Development of an Experimental Platform for Architectural-Scale Robotics: The Digital Construction Platform

by
Julian Leland Bell

B.S., Engineering and Public Policy, Swarthmore College, 2012

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

Master of Science in Mechanical Engineering

at the
Massachusetts Institute of Technology
September 2017

Signature of Author: ....
Department of Mechanical Engineering
August 11th, 2017

Certified by: ....
Neri Oxman
Associate Professor of Media Arts and Sciences
Thesis Co-Supervisor

Certified by: ....
David L. Trumper
Professor of Mechanical Engineering
Thesis Co-Supervisor

Accepted by: ....
Rohan Abeyaratne
Professor of Mechanical Engineering
Chairman, Committee on Graduate Students

Development of an Experimental Platform for Architectural-Scale Robotics: The Digital Construction Platform

by

Julian Leland Bell

Submitted to the Department of Mechanical Engineering
on August 18, 2017 in partial fulfilment of the requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

This thesis describes the development and refinement of the second prototype of the Digital Construction Platform, or DCP. The DCP is a serial-link micro-macro manipulator robot intended for architectural-scale fabrication tasks, originally conceived of and presented by Keating in [1]. It is envisioned primarily as a *platform* for experimentation in automated construction, rather than as a closed, single-application system.

In the work described here, a second prototype of the DCP – referred to as the DCP v.2 – was developed over two distinct periods. During the first period, from September 2015 through August 2016, the DCP v.2 system was assembled and a basic command and control architecture was developed to operate it. A series of experiments were conducted to examine the system's performance, including pose repeatability testing in accordance with the ISO 9283-1998 robot performance characterization standard; and the fabrication of an architectural-scale dome structure from spray polyurethane foam.

During the second period, from September 2016 through August 2016, the DCP v.2 system and command/control architecture were modified in a variety of ways to improve performance, reliability, accessibility to new users, and adaptability to new tasks. These modifications included transition to a modular, hard-real-time control architecture; installation of additional sensor systems on the vehicle; and the refinement and standardization of the system's toolpath generation architecture. The impact of this work was demonstrated through a second set of demonstrations, including large-scale light paintings leveraging the new control architecture's capabilities; and re-characterization of the system's ISO 9283 pose repeatability, demonstrating a 59% improvement in this metric.

Thesis Co-Supervisor: Dr. Neri Oxman
Title: Associate Professor of Media Arts and Sciences, Program in Media Arts and Sciences

Thesis Co-Supervisor: Dr. David Trumper
Title: Professor of Mechanical Engineering, Department of Mechanical Engineering

Acknowledgements

My path through this project begins with the DCP team. Steve Keating is the original inventor of the DCP and Print-in-Place concepts – basically, all the really good ideas here – and led the DCP project through August 2016. It is thanks to him that I have had the opportunity to be a part of this incredible project, and to help turn these visions into reality: I am deeply grateful. Along the way, many other people have lent their talents to the project, including Levi Cai, Selam Gano, John Zhang, Barrak Darweesh, Owen Trueblood, Damien Martin, and Grant Sellers. Thank you all for your help, patience, guidance and good humour.

Second, I would like to acknowledge my advisors: Dr. Neri Oxman and Dr. David Trumper. In coming to MIT, I wanted to grow as an engineer, but also to reach beyond my discipline in my work. The two of you have made this possible, enabling me to pursue a mechanical engineering degree while working on the DCP project, and providing equal parts technical rigor and multidisciplinary grace in your mentorship. It has been an honor to learn from each of you.

The Mediated Matter Group has also played an instrumental role in helping me develop beyond being "just an engineer." I doubt I will ever again have the opportunity to surround myself with such intellectual diversity, and I cannot thank you all enough for inviting me to be a part of this rare team.

MIT is much more than just students and professors. I would particularly like to thank the many shop masters I have bothered over the past two years, including John DiFrancesco and Tom Lutz at the Media Lab shop; Tara Fadenrecht at the DMSE shop; Mark Belanger at Edgerton; and Jen O'Brien at the Architecture shop. Additionally, as with any large institution there are a small set of people at MIT who do a huge amount to keep the entire operation running smoothly: Leslie Reagan in Mechanical Engineering, Keira Horowitz at the Media Lab, and the indomitable Kelly Donovan at Mediated Matter. Thank you all for your patience, help and kindness.

A range of companies and organizations have helped make my work with the DCP possible through their support, including the following:

- Altec, Inc.: I am particularly grateful to David Boger, whose persistent belief in our vision for the DCP has made realizing that vision possible. I would also like to thank the many Altec employees who have generously given their time to support our work, including Ali Khodadadi, Jessica Patel, Dan Nelson, Mike Whittaker, Jesse Thompson, and Tim Mourlam.

- Dow Chemical: Dow Chemical generously donated the FROTH-PAK™ spray foam insulation used in the dome print. I would like to particularly thank EJ Herst, Sven Claessens, and Ed Greer for their support in making this happen.

- Autodesk: Finally, for the past year, the DCP has been housed at the Autodesk BUILD Space. I would like to thank Rick Rundell, BUILD Space mastermind, for giving me the opportunity to work in this amazing environment. I would also like to thank the incomparable BUILD Space shop team for their support and kindness, including Adam Allard, Athena Moore, Taylor Tobin, Tim Brinkerhoff, Joe Aronis, and Josh Aigen.

I have been lucky to have a team of wise engineers as mentors and advisors who have helped me get to – and through – MIT, including Dr. Faruq Siddiqui, Dr. Lynne Molter, Dr. Erik Cheever, Dr. Bill Townsend, Dave Wilkinson, and Scott Harris. Your counsel and support is invaluable, and I look to your example as I begin to mentor the next generation of engineers.

From an early age, I was taught that education is beyond value. Thanks to the hard work and sacrifice of my parents and grandparents, I have had the privilege to learn at some of the finest institutions in the world, for which I cannot overstate my gratitude. I am particularly grateful to my parents, Annette and Jonathan Leland, who have always given me their unwavering support throughout my educational journey.

Finally, to my wife, Avery Davis Bell. The life we continue to build together is my finest project, and I could not ask for a better partner to build it with.

Author's Note

This thesis has turned out to be quite long, which I think is unfortunately more of a reflection of my limitations as a writer than the magnitude of my contributions. However, this length allows for a corresponding degree of detail to describe the work I have done on the Digital Construction Platform (DCP). Particularly for a project where my primary objective has been to provide a tool that others can use, my hope is that this detail will prove useful for future users and developers of the DCP system.

In light of this, there are two ways this thesis may be read:

- For the reader who is interested in learning about the DCP and its contributions to the field of automated construction, I recommend reading Chapter 1, followed by the System Implementations sections in Chapters 2 and 3. This provides a brief review of the history of automated construction as well as the early development of the DCP concept; followed by detailed descriptions of the tasks the DCP v.2 system has been used for. The reader may also wish to refer to [2] and [3] for further information about the project and its history.

- Future users of the DCP v.2 platform are advised to skip directly to Chapter 3, and use it as a guide to explore the dcpctrl_v2 repository, which hosts the DCP's software architecture. Other sections of this thesis may be useful as reference sources, but this chapter contains the material most relevant to the DCP as it exists in August 2017.

I hope that this is helpful.

This thesis was formatted using the *LaTeX 7* template by Sebastian Nilsson [4].

Dedicated to the memory of my grandfather,
Benjamin Towne Leland,
engineer, aviator and eternally curious mechanical mind,
who would have thought this project was pretty nifty.

Table of Contents

List of Figures

List of Tables

1 Introduction

This chapter develops the technical landscape in which the DCP project exists, and describes the prior work that inspired and informed the development of the DCP v.2 prototype. We first provide a brief introduction to the field of automated construction: the domain at the intersection of robotics, machine design, architecture and structural engineering that this thesis lies within. This introduction first surveys historical and contemporary work in the automated construction field, with a particular focus on the range of machine designs that have been developed for automated construction tasks. We then attempt to develop categorizations within automated construction: between schools of thought within the automated construction field, and between different automated construction systems. We then focus specifically on the technologies that make up the DCP, describing the history of the micro-macro serial-link manipulator arm concept used in the DCP; and detailing previous work done at the Mediated Matter Group by Keating on the DCP v.1 system [1], and developing the Print-in-Place manufacturing technique [5].

This chapter incorporates elements previously published in Keating et. al. [2], which have been adapted for this thesis.

1.1 Background

1.1.1 Historical Efforts in Automated Construction

Leveraging the tools and techniques of automation on the construction site has been a goal of researchers in architecture, construction technology, civil engineering and more for decades. The current thrust of research into automated construction systems – referred to in this thesis as ACS – is generally regarded as having originated in the late 1970s in Japan, where major collaborative efforts between academics and both the robotics and construction industries resulted in the development of hundreds of different systems over the following decade ([6]–[8]). As described by Hasegawa in [9], the 1980s also saw the development of similar systems around the world, particularly in Germany and France.

A particularly instructive snapshot of ACS from this era of development was published by the International Association for Automation and Robotics in Construction (IAARC) in 1998 as "Robots and Automated Machines in Construction" [10]. This catalog contains more than 80 different listings, categorized by the type of task each machine is designed for. This catalogue is by no means exhaustive; it lists only systems from the Japanese, German and Swedish markets, and generally excludes systems developed in academic and research laboratories. However, it provides a few useful insights into the state of automated construction at the end of the 20th century:

- Discrete task segmentation, small scales: The vast majority of the ACS described in [10] are single-purpose systems, designed to execute a specific, highly specialized task like concrete slab finishing or teleoperated pile digging. Generally, these are small, peripheral subtasks within the larger construction process, rather than a core construction task like foundation establishment or wall erection. In the context of modern ACS research, these systems are better considered as construction robots rather than true automated construction systems.

- "Building factories": The notable exception to this are the "building factory" automated construction systems developed in Japan. As detailed by Smith in [8], these systems are structured either as "...robotic systems which form a systematic 'factory' that is stationary or fixed in the context of the site; and robotic systems which form a systematic factory that moves itself along as it completes portions of the building." Examples of these systems include the AMURAD Construction System (Kajima Corporation), which "extrudes" a high-rise building up from an automated cell located at the base of the building; and the Automated Building Construction System/ABCS (Obayashi Corporation), where the automated cell climbs the structure of the building as it erects successive floors. These systems are generally used to complete the majority of structural fabrication tasks involved in constructing a building, along with some infrastructural operations (plumbing, wiring) and finishing tasks. They are generally not fully automated systems, but rather require some amount of control/support on the part of human workers. For example, in the ABCS system, Bock reports that Obayashi found that "...human welders were faster (than the integrated welding robots) in conducting simple welding operations...(and) for more complex welding operations, they were...a necessity." [11]

1.1.2 Contemporary Automated Construction

In the past decade, the focus of research into automated construction systems in the United States and Europe has shifted to a new middle ground between the "construction robots" and "building factories" described by Smith. This latest generation of systems takes advantage of recent advances in robotics – particularly in localization and mapping, and in collaborative robotics – to build smaller, standalone robotic systems that are capable of executing major portions of the construction process autonomously.

There are a wide range of metrics that can be used to categorize contemporary ACS:

- The fabrication process used by a given ACS is the most obvious metric available. Most ACS use processes that can be grouped generally into either material extrusion processes, or assembly processes [2]. However, within these categorizations, there is a huge range of diversity present. Systems may extrude structural material directly, extrude supports or formworks for structural material, or perform a hybrid process. Similarly, assembly processes have been implemented using brick [12], wood [13], foam blocks [14] and woven tensile elements [15]. We do not exhaustively detail the wide range of existing fabrication systems in this thesis: instead, the reader is recommended to one of the survey resources listed at the end of this section.

- Alternatively, characterizing the fabrication *intent* of a given ACS as functional or aesthetic provides another useful tool for categorization. The canonical examples of functional vs. aesthetic ACS are the Contour Crafting process and system developed by Khoshnevis ([16], [17]) and the D-Shape process and system developed by Dini [18].

The Contour Crafting process directly extrudes a “filament” of cementitious material in a 2.5D fabrication process similar to conventional thermoplastic fused-filament fabrication (FFF) to create a formworks, which conventional concrete is then poured into (Figure 1). The goal of the Contour Crafting process is the direct fabrication of structural elements: while the aesthetics of the final structure are obviously a consideration for the Contour Crafting process, the limitations imposed by the 2.5D fabrication system and intrinsic to the Contour Crafting process necessarily limit the available architectural design space [19].

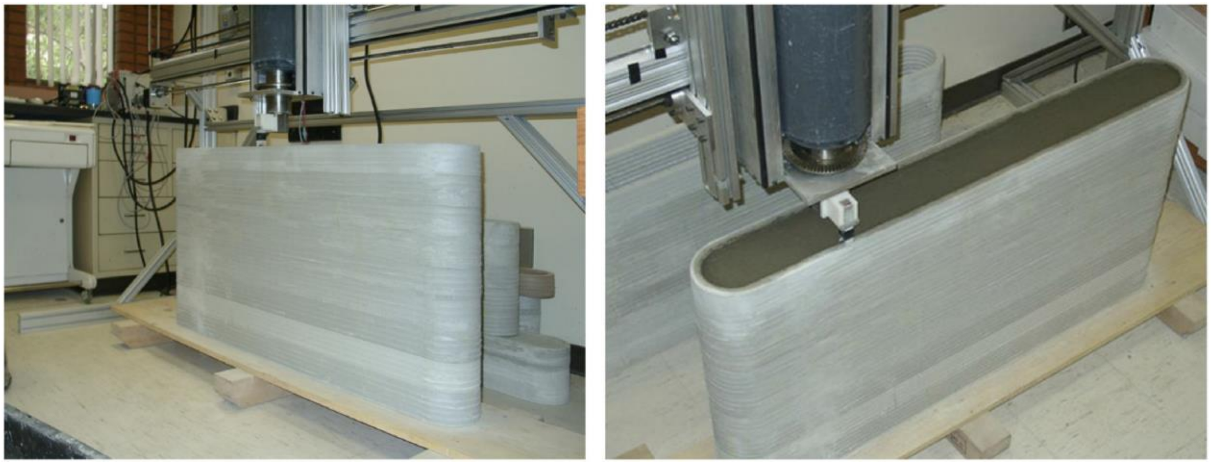


Figure 1: Contour Crafting Wall [20]

Meanwhile, the D-Shape process implements a selective powder-binding process, similar to thermoplastic selective laser sintering (SLS) processes, but at substantially larger scales [21]. The D-Shape process offers impressive mechanical properties, high feature resolution, and tremendous geometric complexity. However, the shortcomings of the process – particularly the slow fabrication rate, and fundamental limitations on print size – have largely relegated the process to aesthetic tasks such as the creation of largely sculptural structures like the *Radiolaria* series (Figure 2), or the Alcobendas Footbridge (Figure 3).

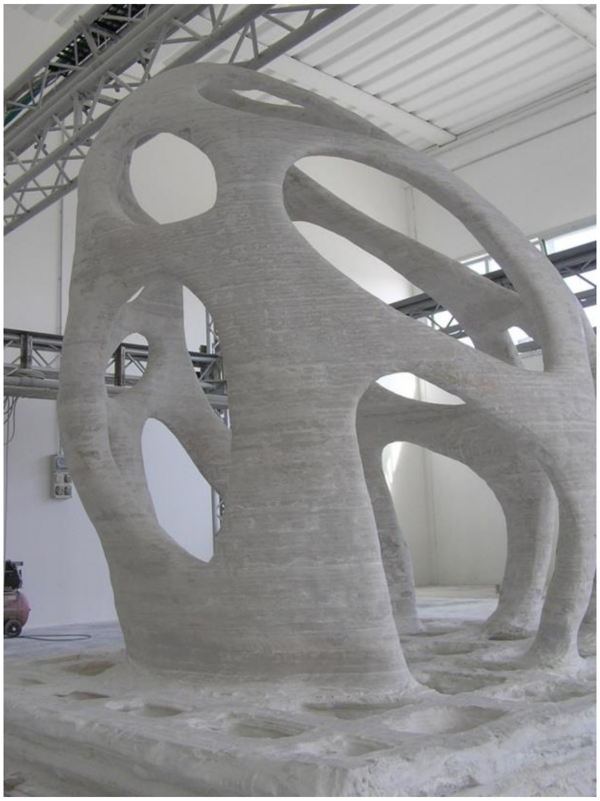


Figure 2: D-Shape *Radiolara* Sculpture[1]

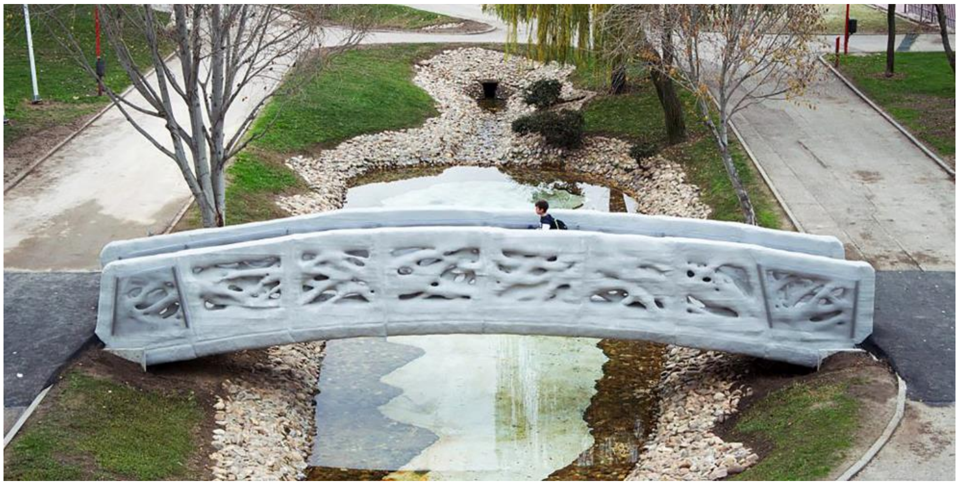


Figure 3: Alcobendas Footbridge[2]

[1] Image from: http://shiro-studio.com/i/size/1300x/http://shiro-studio.com/images/jpeg/new/Radiol_05.jpg. Accessed 2017-07-27

[2] Image from: https://www.designboom.com/wp-content/uploads/2017/01/catalonia-3D-print-bridge-designboom-header.jpg. Accessed 2017-07-27.

This categorization is obviously more qualitative and subjective than categorization based on a physical feature of a given ACS, but still provides useful insight into different schools of thought within contemporary ACS development. As examples of how it may be applied to different systems, the *Sequential Roof* project at ETH Zurich (Figure 4, [22]), executed using the Guedel gantry robot developed collaboratively between ERNE AG and ETH Zurich, represents a primarily functional ACS: its product is primarily a structural element, and the freedom of geometry available through this process is relatively limited. Meanwhile, the Gantenbein Vineyard Façade, fabricated by Gramazio & Kohler (Figure 5, [21]) prioritizes aesthetics over structural functionality: the brick façade does not carry structural loads, but provides a striking visual effect that could not easily be achieved without the use of an automated system.

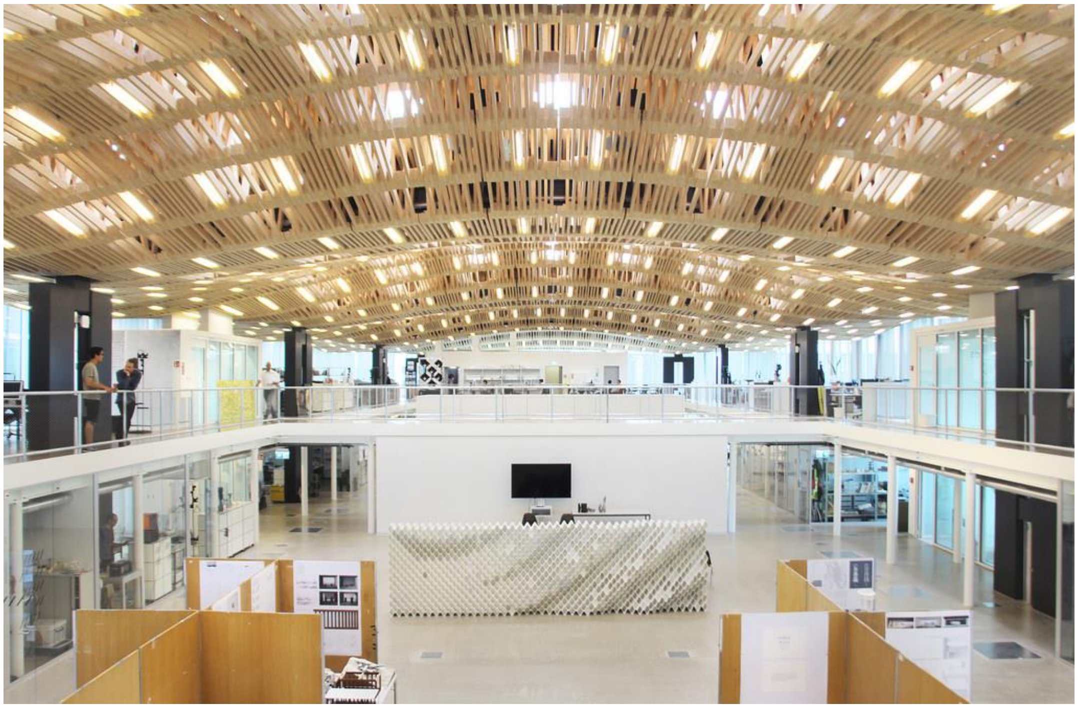


Figure 4: The Sequential Roof[3]

[3] Image from: https://www.flickr.com/photos/trevorpatt/34337822063/in/photostream/. Accessed 2017-07-27. CC BY-NC-SA

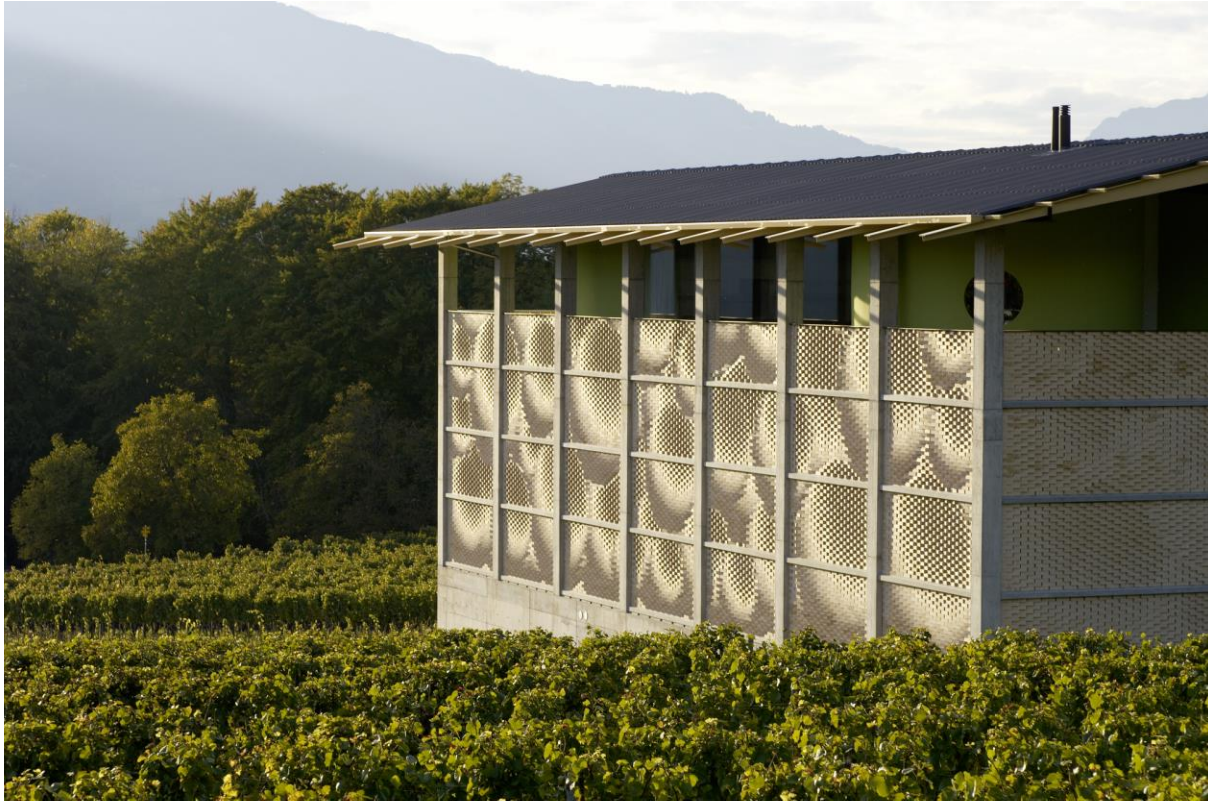


Figure 5: Gantenbein Vineyard Façade[4]

For the purposes of this thesis, however, the most useful metric for categorizing ACS is kinematic structure. Kinematic structure is one of the central properties of an ACS, defining the system's available work volume; the kinematic flexibility[5] of the system; its load capacity; the complexity associated with its design, fabrication, and control; and more. Furthermore, classification by kinematic structure is largely robust. Unlike manufacturing processes, which can often be trivially adapted between different ACS, a device's kinematic structure is almost always fixed. Finally, classification by kinematic structure is generally quick and intuitive, not requiring in-depth knowledge of specifics of process or device design.

[4] Image from: http://images.adsttc.com/media/images/501f/4a37/28ba/0d02/4200/0054/large_jpg/stringio.jpg?1414257261. Accessed 2017-07-27.

[5] Instead of the traditional robotics definition, we use kinematic flexibility here to indicate the range of pose control available to an ACS: whether it can just control position; control position and orientation; or even leverage joint redundancy to perform the same task in multiple poses.

However, these advantages do not necessarily mean that classification by kinematic structure is always trivial. In [21], Labonnote et. al. segment ACS into five categories: 1) gantry-style systems; 2) cable-suspended systems; 3) swarm-based systems; 4), multi-purpose robotics, and specifically robot arms; and 5) self-folding or self-assembling systems. Labonnote provides these classifications specifically in the context of additive manufacturing systems, but they are equally appropriate for non-additive ACS. In [2], we simplify Labonnote's categorizations from five to three – gantries, aerial drone systems, and robotic arms – by collapsing cable-suspended systems into gantries, and discarding self-folding and self-assembling systems, which have largely not seen implementation at true architectural scales. This categorization emphasizes intuitiveness, particularly for the broad scientific audience that [2] targets. However, it excludes a number of important systems, and its framework is arguably insufficiently rigorous.

In this thesis, we take inspiration from machine design terminology, and instead propose a slight modification to the classification described in [2]: between systems with 1) largely parallel kinematic structures; 2) largely serial kinematic structures, and 3) plural kinematic structures. We believe that this classification method more accurately separates systems by the degree of kinematic flexibility available to them. For instance, a gantry and a suspended-cable printing system are both 2.5D fabrication operations, and will frequently encounter kinematic conflicts when multiple systems are operating in the same workspace. A serial-link arm does not suffer from these limitations to the same degree, but will typically have a substantially reduced mass to capacity ratio in comparisons to the parallel kinematic systems. Examples of each category are given below to elucidate the distinction:

- Parallel kinematic structure: In a system with a parallel kinematic structure, multiple substantial closed paths between the workpiece or work surface and the end-effector can be drawn. This class includes traditional Cartesian gantry systems like the printer systems used by the D-Shape and Contour Crafting projects; suspended cable systems like the NIST RoboCrane [23]; as well as robotic manipulators with parallel kinematics like the WASP BigDelta [21] printer.

