and is free to return to its original task.

Both module extraction and module insertion are essential for a successful repair process. The nature of design for the elements of the HexDMR III system (Sec. 6.3) dictates a more robust extraction process than insertion process due to the fact that during extraction, alignment and docking are completed for just the manipulator and not a manipulator holding a module (less complex) as well as the fact that mating a module to the manipulator can be less accurate and still successful.

7.3.3 Demonstration of Autonomous Team Repair

Similar to other generations of HexDMR systems, repair was first attempted by teleoperation to verify its efficacy. After these initial successful trials with the HexDMR III system, autonomous repair was demonstrated by independently conducting extraction and insertion procedures.

7.3.3.1 Experimental Setup

The experimental setup for both of these repair maneuvers was fairly similar. The experiments were conducted on top of 0.25 inch thick black foam board which provided better contrast than other materials to the structural clear acrylic plastic of the HexDMR agents. This contrast greatly aided in capturing clear and detailed

images and video during the experiments. For both repair procedures, only two agents were employed. These agents were placed 61 cm apart and the RCR was rotated 90° CCW with respect to the RIF. The nominal configuration of each agent included three drive modules, a control module, a manipulator module, and a power module on the lower layer. The drive modules were installed in an alternating fashion such that there was a non-drive module in between each drive module. Additionally, a single camera module was installed on the top layer directly above the manipulator module. During the extraction procedure, the simulated "faulty" module was placed directly adjacent to the drive module that the RCR was facing in the CCW direction when looking down on the RIF from above. During the insertion procedure this same module was placed on the end-effector of the RCR's manipulator module instead of in the RIF's bottom layer.

7.3.3.2 Repair Validation

Once each robot was configured according to Sec. 7.3.3.1, validation began for both the autonomous extraction and insertion of a module. For these experiments the "faulty" module was chosen to be the RIF's power module. This choice had the added advantage of providing a clear indicator as to when the power module was successfully inserted and extracted due to the status LED changes on the drive modules. The experiments were conducted until there were at least three successes for

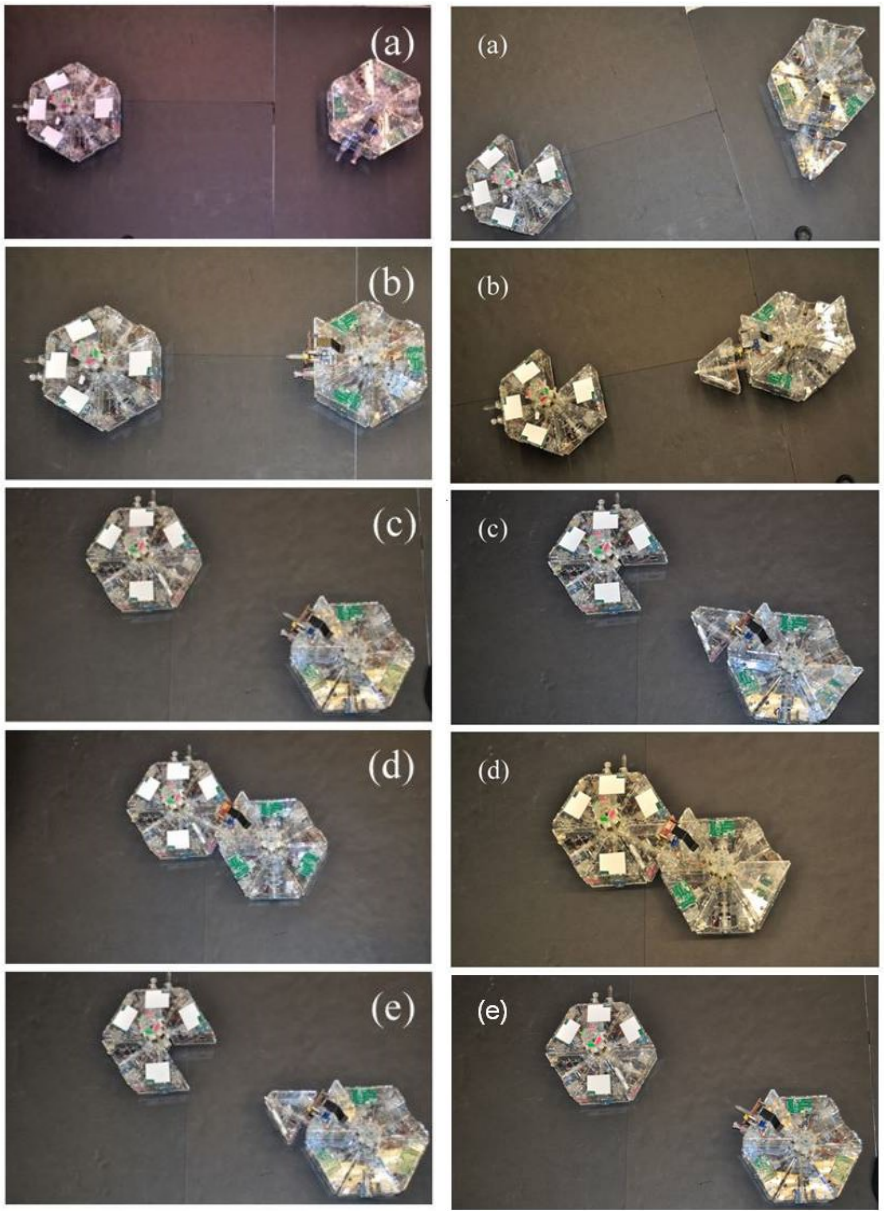


Figure 7.7: Repair Procedures: The first column demonstrates an extraction procedure, while the second column demonstrates an insertion procedure. The corresponding text that outlines each of these procedures is included in Sec. 7.3.3.2

each procedure. Still images from the autonomous removal of the power module are illustrated in the first column of Fig. 7.7, while images from the autonomous insertion

of the module are displayed in the second column. The procedure for extraction is as follows: (a) the RCR rotates CW until the RIF is located; (b) the RCR approaches the RIF and checks the barcode of the module it is facing; (c) the RCR orbits until it finds another module and confirms that the module is slated for repair; (d) the RCR docks with the RIF and unscrews the module; (e) the RCR drives backwards to extract the "faulty" module; and the procedure for insertion is as follows: (a) identical to extraction; (b) the RCR approaches the RIF and checks to see if the module it is facing contains a barcode; (c) the RCR orbits until it finds another module location and confirms that there is no barcode; (d) the RCR inserts the module; (e) and drives away. A video of both the autonomous insertion and extraction procedures is available at `https://www.youtube.com/channel/UC11bvIH6byvI1ecPsgr0XGA`.

Experiments continuously and repeatably demonstrated that an agent of the HexDMR III system can locate the RIF, identify modules by their barcodes, locate the "faulty" module or the absence of a module, and either remove or insert modules in the central hub. Although both procedures were ultimately successful the limited functionality and minimal performance of the CMUcam4 made repair possible but difficult. In addition to this issue, the success rate of the insertion process was noticeably lower than that of extraction due to reasoning provided at the conclusion of Sec. 7.3.2.

7.4 Conclusions

In this chapter, the first fully autonomous repair process for a modular robotic system was demonstrated. In the past other researchers and even previous generations of the HexDMR system demonstrated teleoperated repair but stopped short of fully automating the process. Repair was achievable mainly due to the unique heterogeneous modular design of the HexDMR III system as well as it novel mating interfacing and the presence of a camera sensor to provide limited feedback. Additionally, a method was presented to identify the address of all actuated modules for use in identifying newly added modules and preserving the holonomic drive of the agent after repair.

Although repair was successfully demonstrated, several improvements can be made to future generations of the HexDMR system to enable an even more robust procedure. First, a different camera module should be chosen that provides higher fidelity data at a higher frame rate. Second, the alignment pins and mating surfaces on the HexDMR III system should be constructed with tighter tolerances to ensure more precise placement of the modules. Lastly, the transparency of the acrylic used in the construction of agents should be reduced to increase the contrast in camera images.

With the design of agents in the HexDMR III system verified through the repair process, a diagnosis procedure must be created to identify the faulty or damaged modules in the system.

Chapter 8

HexDMR: Diagnosis

Autonomous repair procedures were demonstrated in the previous chapter with the assumption that damaged or faulty modules were known a priori. This chapter provides the methods and algorithms required to be able to autonomously diagnose or identify a faulty module through a two-step diagnosis procedure that relies on both qualitative and quantitative measures. Before the diagnosis procedure is outlined, a new, more powerful, camera module is introduced and the overall software architecture of agents in the HexDMR III system is discussed. Then, the failure modes and fault states of agents in the HexDMR III system are presented which is then closely followed by a general outline of the diagnosis procedure. The qualitative portion of this procedure describes an interface by which one agent initiates the diagnosis behavior with another agent while receiving health or diagnostic information about the

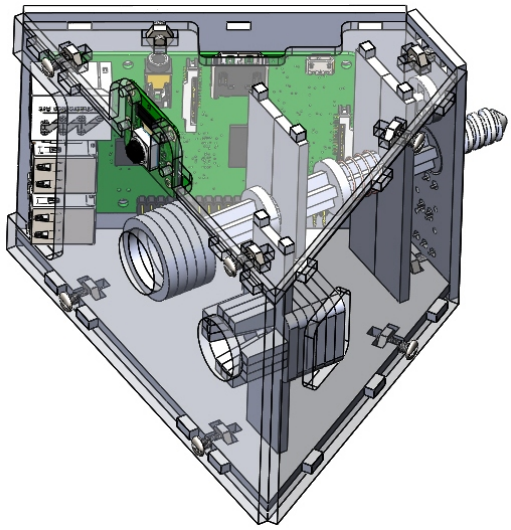
status of camera, elevator, and manipulator modules. Assuming no fatal faults are identified such as a "dead' power module, the quantitative portion of the diagnostic procedure continues to identify specific issues with individual drive modules. For this portion, an unscented particle filter (UPF) is used to compare the performance of the agent to nominal states while it completes several motion-based diagnostic maneuvers. Simulation results for this procedure are then provided for both the three- and six-wheeled agents. Finally, experiments are conducted to validate the two-step autonomous diagnosis procedure.

8.1 The New Camera Module

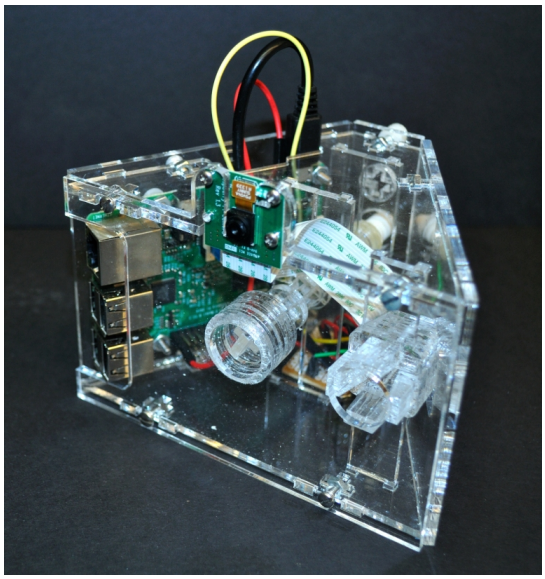
The original camera module was equipped with a CMUcam4 sensor that could only track a single color at a given time to provide basic feedback such as bounding boxes, centroids, and confidence levels. Even though it was with this sensor that an autonomous repair procedure was demonstrated, there were several drawbacks. For instance, the camera data was very sensitive to lighting conditions so much so that readings could vary day to day and even hour by hour. Additionally, it was impossible to extract the raw camera image and, finally, the camera data was specified in the camera's frame of reference (i.e., a two-dimensional set of coordinates corresponding to the physical pixels of the camera's sensor) which led to approximations of higher-

level data such as the orientation and range to an observed object. In the context of diagnosis, each of these drawbacks is untenable, as more information is required to determine the underlying state of each agent. As a result, a new camera module was designed with a more powerful sensor at approximately the same cost.

As opposed to the CMUcam4 sensor which included an integrated microprocessor and camera, this new camera module contains a Raspberry Pi (RPi) 3 quad-core 1.2 GHz microprocessor and a separate RPi camera sensor. The camera sensor can stream raw images up to 90 frames per second at a resolution of 640 by 480 pixels and provides almost a twofold increase in the field-of-view over its predecessor. Since both the computer and camera sensor are still contained in a single module the resolution of repair has not changed between the two versions. Additionally, the physical serial communication interface between the control module and the camera module remained the same as well. The new camera module is approximately the same complexity as the original as the increase in the field-of-view enabled the removal of the tilt servo motor. Moreover, this change allowed the reduction of the vertical footprint of the new module which decreases its cost to homogeneity. A computer-aided design rendition and physical image of the new camera module (now known as the RPi camera module) are included in Fig. 8.1.



(a) A computer-aided design representation of the new RPi camera module



(b) A physical representative of the new RPi camera module

Figure 8.1: Representations of the RPi camera module

8.1.1 Software Architecture

In addition to the component changes between the two camera modules, the RPi camera module enables more freedom in the choice of operating system and available data. Specifically, the RPi camera module was installed with a Linux kernel and equipped with the Robot Operating System (ROS), an open-source robotics middleware that provides higher-level capabilities and interprocess communication (even with other robots) through a messaging framework [93]. Additionally, ROS exposes communication to other processors such as the AtMega168 microcontroller in the control module through a header file. Therefore, the control module can support bidirectional communication with the RPi camera module through specialized mes-

sages over the serial communication interface. This interface enables the RPi camera module to process the computationally heavy camera data and provide only compact and discrete actionable data to the control module.

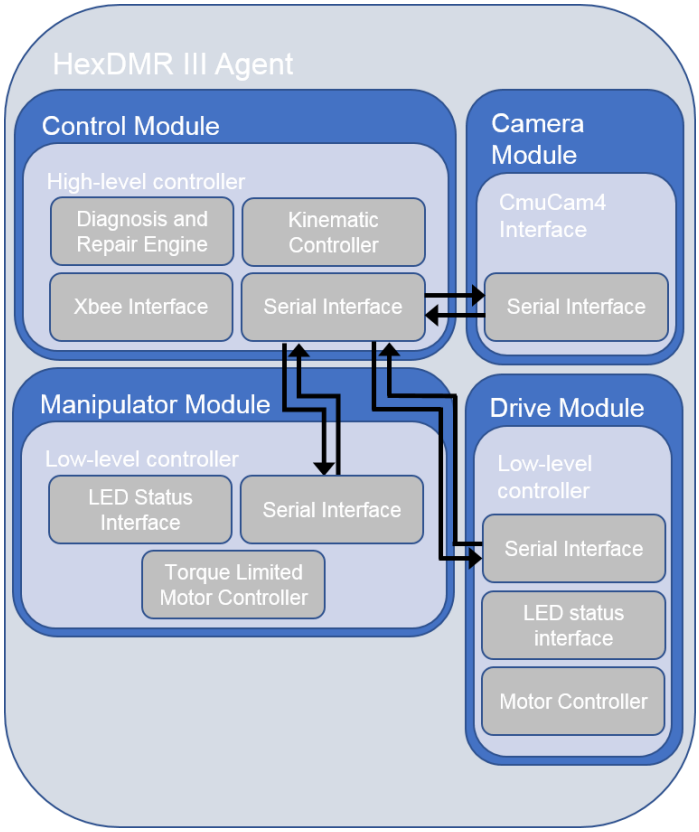


Figure 8.2: Software architecture for HexDMR III agents equipped with the CMUcam4 camera module

In particular, by taking advantage of one of the main benefits of ROS, the camera module can determine the identification (ID) number of a fiducial marker and pass it to the control module to initiate a diagnosis procedure. Specifically, the RPi camera module is installed with the AprilTag framework developed by Edwin Olsen in [94].

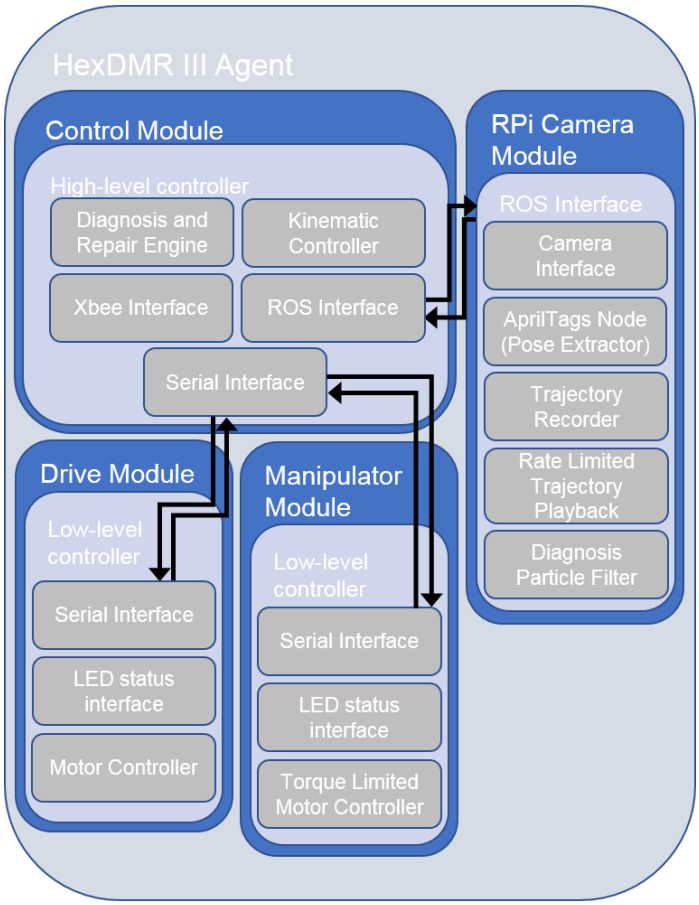


Figure 8.3: Software architecture for HexDMR III agents equipped with the RPi camera module

In this framework, the pose (position and orientation) relative to the camera lens' focal point as well as an ID number can be determined from raw camera data given a set of fiducial markers or tags of a known size and design. This data is essential to the diagnosis framework discussed in subsequent sections and as a result the overall software architecture of agents in the HexDMR III system was modified to accommodate these changes. Figure 8.2 reflects the software architecture of agents equipped with the original camera module and Fig. 8.3 highlights the changes for the RPi camera

module.

In these figures, the software is represented first at the agent level and then at the module level. The first layer in each module represents the main software functionality presented by each module. Then the specific software layers present in each layer are separated. The black arrows in each figure represent the interfaces in which information or data flows from one module to another. Note that connections only feed into and out of the control module and not between other modules. This behavior matches the previously described hierarchical control structure employed by the HexDMR III system.

In general, the majority of the software architecture is unchanged between different versions of the camera module. In the newer version, the control module now includes limited ROS functionalities through an included header file that facilitates communication with the RPi. This data, however, is still transfered over the same physical serial communication bus. Additionally, the serial communication interface, outlined in Sec. 6.3.2.1, to every actuated module is left unchanged. By far, the largest change is in the type of data provided by the camera module as well as the capabilities of the camera module as a whole. As stated before, the control module is still able to delegate more computationally intensive work to the camera module while it concentrates on higher-level behaviors.

8.2 HexDMR III Failure Modes

Before an effective diagnosis procedure can be designed, the frequency and modes of failure in the system must be determined. Many tools exist to conduct this analysis but perhaps the most well known among quality literature is failure modes and effects analysis or FMEA. A failure in FMEA is classified as any deviation from nominal behavior such as a drive module performing at only 50% of its maximum capacity or a single packet being lost in the communication layer. It is not necessary or required that the failure cause harm to overall system to be deemed a failure mode. Furthermore, in this approach only a single point of failure is considered at a time and software is expected to perform nominally. For each failure, the probability and severity as well as the likelihood of detection of the failure are recorded and combined to calculate the risk level. Upon completion, the failure modes can be sorted by risk level to identify which items need immediate corrective action. However, for the diagnosis procedure, FMEA is primarily used to identify the most likely failure modes as well as the best method in which to detect a failure.

In this work, individual modules are the smallest replaceable unit in the HexDMR system; therefore, FMEA was conducted independently on each module type. The results of the FMEA for each module are included in individual tables in Appendix B. The data in each table is populated with extensive firsthand knowledge from interacting with multiple generations of HexDMR systems as well as the documented failure

rates of the physical hardware. From the analysis it is clear that the power module is at the highest risk of failure. Intuitively this makes sense as batteries can only provide limited amounts of energy before needing to be recharged. It is important to note that the severity ratings are mostly predicated on the functionality provided by the individual module. When system level faults are considered in relation to repair it becomes abundantly clear that failure of either the manipulator, camera, or drive modules can cripple the entire agent and be catastrophic without spare modules. Additionally, although failure modes in other modules may be unlikely they must still be detected. Therefore, individual tests or metrics must be defined to identify faults in individual types of modules.

When considering the details of the failure modes in Appendix B it becomes clear that there are two different types of failures. The first is a binary failure in which there are only two states for a module (nominal or faulty) and the second is continuous failure where the failure can be categorized as some percentage of a nominal behavior. For instance, determining if a power module is functional can be a simple yes or no while damage to a drive module may be more nuanced. In particular, referencing item D.2 in Table B.4 if only one out of eight rollers is stuck, the performance of the drive module may be limited to 87.5% of its nominal performance level. This distinction between binary and continuous failure modes leads to the need for both qualitative and quantitative measures in the diagnosis process.

8.3 HexDMR Diagnosis Procedure

As stated in Chapter 5, there are several different approaches to conducting fault diagnosis on mobile robots. In particular, robots can employ self diagnosis where sensor measurements and execution performance are aggregated to determine system faults or external diagnosis where a separate entity observes known behaviors and identifies faults. The approach developed in this chapter is a hybrid method which relies on both qualitative and quantitative measures to successfully ensure that an agent in the HexDMR III system is fully functional. Since individual modules are the smallest replaceable feature in the HexDMR III system, failures only have to be localized to the module level instead of the individual failing components. In essence, modules that exhibit binary failures modes will be grouped under the qualitative diagnosis procedure, while modules that exhibit continuous failure modes (in this case solely the drive modules) fall under the quantitative diagnosis procedure. Each of these procedures is developed in detail below.

8.3.1 Assumptions

This implementation of the diagnosis procedure assumes that at least two robots are always present. The first is the "diagnoser" or the agent that conducts the diagnosis and the second is the "diagnosee" which may move or provide information to the

diagnoser. Additional assumptions regarding the operating condition of each agent are also required before developing the individual portions of the diagnosis procedure. These assumptions are as follows.

1. There are no software errors.
2. The connections between each module and the central hub do not fail.
3. It is more likely that individual modules will fail than the central hub, so the central hub is omitted from the diagnosis procedures.
4. The number of wheels on the base layer of the diagnosee is known as well as their location relative to the AprilTag on the top of the robot.
5. The diagnoser is entirely healthy (i.e., the camera and control modules are functional as well as the external communication wireless radio).
6. The state of spare modules will not be diagnosed.
7. The qualitative diagnosis procedure is conducted before the quantitative procedure.
8. The agents operate in the plane that is perpendicular to the direction of gravity.

8.3.2 Qualitative Diagnosis Procedure

The qualitative portion of the overall diagnosis procedure will be used to determine if either a camera, control, elevator, manipulator, and/or power module has failed. In order to accomplish this task, each agent in the HexDMR system continually keeps an internal representation of its own state. That is, the control module in each agent periodically checks the status of the camera, elevator, and manipulator modules.

The health of the camera module is checked in a two-stage process. Every minute a heartbeat is sent from the camera module to the control module through the aforementioned ROS messaging interface. If a heartbeat is not received within two minutes, the camera module is deemed to have failed. The second stage of this procedure verifies that the camera is functional. Again, once a minute the camera module checks to ensure that the image from the camera has changed from its last reading and that the individual pixel readings are not all the same. This check is sufficient because noise is present in the camera sensor so even if the robot does not move at least one pixel in the camera image should be different.

Although different, the elevator and manipulator modules follow the same procedure to check their health. Specifically, the most important feature of these modules is their ability to manipulate external modules; therefore, the DC motors must be monitored. As long as the modules are not in use, the control module requests a status update once a minute. Internally, the modules actuate their motors and monitor

the current draw using the analog sensor installed on the PIC board. The current draw is then compared to experimental thresholds for free rotation to ensure that the motors and hence modules are functional. Obviously, if a response is not received the modules are also deemed to have failed.

This information is aggregated into a message that contains the status of each camera, elevator, and manipulator module in addition to two timestamps. The first timestamp corresponds to the latest set of data and the second corresponds to the last time that the agent was diagnosed by a second agent. The actual qualitative diagnosis procedure is executed as follows.

1. One agent, upon identifying a second agent, requests its health status.

2. If no response is received, either one of two modules may have failed, the control or power module. The state of the power module is verified by determining if any of the green LEDs on any of the actuated modules is on by matching pixels colors in the camera image. If no active LEDs can be located, the power module has died. Otherwise, some portion (most likely the XBee interface) of the control module has failed.

3. If a response was received and the message identified that a module has failed the diagnosis procedure stops and repair is initiated. If the agent is found to be healthy and the last diagnosis was performed within a certain time threshold the

diagnosis procedure is aborted, otherwise the quantitative diagnosis procedure begins.

8.3.3 Quantitative Diagnosis Procedure

Once it has been observed that the battery, camera, control, and manipulator modules are operating nominally through the qualitative diagnosis procedure, the state of the drive modules must be determined. Following the design constraints outlined in Chapter 6, modules were designed to reduce overall complexity. As a direct result of this decision, agents in the HexDMR III system are equipped with a single sensing modality, a camera, that can track the pose of fiducial markers. Additionally, due to the lack of others sensors such a accelerometers, gyroscopes, and encoders, each agent must employ a dead-reckoning or open-loop control based on a kinematic model. Therefore, the question becomes, how does one determine the state of each drive module with only these simplistic models and measurements?

One of the more popular tools for tackling this problem is filtering or state estimation. In essence, given a set of control inputs and a distribution representing an initial state, the filter will propagate this state through a process model and will pass acquired observations through a measurement model to estimate the state of the system at a given time. Two of the more commonly used filters in robotics for nonlinear systems are the extended Kalman filter (EKF) [95] and the unscented Kalman filter

(UKF) [96]. These filters produce Gaussian approximations of the posterior distributions of the system state given a set of observations. The EKF essentially linearizes the process and measurement models through first-order Taylor series expansions around the estimates of the state. Unfortunately, this linearization can fail to capture the true nonlinear nature of the system and can lead to poor approximations. On the other hand, the UKF does not approximate the process and measurement models, but instead approximates the distribution of the state with a minimal set of deterministically chosen sample points. These points are then propagated through the nonlinear process and measurement models to capture the posterior distribution with up to second-order accuracy. Both of these methods assume that the state can be represented by a single Gaussian random variable, which is not the case for the HexDMR III system. During diagnosis, the continuous states or even a more limited discrete set of states for each drive module must be monitored simultaneously. Luckily, sequential Monte Carlo methods, which are more commonly known as particle filters, do not suffer from this limitation.

In general, particle filter (PF) methods estimate the posterior distribution of a Markov process through the use of partial observations. A Markov process or Markov chain is a process in which the probability of the current state depends only on the probability of the previous state or in other words the memory of the process is one. As the number of particles increases in the filter, the estimate approaches the

actual posterior distribution of the underlying process. A simple implementation of a PF contains three iterative steps after initialization. During initialization, N particles are sampled from the prior distribution and assigned weights of $1\backslash N$. The first repetitive step is importance sampling where a sample is drawn for each particle from the proposal distribution $q\left(\overset{\wedge}{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}\right)$. As it is difficult to sample from the posterior density function, the proposal distribution is used as an approximation and can be designed or chosen by the user as long as two conditions described in [97] are met. After sampling from the proposal distribution and given an observation $\mathbf{z}_t$ at time t, importance weights are calculated for each particle through Bayesian interference by

$$w_t^{(i)} = \frac{p\left(\mathbf{z}_t \mid \overset{\wedge}{\mathbf{x}}_t^{(i)}\right) p\left(\overset{\wedge}{\mathbf{x}}_t^{(i)} \mid \overset{\wedge}{\mathbf{x}}_{t-1}^{(i)}\right)}{q\left(\overset{\wedge}{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}\right)} \tag{8.1}$$

where $w_t^{(i)}$ is the weight of particle i at time t, $p\left(\mathbf{z}_t \mid \overset{\wedge}{\mathbf{x}}_t^{(i)}\right)$ is the likelihood of observation z_t given the state sampled from the proposal distribution, and $p\left(\overset{\wedge}{\mathbf{x}}_t^{(i)} \mid \overset{\wedge}{\mathbf{x}}_{t-1}^{(i)}\right)$, or the transition prior, is the probability of the sampled state given the previous state. The weights are then normalized through the application of the l_1 norm. Note, in this chapter, $p(\mathbf{x})$ will represent the scalar-valued probability of $\mathbf{x}$ unless stated otherwise. The next iterative step involves resampling or suppressing particles with low weights and replacing them with copies of particles with higher weights to obtain N random samples that approximate the posterior distribution. Depending on the

importance weights, it is possible to introduce severe degeneracy during resampling. In the worse case, all particles are replaced by a copy of a single particle. The last iterative step, or the Markov Chain Monte Carlo (MCMC) step, is optional and seeks to prevent degeneracy. In this step, a Markov transition kernel is applied to move particles to different portions of the state space while maintaining the underlying statistics of the approximation to the posterior distribution. The mean and covariance can be calculated from this approximation of the posterior distribution to gain general information about the underlying process.

For the quantitative diagnosis procedure in the HexDMR III system, the state (position and orientation) of the agent is not as important as the underlying state of each drive module. However, given observations from a camera that estimates the state of an agent and stochastic simulations that emulate the state of an agent given known inputs, the underlying state of each drive module can be determined through the use of a PF. Specifically, this work utilizes an UKF to generate the proposal distribution and closely follows the derivation in [97] with a slight modification to the process model. The implementation of the PF, including the process and measurement models, resampling strategy, and overall strategy are outlined in the next section.

8.3.3.1 Implementation

Agents in the HexDMR III system may be equipped with three, four, five, or six drive modules on their base layer. In order to reduce variability and simplify the diagnosis process, a single, quantitative diagnostic procedure was developed to track the state of each drive module independently. Specifically, the diagnosis procedure consists of two phases. In the first phase, the diagnosee performs a diagnosis maneuver, where each drive module is independently actuated for 1.5 seconds in the clockwise (CCW) direction, pauses for 1.0s, and then is actuated for 1.5 seconds in the counterclockwise (CCW) direction while the diagnoser records the pose of the robot. For each of these motions, the drive module is actuated at its maximum duty cycle. Assuming that failure affects the drive module equally regardless of actuation direction, the robot should approximately return to its original starting position and orientation at the conclusion of the maneuver. This process is repeated for each drive module on the base layer of the diagnosee. These commands produce unique trajectories for each drive module regardless of the number of drive modules installed on a HexDMR III agent as long as the starting pose of the agent remains the same for each segment of the procedure. In essence, each maneuver tracks a circle in the plane with radius equivalent to the instantaneous center of rotation created by the remaining stationary wheels.

During the second phase, the diagnoser processes the observed camera data to

determine the underlying state of each drive module. Specifically, an UPF is created for each half-phase (one segment of motion, either CW or CCW) of the diagnosis maneuver for each drive module. For each segment of the diagnosis procedure, the first recorded pose $g_0 \in SE(2)$ is set as the origin and the measurement is said to be captured at zero seconds. Each subsequent pose $g_i \in SE(2)$ for $i = 1, 2, \ldots, t$ is then transformed into g_0's coordinate system to capture the relative motion. In order to use the measurements effectively, both the process and measurement model of the UPF must be defined.

Many nonlinear systems are described according to the following continuous model

$$\begin{aligned} \dot{\mathbf{x}} &= f(\mathbf{x}) + g(\mathbf{x})\mathbf{u} \\ \mathbf{z} &= h(\mathbf{x}) \end{aligned} \tag{8.2}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^p$ are the control inputs, and $\mathbf{z} \in \mathbb{R}^m$ are the output observations. Taken together the mappings $f : \mathbb{R}^p \times \mathbb{R}^p \mapsto \mathbb{R}^n$ and $g : \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}^n$ comprise the process model while $h : \mathbb{R}^p \times \mathbb{R}^p \mapsto \mathbb{R}^m$ represents the measurement model of the system. Due to reasonably large time between sensor readings in the HexDMR III system, the following, equivalent discrete model is more

appropriate.

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k) + g(\mathbf{x}_k)\mathbf{u}_k \\ \mathbf{z}_k &= h(\mathbf{x}_k) \end{aligned} \tag{8.3}$$

Applying this model to the HexDMR III system, the discrete, deterministic process model for time $0 < k < k+1 < t$ is specified by the following relationships.

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix}, \quad g(\mathbf{x}_k) = G(\mathbf{x}_k), \quad f(\mathbf{x}_k) = \mathbb{O}, \quad \dot{\phi}_i = u_i, \quad v_i = r_i\dot{\phi}_i, \quad \mathbf{z}_k = c_m \tag{8.4}$$

where $i \in 1, 2, \ldots, 6$ corresponds to the wheel location, u_i are the commands sent to each drive module that physically correlate to $\dot{\phi}_i$ or the angular velocity achieved by each wheel, r_i is the radius of the ith wheel, and v_i is the equivalent tangential velocity for each wheel. Notice that the measurement model $h(\mathbf{x}_k)$ is equivalent to c_m or the pose as tracked from the raw images by the RPi camera module. Additionally,

the derivation of G for a six-wheeled agent is as follows.

$$G(\mathbf{x}_k) = \begin{bmatrix} -\left(\cos\theta_k + \sqrt{3}\sin\theta_k\right) & -2\cos\theta_k & \sqrt{3}\sin\theta_k - \cos\theta_k \\ \sqrt{3}\cos\theta_k - \sin\theta_k & -2\sin\theta_k & -\left(\sin\theta_k + \sqrt{3}\cos\theta_k\right) \\ \frac{1}{b_1} & \frac{1}{b_2} & \frac{1}{b_3} \\ & & \\ \cos\theta_k + \sqrt{3}\sin\theta_k & 2\cos\theta_k & \cos\theta_k - \sqrt{3}\sin\theta_k \\ \sin\theta_k - \sqrt{3}\cos\theta_k & 2\sin\theta_k & \sin\theta_k + \sqrt{3}\cos\theta_k \\ \frac{1}{b_4} & \frac{1}{b_5} & \frac{1}{b_6} \end{bmatrix} \tag{8.5}$$

where b_i is the distance from center of the agent to the intersecting wheel plane of the ith wheel. If a wheel is not installed, u_i is set to zero for that wheel for all time. The only steps remaining before being able to execute the UPF presented in [97] are defining how the PF keeps an internal representation of the state of each drive module, the transition prior probability density function (pdf) $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$, and the likelihood pdf.

During the diagnosis procedure each drive module is actuated at its maximum duty cycle; however, the actual performance may range anywhere from 0% to 100% depending on the internal state of the module. Unfortunately, tracking the continuum of performance states for the drive module is untenable. Instead, the particle filter only tracks three internal states. That is, at the conclusion of the quantitative diagnosis procedure each drive module will be deemed either non-operational, dam-

aged, or fully operational corresponding directly to functionality of 0%, 50%, and 100% respectively. In other words, if a drive module performs constantly at 50% functionality it will travel the same distance as if the drive module was actuated at a 50% duty cycle. As seen later, the untracked, intermediate states tend towards their nearest neighbor.

In order to track these states, each particle in the particle filter is augmented with a system state corresponding to the three performance levels. Upon initialization, the system states are assigned evenly among the particles. After each iteration of the particle filter, the system states will migrate to the actual underlying drive module state during the resampling stage (which is discussed later). During the diagnosis procedure, the expectation is that each drive module operates at the 100% performance level; however, the UKF requires observations relative to the actual state of the particle. Therefore, observations related to mean behavior of a particular drive module operating at the 0%, 50%, and 100% performance levels must be captured.

Many different methods exist to capture this data, but this dissertation takes advantage of stochastic differential equations to simulate the performance. In particular, this implementation uses Euler Maruyama (EM) integration on the HexDMR kinematic model to generate ensembles of trajectories that capture the open-loop performance of the system. The mean pose of the ensemble of trajectories is then calculated at different time steps and finally a camera measurement is generated. The

specifics of this process are derived below.