- Serial kinematic structure: In a system with a serial kinematic structure, only a single substantial closed path between the workpiece and end-effector can be drawn. This class is primarily made up of serial-link robotic manipulator arms using exclusively revolute joints like the In-Situ Fabricator [24] and DCP v.1 system [1], but can also include serial systems with other joint configurations.

- Plural kinematic structure: We propose the term "plural kinematic structure" to capture the range of systems which incorporate multiple end-effectors, each with their own substantial closed paths to the workpiece. This class is made largely up of "drone" systems, such as the aerial drones used in the Flight-Assembled Architecture [14] and Rope Bridge [25] projects at ETH Zurich; as well as the terrestrial drone systems explored in the Minibuilders project at IAARC [26].

It is important to note the use of the word "substantially" included in all of these categorizations. Many ACS do not fall cleanly into these categories, or could be categorized differently depending on what scale they are considered at. For example, the ETH Robotic Fabrication Laboratory [27] combines small-scale industrial arm systems (which demonstrate serial kinematics) with a large-scale gantry system to carry them (a parallel kinematic structure). Similarly, mobile systems like the In-situ Fabricator [24] – which are capable of architectural-scale fabrication on their own – have been designed to operate collaboratively with other similar systems, in a plural kinematic structure. To categorize these systems, it is best to consider the scale at which the system is fabricating as well. For example, to build a relatively small structure like the Mesh Mold demonstrator described in [24], an In-situ Fabricator would likely operate alone. Here, the system is probably best characterized as a serial kinematic system, even if the In-Situ Fabricator is required to move around the structure to fabricate it. However, a team of multiple In-Situ Fabricators fabricating a larger structure – particularly if the systems are aware of the others' presence and actions, or are operating collaboratively – becomes a macro system with a plural kinematic structure.

This classification system also provides a continuum that ACS can be considered along, shown in Figure 6:

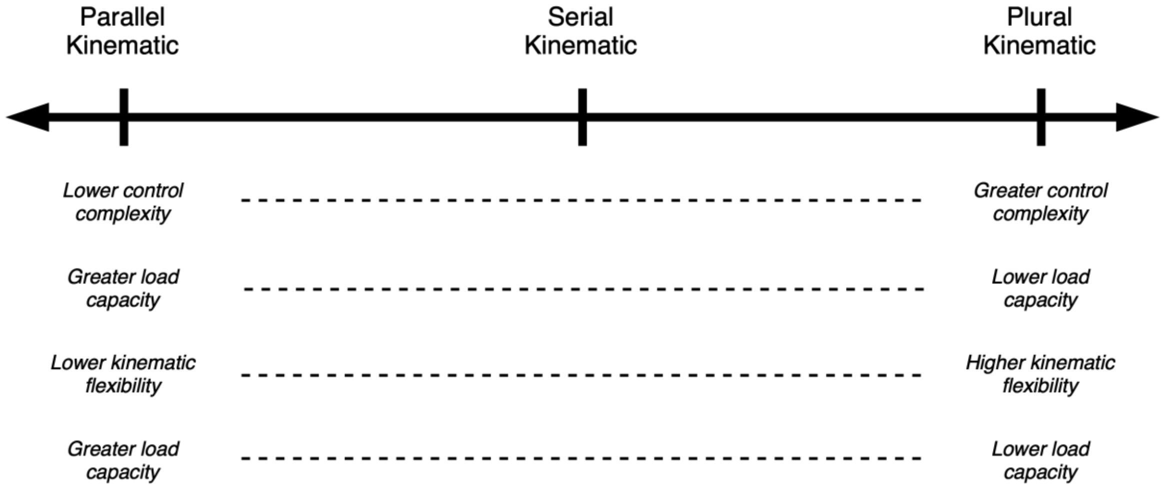


Figure 6: ACS Comparison Axes

Both parallel-kinematic and plural-kinematic systems offer certain advantages. Parallel-kinematic systems are (relatively) easily understood and controlled, and are the most widely adopted kinematic archetype in automated construction today. Plural kinematic systems have also been adopted by the construction industry, although generally in surveying rather than fabrication applications. However, for the majority of construction tasks, we believe that serial-kinematic/arm-based systems provide the optimal balance of robustness and functionality. There are a wide variety of advantages that we identify in arm-based systems, including:

- Fabrication scale: Particularly in gantry-based parallel kinematic systems, fabrication scale is severely limited. The work volume of gantry systems is fundamentally limited to some fixed volume, even if the machine can be expanded in some manner. Furthermore, the costs associated with a gantry system will scale with the size of the machine. Gantry printers that are large enough to print a full building will be tremendously expensive to build, challenging to transport and set up. Because of these limitations, most 3D-printed structures using gantries have been built using prefabricated subcomponents which are built off-site and assembled at the worksite. This introduces additional labor costs and risks to human workers, eliminating many of the promised benefits of 3D printing in construction. Arm-based systems, meanwhile, are already built at enormous scales, as serial-kinematic construction systems like the Putzmeister 70Z (69 m vertical reach) show [28]. Furthermore, most of these systems are designed to be operated from wheeled mobile platforms. The ability to leverage mobility – whether autonomous or requiring user intervention – substantially increases available work volume, making these systems well-suited to true architectural-scale tasks.

- Capability limitations: Parallel kinematic systems can only produce a limited set of geometries. Generally, they are considered 2.5D manufacturing systems, and their kinematic structure does not allow them sufficient control over pose or orientation to perform many operations, such as post-machining the underside of an overhang on a structure, or inserting a component laterally into a printed wall. It is also extremely challenging to have more than one operator/tool head operating within a gantry's workspace at a single time. For very large prints where throughput is critical – or operations requiring more than one device to complete, such as assembly tasks – we believe that parallel-kinematic systems will quickly prove inferior to serial-kinematic systems, which can effectively perform 6D manufacturing operations, as well as work collaboratively/in the same workspace with other systems more easily.

- Energy/capacity performance: We also believe that serial-kinematic systems provide an effective middle ground between parallel-kinematic and plural-kinematic systems in terms of energy efficiency and system capacity. While serial-kinematic systems generally have a substantially poorer payload-to-weight ratio than parallel-kinematic systems, existing serial kinematic construction systems still have the demonstrated capacity for many construction tasks, such as concrete pumping, worker lifting & location, and light-duty manipulation. Furthermore, when energy requirements – or efficiency – are considered, serial-kinematic systems clearly provide better performance than most plural kinematic systems (and particularly aerial drone systems).

- Historical experience: Arm-based manipulator systems have been under development for many decades, not only in traditional robotics applications, but in construction automation as well. For example, the EMIR manipulator [29] and BRONCO bricklaying robot [12] – both hydraulic, serial-link manipulator arm systems – were developed in the late 1980s and mid-1990s, respectively. This long history of development provides a rich literature to support future efforts: in many cases, the problems that arm-based automated construction systems experience have already been investigated (and in some cases, solved) at smaller scales in conventional robotics.

Before continuing on to discuss the performance of different automated construction systems, it is important to note that the description provided in the previous two sections is, of course, by no means a complete survey of the automated construction landscape. For further reading on both historical and contemporary construction automation projects, the reader is recommended to the excellent contemporary surveys conducted by Bock and Linner in the *Cambridge Handbooks on Construction Robotics*, which survey both single-purpose construction robots [30] and site automation systems [11]. Other useful and more easily digestible summaries include Labonnote et. al. [21] and Lim et. al. [31]. Finally, researchers at TU Darmstadt have recently launched an effort to develop an online, open catalogue of additive automated construction systems, available at am4ae.weebly.com: as of July 2017, they have 81 separate projects from 36 different organizations listed.

1.1.3 Comparative Performance Analysis

The high diversity – in kinematic structure, fabrication process, intended application, and more – in the automated construction field makes it challenging to find a common set of metrics with which to compare systems. In addition to variations in system size and intended task, automated construction systems have been designed to leverage a huge range of fabrication processes, which are frequently difficult to compare. For example, a direct comparison between a concrete direct-extrusion process like those implemented in Concrete Printing with a filament winding process like that implemented by the ETH Drone Rope Bridge project is quite difficult. Even between automated construction systems that could reasonably be directly compared – for instance, two gantries using direct-extrusion additive fabrication techniques - the experimental nature of many of these systems means that there is frequently insufficient data published to make these comparisons.

In light of these challenges, we have proposed in [2] a simplistic analysis along two axes, as shown in Figure 7 and Figure 8, that can provide some insight into the relative performance of different automated construction systems. The first axis is the total work volume that the system can reasonably reach during a fabrication operation. This estimates the scale of the structures that a given system can produce, ranging from modules or subcomponents of a structure, to complete monolithic structures. The second axis is the typical maximum volumetric fabrication rate a system can achieve with its default fabrication technology. This provides a measure of how rapidly a given system can produce structures. Taken together, these two metrics give a rough sense of a system's overall performance in executing automated construction tasks.

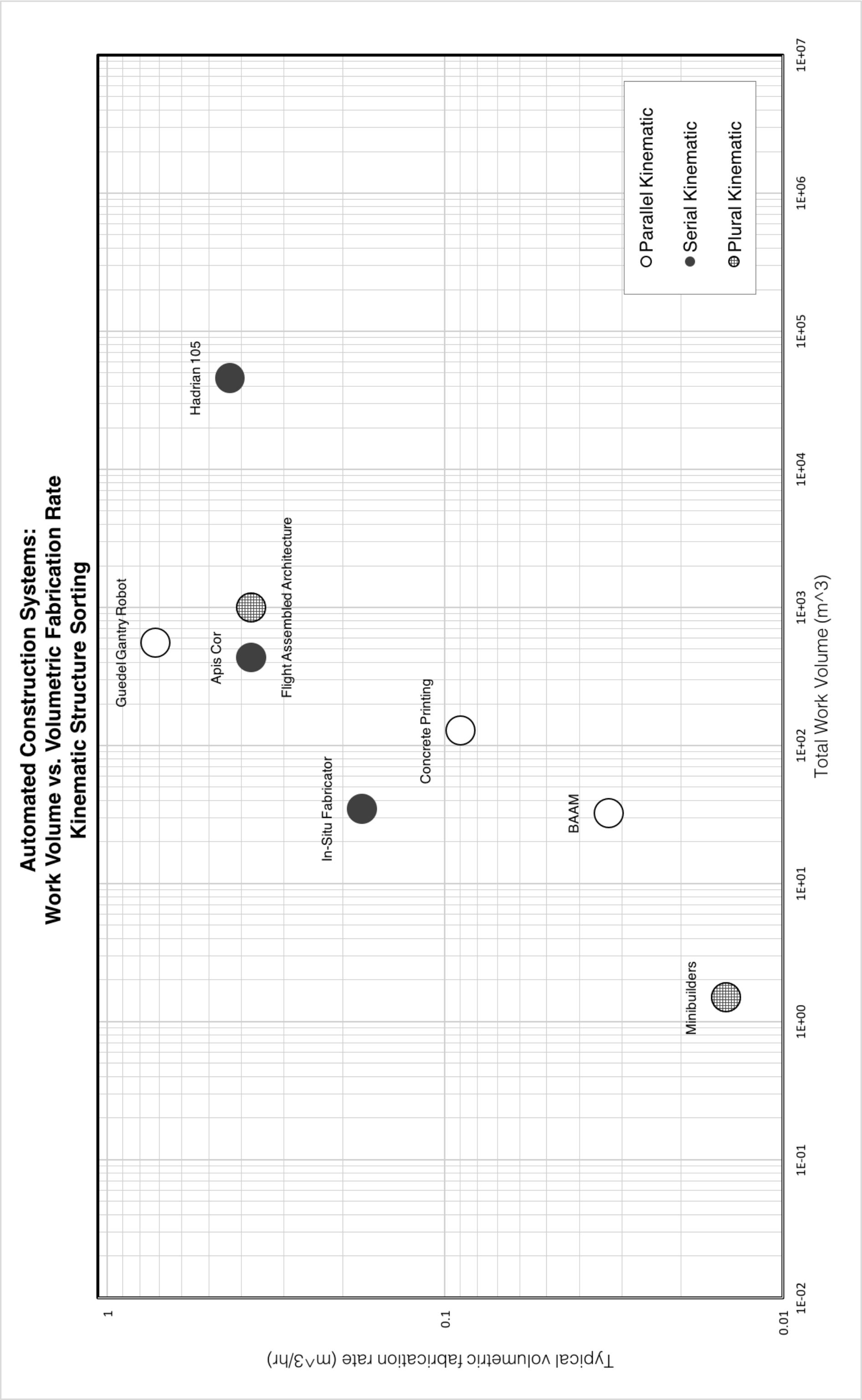


Figure 7: ACS Comparison Mapping by Kinematic Type

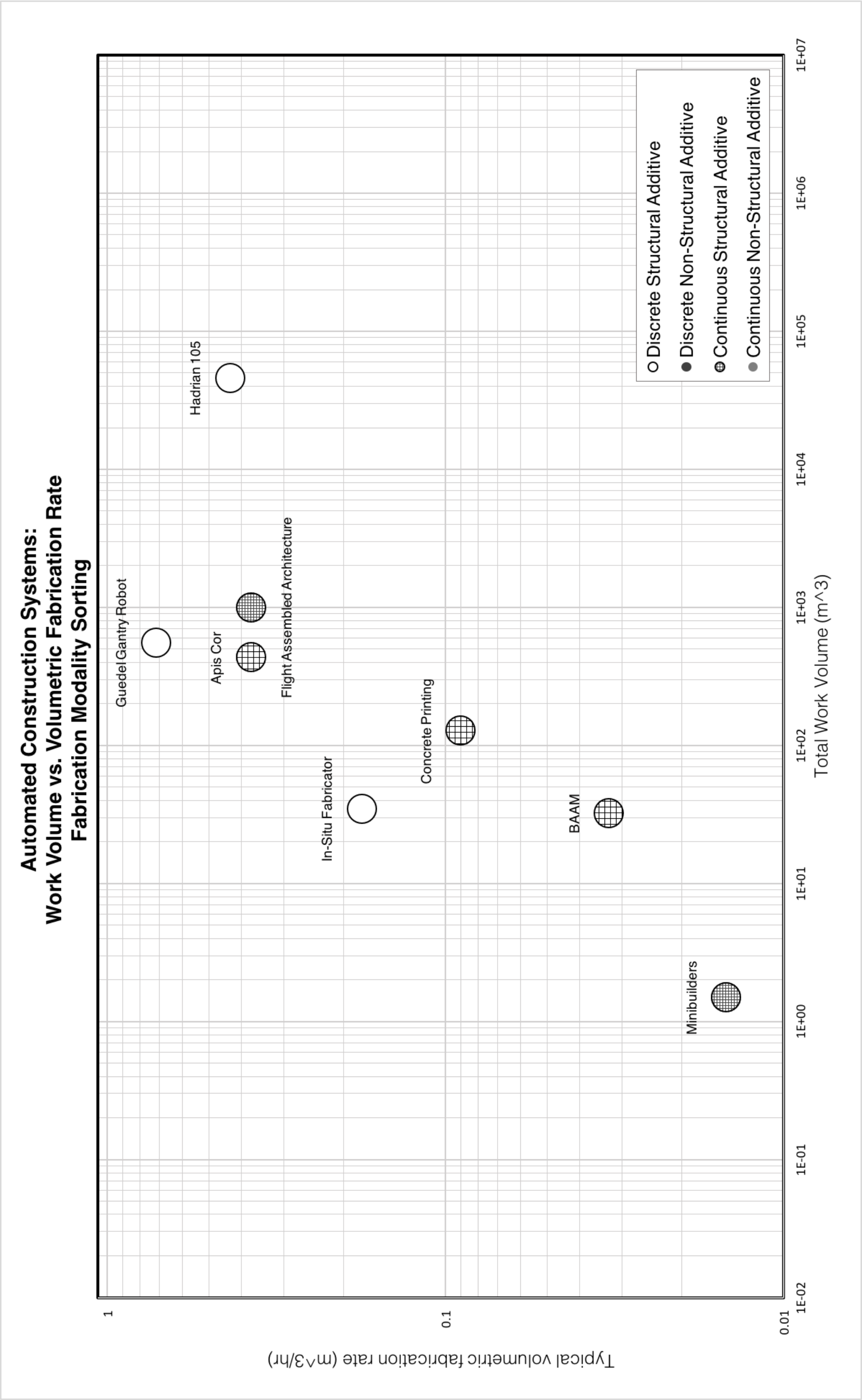


Figure 8: ACS Comparison Mapping by Fabrication Modality

Figure 7 and Figure 8 use these axes to map a (non-exhaustive) selection of automated construction systems that are either in active development, or have been developed within the past 5 years. In addition to work volume and volumetric fabrication rate, systems are also classified in Figure 7 by type using the categories described above, as parallel, serial, or plural kinematic systems; and in Figure 8 by fabrication process, as continuous structural additive, discrete structural additive, continuous nonstructural additive, or discrete non-structural additive. In many cases, these metrics are not explicitly published, and we have had to estimate from existing data. Further details of how metrics were derived for each system, along with a comparison table listing numerical values for each system, may be found in the supplemental materials for [2].

Because of its simplicity, this analysis is necessarily limited in a number of important ways:

- First, the characteristics of systems/processes are not immutable. A system's volumetric fabrication rate is generally a function of fabrication material and process, and is not necessarily intrinsic to the system: a high-throughput fabrication technique – such as Print-In-Place Construction – could reasonably be ported to a wide range of systems. Likewise, the limitations on the work volume of a given system could conceivably modified by increasing the size or mobility of the system. While Figure 7 and Figure 8 are useful for comparing current automated construction systems as they have been reported in the literature, they do not reflect intrinsic performance limits.

- Second, this analysis reduces the dimensions of the performance of a system/process to a point where important characteristics may be obscured. For instance, characteristics like the resolution of a given fabrication process, or the mechanical performance of the produced structure, are not reflected in this analysis. While spray PU foam fabrication processes like Print-In-Place outperform direct concrete or plastic extrusion processes in terms of fabrication rate, they lag substantially in terms of resolution and mechanical performance – which, depending on the fabrication task to be performed, may be substantially more valuable than fabrication rate.

However, even with these limitations, we believe that this analysis provides a useful, "back-of-envelope" heuristic for evaluating the performance of automated construction systems, and provides a few interesting insights. Particularly, it suggests that substantial advances can be made in increasing the fabrication rate of automated construction systems by focusing on materials that are more efficient to work with. This is particularly apparent in Figure 8, where a clear separation can be seen between direct extrusion systems like the BAAM and Concrete Printing printer, and between assembly systems like the Hadrian 105 and Guedel Gantry Robot. This heuristic is clearly by no means a sufficient or complete analytical tool: as discussed in Chapter 3, a comparison method that more directly tests the quantity of interest – how well a given ACS is adapted to constructing buildings – is clearly needed. However, this technique provides a useful starting point for this sort of analysis.

1.2 Prior Work

The work that has led to the development of the DCP v.2 system as presented here can be segmented into three areas: the micro-macro manipulator concept that the DCP system is built around; the work done 2012-2014 by Keating et. al. on the original, DCP v.1 platform; and the development of the Print-in-Place additive fabrication process between 2010-2012 by Keating. Each of these three areas are reviewed briefly, to provide background for the development of the DCP v.2 system.

1.2.1 The Micro-Macro Manipulator

The concept of a micro-macro manipulator - a manipulation system composed of distinct systems that are designed to operate at substantially different scales - was first reduced to practice in robotics by Sharon in the mid-1980s in his MS thesis work [32]. Sharon's original presentation used a five-DOF micromanipulator, capable of XYZ translations and rotation about two axes, and demonstrated the concept's utility in compensating for the flexible modes in a cantilever beam. Further work – particularly [33] – developed this further and established a number of important results, including:

- The micro-macro manipulator architecture can enable both force and position control bandwidths substantially above the fundamental structural frequency of the macromanipulator.

- Dynamic coupling between the micro and macro manipulators can be neglected – and system stability ensured for all gains – if the system can be designed such that the endpoint inertia of the macromanipulator is much greater than the inertia of the micromanipulator plus load.

Subsequent work by other authors has expanded on this core concept substantially, including explorations of task segmentation, such as offline task segmentation between the micro and macro manipulators [34], and manipulability-based real-time task segmentation [35]; and development of new methods for mitigating macro-manipulator vibration, including the Coupling Map Method [36], and controlling reaction forces produced by the micro-manipulator to damp the macro-manipulator [37]. Particularly notable here is the work done by Book and his lab at Georgia Tech, where they focused specifically on the control of large-scale flexible manipulators, such as their RALF (Robotic Arm, Large and Flexible) system. They developed wide variety of techniques for vibration mitigation through use of the micro-manipulator as an inertial damper ([38], [39]), along with related technologies and techniques for control of large-scale, flexible robotic systems ([40], [41]).

In automated construction applications, the micro-macro manipulator architecture has not yet become as ubiquitous as might be expected. Arguably, this is because the majority of explorations in automated construction – particularly over the past two decades – have focused on the development of materials systems, and at small enough scales that the efficiencies a serial-kinematic system can offer do not yet outweigh the additional challenges of the architecture. One notable exception to this, however, is the Hadrian bricklaying robot, under development by Fastbrick Robotics in Australia since the mid-2000s ([42], [43]). The Hadrian is composed of a large hydraulically-actuated arm (historically, a repurposed conventional excavator) with a multi-axis, spherical micro-manipulator mounted at the endpoint. Through use of an external laser tracking system, this micro-manipulator provides dynamic compensation for deflections of the macro arm, in addition to orienting and placing bricks [44]. The success of the Hadrian prototype – particularly when coupled with the 28-meter reach the system offers – is an encouraging result for the wider adoption of the micro-macro manipulator in automated construction.

1.2.2 The DCP v.1

It is in the context of this prior art that the original Digital Construction Platform (DCP v.1) was designed and implemented. The core concept of the DCP – using a large-scale arm system based on construction equipment with a smaller, high-precision arm system – was originally conceived by Keating in his MS thesis [5], as a tool for expanding the Print-in-Place process developed there to architectural scales. Notably, the arm at this point was described not as a micro-macro manipulator in the sense of Sharon, but rather as a compound arm system – similar to what Sharon describes in his work to distinguish between a true micro-macro manipulator (where the micro-manipulator actively compensates for poor performance of the macro-manipulator) and a robot arm with a high-DOF end-effector.

This idea was then developed with the DCP v.1 prototype, described in [1] and [3]. Like the v.2 prototype, this system was constructed around an aerial lift – in this case, an Altec L42M electrical service vehicle. The L42M offered substantial reach (11.3 m) and endpoint load capacity (681 kg) [45], but was otherwise quite limited, with no built-in digital interface to the hydraulic valves. Digital valves were installed by the research team, but they were binary solenoid-controlled valves, and anecdotally provided poor joint control. The micro-manipulator, meanwhile, was implemented using first a KUKA KR 5 sixx R850, and then a KUKA KR10 R1100 sixx industrial robot arm system. Control of the system was implemented through a conventional (non-real-time) MATLAB loop function, which communicated with sensors on the L42M and elsewhere in the system through a LabJack U6 data acquisition interface; and with the KUKA through the KUKA-proprietary Robot Sensor Interface (RSI). It is unclear what degree of higher-level control (path planning, FK/IK, etc.) were developed for this system.

While never directly described using existing micro-macro manipulator literature and terminology, the DCP v.1 is substantially closer in function to the traditional micro-macro manipulator described by Sharon: the KUKA is implemented at least partially to compensate for static and dynamic errors caused by the substantial flexibility of the L42M's boom.

While the DCP v.1 prototype was never successfully used to fabricate a structure, it established a number of core concepts that have substantially shaped the design intent of the DCP project:

- Use of existing commercial equipment: While the DCP v.1 was built around an existing aerial lift vehicle primarily for expediency rather than ideological reasons, the idea of creating an automated construction system around existing serial-kinematic systems and compensating for poor performance/undesirable behaviors is an interesting one. Most conventional industrial robots are built within a design paradigm that does not translate well to architectural scales. They are designed to be extraordinarily stiff (ideally, approximatable as infinitely stiff), and with task-space repeatability requirements in the tens to hundreds of microns throughout the robot's workspace. This limits the size of industrial arms to only a few meters, and at very high cost: for example, the FANUC M-2000iA/1700L robot has a lateral reach of slightly over 4.6 m, and is reported anecdotally to cost around $400,000 [46].

  Conversely, existing serial-kinematic construction systems like the aerial lift unit used in the DCP v.1 achieve radial reaches many times greater than this, usually at a fraction of the cost. For example, the AT40GW aerial lift used in the DCP v.2 system – one of the smallest lift systems that Altec sells – has a radial reach of 8.5 m, and is listed at a base price of $65,000 including its trailer. Other systems, such as the enormous Putzmeister 70Z concrete boom pump, offer lateral reaches of more than 50 m [28]. The companies that build these systems have deep experience in engineering for challenging worksite conditions, and already have a well-established fabrication, sales and service infrastructure. Because of this, we believe that leveraging existing equipment manufactured by the construction and utility industries represents a much faster path to large-scale adoption for serial-kinematic automated construction systems.

- Hybridization of traditional micro-macro manipulator concept: The DCP v.1 also implements a hybridized variant of the micro-macro manipulator concept, where the micro-manipulator's reach is large enough to perform useful tasks on its own. For construction applications, we believe that the combination of these two manipulators with well-separated but still useful operation scales – 10 meters vs. 1 meter for the DCP v.1 – fits naturally with segmentation of tasks and scales found on the worksite. For instance, the scale of the macro manipulator is well-matched to large-scale tasks like the fabrication of fabrications and primary structural components, while the micro manipulator is better adapted for tasks like precision finishing or component installation. However, it is important to note that at least in the configurations of the DCP v.1 and v.2 systems, the micro manipulator's scale means that it is slightly too large to cleanly meet Sharon's stipulation that the micro and macro manipulator inertias be substantially separated. Therefore, dynamic coupling between the two systems remains an important factor to be considered.

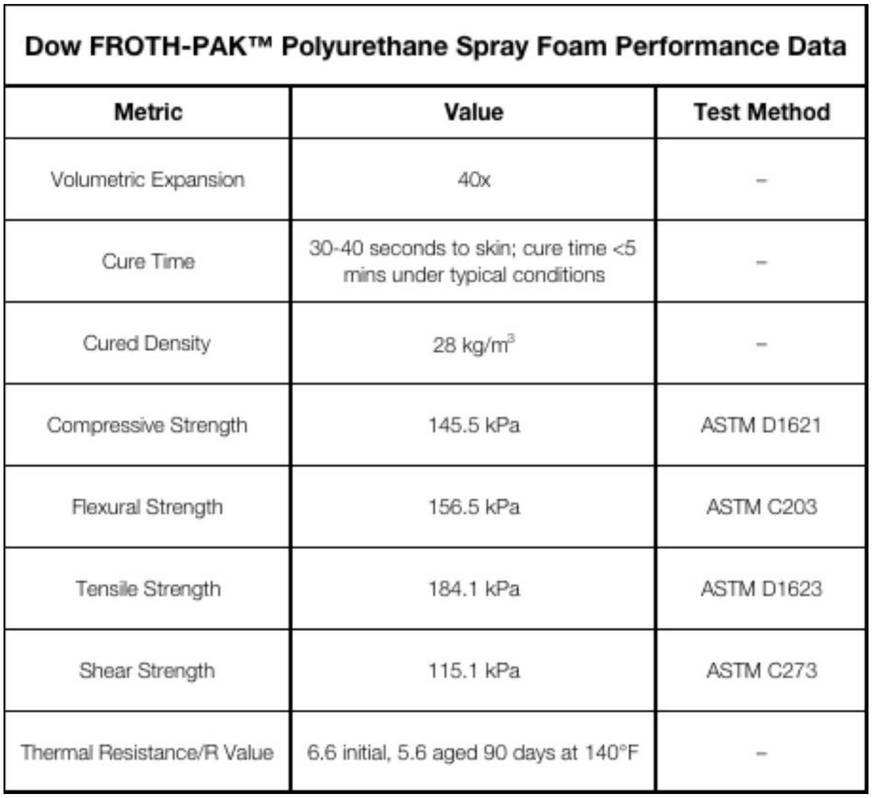
- Platform, not printer: Most importantly, the DCP v.1 established the idea of building a *platform* for fabrication and exploration in automated construction – much like a conventional industrial robot arm – rather than a closed, single-application system. This concept was at the core of the DCP v.1 platform, as described by Keating in [1], and continues to guide the development of the v.2 platform today.