Recalling the kinematic model of a three-wheeled agent presented in Eqn. 6.1 and inverting to obtain the state velocity in the body frame results in

$$\begin{aligned}\begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} &= \frac{1}{b_1+b_3+b_5} \begin{bmatrix} -b_5 & -b_5 & b_1+b_3 \\ \frac{2(2b_3+b_5)}{\sqrt{3}} & \frac{-2(2b_1+b_5)}{\sqrt{3}} & \frac{2(b_3-b_1)}{\sqrt{3}} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} r_1\dot{\phi}_1 \\ r_3\dot{\phi}_3 \\ r_5\dot{\phi}_5 \end{bmatrix} \\ &= \frac{r}{3} \begin{bmatrix} -1 & -1 & 2 \\ \sqrt{3} & -\sqrt{3} & 0 \\ \frac{1}{b} & \frac{1}{b} & \frac{1}{b} \end{bmatrix} \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_3 \\ \dot{\phi}_5 \end{bmatrix}\end{aligned} \tag{8.6}$$

where the substitutions $b = b_1 = b_2 = b_3$ and $r = r_1 = r_2 = r_3$ were made due to the symmetry and homogeneity of the drive modules in the HexDMR system. The state velocity in the world frame is obtained by

$$\begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{\theta}_w \end{bmatrix} = \begin{bmatrix} \cos(\theta_w) & -\sin(\theta_w) & 0 \\ \sin(\theta_w) & \cos(\theta_w) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} \tag{8.7}$$

$$= \frac{r}{3}\begin{bmatrix} -\left(\cos(\theta_w) + \sqrt{3}\sin(\theta_w)\right) & \sqrt{3}\sin(\theta_w) - \cos(\theta_w) & 2\cos(\theta_w) \\ \sqrt{3}\cos(\theta_w) - \sin(\theta_w) & -\left(\sin(\theta_w) + \sqrt{3}\cos(\theta_w)\right) & 2\sin(\theta_w) \\ \frac{1}{b} & \frac{1}{b} & \frac{1}{b} \end{bmatrix}\begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_3 \\ \dot{\phi}_5 \end{bmatrix} \quad (8.8)$$

Eqn. 8.8 can be transformed into an Itô stochastic differential equation by representing the wheel angular velocities $\dot{\phi}_i$ for $i \in [1, 3, 5]$ equivalently as $\omega_i = d\phi_i/dt$ with scaled uniform noise added through a Wiener process

$$d\phi_i = \omega_i dt + \sqrt{D} dw_i \quad (8.9)$$

and substituting back into Eq. 8.8.

$$dX = \begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix} = \frac{r}{3}\begin{bmatrix} (-\omega_1 - \omega_3 + 2\omega_5)\cos\theta + \sqrt{3}\,(-\omega_1 + \omega_3)\sin\theta \\ \sqrt{3}\,(\omega_1 - \omega_3)\cos\theta + (-\omega_1 - \omega_3 + 2\omega_5)\sin\theta \\ \frac{1}{b}\,(\omega_1 + \omega_3 + \omega_5) \end{bmatrix} dt \quad +$$

$$\frac{r\sqrt{D}}{3}\begin{bmatrix} -\left(\cos\theta + \sqrt{3}\sin\theta\right) & \sqrt{3}\sin\theta - \cos\theta & 2\cos\theta \\ \sqrt{3}\cos\theta - \sin\theta & -\left(\sqrt{3}\cos\theta + \sin\theta\right) & 2\sin\theta \\ \frac{1}{b} & \frac{1}{b} & \frac{1}{b} \end{bmatrix}\begin{bmatrix} dw_1 \\ dw_3 \\ dw_5 \end{bmatrix} \quad (8.10)$$

More generically, this process can be repeated with four-, five-, and six-wheeled configurations by taking the pseudo-inverse and substituting. The stochastic differ-

ential equation for the six-wheeled robot is derived in Eqns. 8.11, 8.12, 8.13, and 8.14, while the equations for the four- and five-wheeled agents follow the same process and are omitted.

$$\begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} = \underbrace{(T^T T)^{-1} T^T}_{\mathrm{H}} \begin{bmatrix} r_1\dot{\phi}_1 \\ r_2\dot{\phi}_2 \\ r_3\dot{\phi}_3 \\ r_4\dot{\phi}_4 \\ r_5\dot{\phi}_5 \\ r_6\dot{\phi}_6 \end{bmatrix}, \quad T = \begin{bmatrix} -\frac{1}{2} & \frac{\sqrt{3}}{2} & b1 \\ -1 & 0 & b2 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & b3 \\ \frac{1}{2} & -\frac{\sqrt{3}}{2} & b4 \\ 1 & 0 & b5 \\ \frac{1}{2} & \frac{\sqrt{3}}{2} & b6 \end{bmatrix} \tag{8.11}$$

inverting results in

$$d = 4(e + f + g)$$

$$e = 2b_1^2 - b_1b_2 + b_1b_3 + 2b_1b_4 + b_1b_5 - b_1b_6 + 2b_2^2$$

$$f = -b_2b_3 + b_2b_4 + 2b_2b_5 + b_2b_6 + 2b_3^2 - b_3b_4 + b_3b_5$$

$$g = 2b_3b_6 + 2b_4^2 - b_4b_5 + b_4b_6 + 2b_5^2 - b_5b_6 + 2b_6^2$$

$$H = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix}$$

$$
\begin{aligned}
h_{00} &= -\frac{\begin{array}{r}2b_2^2 - b_2b_3 - b_2b_4 + b_2b_6 - 3b_1b_2 + b_3^2 - b_3b_4 + b_3b_5 + 2b_3b_6 - b_1b_3 \\ + 2b_4^2 + b_4b_5 + b_4b_6 + 2b_1b_4 + 2b_5^2 - b_5b_6 + 3b_1b_5 + b_6^2 + b_1b_6\end{array}}{d} \\
h_{01} &= -\frac{\begin{array}{r}3b_1^2 + 2b_1b_3 + 2b_1b_4 - 2b_1b_6 - 2b_1b_2 + 3b_3^2 - 2b_3b_4 + 2b_3b_6 \\ - 2b_2b_3 + 3b_4^2 + 2b_4b_6 + 2b_2b_4 + 4b_5^2 + 4b_2b_5 + 3b_6^2 + 2b_2b_6\end{array}}{d} \\
h_{02} &= -\frac{\begin{array}{r}b_1^2 - b_1b_2 + 2b_1b_4 + b_1b_5 - b_1b_6 - b_1b_3 + 2b_2^2 + b_2b_4 - b_2b_6 \\ - 3b_2b_3 + b_4^2 - b_4b_5 + b_4b_6 + b_3b_4 + 2b_5^2 + b_5b_6 + 3b_3b_5 + 2b_6^2 + 2b_3b_6\end{array}}{d} \\
h_{10} &= \frac{\sqrt{3}\begin{pmatrix}4b_2^2 - 3b_2b_3 + b_2b_4 + 4b_2b_5 + 3b_2b_6 - b_1b_2 + 5b_3^2 + b_3b_4 \\ +3b_3b_5 + 2b_3b_6 + 5b_1b_3 + 6b_4^2 - b_4b_5 - b_4b_6 \\ + 6b_1b_4 + 4b_5^2 - 3b_5b_6 + b_1b_5 + 5b_6^2 - 5b_1b_6\end{pmatrix}}{3d} \\
h_{11} &= -\frac{\sqrt{3}\,(b_1 - b_3 - b_4 + b_6)\,(4b_2 - b_1 - b_3 + b_4 + 2b_5 + b_6)}{3d} \\
h_{12} &= -\frac{\sqrt{3}\begin{pmatrix}5b_1^2 - 3b_1b_2 + 2b_1b_4 + 3b_1b_5 + b_1b_6 + 5b_1b_3 + 4b_2^2 + 3b_2b_4 \\ +4b_2b_5 + b_2b_6 - b_2b_3 + 5b_4^2 - 3b_4b_5 - b_4b_6 \\ - 5b_3b_4 + 4b_5^2 - b_5b_6 + b_3b_5 + 6b_6^2 + 6b_3b_6\end{pmatrix}}{3d} \\
h_{20} &= \frac{2\,(4b_1 - b_2 + b_3 + 2b_4 + b_5 - b_6)}{d} \\
h_{21} &= \frac{2\,(-b_1 + 4b_2 - b_3 + b_4 + 2b_5 + b_6)}{d} \\
h_{22} &= \frac{2\,(b1 - b_2 + 4b_3 - b_4 + b_5 + 2b_6)}{d}
\end{aligned}
\tag{8.12}
$$

Again substituting for $b_i = b$ and $r_i = r$ for $b, r \in [1, 2, \ldots, 6]$ yields

$$\begin{bmatrix} v_{x,b} \\ v_{y,b} \\ \omega_b \end{bmatrix} = \frac{r}{6} \begin{bmatrix} -1 & -2 & -1 & 1 & 2 & 1 \\ \sqrt{3} & 0 & -\sqrt{3} & -\sqrt{3} & 0 & \sqrt{3} \\ \frac{1}{b} & \frac{1}{b} & \frac{1}{b} & \frac{1}{b} & \frac{1}{b} & \frac{1}{b} \end{bmatrix} \begin{bmatrix} r_1\dot{\phi}_1 \\ r_2\dot{\phi}_2 \\ r_3\dot{\phi}_3 \\ r_4\dot{\phi}_4 \\ r_5\dot{\phi}_5 \\ r_6\dot{\phi}_6 \end{bmatrix} \tag{8.13}$$

and then substituting in Eqn. 8.9 results in

$$dX = \begin{bmatrix} dx \\ dy \\ d\theta \end{bmatrix} =$$

$$\frac{r}{n_w} \begin{bmatrix} (-\omega_1 - 2\omega_2 - \omega_3 + \omega_4 + 2\omega_5 + \omega_6)\cos\theta + \sqrt{3}\,(-\omega_1 + \omega_3 + \omega_4 - \omega_6)\sin\theta \\ \sqrt{3}\,(\omega_1 - \omega_3 - \omega_4 + \omega_6)\cos\theta + (-\omega_1 - 2\omega_2 - \omega_3 + \omega_4 + 2\omega_5 + \omega_6)\sin\theta \\ \frac{1}{b}\,(\omega_1 + \omega_2 + \omega_3 + \omega_4 + \omega_5 + \omega_6) \end{bmatrix} dt$$

$$+\frac{r\sqrt{D}}{n_w}\begin{bmatrix} -\left(\cos\theta+\sqrt{3}\sin\theta\right) & -2\cos\theta & \sqrt{3}\sin\theta-\cos\theta & & & \\ \sqrt{3}\cos\theta-\sin\theta & -2\sin\theta & -\left(\sin\theta+\sqrt{3}\cos\theta\right) & & & \\ \frac{1}{b} & \frac{1}{b} & \frac{1}{b} & & & \\ & & & \cos\theta+\sqrt{3}\sin\theta & 2\cos\theta & \cos\theta-\sqrt{3}\sin\theta \\ & & & \sin\theta-\sqrt{3}\cos\theta & 2\sin\theta & \sin\theta+\sqrt{3}\cos\theta \\ & & & \frac{1}{b} & \frac{1}{b} & \frac{1}{b} \end{bmatrix}\begin{bmatrix} dw_1 \\ dw_2 \\ dw_3 \\ dw_4 \\ dw_5 \\ dw_6 \end{bmatrix} \tag{8.14}$$

EM integration is then used to simulate one thousand trajectories for the three internal states of each drive module during each portion of the diagnosis maneuver at 0.1 second time steps. At each time step, the iterative exponential mean is calculated according to

$$\mu_i = \mu_{i-1} \circ \exp\left[\frac{1}{N}\sum_{j=1}^{N}\log(\mu_{i-1}^{-1}\circ g_j)\right] \tag{8.15}$$

where $\mu_{i-1} \in SE(n)$ is the current estimate of the mean and $g_j \in SE(n)$ is a rigid body transformation or pose. Finally, the camera measurement is sampled from a Gaussian distribution with mean $\mu^{\vee}$ and covariance Σ_c or the physical covariance of the camera sensor. The $\vee$ operator here transforms the Lie algebra into its vectorized form.

The transition prior pdf is defined as a Gaussian distribution with mean μ_m equivalent to the deterministic relative motion between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ and covariance equal

to the covariance of a particle at time k. More specifically,

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k) = \frac{1}{(2\pi)^{1.5}|\Sigma|^{0.5}} \exp\left(-\frac{1}{2}(\mathbf{e}-\mu_m)^T\Sigma^{-1}(\mathbf{e}-\mu_m)\right) \tag{8.16}$$

where $\mathbf{e}$ is the relative transformation from the previous state of a particle to a sample drawn from the distribution of the current predicted state of the particle.

The likelihood pdf is defined assuming that the camera, and hence the measurement model, is subject to zero-mean white noise. Specifically, $p(\mathbf{z}_t|\hat{\mathbf{x}}_t)$ or the likelihood of observation $\mathbf{z}_t$ given predicted state $\hat{\mathbf{x}}_t$ can be determined by first obtaining a predicted observation $\hat{\mathbf{z}}_t$ by applying $\hat{\mathbf{x}}_t$ to the measurement model and then using a multivariate Gaussian distribution to calculate the likelihood of z_t as follows

$$p(\mathbf{z}_t|\hat{\mathbf{x}}_t) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_c|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}\left(\mathbf{z}_t-\hat{\mathbf{x}}_t\right)^T\Sigma_c^{-1}\left(\mathbf{z}_t-\hat{\mathbf{x}}_t\right)\right] \tag{8.17}$$

Now with the process and measurement models defined in addition to the implementation of the prediction and update steps, the resampling strategy must be outlined. In general, there a four common resampling strategies (i.e., multionomial, stratified, systematic, and residual) that rely on sampling from a multinomial distribution. Each of these methods is summarized in [97]; however, this work utilizes residual resampling with a slight twist. Specifically, to avoid complete depletion of a particular drive module state, the top 5% of the most heavily weighted particles for

each drive module state are exempt from resampling. This distinction is incredibly important as the physical states of the agent are clustered together during the nascent stages of the diagnosis trajectory which prevents the proper state from propagating through the particles at a later time. This choice also ensures that the probability of selecting the correct drive module state will never exceed 90%. After resampling, a basic Metropolis-Hastings MCMC step is conducted using the UKF to generate the posterior distribution.

At the conclusion of the quantitative diagnosis procedure, the underlying state of each drive module is determined. In particular, for each drive module the likeliest state for both the CW and CCW runs are compared. If there is a discrepancy between the two states, the more conservative option is chosen. For instance, if the CW run returned 100% and the CCW run returned 50%, that drive module is determined to be operating at 50%. These states are then used to make a determination if any module needs to be repaired.

8.4 Quantitative Diagnosis Procedure Simulation

Simulation was conducted in MATLAB to verify the effectiveness of the quantitative diagnosis approach in diagnosing the underlying state of a single agent in the

HexDMR III system when subjected to independent drive module faults.

Parameters for the simulation were chosen to match the physical limitations of HexDMR III agents as closely as possible. For instance, the time step between different iterations of the UPF was set to 0.1 seconds to match the settings of the RPi camera module. Additionally, the angular velocity of the drive modules was determined for different inputs to ensure that the proper inputs were passed to the UKF during the PF prediction step. Specifically, a camera with a frame rate of approximately 240 frames per second was used to record the position of a wheel on a drive module relative to a fixed position. Video was recorded for 20 iterations (split into 10 CW and 10 CCW samples) at five different commanded inputs. At each input, the angular velocity was calculated according to

$$\omega = \frac{2\pi n}{t_e - t_s} \tag{8.18}$$

where n is the number of complete revolutions and $t_s - t_e$ is the elapsed time of the recording in seconds. The averaged angular velocity at each sampled input value is included in Table 8.1. It is unsurprising that there is a cliff where a non-zero input produces no rotation as each drive module must first overcome the frictional force between the agent's wheels and the driving surface. Due to the nature of the data, a

Table 8.1: Wheel angular velocities at key inputs

Input	Angular Velocity (rad/s)
0	0.0
100	0.0
105	1.509
152	3.423
204	5.071
255	6.701

piecewise function was required to represent the inputs throughout their full range

$$\omega(u) = \begin{cases} 0 & u \leq 100 \\ f(u) & 100 < u \leq 255 \end{cases}$$

where

$$f(u) = 1.27 \exp(0.006521u) - 1.094\mathrm{e}^{+08} \exp(-0.1762u) \tag{8.19}$$

is a combination of exponentials that was fit to the remaining data points to generate the nonzero portion of the range (or image) and $u \in [0, 255]$ is the input sent from the control module to a drive module and corresponds to the duty cycle of the pulse width modulation signal sent to each motor.

The final area in which realism was inserted into the simulation relates to the camera. For the UPF to function properly, a measurement model is required that can generate estimates of the pose as seen by the camera module given drive module inputs, at a specific system state, and the current elapsed time. In simulation, this

is accomplished by applying Euler Maruyama integration, following the method included in [98], to the process model or system kinematics to generate trajectories of the robot, as seen in the camera's coordinate system, over time.

Two separate types of camera observations are required to run a single diagnostic simulation. The first is the baseline or aggregate behavior of an agent in a specific system state and the second is a single random sample of the behavior. For both cases, camera observations are drawn from a multivariate Gaussian distribution. The covariance matrix for both cases is the same, but the mean is slightly different. In general the diagnostic maneuvers are known a priori, therefore ensembles of trajectories can be simulated for each maneuver using the specific system state of interest. For the first type, the mean of the Gaussian distribution corresponds to the mean of the ensemble of trajectories for a specific system state. This mean was calculated using Eqn. 8.15. For the second type of camera observation, the mean of the Gaussian distribution was selected as the pose of an agent during a single random trajectory in a specific system state. The covariance matrix for both cases was calculated from camera data captured by an HexDMR III agent. In particular, a stationary agent gathered pose data for a second agent at eleven different locations in the workspace. For each location, approximately 30 samples were collected and the mean was calculated according to Eqn. 8.15. Then relative transformations from the mean to each sample were aggregated for each location and the unbiased sample covariance of this

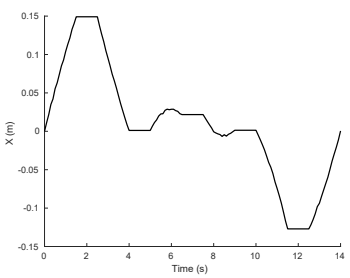
combined data set was captured by

$$\Sigma = \frac{1}{N-1} \sum_{i=1}^{N} T_i T_i^T \tag{8.20}$$

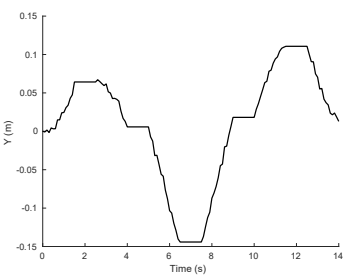
where $T_i = [\log(\mu^{-1} \circ g_i)]^\vee$.

The quantitative diagnosis procedure was tested for both the three- and six-wheeled variants. The first test was designed to verify the functionality of the simulation while the second was developed to test its robustness. For the first test, 20 iterations of the full diagnosis procedure were performed for both variants at each of the known functionality levels. That is, inputs were applied to simulate drive module functionalities of 0%, 50%, and 100% which correspond directly to the three states tracked in each PF. The results of these tests were split to display the results for each drive module as well as the aggregate result for the diagnosis procedure. A sample trajectory of one iteration of the diagnosis procedure is included in Fig. 8.4. Note that since each portion of the diagnosis procedure is analyzed separately that there is no need to randomly set drive module states for each iteration as long as the probability of successful identification of each system state is robust. The results for the first test are included in Tables 8.2 and 8.3 for the three- and six-wheeled variants respectively. In these tables, the + sign represents a CCW rotation while a − sign represents a CW rotation.