1.2.3 Print-in-Place Fabrication

The last major source of prior art relevant to the development of the DCP v.2 system is the Print-in-Place additive fabrication technique, originally described by Keating in [5]. Here, Keating recognized the limitations of concrete fabrication systems – namely, the challenges associated with extruding concrete. He then inverted the problem, examining instead methods for fabricating formworks for concrete to be poured into, and settling on developing an additive analogue to insulated concrete form (ICF) construction. He reasoned that this would produce a high-strength finished product while still making available the geometric complexity afforded by 3D printing. Additionally, by developing a process where the finished structural element is produced using methods that are already understood in existing building code, Keating proposed that the path to regulatory adoption for this new process would be dramatically shorter than for *de novo* processes like layer-based concrete extrusion.

Early work to find a material suitable for additive fabrication and capable of replacing the standardized Styrofoam panels used in ICF construction focused on commercial spray polyurethane foam compounds. These compounds are available from a wide range of suppliers, including Dow Chemical, BASF, Saint-Gobain, and more. Foam compounds can generally be classified as open-cell or closed-cell foams. They are delivered via low-pressure (~225 PSI) or high-pressure (~1200 PSI) spray systems: generally, high-pressure systems are used for large spray foam installations, while low-pressure systems are used for repairs and smaller installations. Keating reports in [5] that five separate foam varieties were tested during the development of Print-in-Place. Of these, the FROTH-PAK insulating foam manufactured by Dow Chemical – a medium-density, open-cell, low-pressure-delivery spray foam – was found to be the best match for the application, and the simplest to use: it has remained in use throughout the various stages of the DCP project.

Dow Chemical reports the following mechanical properties and other statistics for FROTH-PAK, as shown in Table 1 below:



Dow FROTH-PAK™ Polyurethane Spray Foam Performance Data

Metric	Value	Test Method
Volumetric Expansion	40x	–
Cure Time	30-40 seconds to skin; cure time <5 mins under typical conditions	–
Cured Density	28 kg/m^3	–
Compressive Strength	145.5 kPa	ASTM D1621
Flexural Strength	156.5 kPa	ASTM C203
Tensile Strength	184.1 kPa	ASTM D1623
Shear Strength	115.1 kPa	ASTM C273
Thermal Resistance/R Value	6.6 initial, 5.6 aged 90 days at 140°F	–

Table 1: Material performance data for Dow FROTH-PAK™. Some data taken from [47]

As a 3D printing media, spray polyurethane foams like FROTH-PAK offer a wide range of advantages. First, their rapid cure time and high volumetric expansion ratio enable extremely fast fabrication, with the foam typically curing sufficiently to support subsequent layers with 45 seconds. This rapid cure time also combines with the strong adhesive nature of the foam to enable extremely steep overhangs to be printed, as shown in Figure 11, and even lateral printing to be successful up to moderate limits. Spray foams have also been observed to exhibit some degree of self-levelling, enabling them to be used on rough or uneven substrates.

Work in 2017 by the author also examined the surface characteristics of sections produced with spray polyurethane foams. Like most layer-based additive fabrication techniques, Print-in-Place fabrication produces a striated surface texture after printing. This texture contributes to the resolution of the printing process, as well as the surface finish of the final product, and is important to characterize.

To quantify the typical surface textural properties of Print-in-Place fabricated formworks, a 250 mm x 300 mm sample of material printed in-lab using a conventional robotic arm was analyzed in accordance with ISO 4287 [48] (Figure 9).

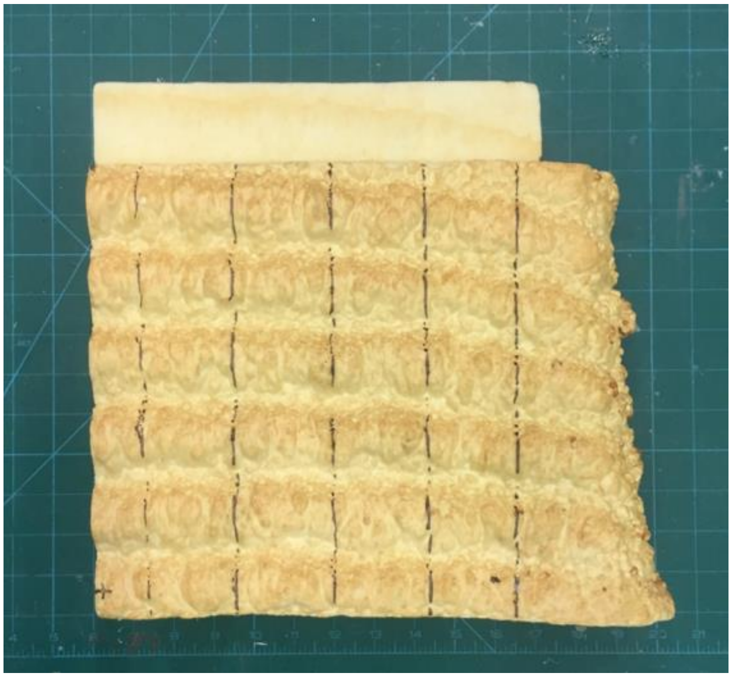
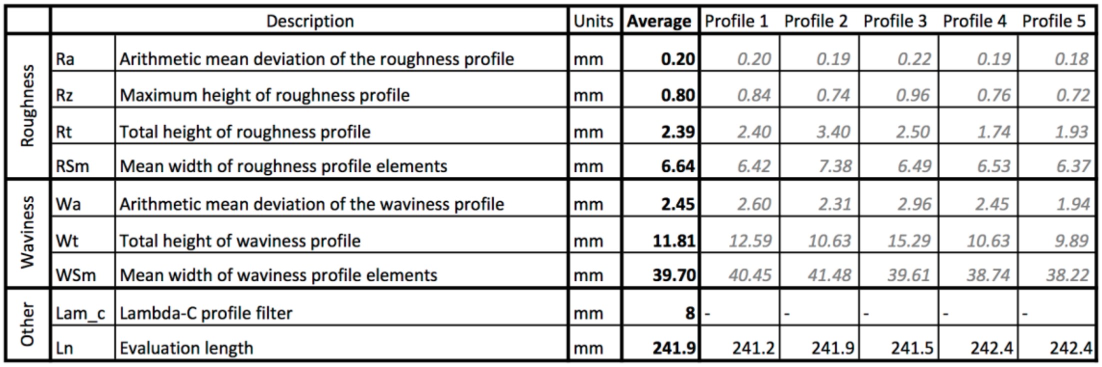


Figure 9: Print-in-Place formwork sample for ISO 4287 roughness characterization

Five surface profile samples were taken using a Microscribe 3DLX digitizer, with a resolution of 0.48 mm between recording points. While this instrument is dramatically lower-resolution than typical surface topography measurement equipment, it is sufficient for this application because of the scale of surface features of interest on the PiP printed sample. Profile data was analyzed using the MountainsMap surface metrology software package [49]. Profiles were denoised (50 μm cutoff) and then segregated with an 8 mm λ_c filter to separate roughness and waviness components. Characteristics from all five profiles were averaged, yielding the data presented in Table 2.



		Description	Units	Average	Profile 1	Profile 2	Profile 3	Profile 4	Profile 5
Roughness	Ra	Arithmetic mean deviation of the roughness profile	mm	**0.20**	*0.20*	*0.19*	*0.22*	*0.19*	*0.18*
	Rz	Maximum height of roughness profile	mm	**0.80**	*0.84*	*0.74*	*0.96*	*0.76*	*0.72*
	Rt	Total height of roughness profile	mm	**2.39**	*2.40*	*3.40*	*2.50*	*1.74*	*1.93*
	RSm	Mean width of roughness profile elements	mm	**6.64**	*6.42*	*7.38*	*6.49*	*6.53*	*6.37*
Waviness	Wa	Arithmetic mean deviation of the waviness profile	mm	**2.45**	*2.60*	*2.31*	*2.96*	*2.45*	*1.94*
	Wt	Total height of waviness profile	mm	**11.81**	*12.59*	*10.63*	*15.29*	*10.63*	*9.89*
	WSm	Mean width of waviness profile elements	mm	**39.70**	*40.45*	*41.48*	*39.61*	*38.74*	*38.22*
Other	Lam_c	Lambda-C profile filter	mm	**8**	-	-	-	-	-
	Ln	Evaluation length	mm	**241.9**	**241.2**	**241.9**	**241.5**	**242.4**	**242.4**

Table 2: ISO 4287 surface topography summary data.

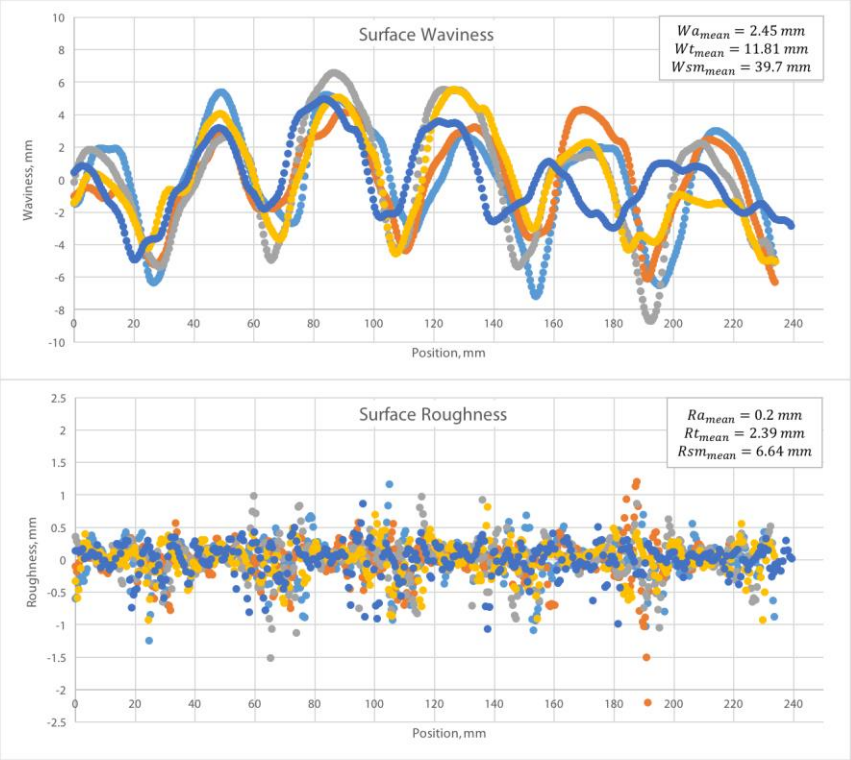


Figure 10: Roughness contours, ISO 4287 characterization

This analysis provides valuable insight into the impact of the Print-in-Place formwork surface on the performance and aesthetics of a completed structure:

- The surface produced by the Print-in-Place process exhibits *roughness* with features on the order of ~1 mm. This has been observed in printing to be affected by a number of print parameters, including isocyanate/polyol mixture (a less stiff mixture spatters less);

distance from spray nozzle to print surface; spray pressure; spray flowrate; and nozzle cleanliness. Smoother surfaces have been observed on some prints, although obtaining these reliably through control of the above variables would be challenging.

- Meanwhile, the *waviness* of the PiP surface is dictated primarily by foam layer structure. In the test sample measured, layer height (which corresponds approximately to W_{sm}) is on the order of 40 mm. At this layer height, the total lateral deviation of the printed surface – captured in W_t – is on the order of 12 mm. Further characterization is needed to explore how changes in layer height and other parameters affect this relationship.

It is important to note that these estimates are likely best-case estimates, since the sample used was printed in optimal conditions (specifically, on a flat print surface) using a much less complex robotic system. Larger-scale components printed with the DCP have been observed to exhibit substantial Z-waviness as well (for examples, see Section 2.2.3), generally resulting from imperfections in the starting print surface. Further work is required to characterize these behaviors. However, these numbers provide a useful starting point for understanding the performance of PiP-fabricated surfaces, and comparing the process to other automated construction processes. Most usefully, they provide a lower bound on the achievable geometric accuracy of the process, indicating what architectural components can be produced directly with PiP, and what will require additional post-processing via subtractive machining, surface smoothing or other processes.

Keating's early experiments, described initially in [5] and [3], demonstrated that this type of post-machining to improve surface finish and geometric accuracy was viable. He also showed that Print-in-Place formworks could be reinforced and finished using conventional ICF techniques; colorized; and even cast into. Most importantly, he conducted basic hydrostatic pressure testing which confirmed that formworks fabricated using Print-in-Place techniques could support comparable pressures to conventional ICF panels. The test specimen described in [3] failed at 53.4 kPa, corresponding to roughly a 2-meter pour of concrete.

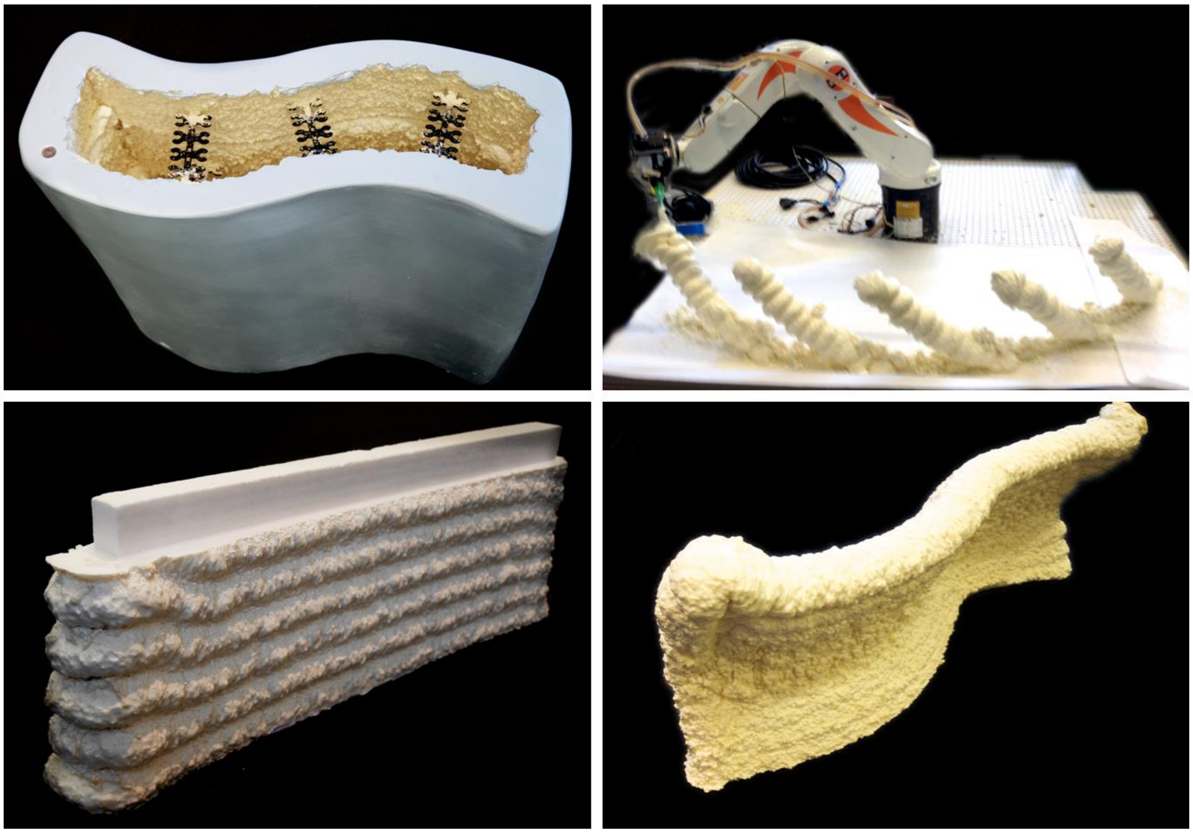


Figure 11: Early experiments with Print-in-Place fabrication. Clockwise from top left: 1) Wall, finished with conventional ICF finishing techniques. 2) Tilted columns fabricated with PiP process. 3) PiP foam can be machined to improve surface finish and geometric accuracy. 4) Example of double curvatures achievable with PiP. Photography credit (all images): Steven Keating. Images a and c originally published in [50].

The suitability of spray foams for additive fabrication has subsequently been validated by a range of other groups in academia and industry. In academia, additive foam has been fabricated at larger deposition widths and faster rates by researchers at the University of Nantes in their BatiPrint3D project, where they report extrudate cross sections of up to 8000 mm^2, and feed rates of up to 300 mm/s [51]. Higher-resolution results have also been achieved by Barnett and Gosselin in [52], where they use a suspended-cable robot to extrude polyurethane foam (along with shaving cream that acts as a support material, since it apparently does not bond to the polyurethane, or impair its blowing process.) They report the successful fabrication of a 2.16 m high statue of a person using this method, and achieve impressive geometric complexity including substantial overhangs. Finally, in a more speculative vein, Hunt et. al. have recognized the utility of the low transportation volume of spray foams, and have begun in [53] to explore using them with quadrotor drones to implement additive fabrication processes.

2 DCP v.2 System Development

This chapter describes the first stage of development of a second prototype of the Digital Construction Platform – referred to here as the DCP v.2 – along with a more fully developed motion planning and control tool chain. We then re-implement the polyurethane-foam additive manufacturing process – the “Print-in-Place” process – originally described by Keating in [5], and use this process with the DCP v.2 to produce an architectural-scale, monolithically printed structure.

This chapter first summarizes the system concept, hardware and software that make up the DCP v.2 system, including the pre-existing commercial systems that are used to implement the micro- and macro-manipulator arms; additional sensing hardware used to instrument these systems; and the command generation and control architecture developed to support the system. We then describe a series of different implementations that were conducted with the DCP v.2 system, such as large-scale light paintings, additive manufacturing with polyurethane spray foam, and performance characterization of the DCP v.2 system in accordance with robotic performance standards. Finally, we review the DCP v.2 system’s performance, both in the broader context of automated construction systems, and also to identify specific weaknesses with the architecture that future work should address.

The work described in this chapter was conducted between August 2015 and July 2016, at both MIT and at the Google “Leghorn” facility in Mountain View, CA. This phase of the DCP’s development was led by Steven Keating, and conducted in collaboration with Levi Cai, Selam Gano, Grant Sellers and Damien Martin. Some elements of the work described here, along with some figures, have been published previously, in either Keating [3] or Keating et. al. [2]. Some text has been adapted from [2] for inclusion here, although that text is the author’s original work.

2.1 System Development

The DCP is a relatively complex robotic system. It incorporates two distinct robotic systems – one commercial, one effectively custom-built – to create a single, highly redundant serial-link manipulator arm. Development of the DCP platform required work at every level of robotic system design, from low-level mechatronics hardware and control code, to high-level task planning and trajectory generation.

This section describes the selection and development of the hardware and software used in the DCP. We first describe the arm systems used to instantiate the micro-and macro-manipulator arms, along with modifications and additions made to these systems to enable their use. We then describe the control and software architecture used in the DCP between August 2015 and July 2016, known as dcpctrl_v1 [54].

2.1.1 DCP v.2 Hardware

The DCP v.2 system borrows heavily from the DCP v.1 system originally developed by Keating [1], both in concept and in some specific choices of hardware and control modality. Like the DCP v.1, the DCP v.2 is a micro-macro serial-link manipulator arm. The macro arm is implemented using an Altec AT40GW aerial lift vehicle, while the micro arm is implemented using a KUKA KR10 R1100 sixx WP electric robotic arm. Each subsystem is described in detail below.

AT40GW Aerial Lift Vehicle

The AT40GW is an open-center hydraulic aerial lift vehicle, manufactured by Altec Inc. Designed for electrical utility service work, the AT40GW is particularly specialized for operation in confined, limited-access areas such as alleyways and dense environments, offering a narrow footprint, a tracked mobile base, and relatively light total system weight (3,447 kg) [55]. While the reach of the system is considerably smaller than that of Altec's other aerial lift systems, it is still dramatically larger than that of conventional industrial robot arms, as shown in Figure 12.

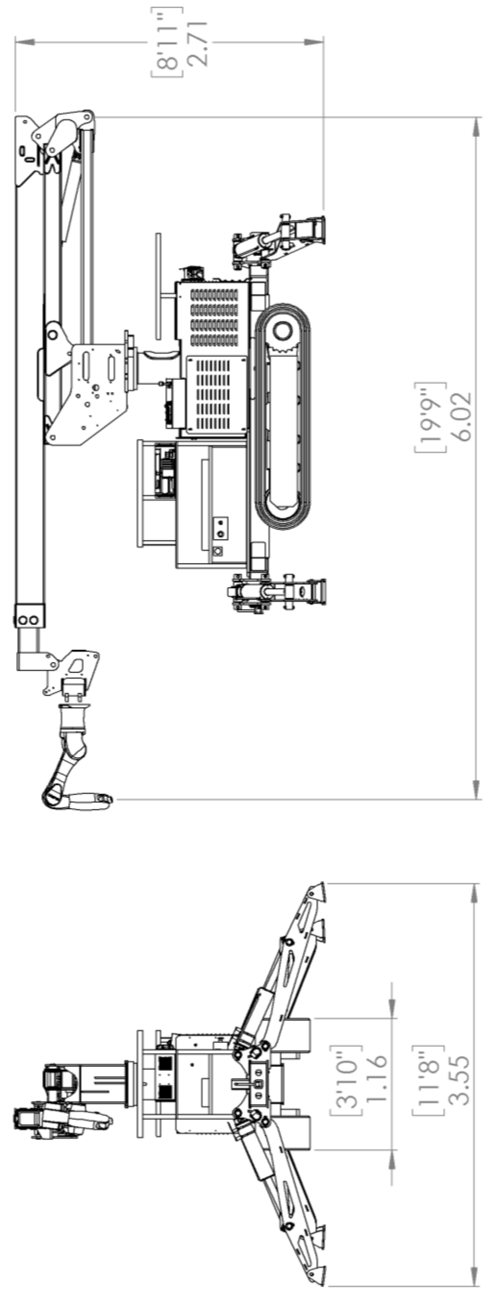


Figure 12: DCP approximate system dimensions, meters and [feet inches].

AT40GW Kinematics

Kinematically, the AT40GW has a total of six controllable joints, in an RRRPRR configuration (R = revolute joint, P = prismatic joint). Four of these joints are digitally controllable through the PWM interface (described further below). These joints, along with their defined zero positions, are shown in Figure 13 below.

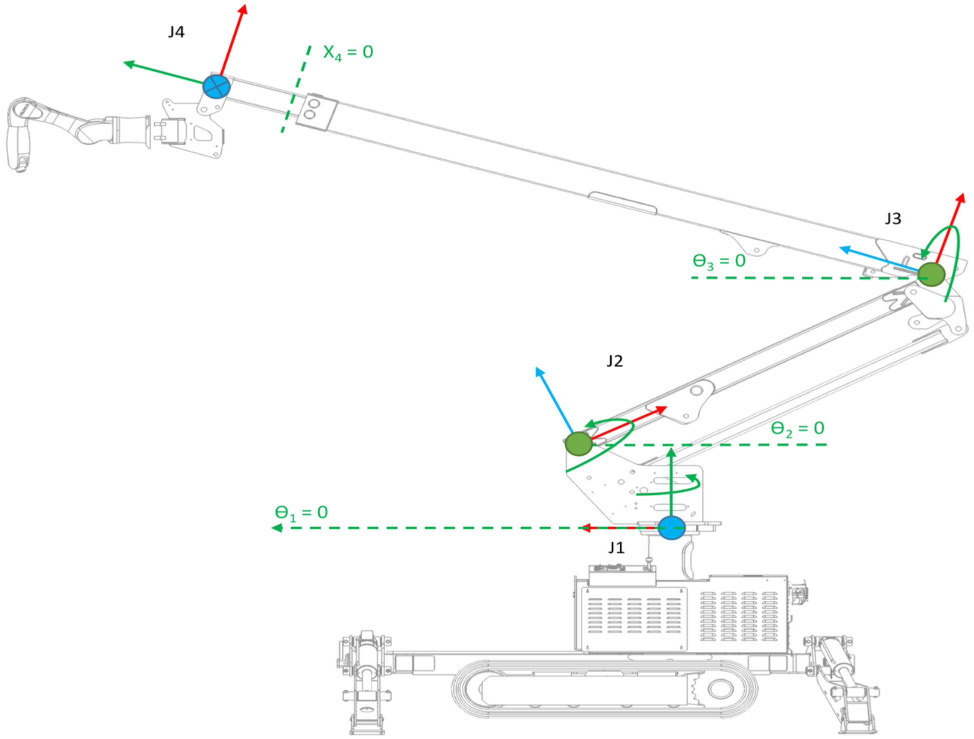


Figure 13: AT40GW controlled joint definitions

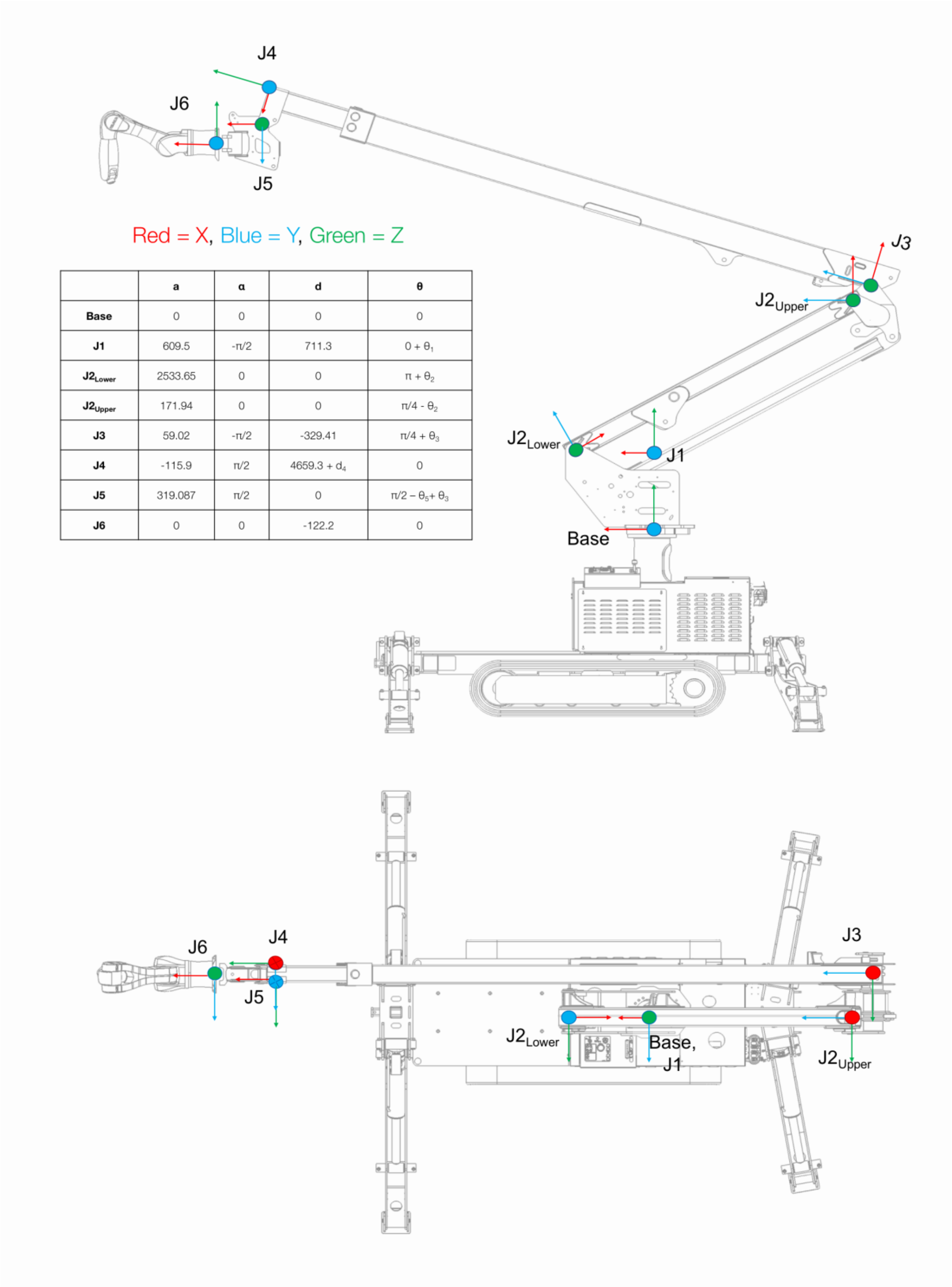
Because the AT40GW has been designed as an electrical service vehicle, rather than as a robotic manipulator, it exhibits a number of unusual kinematic features:

- The second joint drives a parallel linkage, such that a rotation of this joint is cancelled by an equal and opposite rotation at the end of the AT40GW inner link. When used as an electrical service vehicle, this allows the AT40GW operator to elevate the main boom arm, allowing the device to reach over obstacles or increase its maximum vertical reach.

However, when the AT40GW is used as a robot the utility of this joint is limited, as its motion is redundant with Joints 3 and 4.

- The third and fifth joints are linked through a hydraulic circuit into a virtual parallel linkage. Any motion at the third joint will be matched by an equal and opposite rotation at the fifth joint, but the fifth joint can be moved independently as well using a mechanically-operated valve. In use as an electrical service vehicle, this mechanism allows the endpoint of the arm to maintain a fixed angular orientation relative to the ground, avoiding dumping the operator out of the vehicle bucket. In our application, this feature simplifies orientation control of the DCP endpoint, and reducing the amount of compensatory motion required from the micro-manipulator.

To create a kinematic model of the AT40GW, the proximal DH parameter notation developed by Craig [56] was used. The complete DH parameter set, with link locations and joint definitions, are shown in Figure 14. DH parameter values were initially determined from CAD data, and later confirmed from direct measurements of the AT40GW. To enable use of the J2 joint during path planning, two separate DH frames are used to describe the J2 parallel link mechanism, with the joint angles linked. Similarly, the J3 joint angle is applied to both the J3 and J5 joints to reflect the operation of the virtual parallel linkage structure implemented in hydraulics.



Red = X, Blue = Y, Green = Z

	a	α	d	θ
Base	0	0	0	0
J1	609.5	-π/2	711.3	$0 + \theta_1$
J2$_{Lower}$	2533.65	0	0	$\pi + \theta_2$
J2$_{Upper}$	171.94	0	0	$\pi/4 - \theta_2$
J3	59.02	-π/2	-329.41	$\pi/4 + \theta_3$
J4	-115.9	π/2	$4659.3 + d_4$	0
J5	319.087	π/2	0	$\pi/2 - \theta_5 + \theta_3$
J6	0	0	-122.2	0

Figure 14: AT40GW DH parameters

During work conducted 2015-2016, only DH parameters through Frame 4 were measured and used in the control of the DCP. This introduces substantial deviation between the actual and desired endpoint trajectory, since the endpoint trajectory is actually being planned using the Frame 4 coordinate system (the end of the AT40GW boom) as the "endpoint" of the robot. While this was tolerable for the experiments conducted during this phase, it was a major shortcoming of the DCP's architecture, and was remedied in later work as described in Chapter 3.

AT40GW Structural Dynamics

The AT40GW's design as a utility vehicle, rather than as a robot arm, also has repercussions for the structural dynamics of the vehicle. As with many micro-macro manipulators, the AT40GW's links are too long and flexible to be reasonably approximated as rigid, with even relatively small impulses at the joints resulting in substantial oscillations at the endpoint.

To develop a basic understanding of the natural frequencies of the AT40GW structure and the approximate range of endpoint accelerations exhibited during oscillatory behavior, a preliminary characterization of the AT40GW's endpoint response to a velocity impulse at each joint was conducted. Tests were conducted on a "floor model" AT40GW system at the 2015 International Construction and Utility Equipment Exposition (ICUEE) show by the author, with support from Altec engineering staff. A mBientLabs MetaWear RPro Bluetooth-enabled accelerometer/gyroscope module was attached near the end of the DCP boom, approximately 140 mm inwards along the boom from the J4 DH frame location, and oriented as shown in Figure 15. Acceleration measurements were recorded using the MetaWear Developer app, at 12.5 Hz or higher and with a measurement range of ±16g.

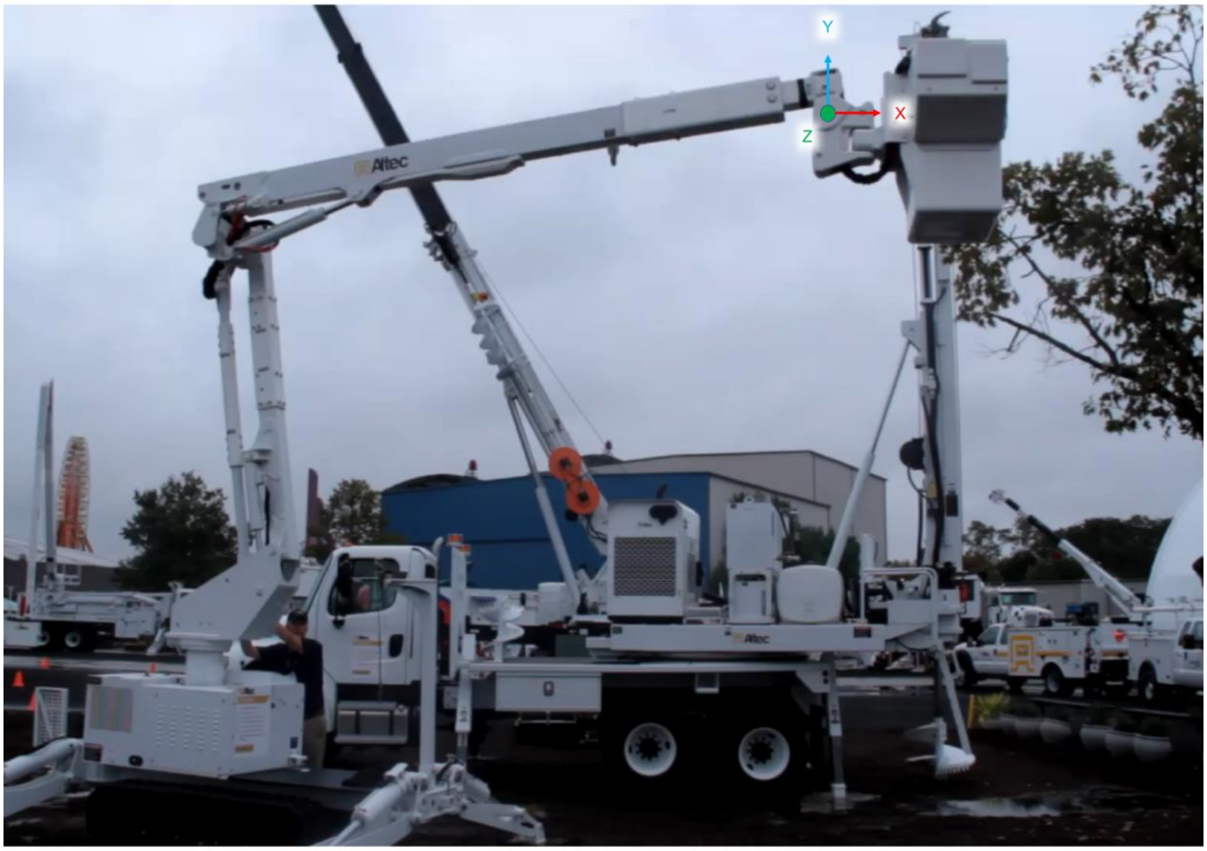


Figure 15: AT40GW natural frequency characterization – sensor orientation.

Fundamental structural frequencies of the AT40GW in a variety of poses were excited by manually applying impulse inputs using the AT40GW's hydraulics. This was accomplished by moving the valve position of the joint under test to the full-open position, and then rapidly actuating the hydraulic dump valve from open (default position) to closed and back again, rapidly applying and then removing full pressure from the hydraulic circuit. While this does not produce an identical impulse about each joint, or even a precisely repeatable impulse between tests, it provided a sufficient approximation of the impulse function for this test. The following tests were conducted:

- Excitation about J1 joint, J4 extended and retracted. This test primarily excites lateral vibrations (Y direction in DCP base coordinate system) in the outer link.

- Excitation about J2 joint, J4 extended and retracted. This test excites both in-out and vertical vibrations in the system (X direction and Z direction in DCP base coordinate system), mostly in the inner link and J2 joint.

- Excitation about J3 joint, J4 extended and retracted. This test excites vertical vibrations (Z direction in DCP base coordinate system) in the outer link.

Results from these tests may be seen in Figure 16, and are explained further below. It is important to note that the coordinate system used by the MetaWear RPro – shown in Figure 15 – has a different orientation than the DCP's base coordinate system (where the Z axis points along the axis of the J1 joint, and the X axis points towards the end of the AT40GW boom when in its stowed position).

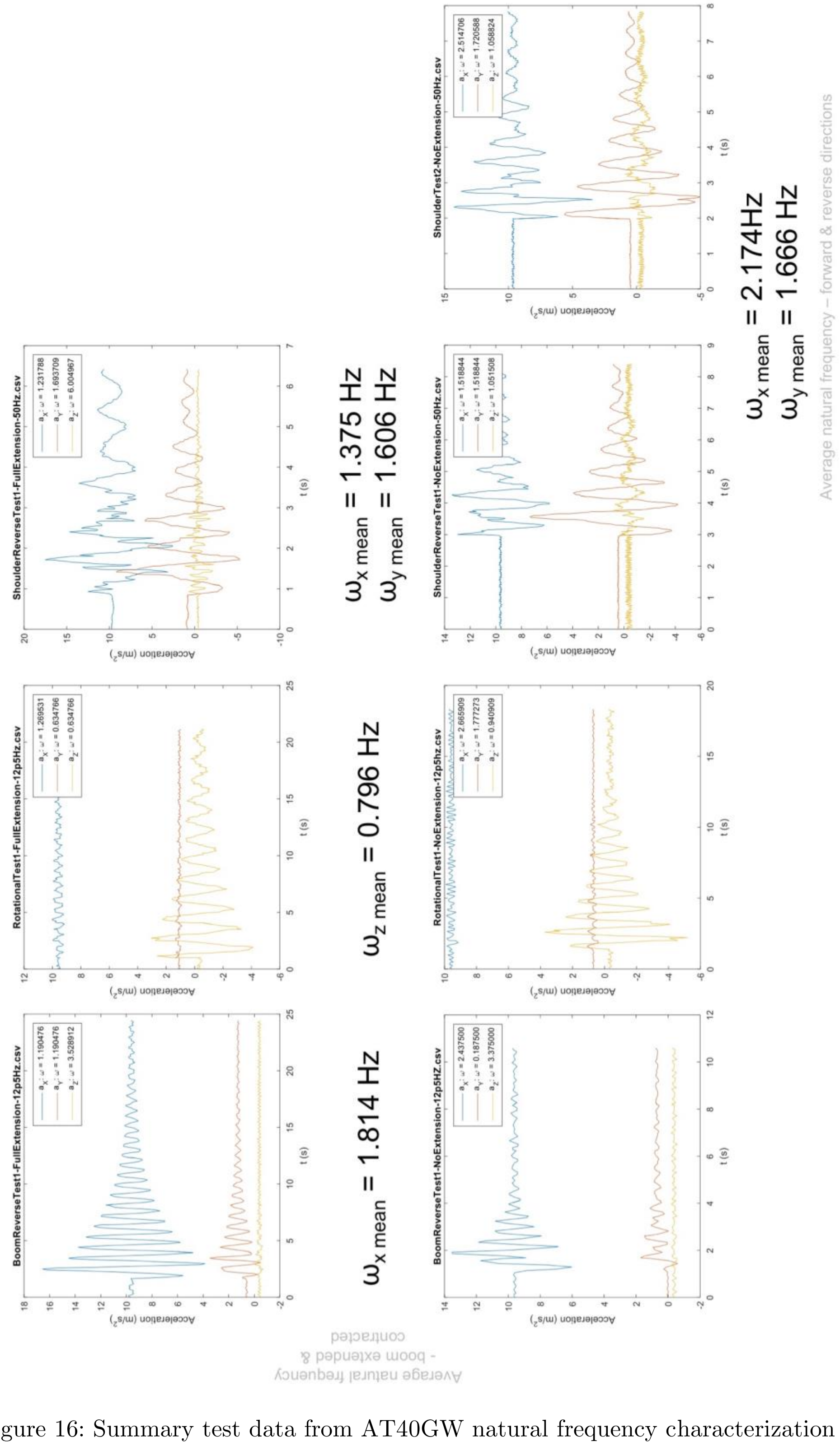


Figure 16: Summary test data from AT40GW natural frequency characterization

These experiments found that endpoint oscillation frequencies ranged between 2.5 Hz for in-out vibration in response to impulse at J2 with J4 fully retracted; and 0.6 Hz for lateral vibration in response to impulse at J1 with J4 fully extended. Peak accelerations observed during this test reached as high as 8 m/s^2. Interestingly, this range of fundamental structural frequencies has been confirmed by Altec anecdotally as characteristic of their aerial lift systems broadly, suggesting that this is a reasonable "first guess" for the development of control systems for this type of aerial lift system. Since control of this type of oscillatory behavior is critical to maximizing the performance of the DCP, having even first-order estimates of the AT40GW's natural structural frequencies is valuable for determining the viability of various oscillation control techniques, such as input shaping, micro-manipulator position compensation, and others.

AT40GW Power, Control & Sensing Hardware

The AT40GW system used in the DCP v.2 has had a number of additional modifications made to it to facilitate use as a robotic arm.

- First, the AT40GW's stock diesel-driven hydraulic pump has been supplemented with an electrohydraulic drive system, designed and installed by Altec. This drive system incorporates battery packs, power management circuitry and an electric pump circuit, and can be charged from a standard 120V outlet. This drive system makes the AT40GW substantially easier to operate, allowing indoor operation, and reducing audible noise and vibration in the system. Additionally, since this allows the DCP to operate fully from electric power, it creates potential for autonomous operation of the system in conjunction with photovoltaic packs or other mobile power sources.

- Second, the stock hydraulic valves on the AT40GW have been replaced with an updated valve and valve driver pack, along with a custom controller interface. The controller interface allows the four primary joint valves – J1 through J4 – to be controlled using 500 Hz PWM signals, with duty cycle corresponding to valve spool position/hydraulic velocity. While this interface is far more convenient and robust than interfacing directly with the valve drivers, it also masks substantial complexity between the PWM input and the valve driver output, including internal control loops, tuned valve driver deadband, and other response characteristics.

- Finally, there is no feedback on joint position available in the stock AT40GW. To implement closed-loop position control, we instrument the four primary axes of the AT40GW. All sensors were monitored through a LabJack T7 PRO DAQ, which also provides additional analog and digital I/O for controlling other system components, such as outrigger supports, mobile platform controls, and end effectors.

  The J1 axis position is monitored with a 1024 pulse/rev YUMO relative rotary encoder, mounted directly to the J1 hydraulic motor. When this sensor is measured in 4x resolution mode and combined with the J1 gear drivetrain (which provides a reduction of 180:1), it theoretically provides .0005-degree angular resolution on output position. This is, of course, not functionally realizable. The backlash in the J1 gear drivetrain is far more substantial than that found in traditional industrial robot geartrains, and is easily detected by pushing laterally on the boom about the J1 axis when the boom is extended. The J1 backlash is also expected to vary as a function of joint position, due to errors introduced during the fabrication of the J1 output gear; and system pose, as the J1 bearing deflects when the AT40GW boom is extended. This backlash variability has not been exhaustively characterized. Since the backlash could not be detected using the sensors available during work completed 2015-2016, it was disregarded. Subsequent work in 2016-2017 has focused on implementing sensors to capture this backlash, and on use of adaptive control techniques to accommodate variability in the backlash spacing, as described in Chapter 3.

  The J2-J4 axis positions are monitored using Balluff magnetostrictive linear position sensors, mounted coaxially with each hydraulic cylinder. These sensors provide an analog position signal between 10 and -10 VDC proportional to sensor carriage position. Nominally, they have a read resolution of 350 μV. However, there are substantial sources of electromagnetic interference present in the DCP, including the motor power cable for the KUKA and the electrohydraulic drive system. Additionally, the LabJack T7 analog inputs offer an adjustable read resolution, enabling the user to adjust between measurement stability and read time. With EMI sources active and the LabJack T7 analog interface set to a read resolution index of 10 (corresponding to ~20.5-bit resolution at the interface), the Balluff sensors were measured to yield worst-case angular position resolutions of 0.15 degrees at J2 and J3, and 3 mm at J4.

With these modifications made, a number of characterization tasks were required before successful closed-loop control of the AT40GW could be implemented, including characterization of the relationship between sensor output and joint motion; and between PWM duty cycle and joint velocity. These characterization processes were generally straightforward, and are not described exhaustively here. However, the following notable observations were made during characterization:

- <u>Sensor to joint position/velocity relationships</u>: For J1 and J4, the relationship between sensor output and joint position is linear, at least in the ideal case. In the J1 joint, joint output state and measured sensor state are related by the J1 transmission ratio, although it is important to remember that this neglects the substantial, variable backlash in the J1 joint. In the J4 joint, the relationship is directly linear, as the sensor is mounted coaxially with the joint output. There may be some error between the measured joint state and actual joint state introduced as the boom extends and deflects, but this is a second-order effect. Moreover, in the case of both J1 and J4, joint state can be directly differentiated, allowing measurement of joint velocity and acceleration if a sufficiently clean position signal is available.

  In J2 and J3, however, the joint is actuated by a linear hydraulic cylinder which forms the hypotenuse of a triangle with the joint opposing it, as shown in Figure 17.

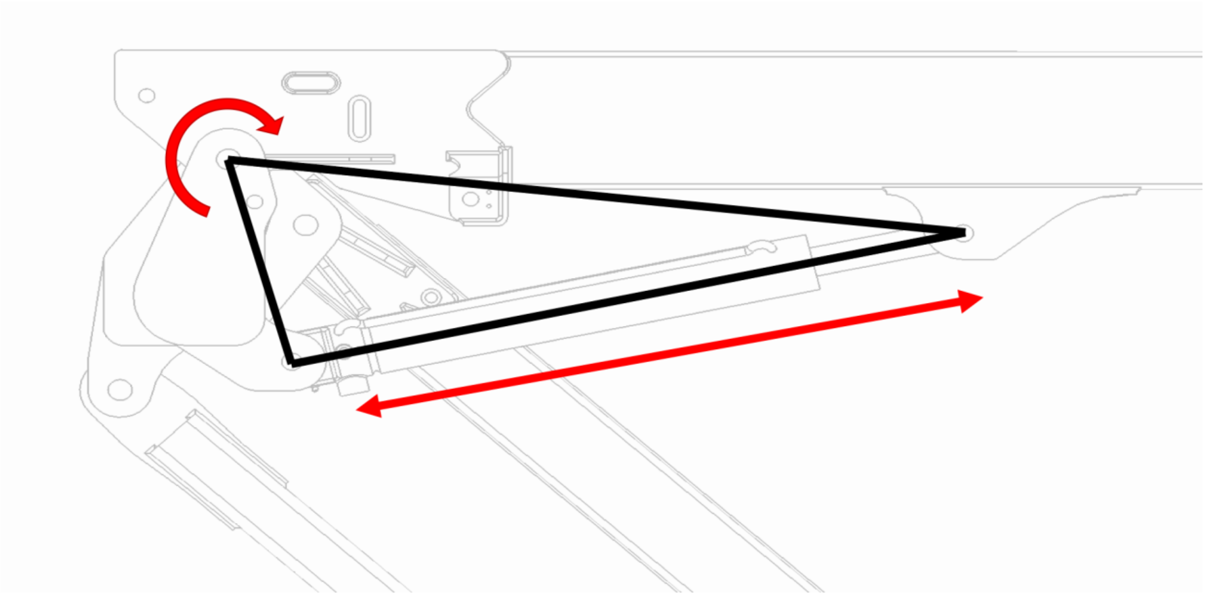


Figure 17: J3 joint sensor configuration (measurement along red line)

For J3, the joint angle γ and joint position x may be related using the law of cosines as:

$$\gamma = \cos^{-1}\left(\frac{c_0^2 - a^2 - b^2}{-2ab}\right) - \cos^{-1}\left(\frac{x^2 - a^2 - b^2}{-2ab}\right)$$

and

$$x = \sqrt{a^2 + b^2 - 2ab\cos(\gamma - \gamma_0)}$$

where x = the current measured length of the cylinder (which may be greater or smaller than c_0, the cylinder length at $\gamma = \gamma_0 = 0$ deg). While these are relatively fast calculations to implement, their derivatives are less straightforward. Particularly, hydraulic cylinder velocity is a nonlinear function of both angular position and angular velocity, making control of joint velocity from measured cylinder position challenging.

A preliminary attempt to model the relationship between desired joint velocity and cylinder velocity for J3 involved calculating linear velocities as a function of angular velocities for the entire range of joint angles, and produced the following discrete surface (Figure 18) relating joint angle, joint angular velocity and joint sensor velocity (which is proportional to linear velocity). The flat sections of the surface show where the maximum joint velocity has been reached.

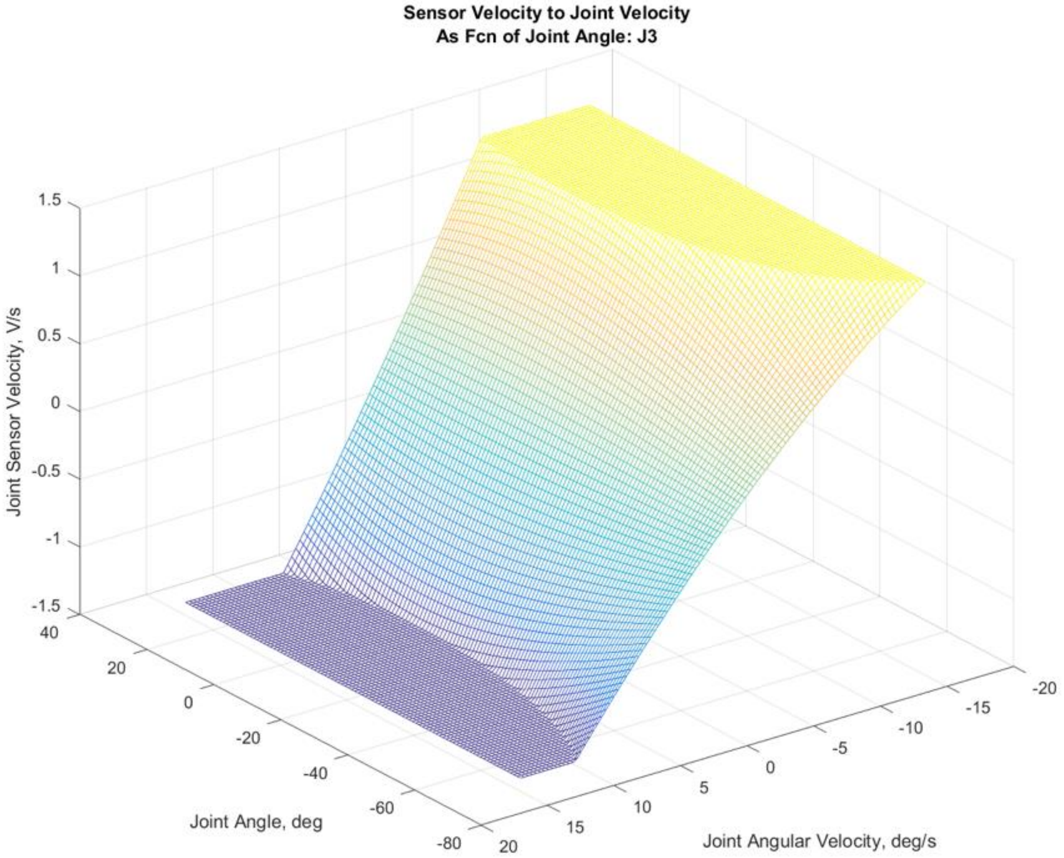


Figure 18: J3 sensor velocity to joint velocity mapping as function of joint position

A continuous surface was then fit to the non-planar sections of this surface, using a third-order polynomial function. Unfortunately, challenges associated with ensuring consistency of this surface function – for instance, ensuring that the function returned a sensor velocity of zero for angular velocities of zero – led to this 3D mapping being replaced with a single, 2D mapping function (Figure 19) that is valid at the joint angle with the smallest permissible range of angular velocities (-2.62 deg).

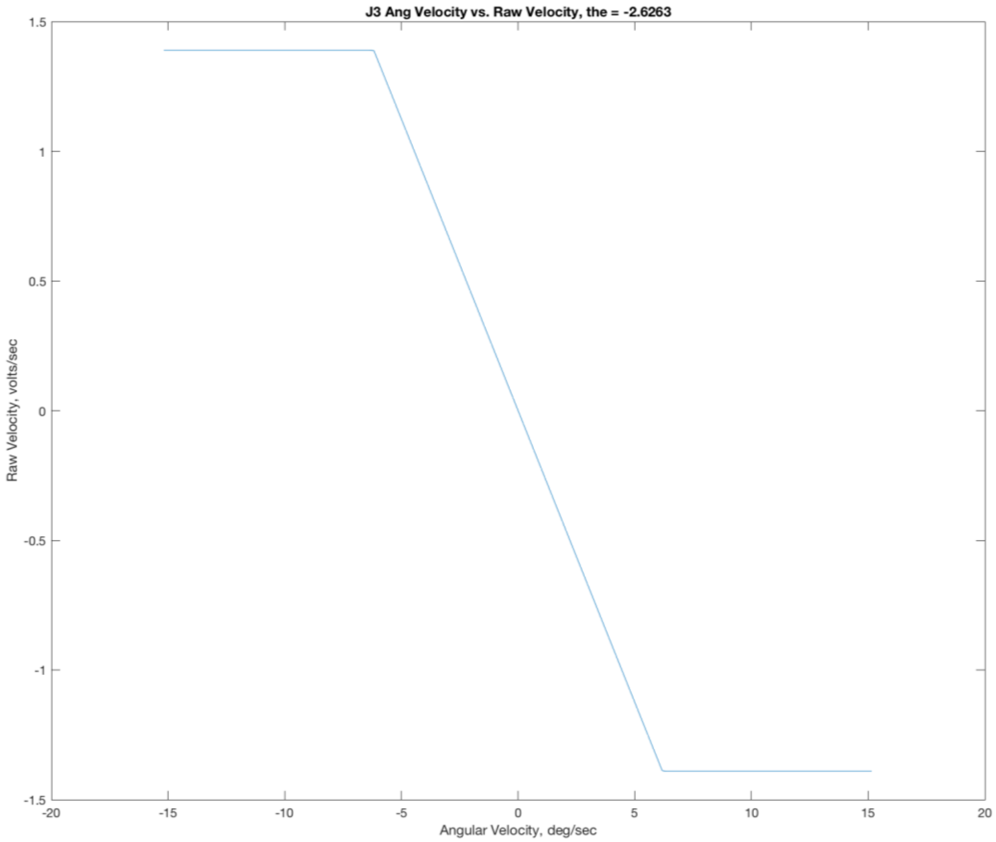


Figure 19: J3 angular velocity to sensor velocity mapping

This approach is simpler, but does lose accuracy at the ends of travel. In the control architecture used during this phase of the DCP project, this did not pose a major problem, as sensor/linear and angular velocities were only used as feedforward terms in a feedback control loop. Subsequent work has developed more efficient methods of calculating and controlling joint velocity, as described in Chapter 3.