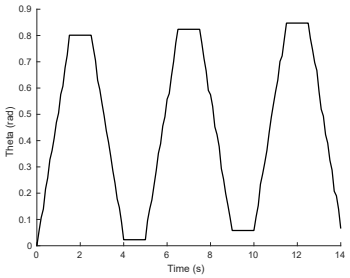
In addition to the raw results, it is helpful to demonstrate the evolution of the



(a) The x-coordinate of the trajectory in meters over time



(b) The y-coordinate of the trajectory in meters over time



(c) The θ-coordinate of the trajectory in radians over time

Figure 8.4: A single trajectory from a full iteration of the diagnostic maneuver for a three-wheeled agent from simulation

Table 8.2: Number of predicted system states at different levels of drive module degradation for a three-wheeled variant

Drive Module Functionality (%)	**Probability of Correctly Predicted States**						
	D1+	**D1-**	**D3+**	**D3-**	**D5+**	**D5-**	**Diagnosis Procedure**
0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
50	1.0	1.0	1.0	1.0	1.0	1.0	1.0
100	1.0	1.0	1.0	1.0	1.0	1.0	1.0

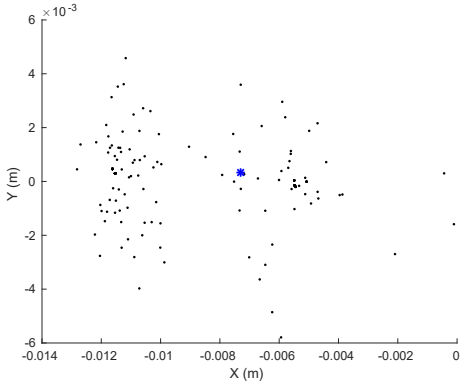
particles over a single portion of the diagnosis procedure. Figure 8.5 demonstrates the evolution of the particles for a single maneuver in the quantitative diagnosis procedure for the three-wheeled variant as well as the corresponding measurements along the trajectory. In this simulation, observations follow a drive module operating at 100% capacity. Each black dot is an individual particle and the blue asterisks is the observation at that time step. From these figures, it is obvious that the particles immediately migrate to the correct drive module state. The groups of the particles from left to right are the 100% state, then the 50%, and finally the 0% state.

Table 8.3: Number of predicted system states at different levels of drive module degradation for a six-wheeled variant

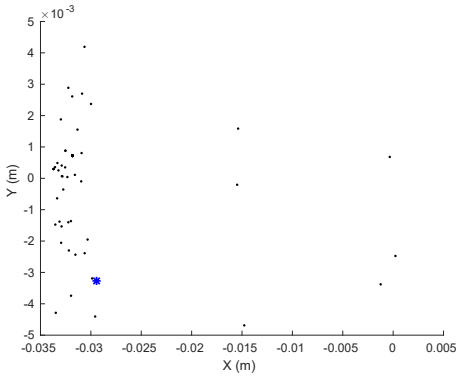
Drive Module Functionality (%)	**Probability of Correctly Predicted States**						
	D1+	**D1-**	**D2+**	**D2-**	**D3+**	**D3-**	
0	1.0	1.0	1.0	1.0	1.0	1.0	
50	1.0	1.0	1.0	1.0	1.0	1.0	
100	1.0	1.0	1.0	1.0	1.0	1.0	
	D4+	**D4-**	**D5+**	**D5-**	**D6+**	**D6-**	**Diagnosis Procedure**
0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
50	1.0	1.0	1.0	1.0	1.0	1.0	1.0
100	1.0	1.0	1.0	1.0	1.0	1.0	1.0

The results from the first test verified that the UPF performed properly; however, it assumed that the simulated system states matched the actual state of the system exactly. In reality, the actual performance of a drive module can range continuously from 0 to 100 percent (where 0% is completely broken and 100% is fully operational). Therefore, the second test was designed to test the robustness of the diagnosis algorithm by predicting the system state at different levels of drive module degradation. Since the previous simulation results demonstrated that the UPF characterization did not differ between the number nor location of drive modules, this test was conducted on only a single drive module of the three-wheeled variant. Table 8.4 provides the results of this approach for 20 iterations at each performance level.

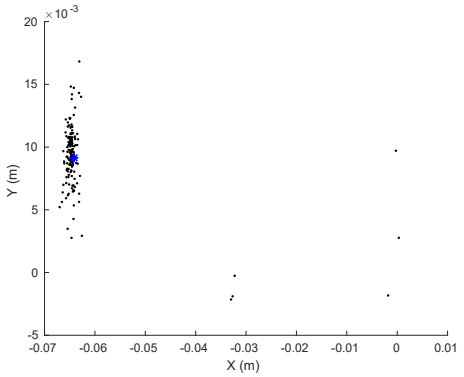
As expected, the diagnosis algorithm correctly identified the system state at the known levels without any false-positives. Additionally, the UPF produced reasonable



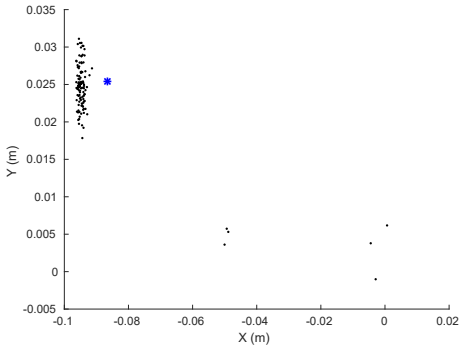
(a) Particles at the beginning of the trajectory



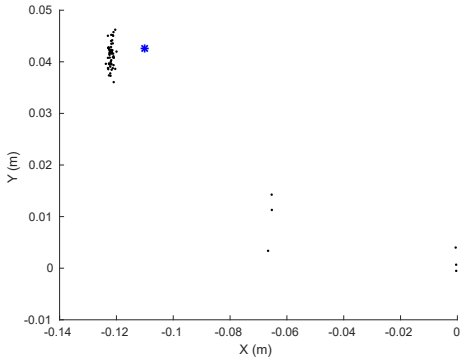
(b) Particles 0.3 seconds into the trajectory



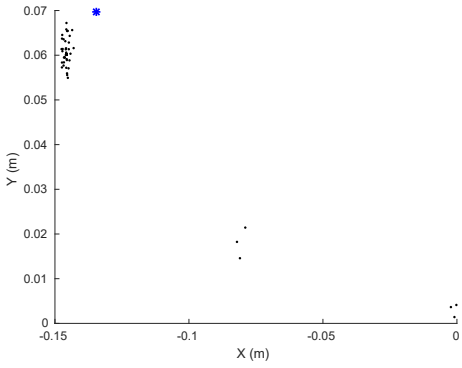
(c) Particles 0.6 seconds into the trajectory



(d) Particles 0.9 seconds into the trajectory



(e) Particles 1.2 seconds into the trajectory



(f) Particles at the conclusion of the trajectory

Figure 8.5: Planar particle evolution and observations over time for the CW portion of a single drive module diagnosis maneuver

Table 8.4: Number of predicted system states at different levels of drive module degradation

Drive Module Functionality (%)	Number of Predicted States		
	0(%)	**50(%)**	**100(%)**
0	20	0	0
40	0	20	0
45	0	20	0
50	0	20	0
65	0	19	1
70	0	14	6
75	0	9	11
80	0	2	18
85	0	2	18
100	0	0	20

results at the transitions between the tracked drive module states. Specifically, the probability of predicting the 50% or 100% state is essentially even at the 75% mark. Moreover, the predictions before and after the transition drastically favored the correct state. Therefore, although the specific state of an individual module may not reflect the state tracked in the UPF, the nearest state will still be chosen. Practically, these results imply that drive modules are more likely to be replaced when their performance drops below the 75% threshold.

8.5 Experiments

The quantitative portion of the diagnosis procedure was verified in the previous simulations with data generated through Euler Maruyama integration. Now, experi-

ments are conducted, with data captured directly from the camera module, to validate the performance of the quantitative diagnosis procedure. Note that experiments to validate the qualitative portion of the diagnosis procedure were omitted due to their straightforward, well-known, and simplistic implementations.

Unfortunately, the code generated for the simulations cannot be used directly during the experiments. Specifically, unlike simulation where the time step between observations can be set to a constant interval, observations during physical applications are more varied due to processor demands. Therefore, interpolation must be used to generate the correct observation for the UKF for each drive module state. Although the state of the agent evolves on SE(2), a cubic Cartesian interpolator was used for simplicity. For a given observation at time t, four points were selected from a pregenerated data set corresponding to the iterative exponential mean of an ensemble of trajectories executed by an agent with a drive module in a particular state at fixed time intervals t_s. If $t > 2t_s$, then the first two points before and after t are selected for the interpolation. Otherwise, the first or last possible points are selected to conduct the interpolation.

In order to validate the quantitative diagnosis procedure, several experiments were conducted. Specifically, the diagnosee executed the diagnosis procedure eight times for each drive module functionality level. For each run, each drive module was commanded inputs to simulate a fault that produced motion according to one of the

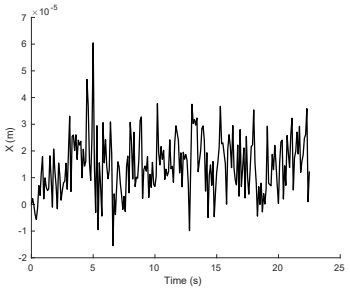
Table 8.5: Number of predicted system states at different levels of drive module degradation for a three-wheeled variant during experimentation

Drive Module Functionality (%)	Probability of Correctly Predicted States					
	D1+	D1-	D3+	D3-	D5+	D5-
0	1.0	1.0	1.0	1.0	1.0	1.0
50	1.0	1.0	1.0	1.0	1.0	0.875
100	0.875	1.0	0.75	1.0	0.75	1.0

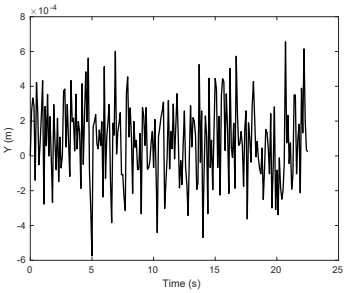
three performance levels. The results for the quantitative diagnosis procedure are included in Table 8.5.

Although not as robust as pure simulation, the results from the experiments appear promising. This method correctly identified the underlying state of each drive module in a single diagnosis procedure 91.7% of the time. Breaking these results down further, the UPF correctly identified all modules operating at the 0% level as well as the 50% level. The one discrepancy seen at the 50% level in Table 8.5 is an artifact of how the data is presented. For the last segment of that diagnosis maneuver the drive module failed to actuate and the UPF successfully identified the state as 0%. The discrepancies seen at the 100% level occurred during two separate trials at the end of the experimental run. For the incorrectly identified states the UPF predicted 50% as the likeliest system state. Anecdotal evidence suggests these results may be accurate as the battery in the power module had to be recharged shortly thereafter.

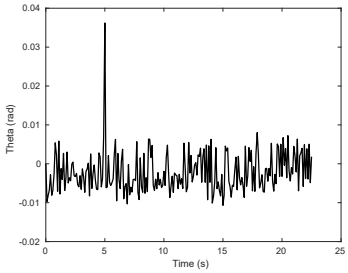
Additionally, Figs. 8.6, 8.7, and 8.8 include representative trajectories from the experimental data for a single trial at each drive module system state. The figures also include histograms at different time steps comparing the number of particles



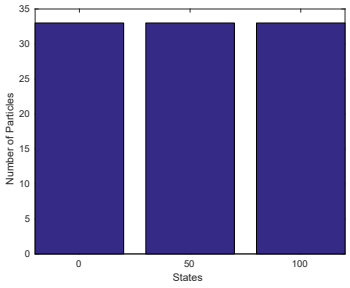
(a) The x-coordinate of the trajectory in meters over time



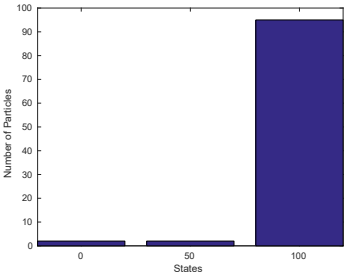
(b) The y-coordinate of the trajectory in meters over time



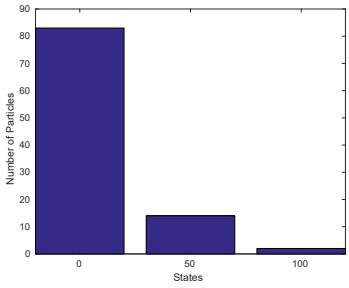
(c) The θ-coordinate of the trajectory in radians over time



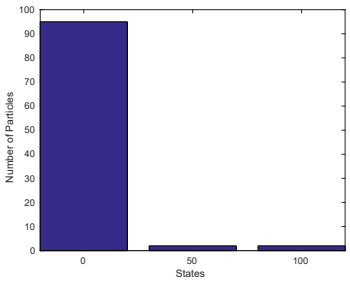
(d) Histogram of particles at the beginning of the trajectory



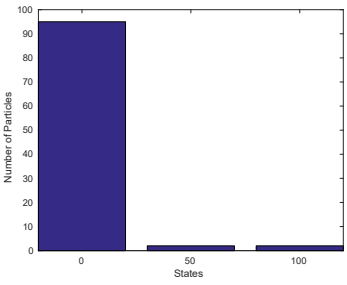
(e) Histogram of particles 0.3 seconds into the trajectory



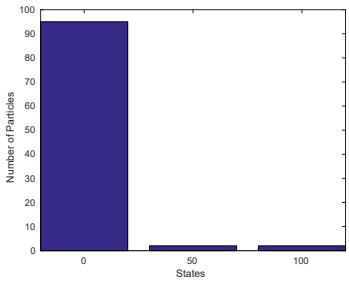
(f) Histogram of particles 0.6 seconds into the trajectory



(g) Histogram of particles 0.9 seconds into the trajectory



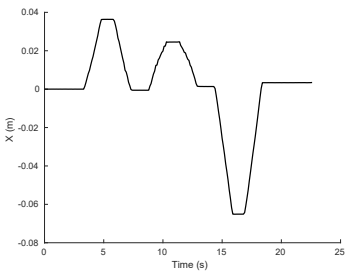
(h) Histogram of particles 1.2 seconds into the trajectory



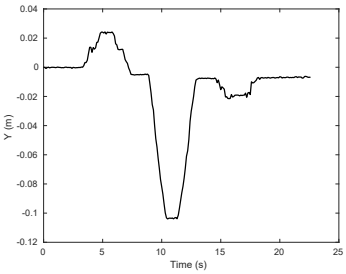
(i) Histogram of particles at the conclusion of the trajectory

Figure 8.6: Experimental data over time for the CW portion of a single drive module diagnosis maneuver while operating at 0% capacity

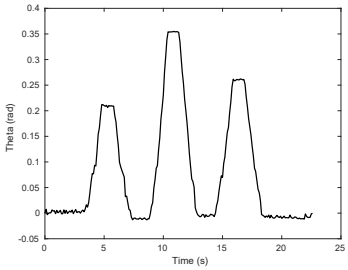
assigned to each system state over the course of execution of a single portion of the diagnosis maneuver. The 0%, 50% and 100% states are represented left-to-right.



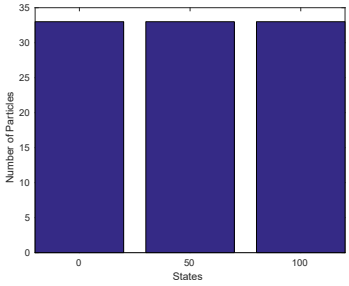
(a) The x-coordinate of the trajectory in meters over time



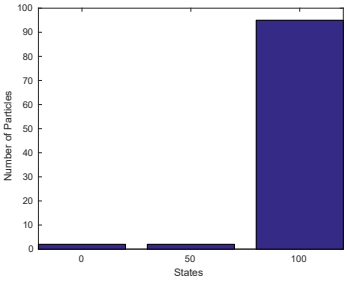
(b) The y-coordinate of the trajectory in meters over time



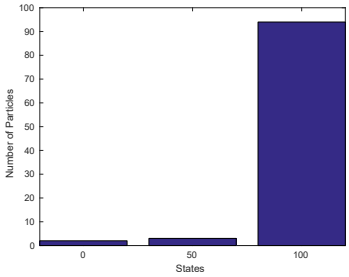
(c) The θ-coordinate of the trajectory in radians over time



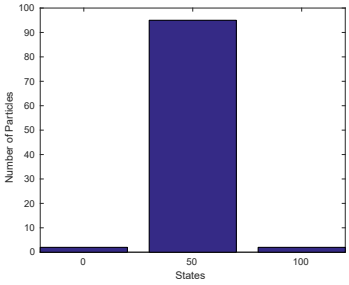
(d) Histogram of particles at the beginning of the trajectory



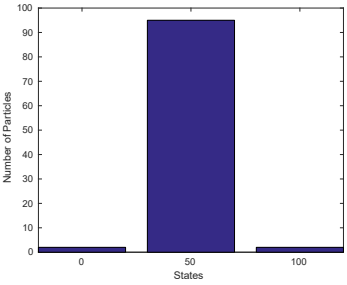
(e) Histogram of particles 0.3 seconds into the trajectory



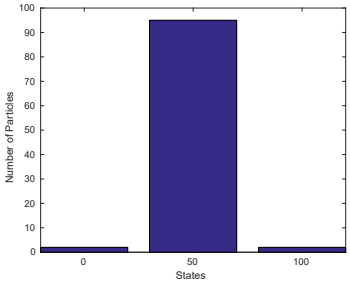
(f) Histogram of particles 0.6 seconds into the trajectory



(g) Histogram of particles 0.9 seconds into the trajectory



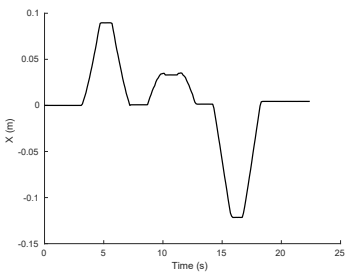
(h) Histogram of particles 1.2 seconds into the trajectory



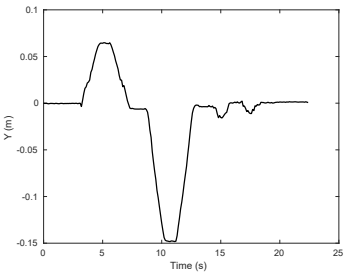
(i) Histogram of particles at the conclusion of the trajectory

Figure 8.7: Experimental data over time for the CW portion of a single drive module diagnosis maneuver while completely damaged while operating at 50% capacity

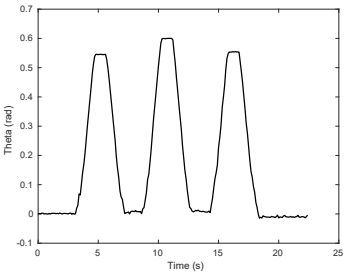
The trajectory in Fig. 8.6 was captured as the diagnosee remained stationary and is a good indicator of the noise in the measurement system. The scale of the axes



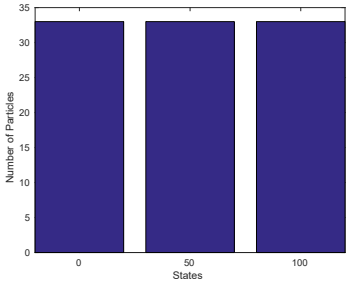
(a) The x-coordinate of the trajectory in meters over time



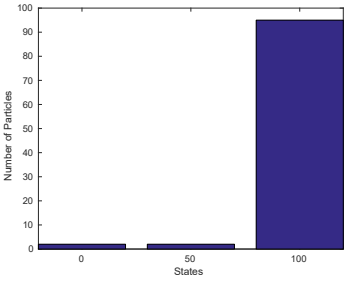
(b) The y-coordinate of the trajectory in meters over time



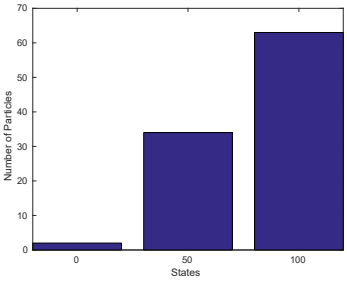
(c) The θ-coordinate of the trajectory in radians over time



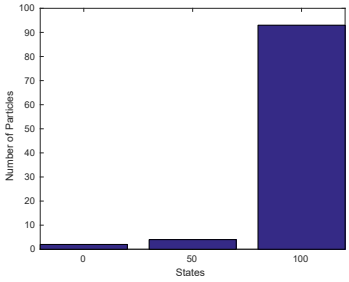
(d) Histogram of particles at the beginning of the trajectory



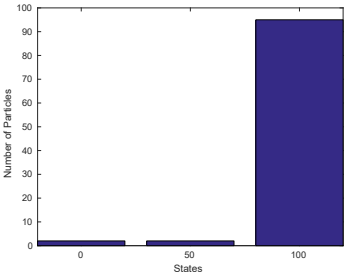
(e) Histogram of particles 0.3 seconds into the trajectory



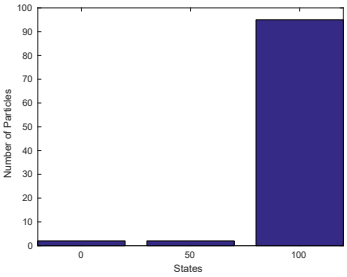
(f) Histogram of particles 0.6 seconds into the trajectory



(g) Histogram of particles 0.9 seconds into the trajectory



(h) Histogram of particles 1.2 seconds into the trajectory



(i) Histogram of particles at the conclusion of the trajectory

Figure 8.8: Experimental data over time for the CW portion of a single drive module diagnosis maneuver while operating at 100% capacity

from left-to-right is $10e^{-5}$, $10e^{-4}$, and $10e^{-2}$ respectively. For each maneuver the 99 particles begin evenly distributed between the underlying drive module states. In the nascent portion of the trajectory, the particles jump around between the states

as the results are almost indistinguishable; however, midway through executing the trajectory (around the 0.7 second mark) the particles have decidedly transitioned to the underlying state and remain mostly stationary. In each of these examples the quantitative diagnosis procedure robustly and accurately identified the underlying state of a drive module. Now, combining both the qualitative and quantitative diagnosis methods, any defective module in an agent of the HexDMR III system can be successfully identified.

8.6 Conclusions

This chapter introduced a new RPi camera module that is necessary to conduct diagnosis in the HexDMR III system. The new camera module not only enables tracking the pose of an agent over time but also captures important heuristic data that can be used during diagnosis. The corresponding new software architecture associated with this module is also included and compared to the previous module. Next, the failure modes of modules in the HexDMR III system were identified through a modified failure modes and effects analysis. The results of this analysis indicated that modules in the HexDMR III system are subjected to either binary or continuous failures. For instance, a power module can only be functional or damaged depending on the remaining charge of the battery, while the performance of a drive module may

vary continuously from 0% to 100% for a given input. As a result, a novel two-part diagnosis procedure was developed that relies on both qualitative and quantitative measures.

The qualitative portion of the diagnosis procedure identifies faults in the camera, elevator, and manipulator modules through an internal messaging protocol and presents this state to an external diagnoser. In addition, the diagnoser also makes determinations about the state of the power module and the responsiveness of the other agent through communication inquiries and observations related to the external, indicator LEDs. The quantitative portion of the diagnosis procedure identifies the underlying state of each drive module through the use of an UPF. In essence, the pose of the diagnosee is tracked while a single drive module is actuated. The UPF then uses the observations from these maneuvers to determine if the drive module is performing at 0%, 50%, or 100% of its capacity. This portion of the diagnosis procedure was verified through simulation for both the three- and six- wheeled variants. When the physical state matched one of the representations in the UPF the simulation correctly identified the underlying state of each drive module 100% of the time. Further simulation was conducted on the three-wheeled variant to test the robustness of the UPF when exposed to a state not directly tracked by the filter. Finally, the entire quantitative diagnosis procedure was validated through experimentation with a three-wheeled agent. Overall, the method correctly identified the underlying state

of each drive module in a single diagnosis procedure 91.7% of the time.

Although the quantitative portion of the diagnosis procedure was developed to determine the underlying state of each drive module, it can also be used to identify the location of drive modules relative to a camera module as long as an observable fiducial marker exists. In this case, it is assumed that each drive module is fully functional and only one agent is present meaning that the agent performing the diagnosis maneuver is also simultaneously capturing the motion relative to a fixed fiducial marker. Now, instead of only tracking three internal states in the particle filter for each individual maneuver, all of the 100% states are tracked for each wheel in each maneuver. At each stage, the particle filter will converge to the state representative of the actuated drive module. Finally, the address of each actuated drive module is stored in the control module and subsequently used to achieve holonomic motion.

The work in this dissertation differs from the method presented by Kutzer et al. in [81] in many facets. First, although the UPF developed here was for a holonomic system it can be extended to any system with independently actuated joints. Second, the procedure developed here does not rely on model-based training but rather stochastic simulation to identify the underlying system state. Furthermore, this method is able to identify faults in all modules utilized by the system as opposed to only the drive modules. Finally, this UPF can identify faults in the drive modules to a finer granularity.

Looking forward, the diagnosis procedures presented here can be extended to the self diagnosis case as long as there is a trackable fiducial marker available. Conversely, these methods can also be applied to teams of larger than two agents. In particular, each agent conducts the diagnosis procedure independently while observing the same agent. The results are then pooled together using either a voting algorithm or another consolidation process to identify the underlying states of each module in the diagnosee. One of the more promising applications of the quantitative diagnosis method may be in identifying faults in commercial robotic arms. In this case, commands are sent to individual joints on the arm and the camera measurements are replaced by the joint positions read from the encoders. Particle filters are then initiated for each joint to identify failures.

Chapter 9

Conclusions

Additive manufacturing and autonomous robotics systems are two exceedingly popular industries that are beginning to secure substantial market share. As these technologies grow consumers will expect and demand even more featureful capabilities. To that end, this dissertation makes modest contributions in both of the aforementioned areas. Specifically, additive manufacturing or the iterative deposition of material in planar layers is extended to a truly conformal process where instead material is deposited normal to the surface of a preexisting object. In the case of autonomous robotic systems, this work presents a novel heterogeneous modular robotic system that is capable of cooperative team repair and diagnosis to extend operational lifetimes. Although these topics are seemingly unrelated, they are strikingly similar analogues if one thinks of the layer generation process as diagnosis and the actual

deposition of material as the repair process. The specific contributions, limitations, and future work of each of these topics is detailed in the following sections.

9.1 Layer Generation in Conformal Additive Manufacturing

Although current additive manufacturing is extremely popular and in wide use, it still has several limitations including the need for sacrificial support material during the build process and the inability to enclose objects or create hollow features. One solution to these limitations is the introduction of a conformal additive manufacturing (CAM) process where material is deposited normal the surface of a preexisting object as opposed to simplistic planar layers.

9.1.1 Discussion

Several researchers have demonstrated limited CAM processes in real world applications such as printing antennas on hemispherical glass substrates or the sensing mechanism in a piezoresistive tactile sensor. These methods are designed to print single layers in specific patterns as opposed to generating full, featureful objects. The main contribution of this work is not the methodology to construct objects, but

rather the development of two novel methods to generate enveloping layers that are conformal to an initial object's boundary and evolve to a desired objects periphery.

The first method relies on variable offset curves and is limited to convex initial objects and desired objects that obey mild compatibility conditions. The second method reparametrizes solutions to Laplace's equation and is only limited to the performance of the numerical solver. Both methods were originally designed for closed spaces or, in other words, uninterrupted free space between the initial and desired objects. A secondary method was introduced that manipulates the solution to the previous two methods to incorporate voids or hollow features into the build volume. Finally, a theoretical testbed was described that could implement these methods in practice.

Ultimately, with proper development, these methods will remove the need for sacrificial support material during the printing process and enable printing directly onto the surface of preexisting objects. The latter capability opens up a whole host of new opportunities previously not afforded to additive manufacturing. For instance, hollow features can be placed below the surface of the desired object to create controlled, deformable surfaces. Additionally, these techniques can be used to repair or retrofit preexisting objects with new features as long as a model exists. Perhaps even more valuable is the ability to embed strengthening materials, akin to rebar in concrete, into the core of larger objects for increased durability and performance. Finally,

these techniques enable unique packaging opportunities where an existing object is completely encased by a lightweight protective shell.

Outside of additive manufacturing, these methods can be utilized in robotic motion planning to identify the entire collision-free configuration space. In this space, the initial object is replaced by an infinitesimally small circle or sphere and the configuration space boundary represents the desired object. The convex hull of the robot is then used to dilate the obstacles and shrink the representation of the robot to a single point. Then, one of the layering methods developed in this dissertation is employed to generate the entire collision-free configuration space. Obstacles, in this case, are treated as voids. If an obstacle changes location only the latter portion of the algorithm is rerun to regenerate the collision-free space. It is important to note that this method generates the entire collision-free collision space unlike traditional, sampling-based, probabilistic methods that only generate a selective portion of the collision-free configuration space.

9.1.2 Future Work

The methods and theoretical testbed discussed in this dissertation are the first step in enabling a truly conformal additive manufacturing process; however, a large amount of work remains to implement this system in practice. In particular, the methods presented in this dissertation described the layer generation process for

CAM. Although individual layers are equidistant along normals originating on the surface of the initial object, each section along the layer varies in size. This discrepancy necessitates a velocity-based control law that ensures the print head dwells in regions of larger separation to achieve full coverage. Moreover, the methods in this dissertation can only handle voids of mild, compatible convexity conditions. Effort should be expended to generalize this method even further in addition to expanding the applicability of the variable offset curve method and reducing the computational burden of numerically solving Laplace's equation.

9.2 Cooperative Robotic Repair and Diagnosis in the HexDMR System

Similar to how opportunities to extend the operational use of items was severely lacking in additive manufacturing processes, the same can mostly be said of robotic systems. Traditionally, repair in robotic systems is performed by highly trained and proficient professionals. This process necessitates a human-in-the-loop and prohibits repairs in remote applications and inhospitable environments. Many of these situations such as the exploration of outer space or the mapping of the power plant at the Fukushima Daiichi nuclear disaster involve expensive systems where a single failure results in the loss of an entire agent. Conversely, modular robotics agents are con-

structed from lower cost, redundant components. In these systems an autonomous "repair" process is employed where damaged modules are simply discarded. The loss of an individual module does not impact the performance of the system; however, the loss of several modules may reduce overall systems capabilities over time. One solution for a truly autonomous repair process that does not reduce system capabilities, and the approach outlined in this dissertation, is the development of a team of heterogeneous modular robots that are capable of diagnosis and repair. Although the development of these methods follow a specific implementation, the designs, methodology, and algorithms can more generally be applied to other robotics systems.

9.2.1 Discussion

In this space, a few researchers have presented piecemeal solutions to this problem. For instance, Bereton and Khosla developed a heterogeneous modular robotic system following a few design constraints that only demonstrated a limited teleoperated repair. On the other end of the spectrum, Kutzer et al. described and implemented an off-board (i.e., the observations and computation were computed separately on a external, unrelated system) autonomous diagnosis procedure to identify some system states of a heterogeneous modular robot. HexDMR III is the first system to fully integrate and demonstrate an autonomous repair and diagnosis procedure into a single enclosed system.

Autonomous repair and diagnosis were achieved through an iterative design process that sought to maximize the homogeneity among modules, the completeness, independence, and robustness of repair, the ubiquity of repair capabilities, and the versatility of agents while minimizing the resolution of repair. To this end, although system capabilities were split into different modules to minimize the resolution of repair the interfaces and designs of each module remained mostly the same. The final system, HexDMR III, encapsulated these requirements into a novel heterogeneous modular robotic system. Each agent in this system is comprised of up to seven different modules types (i.e., central hub, camera, control, drive, elevator, manipulator, and power modules) in two hexagonal layers of six modules. At a minimum, each agent must contain a central hub, three drive modules, a power module, and a control module to achieve basic functionality. In order to perform diagnosis and repair the agents must also be equipped with a camera and manipulator module. The common interfaces and designs of each of these modules, as well as design deficiencies and updates through the different generations of the system, are thoroughly discussed. The general design methodology developed in this dissertation can easily be applied to other robotic systems to move towards more automated repair processes. In fact, Universal Robots already incorporates universal interfaces on their robot arms to reduce costs and streamline the repair process.

Returning to HexDMR III, the modularity of this system as well as the physi-

cal capabilities of each module afford great freedom in selecting a configuration that may suit a particular task. Specifically, in order to determine every unique system configuration, a general method was developed that enumerates each individual configuration into a forest of trees. Each tree is related to functional requirements of the system. In particular, agents in the HexDMR III system require a holonomic drive which mandates that there are at least three drive modules on the base layer. Therefore, the root of each tree in the forest begins with either three, four, five, or six drive modules. Unique configurations in this setting correspond to non-isomorphic, functional agents or, in other words, the functionality and redundancy provided by an agent is unchanged irrespective of the location of the modules on a single layer of the agent. Basically, this requirement prevents double counting of configurations that are solely due to permutations of the placement of modules on a single layer. Following this process, the HexDMR III system contains 10,503 functional, non-isomorphic configurations.

With configurations and design of the HexDMR III system fully characterized, a repair process was created to verify the design. Unlike previous generations of the system, the first truly autonomous repair process was demonstrated when a single agent extracted and then inserted a power module into a second agent to restore its functionality. Repair was accomplished using feedback from the camera module to augment the open-loop kinematic model of the system. It was clear from experi-

mentation that the extraction process was more robust than the insertion process. Intuitively, this discrepancy makes sense as during insertion a second module is effectively cantilevered off of the end of the manipulator module. Poor tolerances during the construction of the friction mechanism coupled with the weight of the module and the strength of the acrylic allow this module to sag which then utilizes the full corrective behavior of the alignment pin during mating with the central hub. Even with these difficulties, the repair process and design of agents in the HexDMR III system were validated.

Once the design was verified through the repair process, the diagnosis procedure was fully developed and implemented. First, a failure modes and effects analysis was conducted to determine the likelihood and cause of the most common failures in each module. This analysis revealed that certain modules in the HexDMR III system experience binary failure while other modules such as the drive module experience a continuous decline in performance. As a result, a novel two-step diagnosis procedure based on both qualitative and quantitative measures was developed. The qualitative portion relies on both an internal state representation of modules in an individual agent as well as external observations to determine failures in the camera, control, elevator, and/or manipulator modules. The quantitative portion of the diagnosis procedure identifies the underlying state of each drive module through the use of an unscented particle filter. For each drive module, three system states are

tracked related to the 0%, 50%, and 100% performance levels. Each drive module is actuated in both the clockwise and counterclockwise directions and the resulting camera measurements are propagated through the particle filter to determine the state. In contrast to the work presented by Kutzer et al. this procedure does not rely of model-based training but rather stochastic simulation to identify the underlying system state. Furthermore, it is able to identify faults in all modules utilized by the system as opposed to a small subset.

The quantitative diagnosis portion of this procedure was verified through simulation for both the three- and six- wheeled variants. The results indicated that the correct underlying state of each drive module was identified 100% of the time as long as the state matched one of the representations in the particle filter. Further simulation was conducted for the three-wheel variant to expose the likelihood of an identifying an intermediate state that was not tracked by the filter. The quantitative diagnosis procedure was then validated through experimentation with a three-wheeled agent.

9.2.2 Future Work

A fully autonomous repair and diagnosis procedure was developed and proven for the HexDMR III system; however, these implementations were not without their own challenges. Several improvements to a future generation HexDMR system would

greatly improve the process. For instance, with constant technological advancement it is possible to upgrade the processor in the control module to a multi-core model at basically the same price point. This improvement enables the HexDMR system to execute its control loop more frequently for better control of the agent. Additionally, the kinematic performance and control of the agent could be improved by the simple addition of encoders to each of the actuated modules.

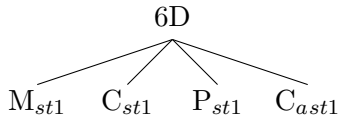
On a more algorithmic side, the diagnosis procedure was implemented for one robot observing a second robot. These methods can also be applied to self diagnosis as long as a fiducial marker is available to track. On the other side, these methods can be applied to teams of more than two robots where several robots observe a robot as it conducts the diagnosis maneuver. The team then votes, or uses a consolidation process, to determine the underlying state of agent being observed. Additional work can also be expended to determine an optimal trajectory that exposes the state of each drive module in a single motion as opposed to two separate motion per module. Furthermore, although the quantitative diagnosis procedure was developed for the HexDMR III system it could easily be extended to detect faults (e.g., errors in calibration or failing actuators) in multiple degree-of-freedom robotic arms. In this case, commands are sent to individual joints on the arm and the camera measurements are replaced by the joint positions read from the encoders. Particle filters are then initiated for each joint to identify failures.

Finally, many of the methods presented in relation to cooperative robotic repair and diagnosis were developed for planar systems; however, with a few adaptations this work can be extended to systems that operate in SE(3). As an example, consider an autonomous unmanned aerial vehicle, or drone, that is comprised of several different modules. The design of repairable interfaces and docking mechanisms can readily be employed to increase the robustness of the system. Additionally, the repair process outlined in Chapter 7 can still be followed albeit it with a more complicated control structure. Furthermore, the two-part diagnosis procedure introduced in Chapter 8 is still applicable although the implemented kinematic model would need to be updated to a dynamic model to ensure proper estimation in the particle filter.