- PWM to joint velocity characterization: To facilitate effective control of the AT40GW, the relationship between PWM duty cycle input and cylinder (and/or joint) velocity output was characterized. While this characterization combines the dynamics of a number of subsystems – such as valve drivers and internal control loops, as previously discussed – it does capture the most important features of the PWM-velocity relationship, including saturation limits, deadband and function shape. Generally, the relationship is described by a piecewise function, as shown below in Figure 20.

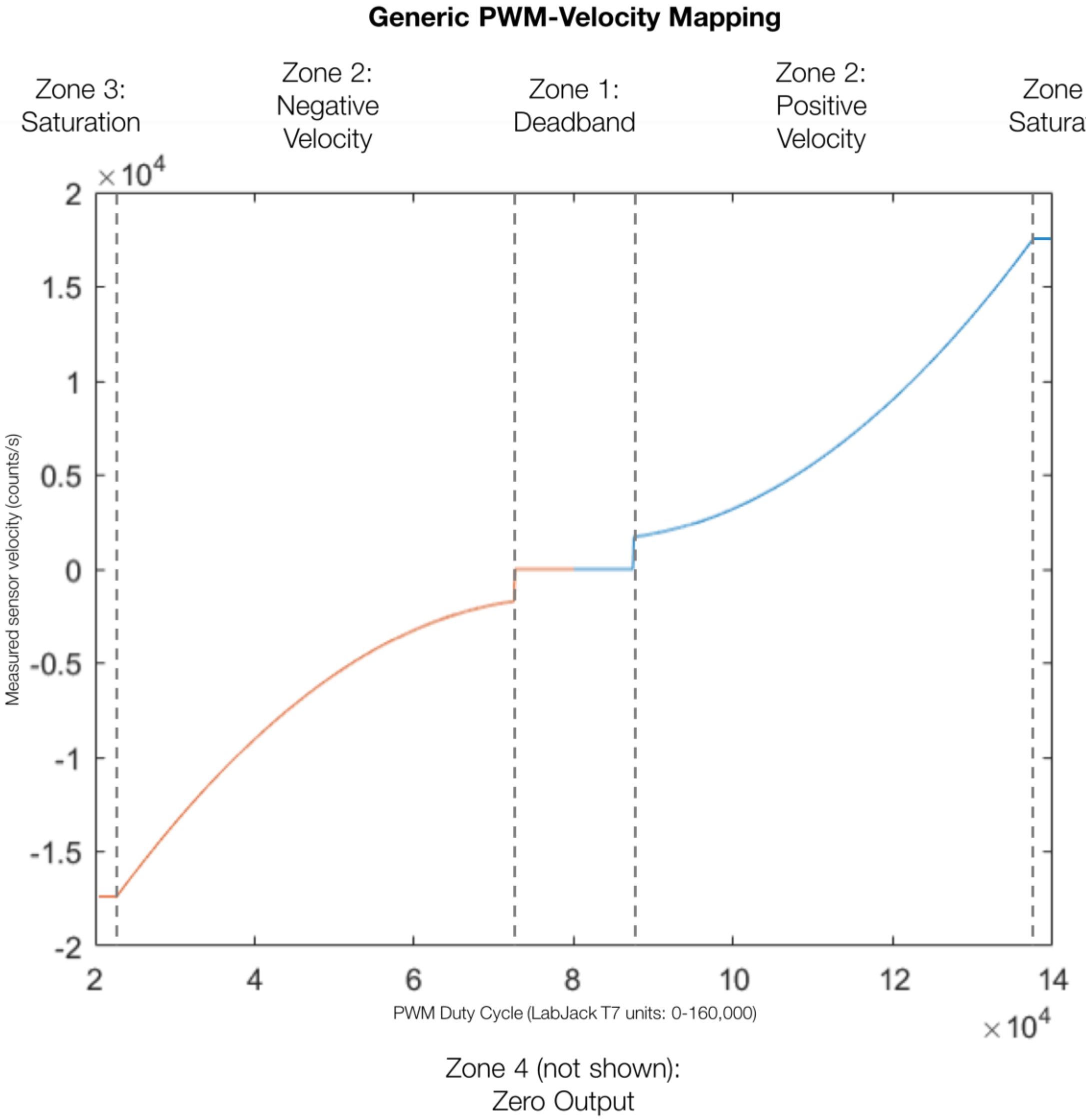


Figure 20: Generic PWM-joint velocity mapping. PWM duty cycle defined for LabJack T7, with 0-160,000 range corresponding to 0-100% duty cycle. Image created in collaboration with Selam Gano.

As the commanded PWM duty cycle moves away from the neutral position (50%), the first region that the system moves through is a deadband region, which reaches between roughly 54.7% duty cycle in the positive direction and 43.8% duty cycle in the negative direction. The width of this deadband is tunable in a limited fashion, but cannot be totally eliminated with current valve hardware & electronics. At the end of this deadband region, the system immediately jumps to a low velocity. Velocity then increases nonlinearly with PWM duty cycle up to some maximum value, beyond which larger PWM duty cycles produce no additional change in velocity. Finally, the system rejects very large/small PWM duty cycles ($> 96.8\%$ or $< 3.1\%$) as a safety feature, and produces zero output when sent these values (not shown in plot).

The PWM-joint velocity relationship was characterized for Joints 1, 3 and 4. This process comprises two parts: 1) characterizing the relationship between input PWM duty cycle and resultant joint sensor measurement, and 2) correlating that joint sensor measurement to a true joint velocity. Finally, for actual use on the vehicle, this relationship was inverted, giving a mapping from desired joint velocity to PWM duty cycle. Each relationship is different, due to differences in joint construction, so the process is described for each joint below.

- Joint 1: To correlate sensor velocity to PWM duty cycle, the J1 joint was cycled between two endpoints at a wide range of different PWM duty cycles, and the average joint velocity measured during each cycle was measured. Figure 21 shows the results of two separate characterization tests. The first test (in red) is a preliminary characterization conducted using the diesel pump with the J4 joint fully extended. The second test (in blue) is a subsequent characterization conducted after the electric hydraulic pump was installed, with the J4 joint fully retracted. The agreement between the two tests is notable, despite the significant difference in rotational inertia about J1 with J4 extended versus contracted, and the change in drive source.

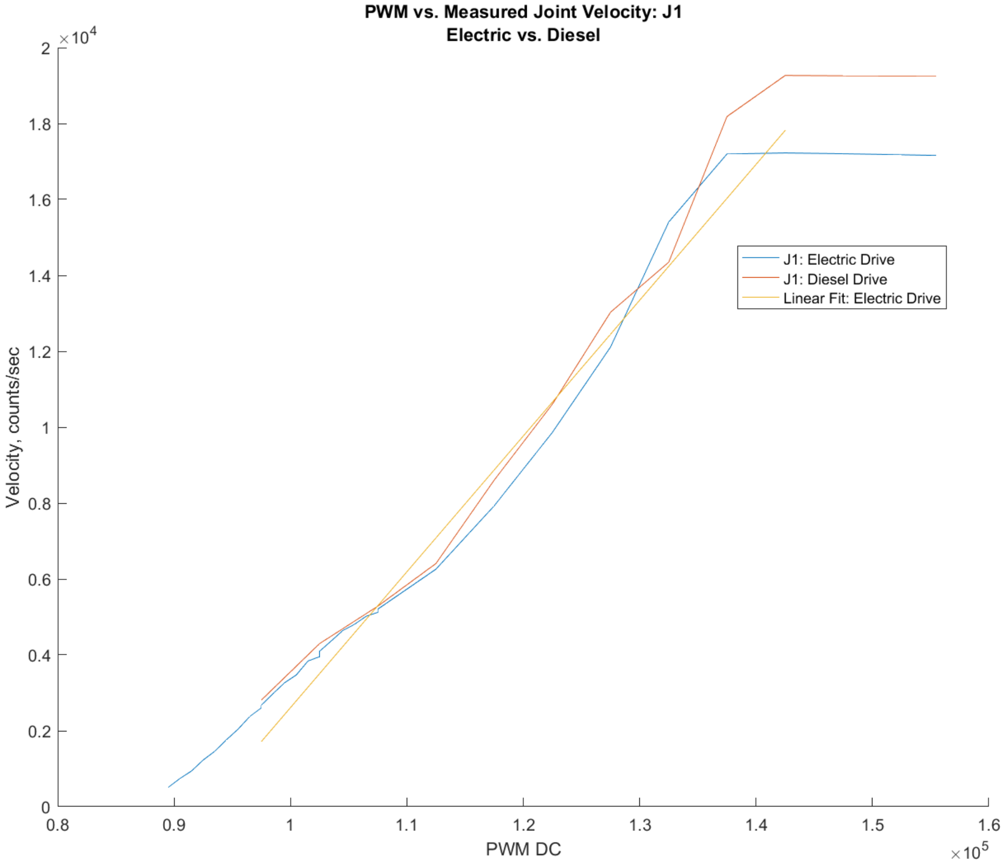


Figure 21: J1 sensor velocity-PWM relationship: diesel vs. electric pump.

A curve fit was performed to the data collected above, generating a fourth-order polynomial that describes the curve shown above on the velocity range [0, 17200] counts/sec. Separately, this data was inverted about the X and Y axes, and another curve fit was generated, giving a mapping between sensor velocity and PWM on the range [-17200,0]. These curve fits were combined into a piecewise function that fully describes required PWM output for all valid sensor velocity inputs. The output of this function for a range of different desired input velocities is show below in Figure 22.

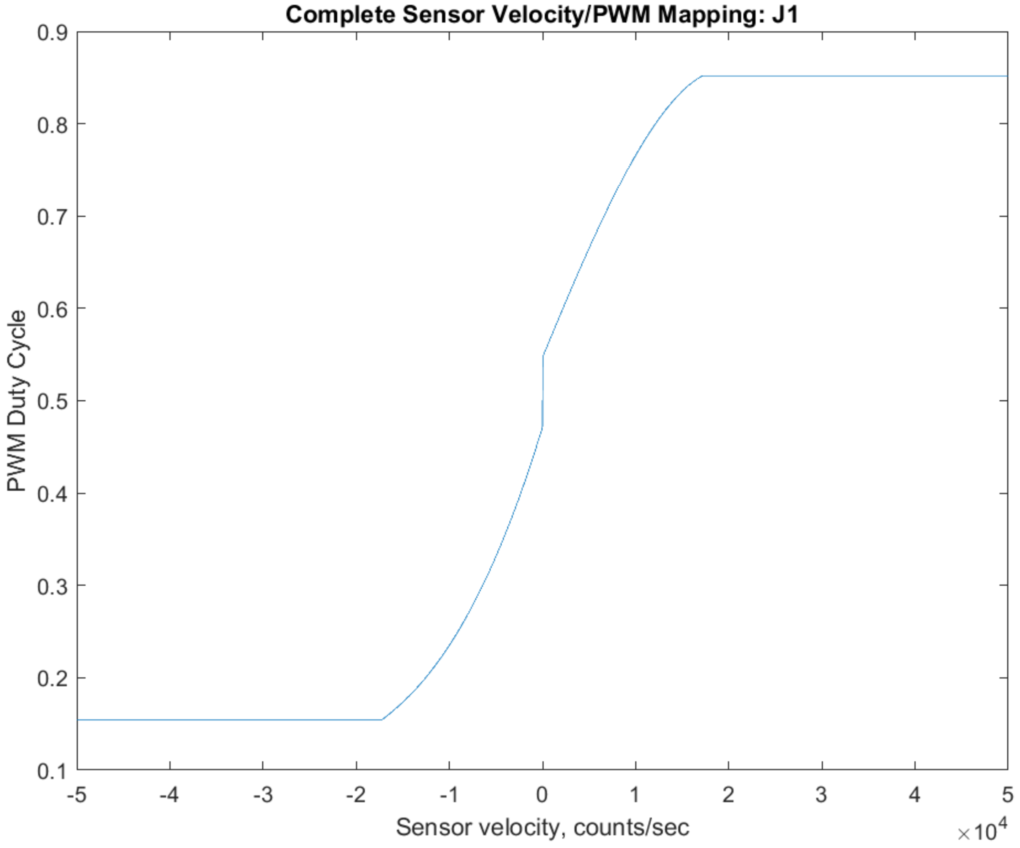


Figure 22: J1 sensor velocity-PWM mapping

- Joint 3: The relationship between sensor velocity and PWM DC was mapped using a similar process to J1, described above. Like J1, J3 was tested with J4 set at three different poses: fully extended, 50% extended, and fully contracted. Despite the significant change in rotational inertia seen at the joint, the system was again observed not to exhibit significantly different performance between tests, as shown in Figure 23.

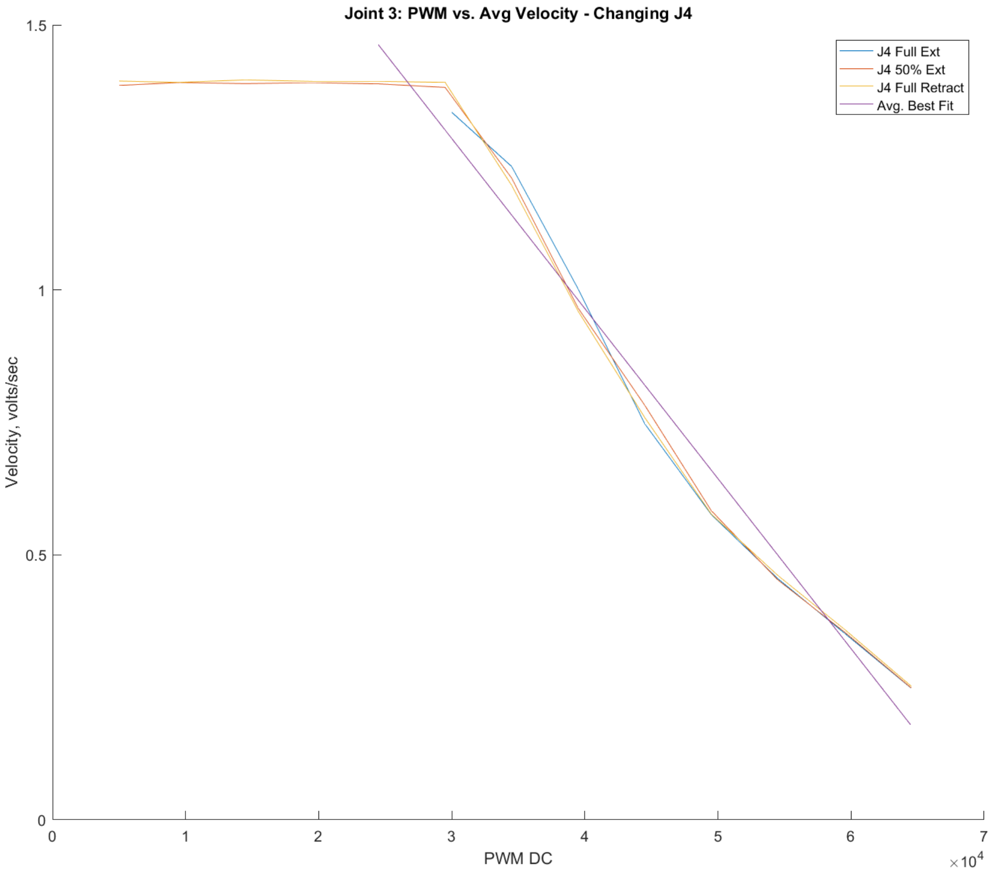


Figure 23: J3 sensor velocity-PWM relationship, multiple J4 poses.

- Joint 4: The relationship between sensor velocity and PWM DC was mapped using the same procedure as used in J1 and J3, with the J3 angle (which adjusts the inclination of the J4 axis) moved between 20 degrees (pointing towards the ground) and -80 degrees (pointing upwards). Results from this mapping are shown in Figure 24.

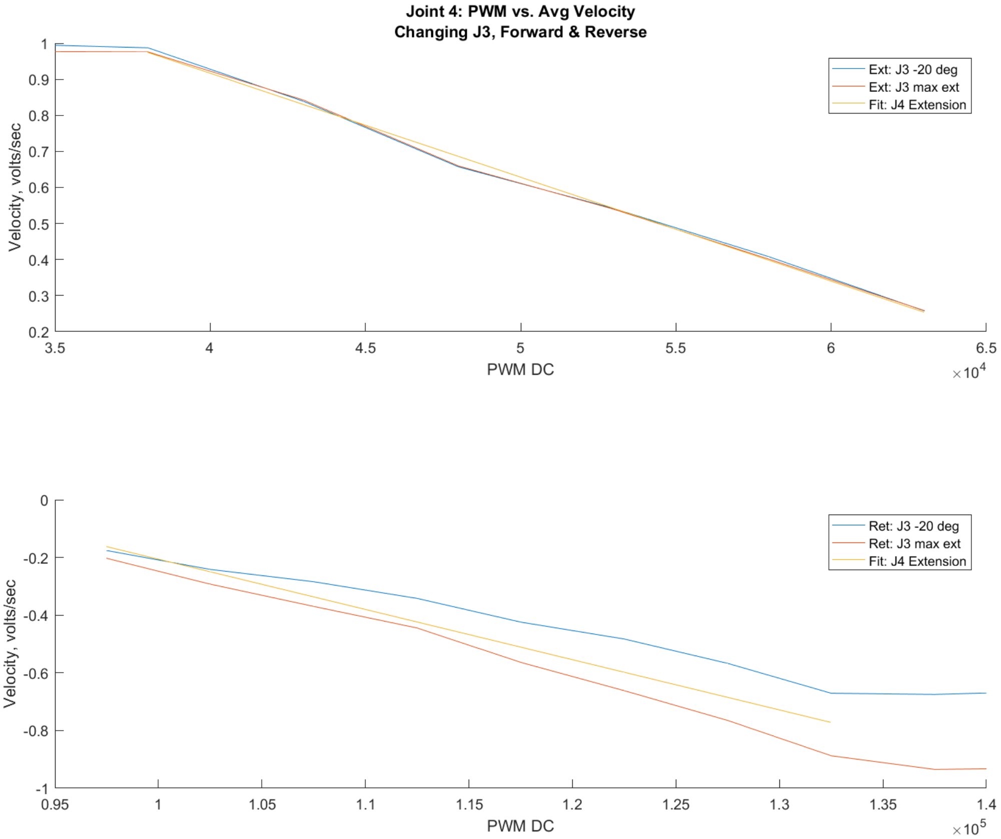


Figure 24: J4 sensor velocity-PWM relationship, multiple J3 poses

Interestingly, while the system extended at similar rates in both poses, the retraction rate differed significantly between poses, with retraction occurring much faster in the upwards-pointing position. This is believed to be caused by the dynamics introduced by the dual counter-balance valves present in the system. These valves are used to ensure that in the event of a hydraulic system failure such as a ruptured hose, the AT40GW joints do not collapse. However, they can cause asymmetric hydraulic responses, thanks to differing pressures required to actuate the valve in each direction.

KUKA KR10 R1100 sixx WP

The KUKA KR10 R1100 sixx WP is a light-payload industrial robotic arm manufactured by KUKA GmBh. It has a maximum payload of 10 kg, a maximum lateral reach of 1.1 m, and a total work volume of 5.2 m^3 [57]. The particular KR10 R1100 variant used in the DCP is IP-67 rated, to enable safe use in worksite conditions.

Real-time control of the KUKA was implemented through the KUKA Robot Sensor Interface (RSI) module [58]. RSI enables direct control of a wide range of behaviors on the KUKA, including control of joint- and task-space position; adjustment of system variables; and triggering of KUKA Robot Language (KRL) programs, at control rates up to 250Hz. A series of Beckhoff analog and digital I/O modules were additionally installed on the KUKA. These modules connected to RSI over EtherCAT, which enabled external signals to trigger KUKA behaviors, and vis-versa: these are described in further detail later.

2.1.2 DCP v.2 Control & Software Architecture

The DCP is a high-DOF robotic system composed of multiple, independent systems, and developing a software and control architecture to manage this system presented a number of challenges.

Command Generation

One major advantage of using a micro-macro manipulator architecture with architectural-scale robots and applications is the ability to operate the robot in multiple different modes depending on the task being completed. In contrast to normal micro-macro manipulators, where the micro manipulator is generally too small to perform any useful tasks autonomously, the micro manipulator in a system like the DCP is a complete robot arm in its own right, with a large usable workspace, six degrees of freedom, and substantial manipulation capacity. By considering the system as both a single monolithic robotic arm, as well as two individual separate arms, three separate operation paradigms can be envisioned:

1) Serial Operation: In this mode, the micro- and macro-manipulators are treated as separate entities. They execute trajectories in series: generally, the macro-manipulator will move the micro-manipulator to a location, and the micro-manipulator will complete a task at that location.

2) Error Compensation Operation: In this mode, the micro- and macro-manipulators are treated as a single unit, as is commonly practiced in the literature (for example, as Yoshikawa describes in [59]). Trajectories are executed by the robot arm as a whole. The macro-manipulator is responsible for trajectory tracking: the micro-manipulator responds only to compensate errors in position or force output.

3) Parallel Operation: In this mode, the micro- and macro-manipulators are treated as a collaborative unit. Trajectories may be assigned to the complete system; and/or to either manipulator individually. The micro-manipulator may provide compensation for errors introduced by the macro-manipulator, in addition to executing its own toolpaths. In the literature, this type of control architecture has been implemented in "arm-finger" robots, which attempt to replicate the operation of the human arm and finger [35]. For systems like the DCP, this mode of operation shows particular promise, as it further supports the use of slow, low-performance arms like the AT40GW as the macro-manipulator by allowing the micro-manipulator to take some responsibility for trajectory tracking.

We have used the DCP to experiment with implementation of each of these modes of operation to at least a limited degree: this is described further in Section 2.2.1 below. For the experiments described in this chapter, and particularly the full-scale case study described in Section 2.2.3, we relied primarily on a serial operation mode. This was done for a variety of reasons. First, the implementation of communication with the KUKA over RSI used in this work was insufficiently robust over the long periods of time required for a full print. Furthermore, in many prints (and particularly the full-scale case study), a clean segmentation between "large" moves and "small" moves is easily found. For example, a layer in a 3D print may be viewed as a "large" move; the vertical step between layers is a "small" move. This segmentation can be exploited by use of a serial operation mode to reduce print time.

Trajectory generation for the DCP was performed using the dcpctrl_v1 MATLAB toolchain [54], available under the MIT license from https://github.com/mitmedialab/dcpctrl_v1. Some components of Peter Corke's RVC Toolbox [60] are used in this toolchain, particularly for task-space trajectory generation. A general schematic of the workflow used to generate a complete trajectory is shown in Figure 25, below.

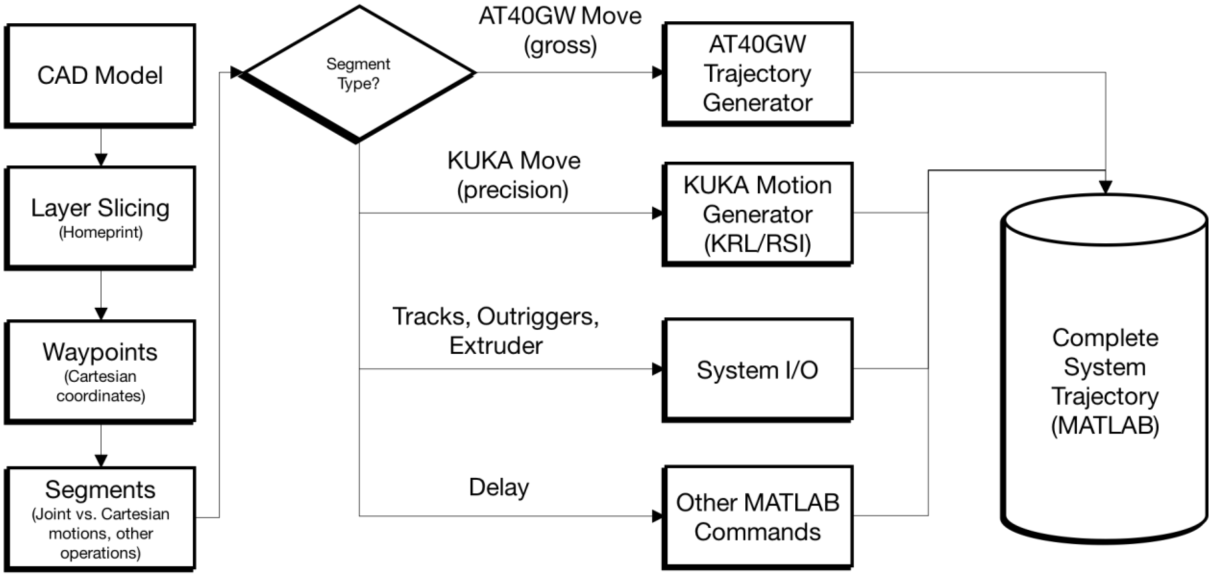

CAD Model
Layer Slicing
(Homeprint)
Waypoints
(Cartesian coordinates)
Segments
(Joint vs. Cartesian motions, other operations)
Segment Type?
AT40GW Move (gross)
KUKA Move (precision)
Tracks, Outriggers, Extruder
Delay
AT40GW Trajectory Generator
KUKA Motion Generator (KRL/RSI)
System I/O
Other MATLAB Commands
Complete System Trajectory (MATLAB)

Figure 25: DCP trajectory generation workflow. Image credit: Steven Keating. Image originally published in [2].

While this figure demonstrates an example of trajectory generation for additive manufacturing of a structure from solid model data, other data sources may be used to provide waypoints for the trajectory generation process, including extraction of edges from 2D images, and algorithmic generation of waypoints. Regardless of source, the procedure for generating a completed trajectory generally involves the following steps:

1) A series of waypoints are generated, either originally or from some type of input file. These waypoints may be arbitrarily sparse, although accuracy suffers at high sparsity. They may be in task-space or (rarely) in joint-space.

2) Waypoints may be on a single continuous path, or alternatively may be broken into distinct segments of motion. These motion segments may be combined with non-motion segments, which trigger other operations such as triggering of KUKA KRL programs; actuation of other subsystems on the AT40GW, including tracks and outriggers; delays; and other commands.
3) Segments which contain move commands are converted to time-parametrized trajectories.
4) Trajectories in task-space are then converted to joint-space.