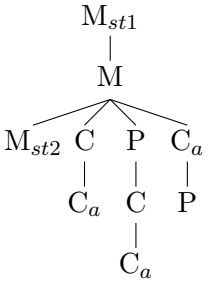
Appendix A

Configurations of the Hex-DMR III System

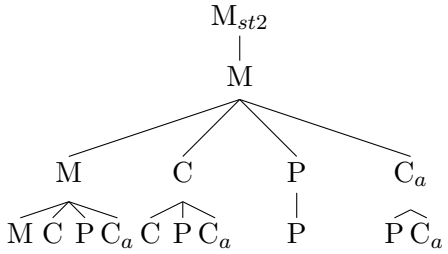
A.1 Six-Wheeled Configurations



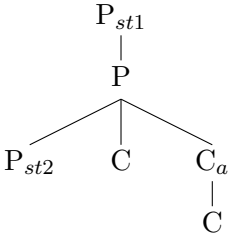
(a) Main Tree



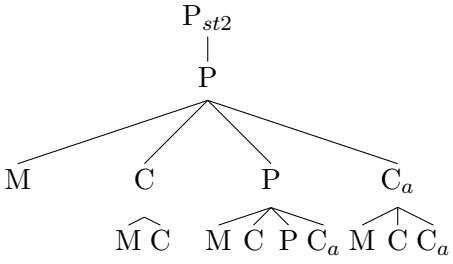
(b) 1st Manipulator Module Subtree



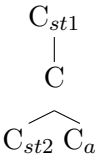
(c) 2nd Manipulator Module Subtree



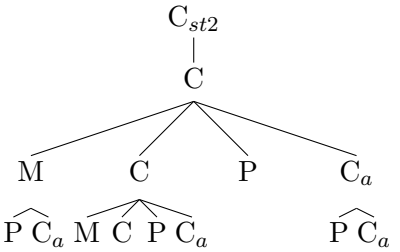
(d) 1st Power Module Subtree



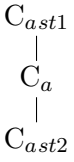
(e) 2nd Power Module Subtree



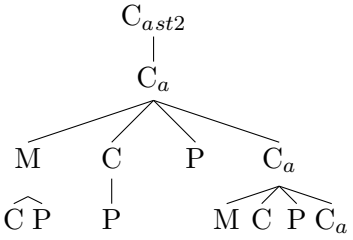
(f) 1st Control Module Subtree



(g) 2nd Control Module Subtree



(h) 1st Camera Module Subtree



(i) 2nd Camera Module Subtree

Figure A.1: Configurational Tree for the Six-Wheeled Base Configuration

A.2 Five-Wheeled Configurations

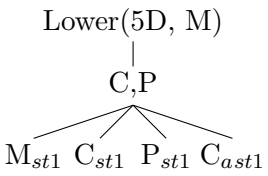


Figure A.2: Configurational Tree for the Five-Wheeled and Manipulator Lower-Level Configuration

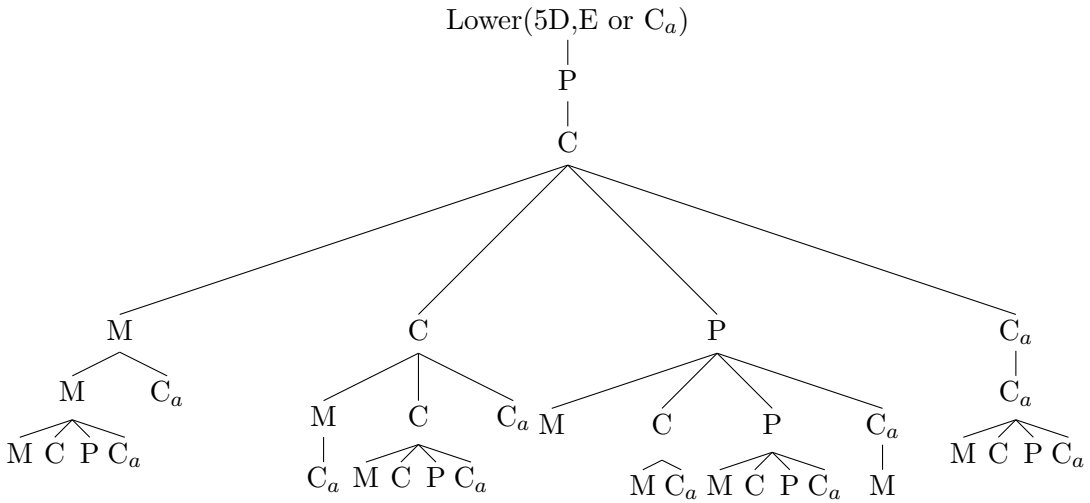
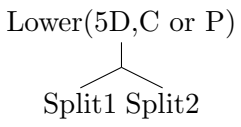
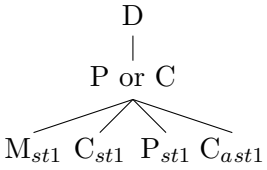


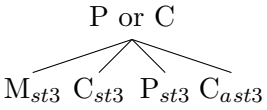
Figure A.3: Configurational Tree for the Five-Wheeled and Elevator of Camera Lower-Level Configuration



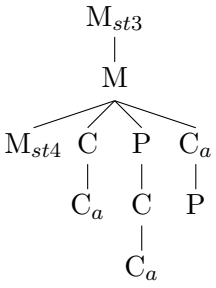
(a) Main Tree



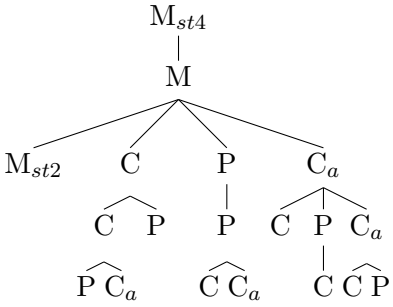
(b) Split 1 Subtree



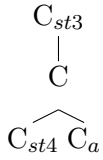
(c) Split 2 Subtree



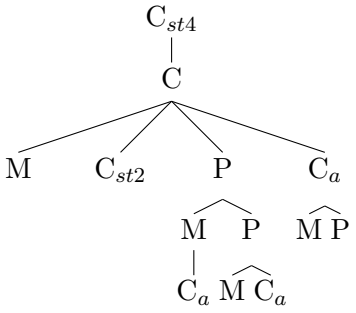
(d) 3rd Manipulator Module Subtree



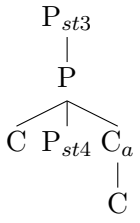
(e) 4th Manipulator Module Subtree



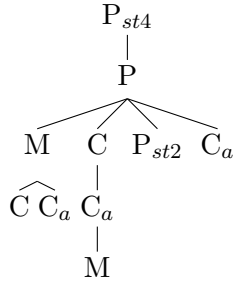
(f) 3rd Control Module Subtree



(g) 4th Control Module Subtree

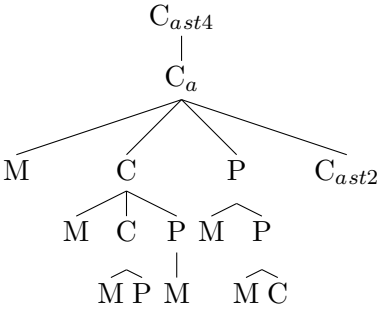


(h) 3rd Power Module Subtree



(i) 4th Power Module Subtree

(j) 3rd Camera Module Subtree



(k) 4th Camera Module Subtree

Figure A.4: Configurational Tree for the Five-Wheeled and Power or Control Lower-Level Configuration

A.3 Four-Wheeled Configurations

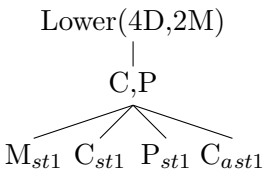


Figure A.5: Configurational Tree for the Four-Wheeled and Two Manipulator Lower-Level Configuration

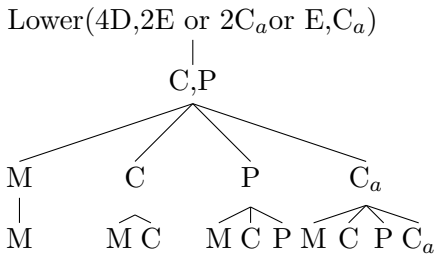
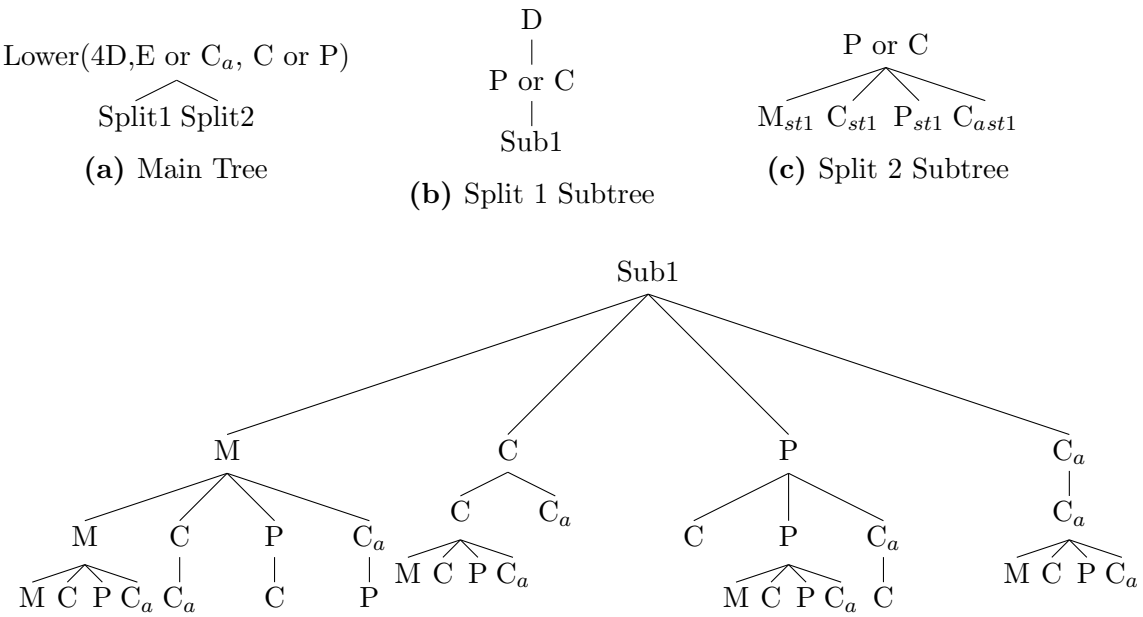


Figure A.6: Configurational Tree for the Four-Wheeled and 2E, 2C_a, or E,C_a Configuration



(a) Main Tree

(b) Split 1 Subtree

(c) Split 2 Subtree

(d) Main Subtree 1

Figure A.7: Configurational Tree for the Four-Wheeled and (E or C_a) or (C or P) Lower-Level Configuration

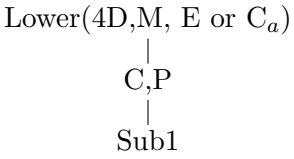
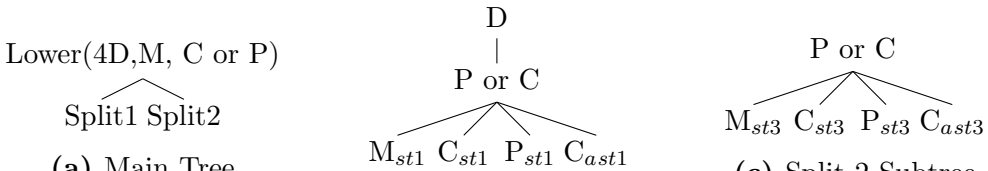


Figure A.8: Configurational Tree for the Four-Wheeled, M, and (E or C_a) Lower-Level Configuration

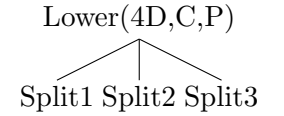


(a) Main Tree

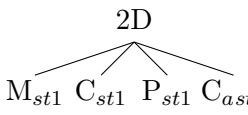
(b) Split 1 Subtree

(c) Split 2 Subtree

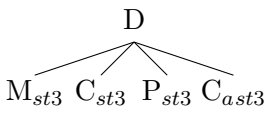
Figure A.9: Configurational Tree for the Four-Wheeled, M, and (C or P) Lower-Level Configuration



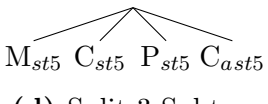
(a) Main Tree



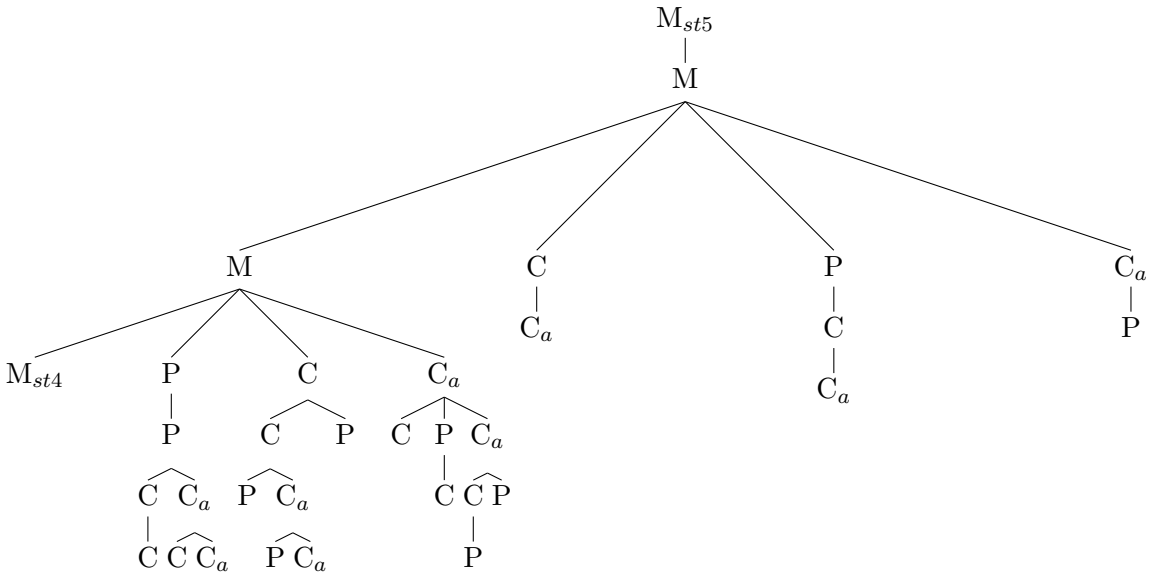
(b) Split 1 Subtree



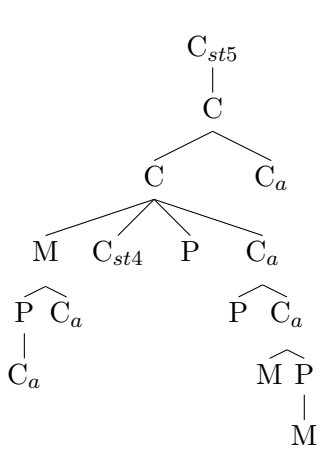
(c) Split 2 Subtree



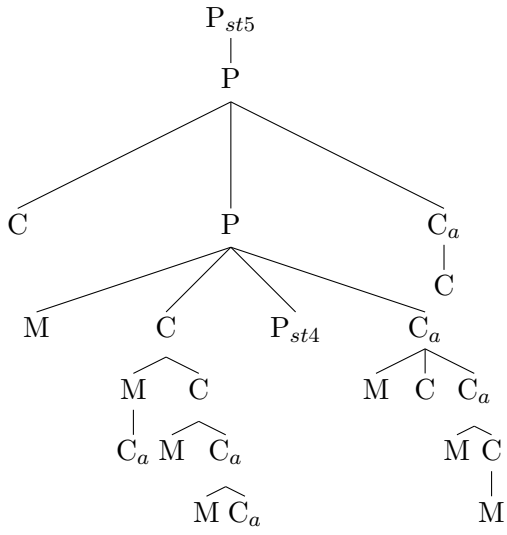
(d) Split 3 Subtree



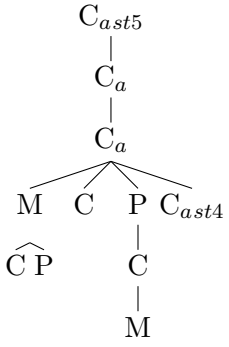
(e) 5th Manipulator Module Subtree



(f) 5th Control Module Subtree

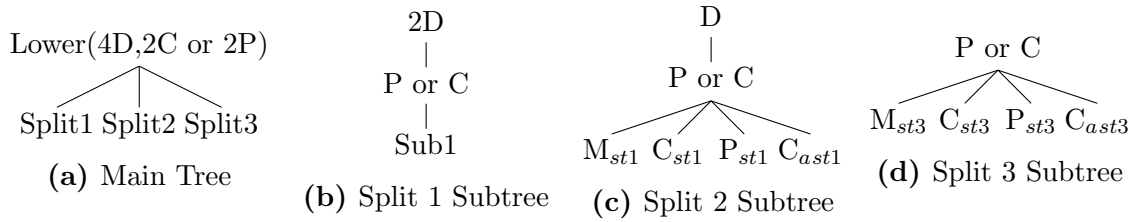


(g) 5th Power Module Subtree



(h) 5th Camera Module Subtree

Figure A.10: Configurational Tree for the Four-Wheeled, C, and P Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree

Figure A.11: Configurational Tree for the Four-Wheeled and (2C or 2P) Lower-Level Configuration

A.4 Three-Wheeled Configurations

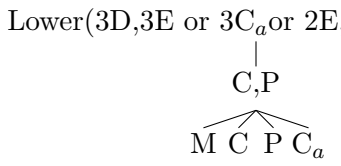


Figure A.12: Configurational Tree for the Three-Wheeled and (3E or 3C$_a$ or 2E,C$_a$ or E,2C$_a$) Lower-Level Configuration

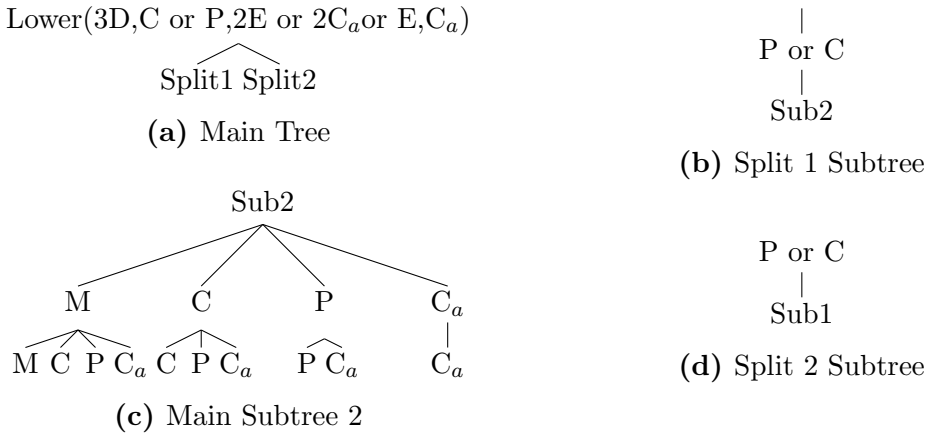


Figure A.13: Configurational Tree for the Three-Wheeled, (C or P), and (2E or 2C$_a$ or E,C$_a$) Lower-Level Configuration