This process yields a complete system trajectory consisting of multiple segments, each of which may contain a joint-space trajectory for the robot; commands for other subsystems, such as the KUKA or system digital I/O; or other command types.

The majority of the steps involved in generating a joint-space trajectory for the DCP v.2 are relatively standard, and in many cases, standard code from existing robotics libraries such as RVC was directly usable on the DCP. However, the AT40GW poses a substantial challenge for existing kinematic solvers, thanks to the presence of either virtual or real parallel linkage joints in the system at J2 and J3/5 (described previously in Section 2.1.1). These parallel linkages result in joints where the joint angle variable is linked to or driven by a different joint, as seen in Figure 14. Most existing robotic kinematic solvers, such as RVC or the more recent MATLAB Robotics System Toolbox[6], do not support linked joints or parallel kinematic systems (another means of representing these linkages). Consequently, a set of custom forward and inverse kinematics routines were developed for the DCP.

The first FK/IK routine was developed by the author. It relied extensively on the MATLAB Symbolic Math Toolbox to create a complete set of forward and inverse kinematics routines directly from DH parameters. This was done by first generating a symbolic forward kinematics transformation matrix from DCP DH parameters; and then calculating a symbolic Jacobian

[6] The MATLAB Robotics System Toolbox kinematics libraries were released as part of R2017a, and were not available during the 2016-2017 period for analysis.

from that forward kinematics transformation matrix, using the MATLAB Jacobian function. During forward kinematics evaluation, or at each step of an iterative inverse kinematics solver, current joint positions could simply be substituted into the symbolic matrices to generate current FK and Jacobian matrices. Use of the Symbolic Math Toolbox allowed the linkages present in the system to be easily modeled, although it substantially increased the computational time required to implement these routines. The author also wrote a naïve iterative inverse kinematics solver, which incorporated logarithmic step-size modification and detection of joint limits, to leverage these tools.

Subsequently, Levi Cai (a graduate researcher who was part of the DCP team during 2016) made substantial improvements to these routines, including:

- Conversion of symbolic MATLAB forward kinematics transformation matrices and Jacobian to MATLAB function representations using the matlabfunction command. This dramatically increased the computational efficiency of forward and inverse kinematics calculations, and provided a Jacobian "function" that can be used effectively for real-time task-space velocity control.
- Calculation of intermediate forward kinematics transformation matrices for all other DH frames in the system. This enabled the development of a more representative system visualization tool.
- Simplifying inverse kinematics routine, based loosely on the ikine implementation in Corke's RVC [60];
- Vectorizing and modularizing codebase, and adapting trajectory generation code from RVC to provide proper trajectory generation for DCP (no trajectory generation existed before this work).

Today, the FK/IK routine used with the DCP v.2 system is functionally identical to Cai's version. It performs well for the limited set of tasks that the DCP v.2 has been applied to, but has a range of limitations that need to be addressed in future work:

- Since the J2 joint is redundant to the J3 and J4 joints, we have designed our current inverse kinematic solver to treat J2 as fixed, applying no change in position to the joint as the solver iterates. An alternative method for managing this joint, which would enable the use of existing, conventional FK and IK solvers, would be to simply calculate a fixed offset between the J1 and J3 frames at the beginning of path planning, once the J2 joint angle is known. However, neither of these solutions allows the J2 joint to be leveraged during operation. Ideally, a future kinematic solver would allow different conditions to be stipulated for the operation of this joint, allowing it to be used in a limited fashion during operation to adjust the DCP's work volume, or at least to select the optimal pose for this joint for a particular task.

- The J3 and J5 joints are also linked through a virtual parallel linkage in hydraulics, and removing this linkage would be both challenging (either requiring modification of the hydraulics, or active control of the J5 leveling cylinder to compensate for the J3 cylinder's motion) as well as inconvenient (the J5 cylinder ensures that the platform which the KUKA is mounted to remains vertical – generally a useful orientation for the system during fabrication tasks). Consequently, any future kinematic solver solution that is implemented needs to be able to represent this joint linkage.

Control

Trajectories were then executed using a custom MATLAB-based controller, which performed both real-time control tasks as well as subsystem management (control of end effectors, triggering of KUKA KRL programs, etc.).

Control of the macro arm was implemented using simple PID control on position with feedforward velocity terms on a per-joint basis, as shown in Figure 26.

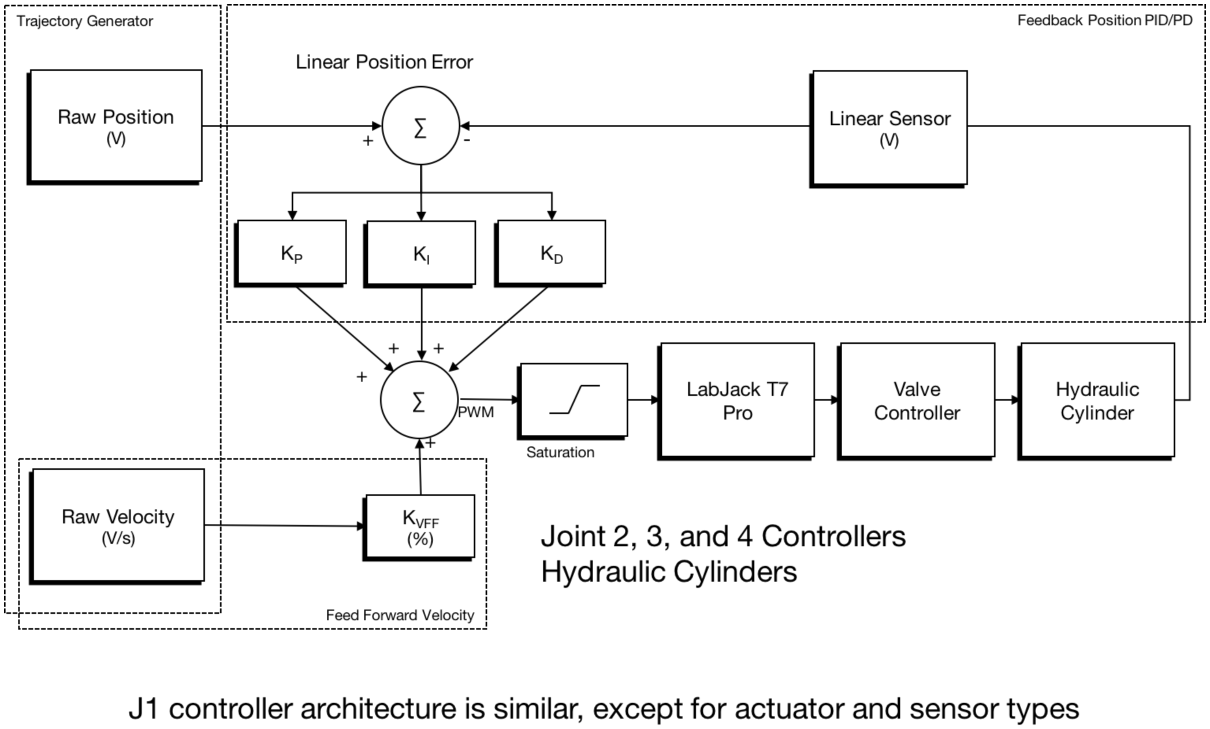


Figure 26: DCP FFPID controller architecture. Image credit: Levi Cai. Variant of this image originally published in [2].

Control was performed in "sensor" space, rather than in joint space. Reference position commands were translated to reference sensor values; feedforward velocity commands were translated to "sensor velocities" using the characterizations of sensor to joint velocity relationships, described in Section 2.1.1 above. PID and feedforward gains were manually tuned, generally on a per-task basis. While simplistic, this architecture was sufficient for early work with the DCP. The relative insensitivity of the AT40GW's actuators to system pose at the low bandwidths that the system was used at allowed us to largely disregard the impact of inertial coupling between joints and still achieve acceptable performance.

For the KUKA, actual joint-space control was relegated to the KUKA's internal controller. References for this control were provided to the KUKA in one of three ways:

- Position Reference Commands from UDP Server: The lowest-level reference protocol used was direct communication of reference position commands to the KUKA via a UDP socket to the KUKA Robot Sensor Interface (RSI). This configuration is the only known means of implementing real-time control of the KUKA from an external source. It is supported in a limited fashion by KUKA[7], and has been used extensively in the robotics and architecture communities, including in previous work by the Mediated Matter Group [61].

In the work described here, we re-implemented the C++ server previously developed by our group. This server sends absolute position reference commands to the KUKA at 83.3 Hz. The KUKA then tries to attain the sent position within the next control cycle, moving as fast as needed to reach this position. This requires that the server calculate the appropriate next position to command based on the KUKA's current position to ensure motion at a specific velocity, creating an additional, rudimentary control loop on velocity here. Desired task-space goal position and velocity commands are sent to the server from a MATLAB function, over a second UDP socket.

The intention of this architecture is that the faster C++ server can provide updates to the KUKA within the 12 ms window that RSI mandates, even when operating in a non-real-time environment; while commands can be generated more slowly – and more sparsely – from MATLAB. However, in practice with the DCP, this link proved unreliable, likely due to differences in system configuration and other sources of system load (e.g. running MATLAB simultaneously). Additionally, the C++ server was not able to take advantage of the faster 4 ms cycle time that RSI can optionally run at. Consequently, for most subsequent experiments, this configuration was abandoned.

- Direct Input from External Sensors: Direct input from external sensors was also experimented with briefly, as described by Keating in [3]. Here, a KUKA RSI configuration was developed by Keating to read position data from an external laser distance sensor (SICK DT35) via an analog voltage signal. This signal was then used to modulate the Z position of the KUKA arm, to maintain a constant distance above the

[7] For example, see the KUKA RSI_Ethernet example in [58].

"ground" surface. Keating demonstrates the KUKA successfully compensating for variations in ground height as the DCP rotates about its J1 joint, over an impressively large range, in [3]. This configuration – which is well-supported by KUKA – is quite robust, and can provide stable, real-time functionality to the KUKA. However, working with the KUKA RSI configuration development software is difficult, and it is challenging to use this type of configuration to implement more advanced functionalities.

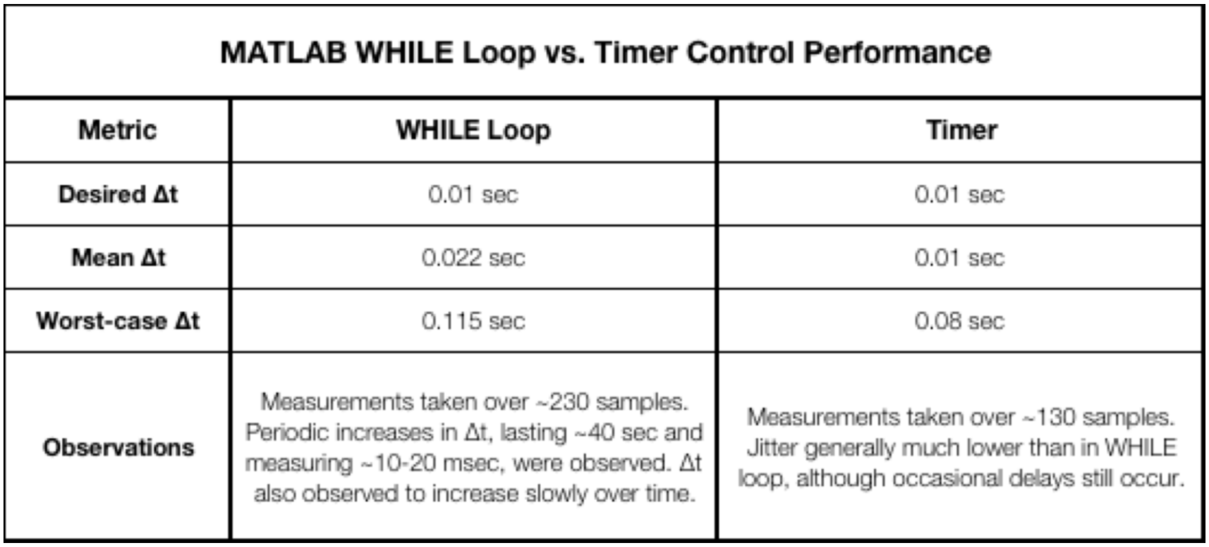
- Digital Input from LabJack to KRL Program: Finally, external analog and digital inputs can be monitored by the KUKA controller and used to trigger previously written programs in the KUKA KRL programming language. This configuration was also implemented by Keating in [3] This configuration provides access to the full complement of trajectory control techniques that KUKA makes available through KRL along with simple, robust communication over digital links – albeit at the expense of narrowly limited functionality. However, for tasks that are repetitive/predictable, this is a viable option for (very) limited real-time system control.

As mentioned earlier, the UDP server that had been developed was insufficiently robust for long-term usage. It was used briefly in a series of small experiments, such as the segmented light painting of the MIT logo described in Section 2.2.1 below. For the primary large-scale demonstration – the dome section print – digital inputs were used to trigger KRL programs to adjust the KUKA's position and orientation as the dome rose.

In general, MATLAB's simplicity and expandability allowed this controller to be reasonably easily adapted to new tasks. For example, when the computer-controllable foam extrusion system described in 2.2.3 was developed by the author, it was straightforward to implement serial communication with the extrusion system through the existing MATLAB control architecture.

The primary shortcoming of the control architecture used during this phase of the system's development was the poor real-time performance of the MATLAB-based controller. The controller was implemented in a MATLAB script, and uses a standard WHILE loop to manage controller timing. Since MATLAB is not designed for real-time operation, the combination of slow execution and high jitter yields a maximum sample rate of 100 Hz – barely adequate for controlling the AT40GW, but too slow to attempt any more sophisticated control techniques, such as input shaping or error compensation using the KUKA.

Experiments were briefly conducted with control loops based around MATLAB's timer functionality. Timers were demonstrated to provide far better performance than MATLAB WHILE loops, as detailed in Table 3:



MATLAB WHILE Loop vs. Timer Control Performance

Metric	WHILE Loop	Timer
Desired Δt	0.01 sec	0.01 sec
Mean Δt	0.022 sec	0.01 sec
Worst-case Δt	0.115 sec	0.08 sec
Observations	Measurements taken over ~230 samples. Periodic increases in Δt, lasting ~40 sec and measuring ~10-20 msec, were observed. Δt also observed to increase slowly over time.	Measurements taken over ~130 samples. Jitter generally much lower than in WHILE loop, although occasional delays still occur.

Table 3: Comparison table – MATLAB WHILE Loop vs. Timer

However, even with these performance improvements, worst-case jitter was still substantial under timer control, with occasional delays of up to 0.08 s observed. When coupled with the challenges associated with implementing timers in the existing control architecture, this line of experimentation was abandoned. Subsequent work with the system, described in Chapter 3, focused on transition to a hard-real-time, Simulink-based control architecture, which enabled much higher sample rates while providing even greater flexibility and accessibility.

2.2 System Implementations, 2015-2016

During the course of 2015-2016, the DCP v.2 platform described above was implemented in a wide variety of experiments, most of which are described by Keating in [3]. Of these implementations, three of the most substantial/complete are showcased in this section: large scale, long-exposure "light paintings" conducted with the system, Print-in-Place fabrication with the system, and finally system characterization using the ISO 9283 robotic performance standard. Each of these implementations provided valuable insight into the DCP v.2 system's capabilities, and guideposts for subsequent development.

2.2.1 Implementation 1: Light Paintings

Throughout the development of both the DCP v.1 and v.2 systems, light paintings – static images or videos that incorporate some degree of light persistence to capture the motion of a light source – have been a valuable tool for capturing system performance and simulating fabrication with minimal overhead and setup. Light paintings may be generated using a wide range of techniques, many of which were developed or expanded upon by Keating in [5]. During work described in this thesis, three primary techniques were used:

- Long-exposure photography: The simplest – and frequently, most aesthetic – means of producing a light painting is to simply take a long-exposure photograph of the system as it executes an operation. A light source is affixed to the point of interest on the system (typically the endpoint), and the camera exposure settings are adjusted so that the light source is visible but the background is obscured. The shutter is then triggered open; the system is run through its operation; and the shutter is closed. Because of the way the camera's exposure has been set, the "trail" created by the light source is captured along with a limited image of the background, but the system is essentially invisible. External light sources such as floodlights can be used to selectively "paint" the scene, illuminating specific areas of the scene or capturing the system at specific times in its operation.

  While this method can produce extremely striking images (as seen below), it has its limitations. It requires that the environment where the painting is conducted be extremely dark. Additionally, this type of light painting is best implemented using a camera with the ability to set an infinite shutter speed (manual open/close) as well as a shutter remote.

- Star-trail stitching: Alternatively, light paintings can be produced from images or videos using software originally intended for star trail photography, such as StarStaX [62]. Sequences of images (which may be extracted from a video using a tool like Apple Compressor [63]) taken of the system as it moves can be imported into StarStaX, which then progressively combines images together. These combined images can be returned as a single image, or saved after each frame is combined and re-stitched into a video again using a tool like Compressor. The returned images or video can then be overlaid and recombined with normally-exposed images or videos of the system, producing compelling images of the system in operation.

  While the images imported into StarStaX must be dark except for the point source of interest, like in the long-exposure photography method, this may be accomplished in post-processing, making this method substantially more adaptable. However, there are limits to what can be accomplished in post-processing, and it is generally advisable to keep the capture environment as dark as possible relative to the point source.

- Video motion tracking with masking: Finally, in extreme cases where ambient lighting cannot be controlled, we have had some success using advanced video editing tools like Motion [64] to dynamically track a point source in a video and mask the video clip around the source. This creates a secondary clip that is dark asides from the point source moving through the scene. This clip can then be used with typical star-trail stitching methods described above.

  This method was not used during work 2015-2016. It is quite time-consuming and generally produces comparatively poor results, but it is a useful technique when ambient light cannot be controlled.

Among other explorations, light paintings were used during development of the DCP v.2 system to visualize the multiple operation modes the DCP v.2 is capable of implementing, described previously in Section 2.1.2. Figure 27 below shows examples of operation paradigms 1 and 3 – serial operation and parallel operation. In Figure 27.a, the AT40GW servos the KUKA between a series of points, while the KUKA draws different segments of the MIT logo, demonstrating serial operation. Meanwhile, in Figure 27.b, the AT40GW rotates about the J1 joint while the KUKA moves its endpoint vertically, producing a sine wave pattern through parallel operation of the two arms[8].

[8] It is important to note that there was no *coordination* between the AT40GW and KUKA during this motion: the two arms executed their movements independently. This image provides an example of how the third operation paradigm might work. Subsequent work, described in Chapter 3, has implemented this concept more fully, with both arms driven by the same controller in a coordinated fashion.

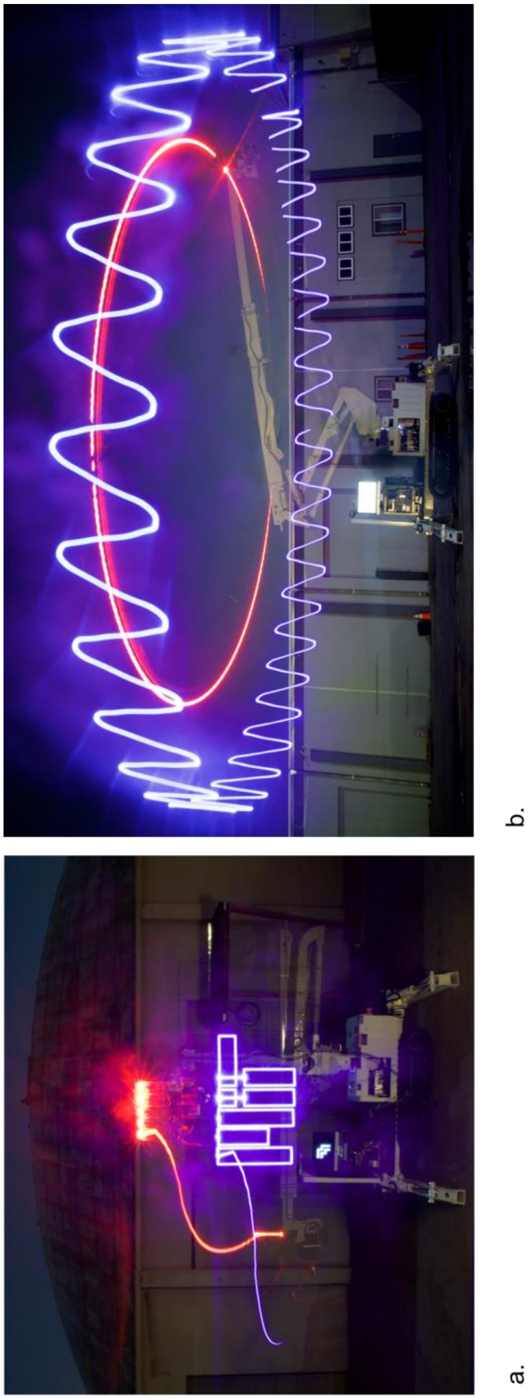


Figure 27: DCP system operation modes. a) Serial Operation – MIT Logo. b) Parallel Operation – Sine Wave. The red light is attached to the AT40GW's endpoint, while the blue light is mounted at the end of the KUKA. Image credit: Steven Keating. Images originally published in [5].

Light paintings have also provided a means of visualizing progress in the DCP v.2 system's development. The following series of figures show how toolpath generation and control for the AT40GW evolved during work 2015-2016.

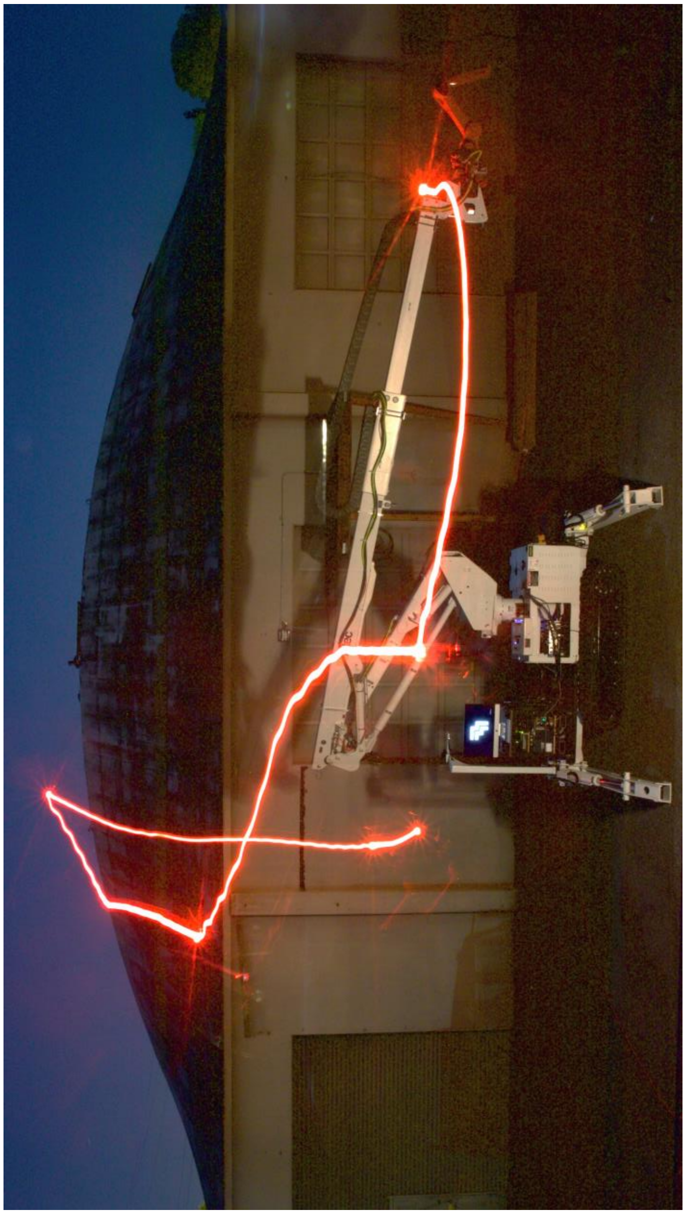


Figure 28: March 28th, 2016 – AT40GW servoing between four locations in joint space. The system is given a desired set point position, and performs proportional control on a per-joint basis to reach these setpoints. Since there is no continuous trajectory or attempt to coordinate joint motions, the joints arrive at their set points at different times. Image credit: Steven Keating.

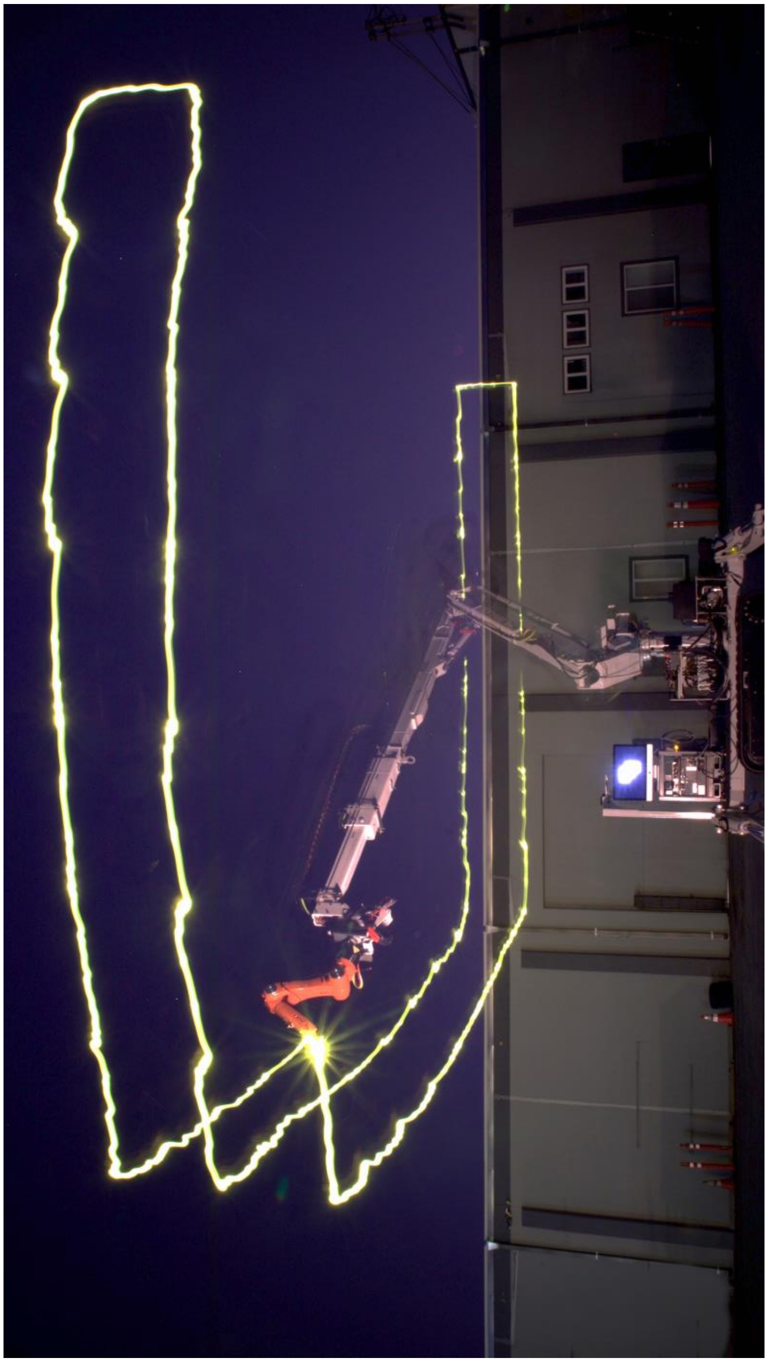


Figure 29: April 20th, 2016 – AT40GW transcribing large-scale rectilinear path using author's initial inverse kinematics solver, with no trajectory generation. The AT40GW is commanded between a series of sparse waypoints, spaced roughly 1 m apart (visible in trajectory), and performs the same type of joint-space proportional control as in Figure 28 above. Image credit: Steven Keating.

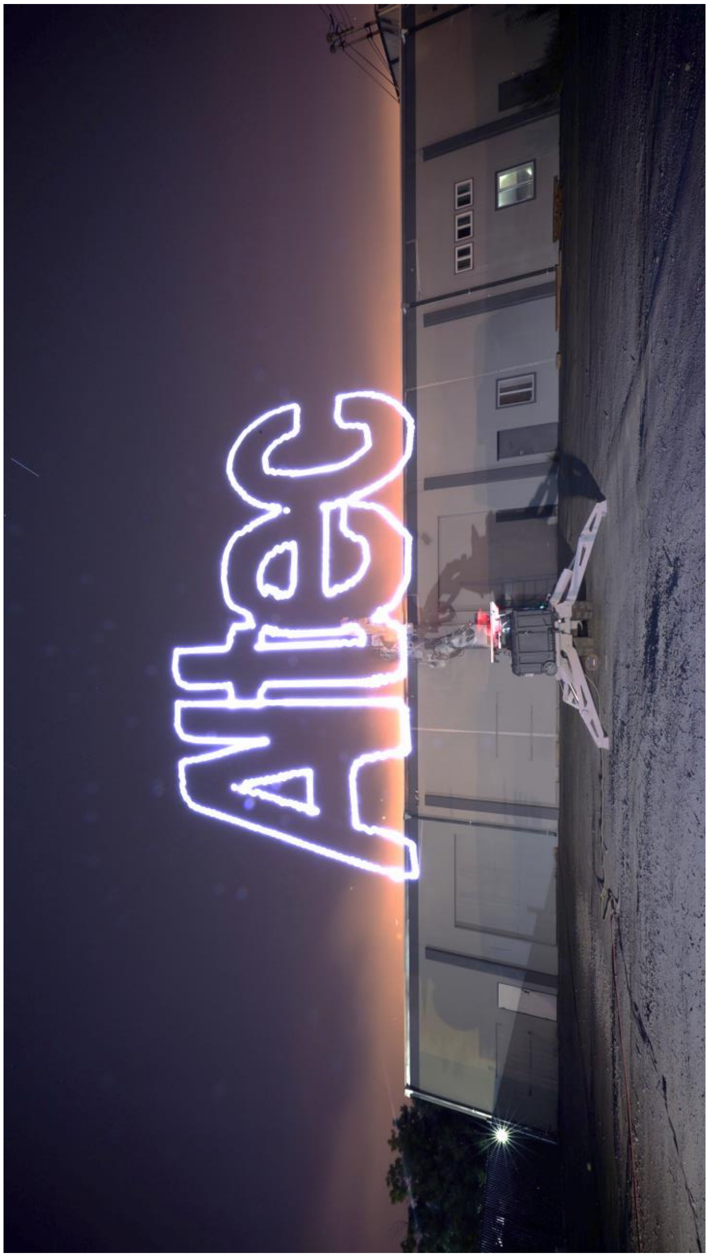


Figure 30: August 8th, 2016 – AT40GW transcribing large-scale complex path, using final inverse kinematics solver, toolpath generation techniques from Corke [11], and image conversion code written by the author. The system is able to provide sufficiently coordinated joint motion to enable curved and straight paths. The impact of system structural dynamics on tracking performance is captured in the painting, particularly where transitions occur between lateral and vertical moves. Additionally, inaccuracies in the robot's kinematic model are apparent in the curvature of the plane of the painting: this is primarily caused by only modeling the robot's kinematics through the end of the boom, as described in Section 2.1.1. Image credit: Steven Keating.

Light paintings have proved to be an invaluable tool during the DCP's development: quick and easy to produce, with minimal overhead and beautiful results. The techniques described here were used throughout 2015-2016, and have continued to be leveraged in subsequent work.

2.2.2 Implementation 2: ISO 9283 Characterization

The second substantial implementation conducted with the DCP v.2 system during this phase of development was the execution of ISO 9283-1998 industrial robot performance characterization test standard, and specifically the pose repeatability test. ISO 9283 [65] is a standard published by the International Standards Organization which provides test protocols for characterizing the performance – and specifically, the interchangeability – of industrial robot arms. It provides a wide range of tests that operators can conduct, including accuracy and repeatability of *pose*; accuracy and repeatability of *path tracking*; cornering deviation; static compliance; and more. Although the ISO 9283 tests are indisputably imperfect[9], the standard remains the only widely-recognized industrial robot performance characterization technique available today, and has been adopted widely by both the robotics industry and the metrology industry.

Because of the substantial errors in our kinematic model of the DCP, described above in Section 2.1.1, we were unable to implement any tests of system *accuracy* during this development phase. At this point, DCP trajectories were being planned at the end of the AT40GW boom rather than the true system endpoint, so any attempt to move accurately would have needed to be characterized at boom end (an inaccessible location) rather than at the true robot endpoint. Instead, we opted to implement a limited version of the ISO 9283 pose repeatability test. This test is only concerned with a robot's ability to re-assume the same pose – regardless of what that pose is, or where it is in the robot's workspace – so it is feasible to

[9] For example, the pose repeatability test specifies that the robot visit the test poses in the same order in each cycle of the test. This fails to capture any asymmetric mechanical defects, such as backlash, that may be present in the robot. While these phenomena are arguably less of a concern for industrial manipulators (which are generally designed to minimize them, and frequently are used to perform cyclical operations similar to the test where they do not pose an issue), for more general-purpose robotic systems like the DCP v.2, the test's limitations are important to consider.

conduct on a robot with errors in its kinematic model. This test provides useful insight into the performance of a robot's sensors, actuators and mechanical design, largely separated from system control or system planning.

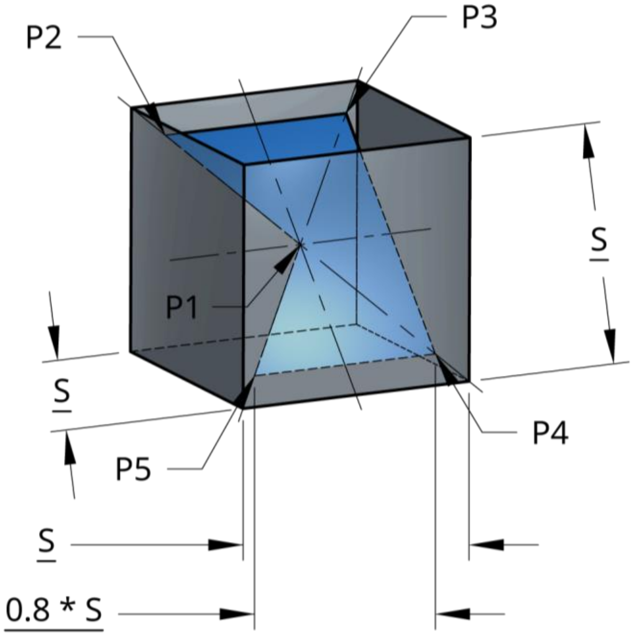


Figure 31: ISO 9283 pose repeatability trajectory specification. Figure reproduced from [65].

In the pose repeatability test, the robot is commanded to move between the center and corner points of the 3D plane shown in Figure 31 above, in the order P1 – P5 – P4 – P3 – P2 – P1, for 30 cycles. The robot must be run at 100% of its rated velocity with 100% of its rated load carried at the endpoint for the test, and may optionally be run at lower velocities/with lower loads if desired. At each pose, the robot is allowed to dwell until it has stabilized, and a measurement of its stable position is taken. Position repeatability is calculated for each pose as:

$$RP_i = \bar{l} + 3S_l = \frac{1}{n}\sum_{j=1}^{n} l_j + 3\sqrt{\frac{\sum_{j=1}^{n}\left(l_j - \bar{l}\right)^2}{n-1}}$$

$$\text{where } l_j = \sqrt{\left(x_j - \bar{x}\right)^2 + \left(y_j - \bar{y}\right)^2 + \left(z_j - \bar{z}\right)^2};$$

$\bar{x}, \bar{y}$, and $\bar{z}$ are the mean x, y and z positions attained at each pose over all cycles;

n is the number of measurement cycles conducted. [65]

For our implementation, we ran the robot at 100 mm/s commanded Cartesian velocity, and with no substantial additional load at the endpoint. Our testing reversed the specified traversal order, although this should have no impact on test results. The test square used was nominally specified to measure 2 m per side (yielding a maximum path length of 1.6 m), although the actual commanded cube measured 2.198 m x 2.091 m x 2.386 m as measured in the DCP's base frame. This is believed to have been caused by the additional, unmodeled kinematic offset provided by the KUKA and J5-6 joint assembly. We did not attempt to control orientation, or characterize orientation repeatability. Test measurements were conducted with a Leica Geosystems Absolute Tracker AT901 laser tracker system, measuring at 3.3 Hz. This tracker is specified to provide a maximum permissible error of better than 500 μm over its full 80 m reach [66].

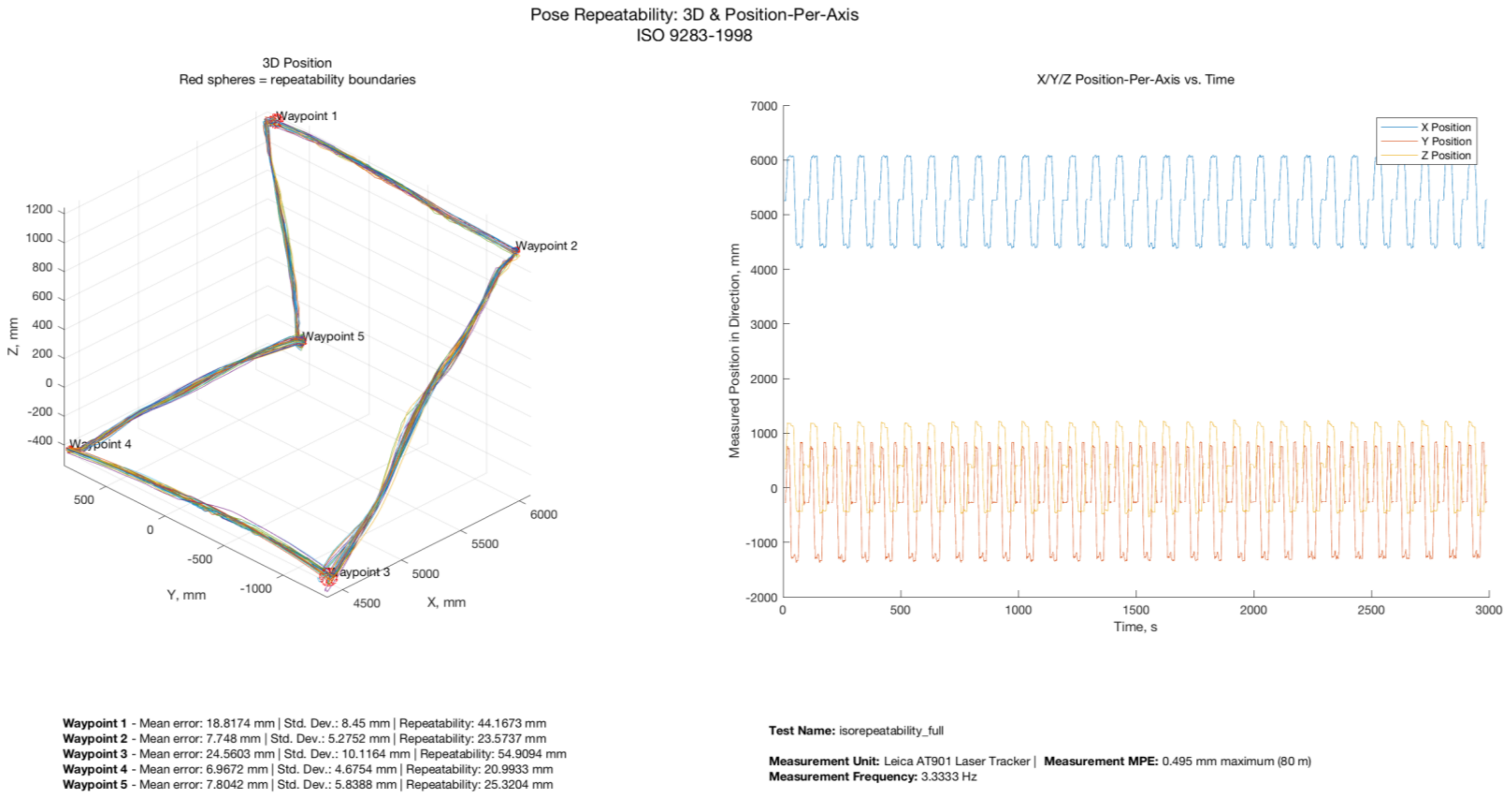
After collection, data was imported into MATLAB for processing and analysis.

- Data Processing: Processing primarily consisted of detecting "segments", corresponding to cycles in the test. This was done by identifying locations within the dataset where both the X/Y/Z position was within some tolerance of the X/Y/Z position at the start of the run, and the derivative of X/Y/Z position (the velocity) was within some tolerance of zero. This analysis created a list of "newmove" flags – locations where a new move has likely begun. These flags were then filtered a second time to remove segments that were too short (not corresponding to entire move segments, but rather created by noise in the data), and the data was finally exported as a MATLAB structure with an entry for each segment, containing measurement index, XYZ position, and time data.

- Data Analysis: Data analysis was performed with the script iso9283analyzer.m, available in the dcpctrl_v1 repository. This script operates by performing the following steps:

- The X/Y/Z position vectors for the first segment of data are loaded, and the derivative of the position for this segment is calculated and filtered to reduce noise.
- Within this first segment of data, locations where the derivative of position is within some tolerance of zero are identified. These are used to generate a set of n positions that are defined as "waypoints," which future runs are compared against. In the case of the ISO 9283 pose repeatability test, there are five waypoints for each segment, with the start and stop position sharing a waypoint.
- Using these waypoints, iterate through all other segments and identify locations where the position is within some tolerance of these waypoints, and the derivative of position is within some tolerance of zero. These are the waypoint measurements for each subsequent segment. They are averages of all points measured while the system is considered to be "at" a waypoint (e.g. meeting the two criteria listed above).
- From all waypoint measurements, generate an average position across all trials for each waypoint.
- Calculate error between waypoint position for each segment and this average waypoint position (lj in ISO 9283).
- Calculate the corrected sample standard deviation for the dataset, as per ISO 9283.
- Calculate final system repeatability measure.

Run data was then plotted as 3D paths as well as individual X/Y/Z trajectories. The final system repeatability for each pose is displayed on the plot as a sphere centered on the mean position of that waypoint, with radius equal to the system repeatability of that pose.



Waypoint 1 - Mean error: 18.8174 mm | Std. Dev.: 8.45 mm | Repeatability: 44.1673 mm
Waypoint 2 - Mean error: 7.748 mm | Std. Dev.: 5.2752 mm | Repeatability: 23.5737 mm
Waypoint 3 - Mean error: 24.5603 mm | Std. Dev.: 10.1164 mm | Repeatability: 54.9094 mm
Waypoint 4 - Mean error: 6.9672 mm | Std. Dev.: 4.6754 mm | Repeatability: 20.9933 mm
Waypoint 5 - Mean error: 7.8042 mm | Std. Dev.: 5.8388 mm | Repeatability: 25.3204 mm

Test Name: isorepeatability_full

Measurement Unit: Leica AT901 Laser Tracker | **Measurement MPE:** 0.495 mm maximum (80 m)
Measurement Frequency: 3.3333 Hz

Figure 32: Results from ISO 9283 pose repeatability characterization, Summer 2016

Figure 32 shows this plot for the pose repeatability test conducted with the DCP v.2 system. The worst repeatability reported for the DCP v.2 over all five waypoints was ±54.90 mm, at Waypoint 3[10]. While the DCP's pose repeatability is many orders of magnitude poorer than what is typical of conventional industrial robots (for comparison, the KUKA used in the DCP v.2 system specifies a pose repeatability of ±0.03 mm [57], and the very large FANUC M-2000iA/1700L specifies a pose repeatability of ±0.27 mm [46]), this is still relatively modest on the scale of construction operations, and moreover, provides a valuable quantitative benchmark for the DCP's performance.

2.2.3 Implementation 3: Print-in-Place Fabrication

The most important experiments conducted with the DCP v.2 system between 2015-2016 examined the system's viability as an automated construction system, through the implementation of the Print-in-Place fabrication process using the system. As described in Chapter 1, the DCP concept was originally conceived by Keating as a means of bringing the Print-in-Place process to architectural scales. Successful implementation of Print-in-Place fabrication using the DCP would both validate Print-in-Place as a competitive option in the architectural-scale AM space, and provide an opportunity to showcase fabrication with the DCP. Consequently, this was made the primary objective of development work during 2015-2016.

Preliminary Print Test: Curved Wall Section

Initial work to implement Print-in-Place on the DCP began early in the system's development process. The first semi-automated print test was conducted in May of 2016, and fabricated a curved wall section measuring 3.3 m in arc length, 1.5 m in depth, and 1 m tall.

[10] It is important to note that ISO 9283 defines system repeatability as mean position error + 3 standard deviations, giving a confidence of 99.7%. By contrast, most other manufacturing system metrology standards, such as the ISO 230-series machine tool metrology standards, only specify a 2 standard deviation confidence interval for most measurements.

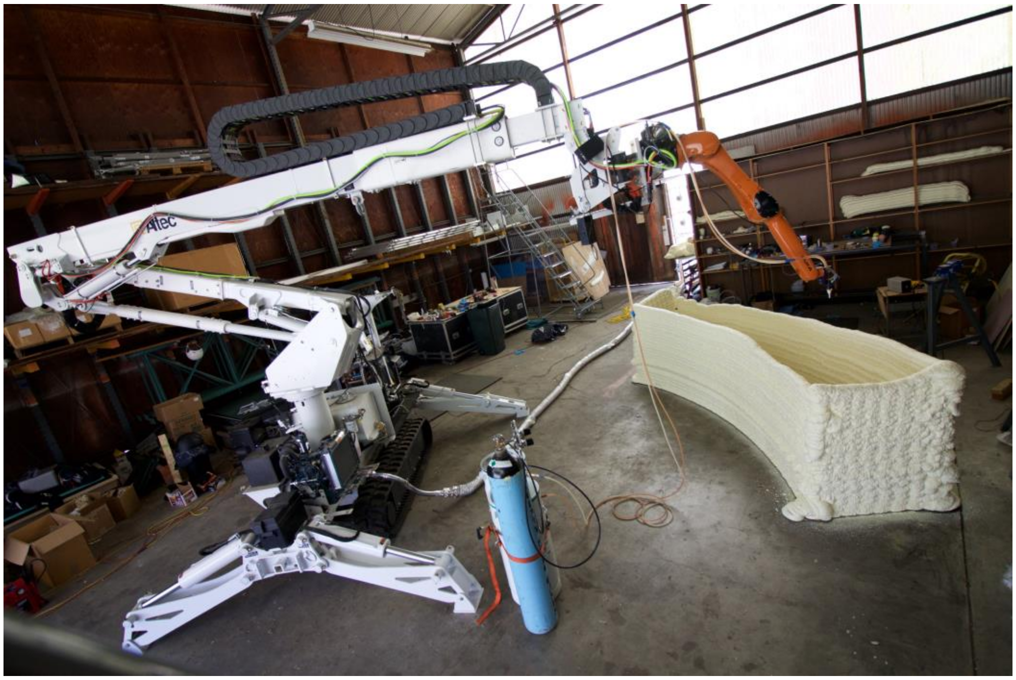


Figure 33: DCP with finished curved wall section. Image credit: Steven Keating.

For this test, the AT40GW was operated using a simple joint-space controller, and control was performed between waypoints at the corner of the structure, rather than along a continuous trajectory. The KUKA was not actively controlled during this test, although it was adjusted manually on two occasions during the test to correct the distance between the spray foam nozzle and the previously printed layer. Control of the foam extrusion system used for Print-in-Place fabrication was initially performed manually, using a pendant controller developed by Keating and shown in [3].

This initial experiment provided a range of valuable observations that informed future work. First, the process successfully executed the print, fabricating 26 layers to create a structure measuring 106 cm tall, as shown in Figure 33. Mean layer height as measured from the outside of the structure was 4.5 cm, with a standard deviation of 1.45 cm[11].

[11] It is important to note that layer height measurement with Print-in-Place fabrication is challenging and prone to inconsistency. Layer height in this experiment was measured between horizontal striations

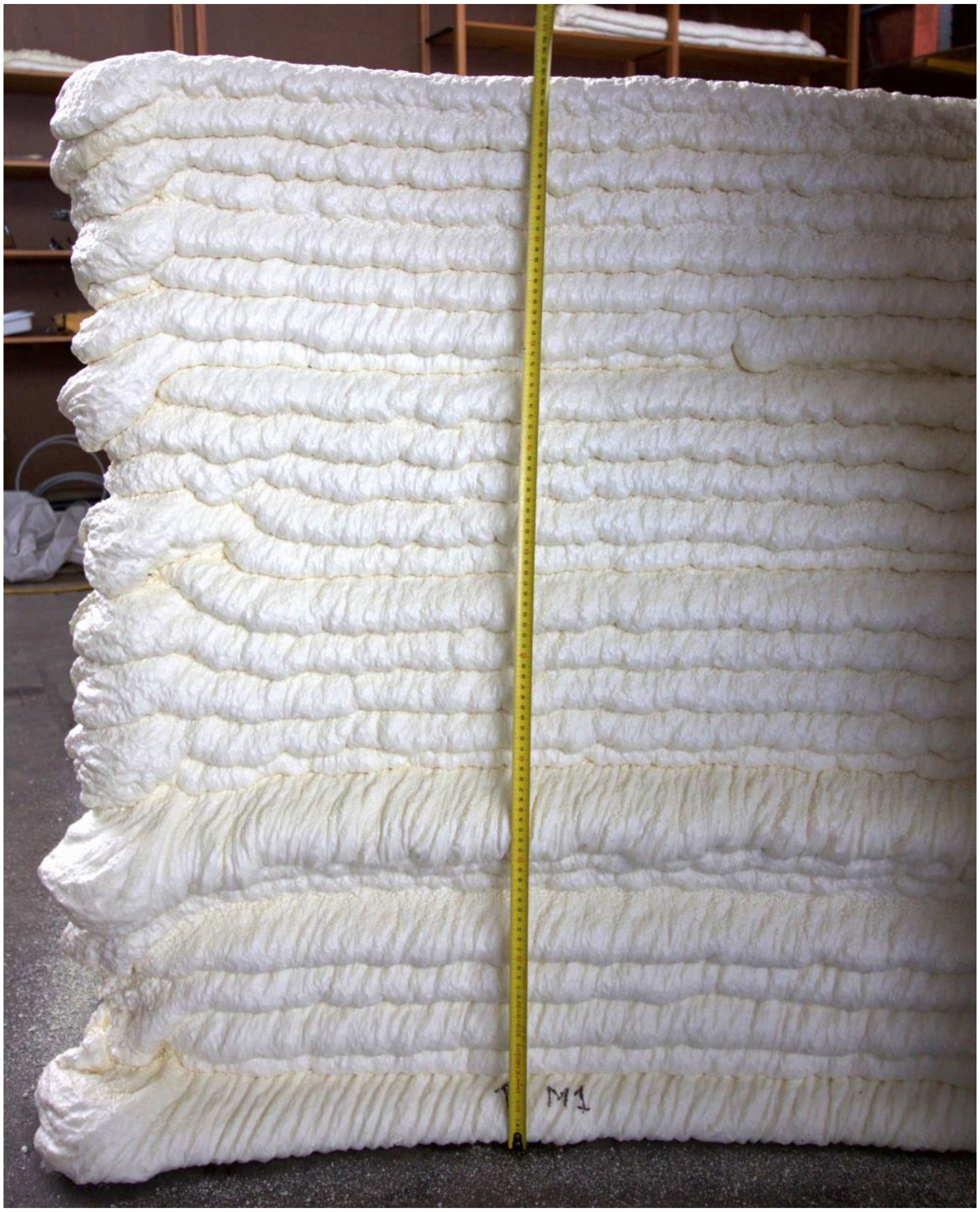


Figure 34: Finished curved wall section layers, with ruler for scale. Image credit: Steven Keating.

between layers, as viewed from the outside of the structure. However, the relative position of sequential layers can cause a layer to “ooze over” the previous layer, potentially obscuring it, and substantially impacting the measurement of layer height. As an example, 26 layers printed at the mean layer height reported here does not correspond to the final measured height of the wall. A more consistent technique defining/measuring layer height would be to cut through the wall and measure the distance between layers along the geometric centerline of the wall.

Second, as Keating reports, binary control of foam flow resulted in buildup at locations on a structure where the DCP's motion slowed (Figure 35). This was partially caused by the control method used for this experiment: the lack of a smooth trajectory along the structure caused the DCP to stop at each corner, producing buildup. However, the importance of being able to continuously modulate foam flowrate to match arm velocity was clear.

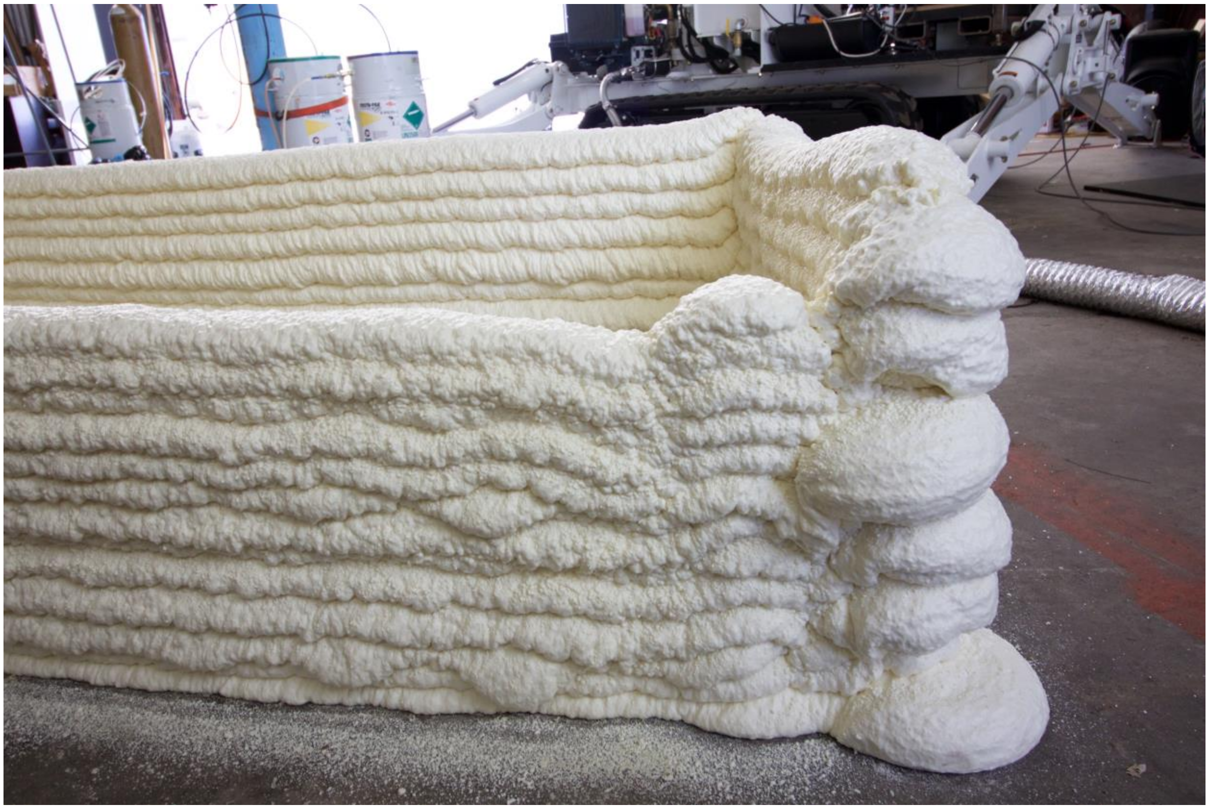


Figure 35: Foam build-up at corner of curved wall section. Image credit: Steven Keating.