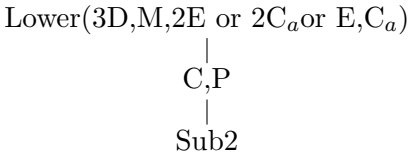
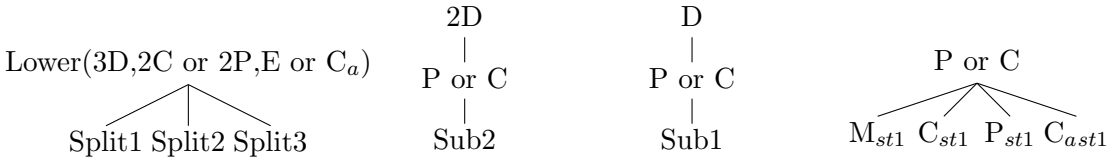
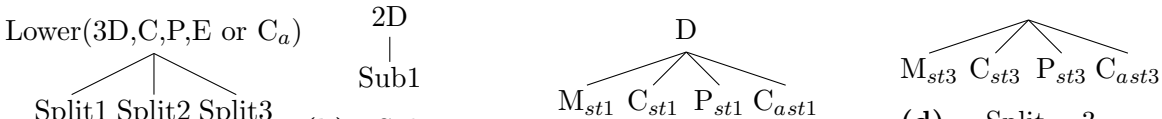


Figure A.14: Configurational Tree for the Three-Wheeled, M, and (2E or 2C$_a$ or E,C$_a$) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree

Figure A.15: Configurational Tree for the Three-Wheeled, (2C or 2P), and (E or C_a) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree

Figure A.16: Configurational Tree for the Three-Wheeled, C, P, and (E or C_a) Lower-Level Configuration

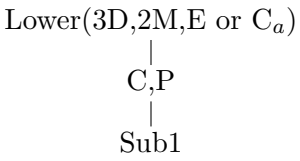
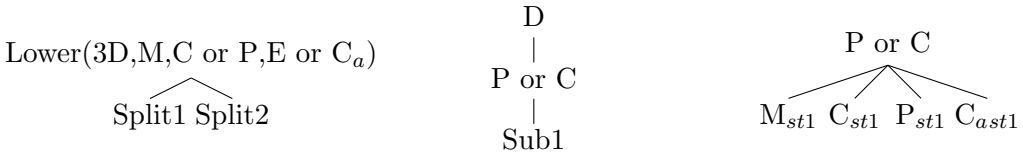
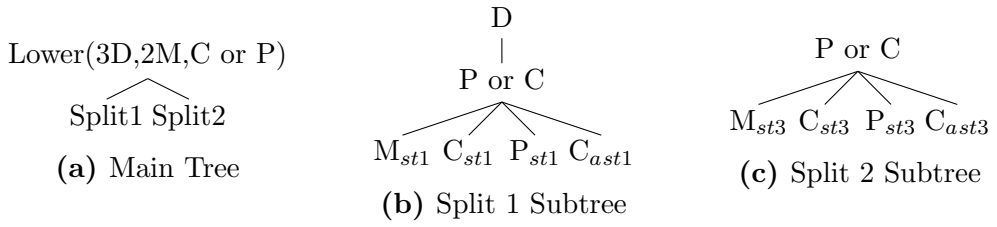


Figure A.17: Configurational Tree for the Three-Wheeled, 2M, and (E or C_a) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree

Figure A.18: Configurational Tree for the Three-Wheeled, M, (C or P), and (E or C_a) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree

Figure A.19: Configurational Tree for the Three-Wheeled, 2M, and (C or P) Lower-Level Configuration

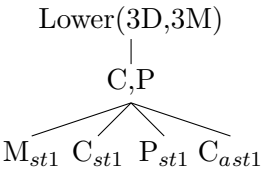
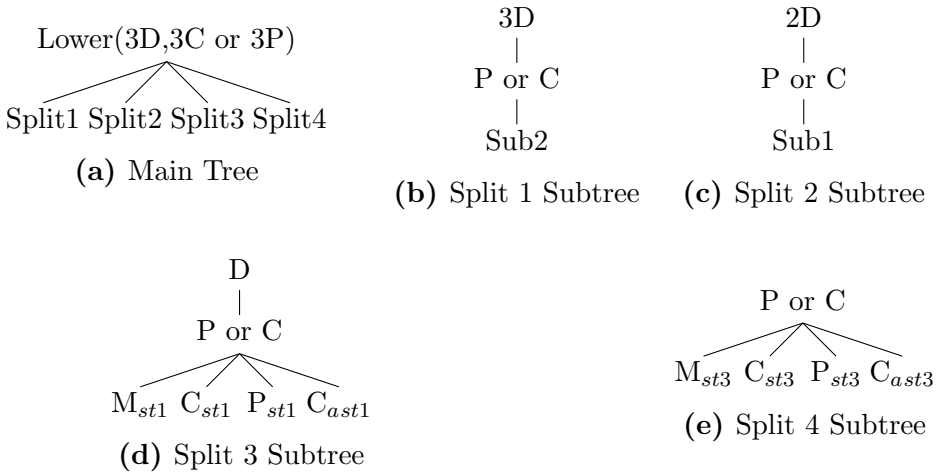
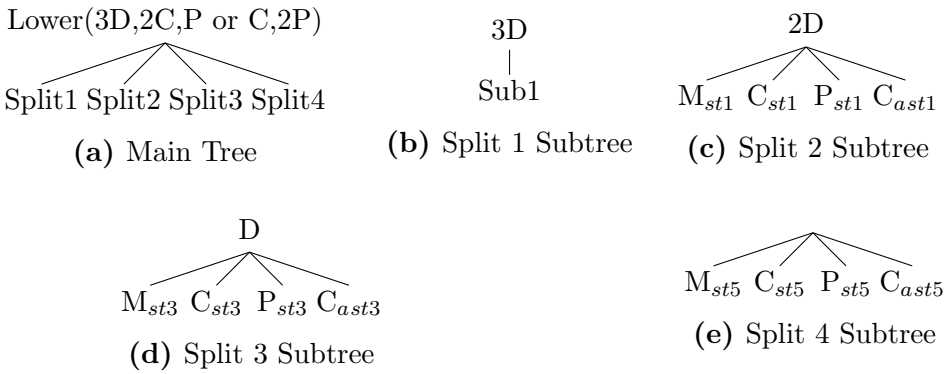


Figure A.20: Configurational Tree for the Three-Wheeled and 3M Lower-Level Configuration



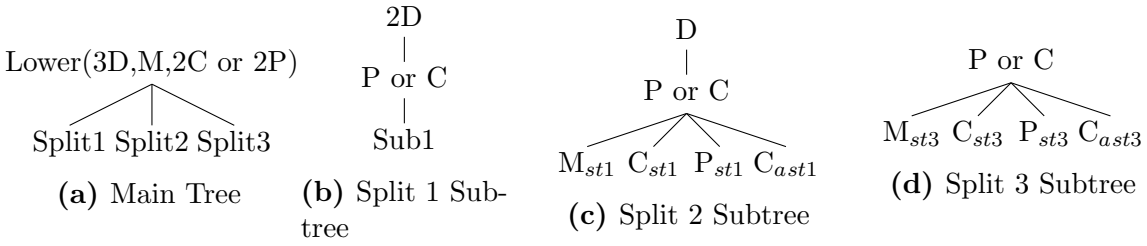
(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree **(e)** Split 4 Subtree

Figure A.21: Configurational Tree for the Three-Wheeled and (3C or 3P) Lower-Level Configuration



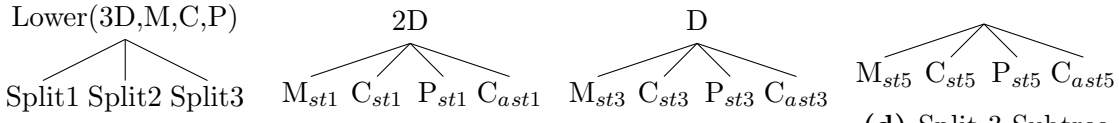
(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree **(e)** Split 4 Subtree

Figure A.22: Configurational Tree for the Three-Wheeled and (2C,P or C,2P) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree

Figure A.23: Configurational Tree for the Three-Wheeled, M, and (2C or 2P) Lower-Level Configuration



(a) Main Tree **(b)** Split 1 Subtree **(c)** Split 2 Subtree **(d)** Split 3 Subtree

Figure A.24: Configurational Tree for the Three-Wheeled M, C, and P Lower-Level Configuration

Appendix B

Failure Mode and Effects Analysis of the HexDMR III System

This appendix utilizes a slightly modified version of the failure modes and effects analysis (FMEA) presented in [99] to identify the failure modes for individual modules in the HexDMR III system. In the analysis for each module, failures are related to system or agent-level deficiencies that may result from a single failure.

Table B.1: FMEA for a central hub in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
$C_h.1$	Severed female connector	Fatigue from repeated contact, impact	The remaining connections are more likely to break	A connected module loses one or more connections
$C_h.2$	Ribbon cable becomes disconnected	Vibration	Connections between layers are lost	An entire layer may stop functioning if power is lost

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
System capability loss	A - Extremely Unlikely	III - Minor	3 - High	Low
The robot may lose sensing capabilities	A - Extremely Unlikely	IV - Critical	3 - High	Low

Table B.2: FMEA for an elevator module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
E.1	The drive train jams	Poorly molded parts	The elevator carriage seizes in place	The DC motor draws its constant stall current and drains the power module or the motor burns out
E.2	A gear breaks and becomes dislodged	Manufacturing defect	The elevator carriage seizes in place	The DC motor draws its constant stall current and drains the power module or the motor burns out
E.3	One of the elevator cutoff sensors breaks	Impact	The elevator carriage continues to drive even past the possible allowed travel	The DC motor draws its constant stall current and drains the power module or the motor burns out

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
Modules can no longer be moved to different layers	A - Extremely Unlikely	III - Minor	5 - Low	Low
Modules can no longer be moved to different layers	A - Extremely Unlikely	III - Minor	4 - Moderate	Low
Modules can no longer be moved to different layers	A - Extremely Unlikely	III - Minor	5 - Low	Low

Table B.3: FMEA for a manipulator module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
M.1	The DC motor burns out	The manipulator tries to continue screwing in a fully mated module drawing its constant stall current	The motor no longer functions	The motor catches fire
M.2	The spring on the end-effector shaft breaks	Fatigue from repeated use	The end-effector no longer protrudes far enough to engage a screwed in module	The spring is free and gets caught seizing the end-effector shaft in place
M.3	The end-effector mating mechanism is damaged	Impact or fatigue from repeated use	No noticeable local effects	The shaft no longer mates with the shaft of other modules
M.4	The alignment mechanism is damaged	Impact	No noticeable local effects	Either the manipulator cannot mate with other modules or modules fall off during extraction or motion

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
Modules can no longer be extracted or inserted	A - Extremely Unlikely	IV - Critical	4 - Moderate	Low
Modules can no longer be extracted or inserted	B - Remote	IV - Critical	4 - Moderate	Moderate
Modules can no longer be extracted or inserted	A - Extremely Unlikely	IV - Critical	4 - Moderate	Low
Modules can no longer be extracted or inserted	B - Remote	III - Minor	4 - Moderate	Low

Table B.4: FMEA for a drive module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
D.1	The H-bridge that supplies power to the motor burns out	Electrical short	The motor can no longer be controlled bidirectionally	The wheel spins freely with enough friction
D.2	One of the rollers on the omnidirectional wheel jams	Dirt build up from use	The roller stops spinning	Additional friction wears the rubber roller faster
D.3	Part of the keyed shaft that holds the wheel in place breaks	Fatigue from use	The wheel is no longer attached to the motor	The wheel becomes lodged sideways

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
Holonomic motion will no longer be possible	A - Extremely Unlikely	IV - Critical	4 - Moderate	Low
Holonomic motion will no longer be possible when that roller is in contact with the ground	C - Occasional	III - Minor	4 - Moderate	Moderate
The robot will be off balance and holonomic motion will no longer be possible	A - Extremely Unlikely	V - Catastrophic	2 - Almost Certain	Moderate

Table B.5: FMEA for a camera module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
$C_a.1$	Bad camera calibration	Lack of tools to verify calibration results	No noticeable local effects	No noticeable higher level effects
$C_a.2$	Ribbon cable to the camera becomes disconnected	Vibration	The cable drops and revels exposed leads	The computer still operates but is unable to obtain camera data
$C_a.3$	Loose power cable to the Raspberry Pi (RPi)	Vibration	The RPi continually restarts as it loses and then regains power	The RPi memory becomes corrupted

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
Data from the camera is incorrect leading to poor conclusions that results in bad navigation or diagnosis	B - Remote	III - Minor	6 - Fault is undetected	Low
Loss of camera data	A - Extremely Unlikely	IV - Critical	4 - Moderate	Low
Loss of camera data and higher-level functions/processing	B - Remote	IV - Critical	5 - Low	Moderate

Table B.6: FMEA for a control module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
C.1	Connection to XBee severed	Vibration and/or loose connector	Exposed leads in the module	Cannot communicate with other robots
C.2	Loose RX wire	Vibration since these are only placed into the Atmega board headers	Exposed wire in the module	Unable to receive camera data or module identifications
C.3	Loose TX wire	Vibration since these are only placed into the Atmega board headers	Exposed wire in the module	Unable to send commands to other modules

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
Other agents may believe this robot is nonoperational	A - Extremely Unlikely	III - Minor	5 - Low	Low
The robot will not be able to navigate or identify other agents	B - Remote	III - Minor	5 - Low	Low
The robot will not be able to move	B - Remote	III - Minor	5 - Low	Low

Table B.7: FMEA for a power module in the HexDMR III System

Reference Number	Potential Failure Mode	Potential Cause(s) Mechanisms	Local Effects of Failure	Next Higher Level Effect
P.1	The battery expends all of its charge	Normal use of the robot	No immediate local effects	No immediate higher level effects
P.2	As the battery ages it fails to hold as much charge	Expected behavior over time	No immediate local effects	No immediate higher level effects
P.3	The battery is shorted	The male connections on the back of the module are accidentally connected, even momentarily	The battery swells or explodes	The module is no longer be usable and cannot be removed

System Level End Effect	(P) Probability (Estimate)	(S) Severity	(D) Detection (Indications to Operator, Maintainer)	Risk Level P*S (+D)
All other modules cease functioning	E - Frequent	V - Catastrophic	1 - Certain	Unacceptable
The expected runtime of the robot is shortened	C - Occasional	III - Minor	1 - Certain	Moderate
All other modules cease functioning and may be damaged	A - Extremely Unlikely	V - Catastrophic	1 - Certain	Moderate

Bibliography

[1] J. D. Davis, M. D. M. Kutzer, and G. S. Chirikjian, "Algorithms for Multilayer Conformal Additive Manufacturing," in *Proceedings of the ASME International Design Engineering Technical Conferences (IDETC) and Computers and Information in Engineering (CIE) Conference*, 2015, p. V01AT02A028.

[2] J. D. Davis, M. D. M. Kutzer, and G. S. Chirikjian, "Algorithms for Multilayer Conformal Additive Manufacturing," *Journal of Computing and Information Science in Engineering*, vol. 16, no. 2, p. 021003, 2016.

[3] J. Davis, M. Kutzer, G. Chirikjian, P. Biermann, R. Matteson, and R. Forrest, "Systems and Methods for Conformal Additive Manufacturing," U.S. Patent Application 15/078,668, Sep 29, 2016.

[4] J. D. Davis, Y. Sevimli, B. R. Eldridge, and G. S. Chirikjian, "Module Design and Functionally Non-Isomorphic Configurations of the Hex-DMR II System," in *Proceedings of the ASME IDETC/CIE Conference*, 2015, p. V05BT08A018.

BIBLIOGRAPHY

[5] J. D. Davis, Y. Sevimli, B. R. Eldridge, and G. S. Chirikjian, "Module Design and Functionally Non-Isomorphic Configurations of the Hex-DMR II System," *ASME Journal of Mechanisms and Robotics*, vol. 8, no. 5, p. 051008, 2016.

[6] J. D. Davis, Y. Sevimli, M. K. Ackerman, and G. S. Chirikjian, "A Robot Capable of Autonomous Robotic Team Repair: The Hex-DMR II System," in *Advances in Reconfigurable Mechanisms and Robots II.* Springer, 2016, ch. 53, pp. 619–631.

[7] T. Wohlers and T. Gornet, "History of Additive Manufacturing," Wohlers Associates, Wohlers Report 24, 2016.

[8] H. Kodama, "Automatic Method for Fabricating a Three Dimensional Plastic Model with Photo Hardening Polymer," *Review of Scientific Instruments*, vol. 52, no. 11, pp. 1770–1773, 1981.

[9] C. W. Hull, "Apparatus forProduction of Three-dimensional Objects by Stereolithography," U.S. Patent 4,575,330, Mar 11, 1986.

[10] M. Feygin, "Apparatus and Method for Forming an Integral Object from Laminations," U.S. Patent 4,752,352, June 21, 1988.

[11] C. Deckard, "Method and Apparatus for Producing Parts by Selective Sintering," U.S. Patent 4,863,538, Sep 5, 1989.

[12] E. Sachs, J. Haggerty, M. Cima, and P. Williams, "Three-dimensional Printing Techniques," U.S. Patent 5,204,055, Apr 20, 1993.

[13] *ASTM F2792 - 12a, Standard Terminology for Additive Manufacturing Technologies*, ASTM International Std.

[14] L. B. Kohlmann and J. Lambeth, "Providing Rapid Response Solutions for the Fleet through 3D Printing," April 2014. [Online]. Available: http://blog.usni.org/2014/04/23/providing-rapid-response-solutions-for-the-fleet-through-3d-printing

[15] S. J. Freedberg Jr., "Navy Warship is Taking 3D Printer to Sea; Don't Expect a Revolution," April 2014. [Online]. Available: http://breakingdefense.com/2014/04/navy-carrier-is-taking-3d-printer-to-sea-dont-expect-a-revolution/

[16] K. Cooper, "3D Printing in Zero-G Technology Demonstration," Nov. 2014. [Online]. Available: http://www.nasa.gov/mission_pages/station/research/experiments/1115.html

[17] B. Compton and J. Lewis, "3D Printing: 3D-Printing of Lightweight Cellular Composites," *Advanced Materials*, vol. 26, no. 34, p. 6043, Sept. 2014.

[18] M. Liang, X. Yu, C. Shemelya, D. Roberson, E. MacDonald, R. Wicker, and H. Xin, "Electromagnetic Materials of Artificially Controlled Properties for 3D

Printing Applications," in *IEEE Antennas and Propagation Society International Symposium*, July 2014, pp. 227–228.

[19] D. Kolesky, R. Truby, A. Gladman, T. Busbee, K. Homan, and J. Lewis, "Bioprinting: 3D Bioprinting of Vascularized, Heterogeneous Cell-Laden Tissue Constructs," *Advanced Materials*, vol. 26, no. 19, p. 2966, May 2014.

[20] M. Hofmann, "3D Printing Gets a Boost and Opportunities with Polymer Materials," *ACS Macro Letters*, vol. 3, no. 4, pp. 382–386, April 2014.

[21] I. Gibson, R. D.W., and B. Stucker, *Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing*. Springer-Verlag, 2010.

[22] D. Radtke and U. Zeitner, "Laser-lithography on Non-planar Surfaces," *Optics Express*, vol. 15, no. 3, pp. 1167–1174, 2007.

[23] Y. Xie, Z. Lu, F. Jingli, Z. Yongjun, J. Zhao, and Z. Weng, "Lithographic Fabrication of Large Diffractive Optical Elements on a Concave Lens Surface," *Optics Express*, vol. 10, no. 20, pp. 1043–1047, 2002.

[24] J. Paulsen, M. Renn, K. Christenson, and R. Plourde, "Printing Conformal Electronics on 3D Structures with Aerosol Jet Technology," in *Future of Instrumentation International Workshop*, Oct 2012, pp. 1–4.

[25] J. J. Adams, E. B. Duoss, T. F. Malkowski, M. J. Motala, B. Y. Ahn, R. G.

Nuzzo, J. T. Bernhard, and J. A. Lewis, "Conformal Printing of Electrically Small Antennas on Three-Dimensional Surfaces," *Advanced Materials*, vol. 23, no. 11, pp. 1335–1340, 2011.

[26] M. Vatani, E. Engeberg, and J. Choi, "Conformal Direct-Print Piezoresistive Polymer/Nanocomposites for Compliant Multi-Layer Tactile Sensors," *Additive Manufacturing*, vol. 7, no. 1, pp. 73–82, 2015.

[27] O. Diegel, S. Singamneni, B. Huang, and I. Gibson, "Curved Layer Fused Deposition Modeling in Conductive Polymer Additive Manufacturing," *Advanced Materials Research*, vol. 199, no. 1, pp. 1984–1987, 2011.

[28] S. Singamneni, A. Roychoudhury, O. Diegel, and B. Huang, "Modeling and Evaluation of Curved Layer Fused Deposition," *Journal of Materials Processing Technology*, vol. 212, no. 1, pp. 27–35, 2012.

[29] J. Hoschek, "Offset Curves in the Plane," *Computer Aided Geometric Design*, vol. 17, no. 2, pp. 77–82, 1985.

[30] R. T. Farouki and C. A. Neff, "Analytic Properties of Plane Offset Curves," *Computer Aided Geometric Design*, vol. 7, no. 1-4, pp. 83–99, 1990.

[31] R. T. Farouki and C. A. Neff, "Algebraic Properties of Plane Offset Curves," *Computer Aided Geometric Design*, vol. 7, no. 1-4, pp. 101–127, 1990.

[32] M. S. Kim, E. J. Park, and S. B. Lim, "Approximation of Variable-Radius Offset Curves and its Application to Bézier Brush-Stroke Design," *Computer Aided Geometric Design*, vol. 25, no. 11, pp. 684–698, 1993.

[33] J. Kao and F. Prinz, "Optimal Motion Planning for Deposition in Layered Manufacturing," in *Proceedings of the ASME IDETC/CIE Conference*, 1998.

[34] C. I. Connolly and J. B. Burns, "Path Planning Using Laplace's Equation," in *Proceedings of the IEEE International Conference on Robots and Automation (ICRA)*, 1990, pp. 2102–2106.

[35] J. Kim and P. K. Khosla, "Real-Time Obstacle Avoidance Using Harmonic Potential Functions," *IEEE Transactions on Robotic and Automation*, vol. 8, no. 3, pp. 338–349, 1992.

[36] D. Keymeulen and J. Decuyper, "The Fluid Dynamics Applied to Mobile Robot Motion: The Stream Field Method," in *Proceedings of the IEEE ICRA*, 1994, pp. 378–385.

[37] J. Guldner, V. I. Utkin, and H. Hideki, "Robot Obstacle Avoidance in n-Dimensional Space Using Planar Harmonic Artificial Potential Fields," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 119, no. 2, pp. 160–166, 1997.

[38] Z. Nehari, *Conformal Mapping*. Dover Publications, 1952.

BIBLIOGRAPHY

[39] Stratasys, "Frequently Asked Questions," Dec. 2014. [Online]. Available: http://www.stratasys.com/3d-printers/technologies/fdm-technology/faqs

[40] Stratasys, "PolyJet Technology," Dec. 2014. [Online]. Available: http://www.stratasys.com/3d-printers/technologies/polyjet-technology

[41] EOS, "EOS M 280," Dec. 2014. [Online]. Available: http://www.eos.info/systems_solutions/metal/systems_equipment/eosint_m280

[42] 3DSystems, "SLA Production 3D Printers," Dec. 2014. [Online]. Available: http://www.3dsystems.com/sites/www.3dsystems.com/files/sla-1113-usen-web_1.pdf

[43] MakerBot, "MakerBot Replicator Mini," Dec. 2014. [Online]. Available: https://store.makerbot.com/replicator-mini

[44] 3DSystems, "Cube," Dec. 2014. [Online]. Available: http://cubify.com/en/Cube

[45] S. Nelaturi, W. Kim, and T. Kurtoglu, "Manufacturability Feedback and Model Correction for Additive Manufacturing," *ASME Journal of Manufacturing Science and Engineering*, vol. 137, no. 2, pp. 021 015–1 – 021 015–9, 2015.

[46] F. Fritsch and R. Carlson, "Monotone Piecewise Cubic Interpolation," *Optics Express*, vol. 17, no. 2, pp. 238–246, 1980.

[47] E. Rimon and D. Koditschek, "Exact Robot Navigation Using Artificial Potential Functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[48] M. D. M. Kutzer and L. D. DeVries, "Testbed for Multilayer Conformal Additive Manufacturing," *Technologies*, vol. 5, no. 2, p. 25, 2017.

[49] L. E. Parker, *Reliability and Fault Tolerance in Collective Robot Systems*, ser. Handbook on Collective Robotics: Fundamentals and Challenges. Pan Stanford Publishing, 2012.

[50] N. Hoang and H. Kang, "A Model-based Fault Diagnosis Scheme for Wheeled Mobile Robots," *ASME International Journal of Control, Automation, and Systems*, vol. 12, no. 3, pp. 637–651, 2014.

[51] M. Brandstotter, M. Hofbaur, G. Steinbauer, and F. Wotawa, "Model-based Fault Diagnosis and Reconfiguration of Robot Drives," in *Proceedings of the IEEE/RSJ International Conference on IROS*, 2007, pp. 1203–1209.

[52] Z. Cai, Z. Duan, J. Cai, X. Zou, and J. Yu, "A Multiple Particle Filters Method for Fault Diagnosis of Mobile Robot Dead-reckoning System," in *Proceedings of the IEEE/RSJ International Conference on IROS*, 2005, pp. 481–486.

[53] P. Goel, G. Dedeoglu, S. Roumeliotis, and G. Sukhatme, "Fault Detection and

Identification in a Mobile Robot Using Multiple Model Estimation and Neural Network," in *Proceedings of the IEEE ICRA*, 2000, pp. 2302–2309.

[54] P. Robertson and B. Williams, "Automatic Recovery from Software Failure," *Communications of the ACM*, vol. 49, no. 3, pp. 41–47, 2006.

[55] G. Steinbauer, M. Mörth, and F. Wotawa, "Real-time Diagnosis and Repair of Faults of Robot Control Software," in *Proceedings of the 2005 Robot Soccer World Cup*, 2005, pp. 13–23.

[56] M. Schwager, P. Dames, D. R. Rus, and V. Kumar, "A Multi-robot Control Policy for Information Gathering in the Presence of Unknown Hazards," in *Proceedings of the International Symposium on Robotics Research*, 2011.

[57] A. Howard, L. E. Parker, and G. S. Sukhatme, "Experiments with a Large Heterogeneous Mobile Robot Team: Exploration, Mapping, Deployment and Detection," *International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 431–447, 2013.

[58] T. M. Roehr, F. Cordes, and F. Kirchner, "Reconfigurable Integrated Multirobot Exploration System: Heterogeneous Modular Reconfigurable Robots for Space Exploration," *Journal of Field Robotics*, vol. 31, no. 1, pp. 3–34, 2013.

[59] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng, "Cooperative Mobile Robotics: Antecedents and Directions," *Autonomous Robots*, vol. 4, no. 1, pp. 7–27, 1997.

[60] D. Kim, "Self-organization of Unicycle Swarm Robots Based on a Modified Particle Swarm Framework," *International Journal of Control, Automation and Systems*, vol. 8, no. 3, pp. 622–629, 2010.

[61] H. Yamaguchi, T. Arai, and G. Beni, "A Distributed Control Scheme for Multiple Robotic Vehicles to Make Group Formations," *Robotics and Autonomous Systems*, vol. 36, no. 4, pp. 125–147, 2001.

[62] A. Loria, J. Dasdemir, and N. Jarquin, "Leaderfollower Formation and Tracking Control of Mobile Robots Along Straight Paths," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 2, pp. 727–732, 2016.

[63] M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal, "Kilobot: A Low Cost Robot with Scalable Operations Designed for Collective Behaviors," *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 966–975, 2014.

[64] J. D. Bjerknes and A. F. T. Winfield, "On Fault Tolerance and Scalability of Swarm Robotic Systems," *Proceedings of the International Symposium on DARS*, vol. 83, pp. 431–444, 2013.

[65] J. Davey, N. Kwok, and M. Yim, "Emulating Self-reconfigurable Robots - Design of the SMORES System," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 4148–4153.

[66] J. Zhao, X. Cui, Y. Zhu, and S. Tang, "A New Self-Reconfigurable Modular Robotic System UBot: Multi-Mode Locomotion and Self-Reconfiguration," in *Proceedings of the IEEE ICRA*, 2011, pp. 1020–1025.

[67] J. Baca, S. G. M. Hossain, P. Dasgupta, C. A. Nelson, and A. Dutta, "ModRED: Hardware Design and Reconfiguration Planning for a High Dexterity Modular Self- Reconfigurable Robot for Extra-Terrestrial Exploration," *Autonomous Systems*, vol. 62, no. 7, pp. 1002–1015, 2014.

[68] M. D. M. Kutzer, M. S. Moses, C. Y. Brown, A. Armand, D. H. Scheidt, and G. S. Chirikjian, "Design of a New Independently-mobile Reconfigurable Modular Robot," in *Proceedings of the IEEE ICRA*, 2010, pp. 2758–2764.

[69] H. Kurokawa, A. Kamimura, and K. Tomita, "Self-Assembly and Self-Reproduction by a M-TRAN Modular Robotic System," in *Proceedings of the International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2012, pp. 205–218.

[70] A. Spröwitz, R. Moeckel, M. Vespignani, S. Bonardi, and I. A. J., "Roombots: A Hardware Perspective on 3D Self-Reconfiguration and Locomotion with a Homogeneous Modular Robot," *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 1016–1033, 2014.

[71] K. Gilpin, K. Kotay, D. Rus, and I. Vasilescu, "Miche: Modular Shape Formation

by Self-disassembly," *International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 345–372, 2008.

[72] H. Wei, Y. Chen, J. Tan, and T. Wang, "Sambot: A Self-Assembly Modular Robot System," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 4, pp. 745–757, 2011.

[73] J. W. Romanishin, K. Gilpin, and R. D, "M-blocks: Momentum-drive, Magnetic Modular Robots," in *Proceedings of the IEEE/RSJ International Conference on IROS*, 2013, pp. 4288–4295.

[74] M. S. Moses, H. Ma, K. C. Wolfe, and G. S. Chirikjian, "An Architecture for Universal Construction Via Modular Robotic Components," *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 945–965, 2014.

[75] S. Yim and M. Sitti, "Soft Cubes: Towards a Soft Modular Matter," in *Proceedings of the IEEE ICRA*, 2013, pp. 530–536.

[76] P. Pagala, M. Ferre, and M. Armada, "Design of Modular Robot System for Maintenance Tasks in Hazardous Facilities and Environments," in *Proceedings of the First Iberian Robotics Conference*, 2014, pp. 185–197.

[77] A. H. Lyder, K. Stoy, R. F. M. Gárcia, J. C. Larsen, and P. Hermansen, "On Sub-modularization and Morphological Heterogeneity in Modular Robotics," in

Proceedings of the 12th International Conference on Intelligent Autonomous Systems, 2013, pp. 649–661.

[78] S. Kernbach, F. Schlachter, R. Humza, J. Liedke, S. Popesku, S. Russo, T. Ranzani, L. Manfredi, C. Stefanini, R. Matthias, and C. Schwarzer, "Heterogeneity for Increasing Performance and Reliability of Self-Reconfigurable Multi-Robot Organisms," in *Proceedings of the IEEE/RSJ International Conference on IROS*, 2011.

[79] P. Hartono and A. Nakane, "Robotics Modules with Realtime Adaptive Technology," *International Journal of Computer Informations Systems and Industrial Management Applications*, vol. 3, pp. 185–192, 2011.

[80] E. Guan, J. Fei, G. Pan, Z. Fu, W. Yan, and Y. Zhao, "Fault Self-diagnosis for Modular Robotic Systems using M-lattice Modules," *International Journal of Advanced Robotic Systems*, vol. 12, no. 4, p. 31, 2015.

[81] M. D. M. Kutzer, M. Armand, D. H. Schied, E. Lin, and G. S. Chirikjian, "Toward Cooperative Team-diagnosis in Multi-robot Systems," *International Journal of Robotics Research*, vol. 27, no. 9, pp. 1069–1090, 2008.

[82] S. G. M. Hossain, C. A. Nelson, K. D. Chu, and P. Dasgupta, "Kinematics and Interfacing of ModRED: A Self-Healing Capable, 4DOF Modular Self-

Reconfigurable Robot," *ASME Journal of Mechanisms and Robotics*, vol. 6, no. 4, pp. 041 017–1 – 12, 2014.

[83] Y. Fei and C. Wang, "Self-Repairing Algorithm of Lattice-Type Self-Reconfigurable Modular Robots," *Journal of Intelligent and Robotic Systems*, vol. 75, no. 2, pp. 193–203, 2014.

[84] S. Abdul and G. Liu, "Decentralised Fault Tolerance and Fault Detection of Modular and Reconfigurable Robots with Joint Torque Sensing," in *Proceedings of the IEEE ICRA*, 2008, pp. 3520–3526.

[85] C. Bererton and P. K. Khosla, "An Analysis of Cooperative Repair Capabilities in a Team of Robots," in *Proceedings of the IEEE ICRA*, 2002, pp. 476–482.

[86] C. Bererton and P. K. Khosla, "Towards a Team of Robots with Repair Capabilities," in *Proceedings of the IEEE ICRA*, 2001, pp. 2923–2928.

[87] C. Bererton and P. K. Khosla, "Towards a Team of Robots with Repair Capabilities: A Visual Docking System," in *Experimental Robotics VII*, 2001, pp. 333–342.

[88] M. K. Ackerman and G. S. Chirikjian, "Hex-DMR: A Modular Robotic Test-Bed for Demonstrating Team Repair," in *Proceedings of the IEEE ICRA*, 2012, pp. 4464–4469.

[89] K. C. Wolfe, M. D. M. Kutzer, M. Armand, and G. S. Chirikjian, "Trajectory Generation and Steering Optimization for Self-Assembly of a Modular Robotic System," in *Proceedings of the IEEE ICRA*, 2010, pp. 4996–5001.

[90] "PIC16(L)F1825/1829 Data Sheet," Microchip Technology Inc., Arizona, US.

[91] I. M. Chen and J. W. Burdick, "Enumerating the Non-Isomorphic Assembly Configurations of Modular Robotic Systems," *International Journal of Robotics Research*, vol. 17, no. 7, pp. 702–719, 1998.

[92] G. Indiveri, "Swedish Wheeled Omnidirectional Mobile Robots: Kinematics Analysis and Control," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 164–171, 2009.

[93] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "ROS: an Open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009, p. 5.

[94] E. Olson, "AprilTag: A Robust and Flexible Visual Fiducial System," in *Proceedings of the IEEE ICRA*, 2011, pp. 3400–3407.

[95] B. Anderson and J. Moore, *Optimal Filtering*. Englewood Cliffs, 1979, vol. 21.

[96] S. Julier and J. Uhlmann, "New Extension of the Kalman Filter to Nonlinear Systems," in *Proc. of the 1997 Signal Processing, Sensor Fusion, and Target*

Recognition VI, vol. 3068. International Society for Optics and Photonics, 1997, pp. 182–194.

[97] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan, "The Unscented Particle Filter," Cambridge University, CUED/F-INFENG/TR-380, August 2000.

[98] D. Higham, "An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations," *SIAM Review*, vol. 43, no. 3, pp. 525–546, 2001.

[99] D. Stamatis, *Failure Mode and Effect Analysis: FMEA from Theory to Execution*, ser. Handbook on Collective Robotics: Fundamentals and Challenges. ASQ Quality Press, 2003.

Vita

Joshua David Davis was born on December 25, 1987 in New Rochelle and raised in Yorktown Heights, New York. He received his Bachelor of Science degree, with Honors, in Mechanical Engineering from the University of Maryland - College Park in 2010 and his Masters of Science in Engineering degree in Mechanical Engineering from Johns Hopkins University (JHU) in 2014.

In addition to Joshua's numerous scholarships and awards at the undergraduate level, he was a 2012 JHU Mechanical Engineering Departmental Fellowship recipient; a 2013 and 2014 ARCS Foundation Metropolitan Washington Chapter Scholar, received a 2014 NSF Graduate Research Fellowships Program (GRFP) Honorable Mention, was the recipient of a best student paper award at the 2015 ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots (ReMar), and was the recipient of the 2015 JHU Department of Mechani-

cal Engineering Creel Family Teaching Assistant Award for his work in mechatronics. His research focuses on developing methods to enable conformal additive manufacturing as well as cooperative robotic design for autonomous team repair and diagnosis. Joshua exhibited his passion for mentoring and teaching by co-leading the Johns Hopkins Robo Challenge for three years and by mentoring over seven students during his tenure at JHU.

Joshua is currently working as the Vice President of Platform for READY Robotics in Baltimore, MD, a manufacturing-based robotics startup company that spun out of Johns Hopkins in 2016, where he is developing the next generation of software to easily integrate and program robotic systems in a variety of industrial verticals.