Finally, the impact of the AT40GW's high structural compliance was clearly apparent in this print. Figure 36 shows images of the foam traces produced near the end of the lateral traverse (Figure 36.a), and immediately after the DCP began a lateral traverse (Figure 36.b). Oscillations introduced into the boom by the abrupt beginning of motion manifest themselves as "blobs" in the foam, created by the endpoint's motion effectively slowing and then speeding up as the boom oscillates. While undesirable – certainly from an aesthetic perspective – these effects were determined to be sufficiently small as to not impact the integrity of the print.

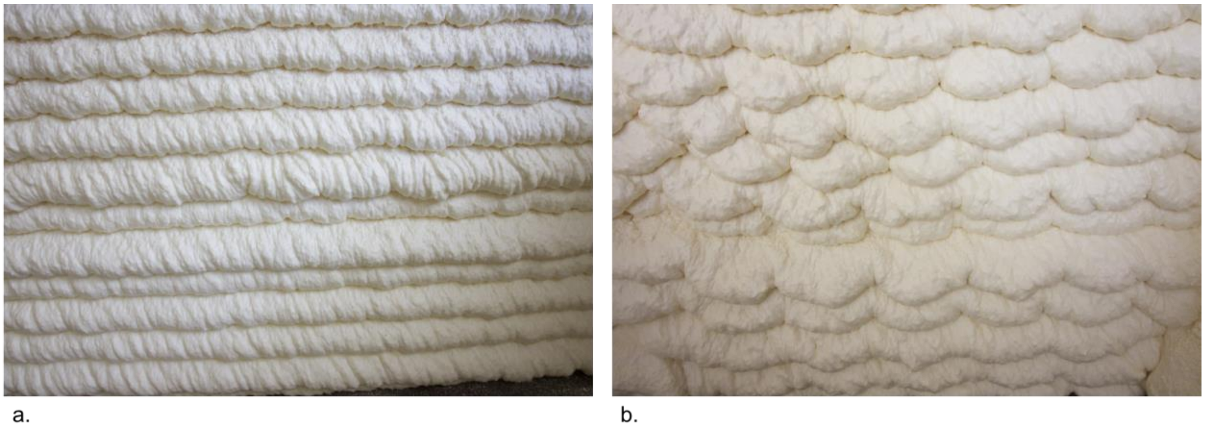


Figure 36: Comparison of layer topography with boom at steady state (a) and with boom lateral oscillation (b). Image credit: Steven Keating.

Print Parameter Calibration and Sprayer Development

This small-scale test provided sufficient confidence to continue work on full-scale fabrication with the Print-in-Place process, but also provided important insight into areas requiring further development. Particularly, it was clear that simple binary control of foam flowrate would not be adequate for a full-scale print.

To address this, a modified version of the spray foam controller originally designed by Keating was developed by the author. This new controller separated actuation of the servomotor controlling the foam sprayer from the command input interface, and connected the two systems over a wired Ethernet UDP link between two Arduinos carrying Ethernet Shields, providing the ability to control the foam sprayer from a considerable distance while providing sufficient robustness to EMI generated by the DCP. In addition to re-implementing the manual controls provided in the original spray foam controller, the new controller also provided a serial input interface, which was connected to MATLAB to allow real-time control of foam flow rate.

Using this new controller, a qualitative characterization of optimal print parameters was conducted. A test pattern was developed to allow variation of flow rate (measured as valve position, rather than using dimensionally appropriate units) and Cartesian feed rate. For each flow rate, an arc-shaped path was transcribed, at feed rates varying between 50 mm/s and 250 mm/s in 50 mm/s increments. The resulting foam traces can be seen in Figure 37, below.

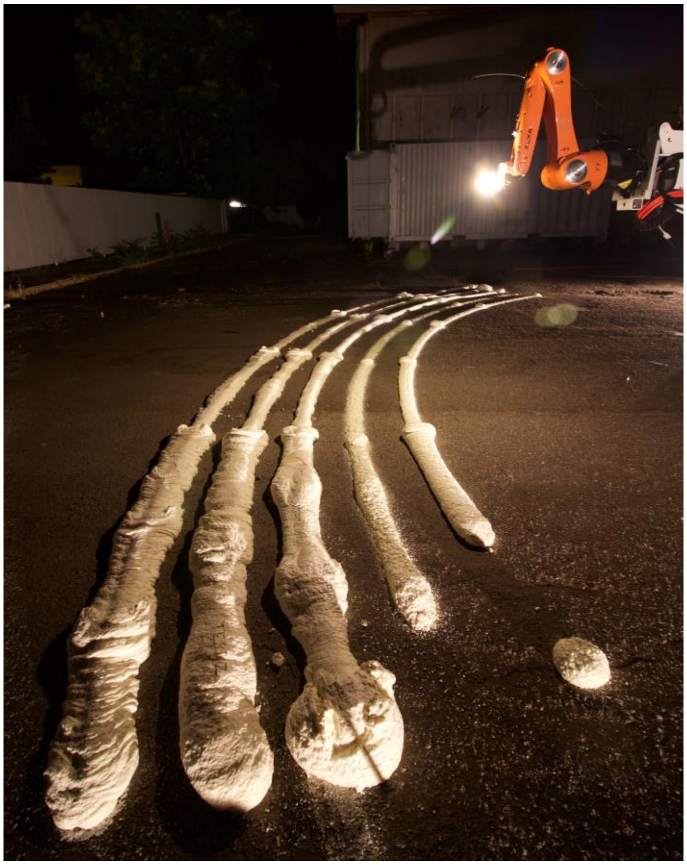
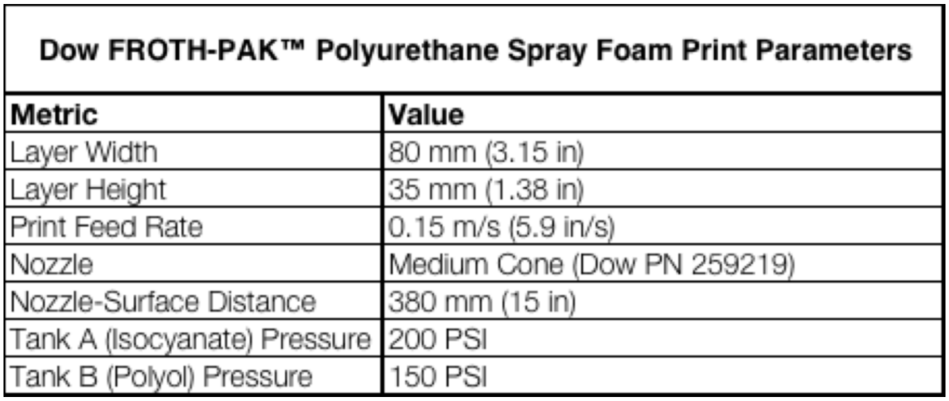


Figure 37: Qualitative determination of optimal feed and flow rate for Print-in-Place process using DCP. Image credit: Steven Keating.



Dow FROTH-PAK™ Polyurethane Spray Foam Print Parameters

Metric	**Value**
Layer Width	80 mm (3.15 in)
Layer Height	35 mm (1.38 in)
Print Feed Rate	0.15 m/s (5.9 in/s)
Nozzle	Medium Cone (Dow PN 259219)
Nozzle-Surface Distance	380 mm (15 in)
Tank A (Isocyanate) Pressure	200 PSI
Tank B (Polyol) Pressure	150 PSI

Table 4: FROTH-PAK™ optimal print parameters

From these tests, a set of viable print parameters was extracted, shown in Table 4. While these parameters were sufficient for the full-scale fabrication experiment described below, it is important to recognize that this is not a complete characterization of the impact of these parameters on the Print-in-Place process. Further quantitative analysis of how best to use spray foam products as print media – in the model of authors such as Barnett and Gosselin [52], who have conducted some limited characterization for their foam printing experiments – is an important topic for further research.

Full-Scale Print: The Dome

To conclude the development of the DCP system during 2015-2016, a full-scale print test was devised to showcase the abilities of both the DCP and the Print-in-Place process. To maximize the size of the fabricated structure, a dome-shaped structure that would closely match the DCP's roughly hemispherical work volume was proposed. This structure was designed to be "hemi-ellipsoidal" in section (a design developed by previous Mediated Matter researchers), with the wall thickness tapering as the structure rises to minimize total structural weight. As described in [2], this structure would be extremely challenging to build with conventional insulated concrete form techniques, but poses no particular challenge for additive manufacturing processes, showcasing the value of additive manufacturing in construction applications.

The toolpath for this dome segment was generated manually in MATLAB, using the script dometraj.m. It was found that as the DCP reached towards the edges of its work volume, the change in joint positions required to produce the vertical offset between layers was too small for the DCP to accurately servo between. Consequently, system movement was divided between the KUKA and AT40GW, with the DCP only moving upwards every three layers, and the KUKA providing vertical adjustment for intermediate layers (as well as lateral adjustment between the inner and outer wall traces).

Fabrication of the dome structure commenced on the morning of July 21st. The actual print time required to print the structure was approximately 13.5 hours, although printing continued through July 22nd. The finished print, shown in Figure 39, measured 14.6 m in diameter, 3.7 m high, and was composed of 306 layers. The dome could have been printed substantially higher, although it is important to note that the dome was never designed to print to full closure because of kinematic limitations of the DCP. However, a combination of factors, including the long duration of the print, and the inaccessibility of the print surface and the DCP nozzle at the final height, motivated the decision to stop printing at 3.7 meters. Finally, the dome was not filled with concrete or any other type of structural material, because of the cost, time and complexity in disposal that this would have added. Further details of the print process may be found in [3] and [2].

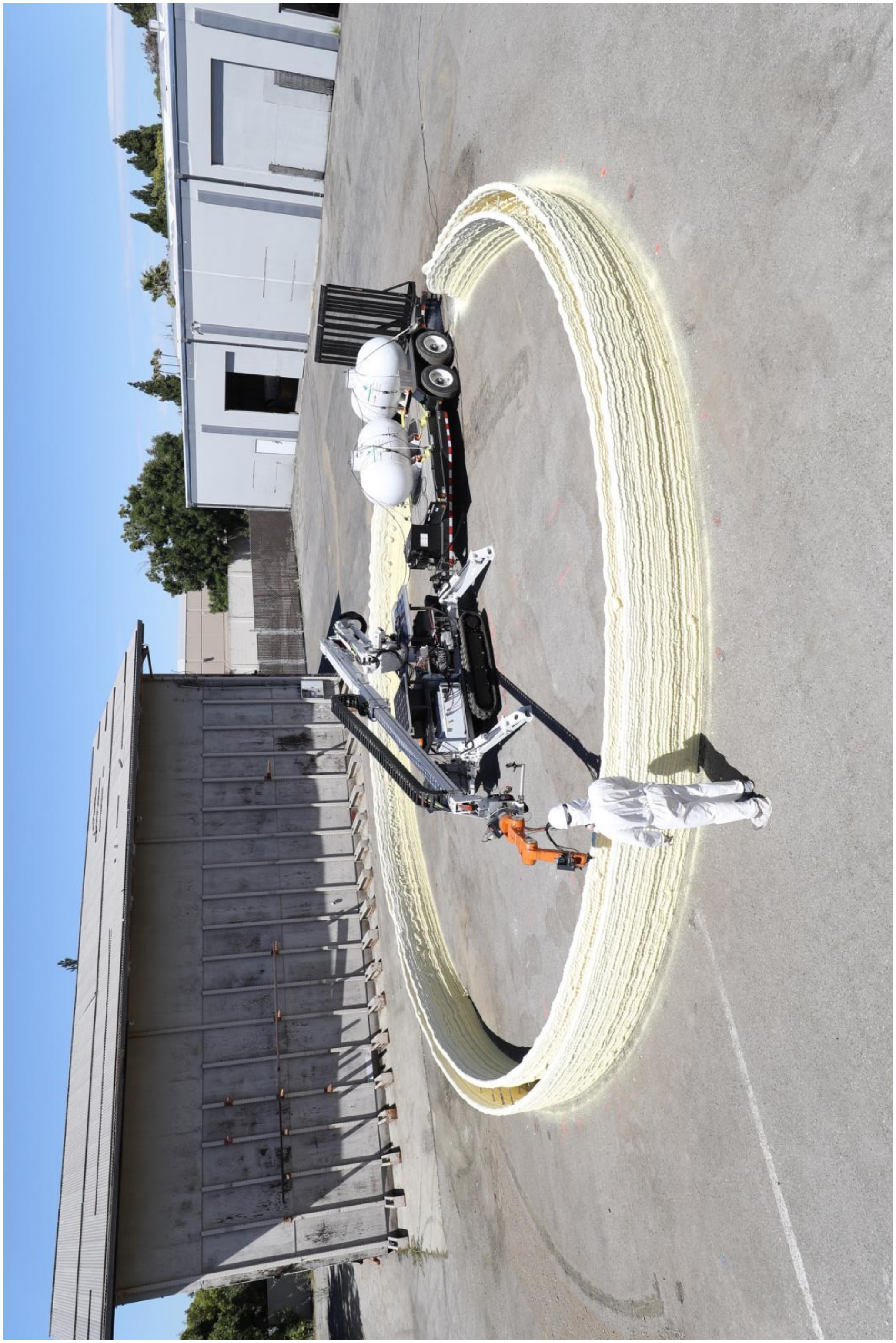


Figure 38: Beginning of dome print. Image credit: Steven Keating.

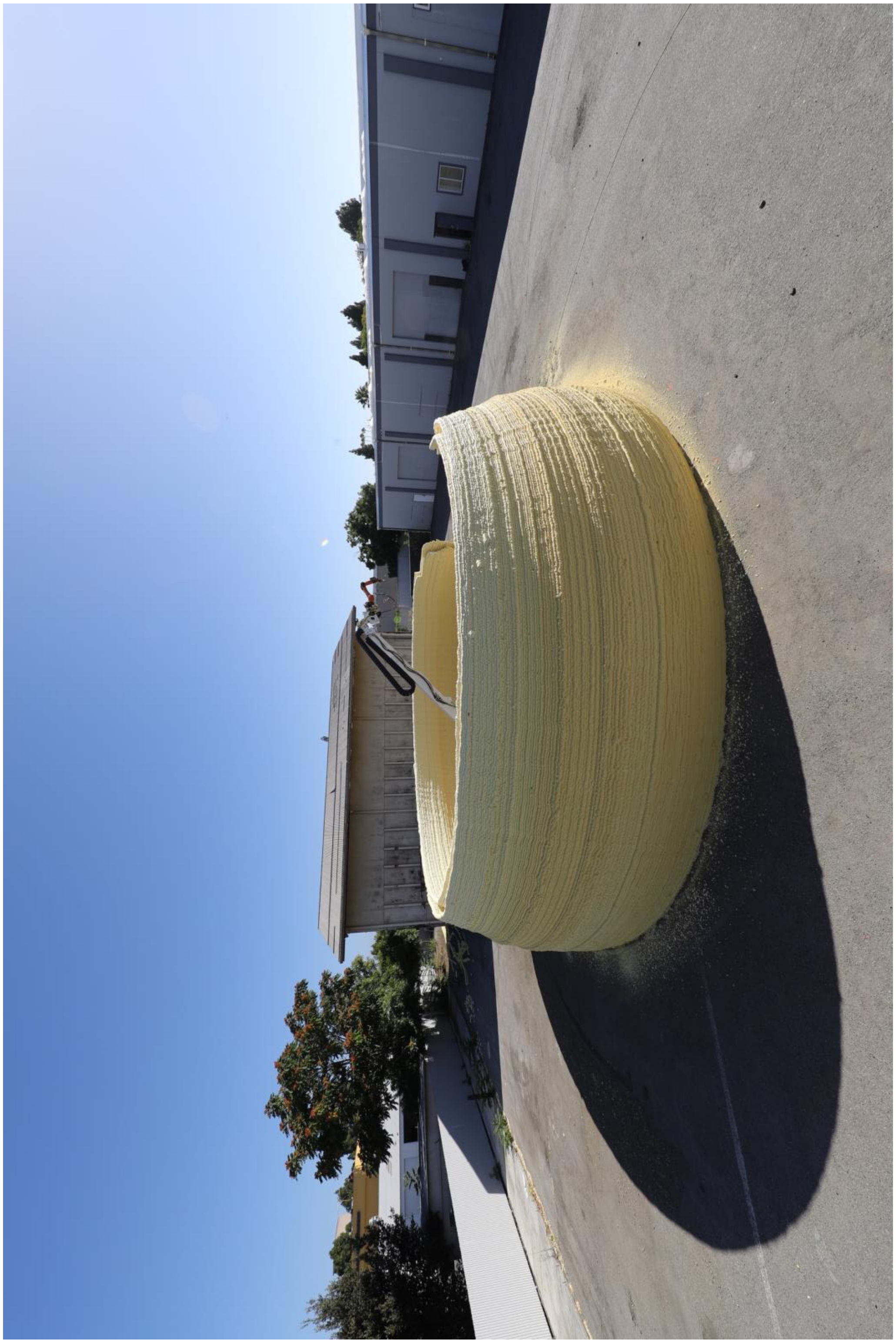


Figure 39: Completed dome. Image credit: Steven Keating.

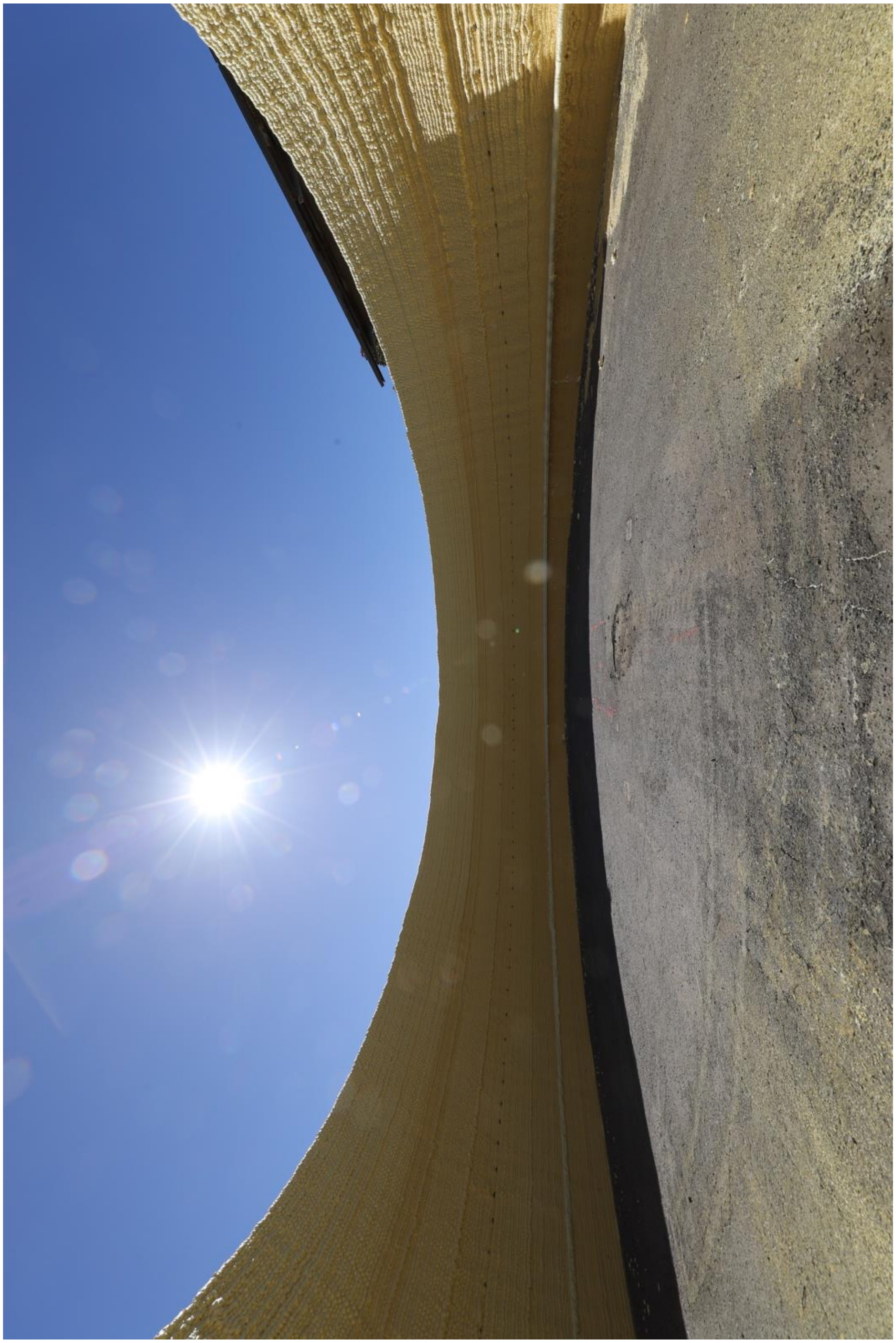


Figure 40: Completed dome, interior view. Image credit: Steven Keating.

In addition to demonstrating the viability of the Print-in-Place process at large scales, the duration and size of this print also provided a number of valuable insights:

- Layer imperfection attenuation/amplification dynamics: As described previously in Chapter 1, the polyurethane spray foam used in Print-in-Place fabrication exhibits some degree of self-leveling, enabling it to tolerate imperfections in the print surface. It is also relatively insensitive to variations in spray height. This is advantageous in construction applications, where it may be challenging or impossible to provide a smooth surface for printing on.

  In preparation for the dome print, the lot where the print was to be conducted was cleaned extensively with powerwashers and blowers to try to reduce the amount of dust and gravel on the print surface. However, the underlying asphalt was still in poor condition, with numerous cracks and pits. Many of these were filled manually, but the print surface was still far from flat or smooth.

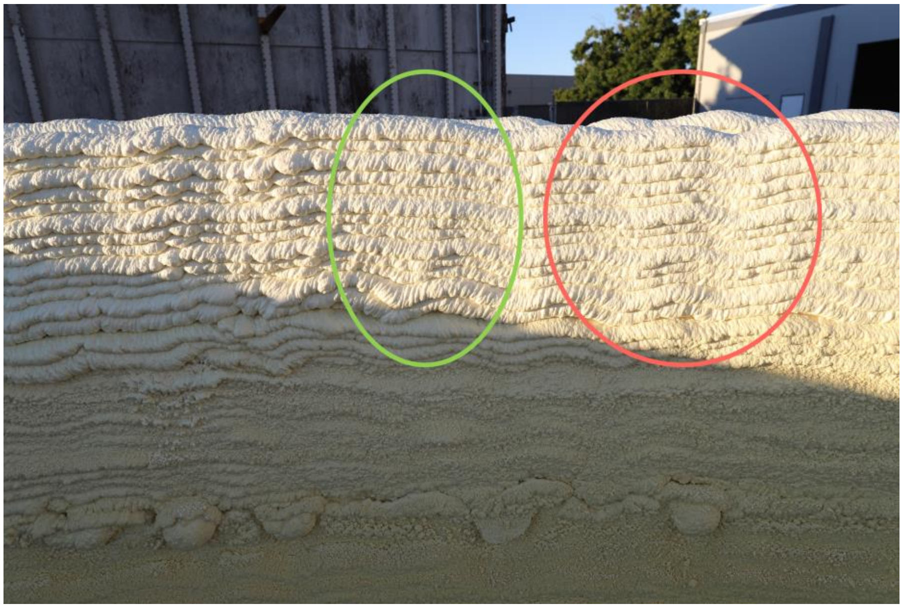


Figure 41: Layer imperfection attenuation (green) and amplification dynamics (red)

As the print progressed, these surface imperfections were transferred into the printed layers, as shown in Figure 41. In some cases, these imperfections attenuated as the print progressed, eventually producing a flat surface. However, in others, the imperfections seemed to amplify as the print progressed. In some cases, these imperfections became so substantial that we took manual corrective action, including filling pits manually with foam from handheld canisters, and on two occasions cutting away a thin layer from the top of the dome to provide a flat print surface. The dynamics behind this behavior have not been examined, and are likely extremely complex. Instead, techniques like the in-situ print height correction implemented by Barnett and Gosselin [52] could be used to address these issues in future prints.

- Layer adhesion and humidity: Another interesting potential failure mode for Print-in-Place fabrication was uncovered on the morning of the second day of printing, when dew settled on the top layer of the dome. The dew acted as a contaminant, impeding bonding between the fresh foam and the previous layer. Furthermore, the water in the dew reacts with the foam's isocyanate component to produce CO2 gas; and can also act as a poor blowing agent when present as the foam components react and heat up. It is believed that some combination of these factors caused extra gas/large voids to be produced at the interface with the previous layer (Figure 42.b), and reduced the bond integrity between the layers. This caused the new layer to fail to bond to the previous layer, and fall off the side of the structure (Figure 42.a)

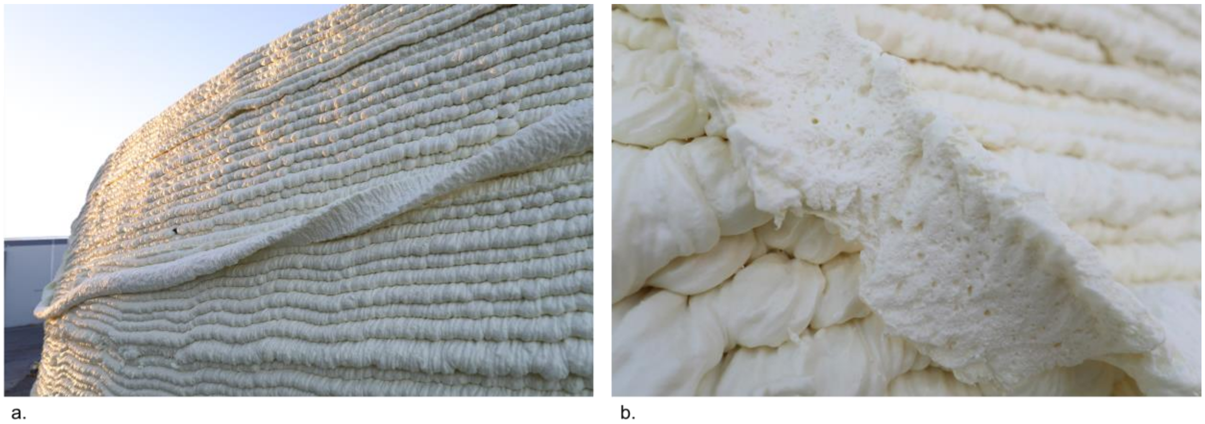


Figure 42: Layer failure – foam layer peeled away from structure (a) and voids in underside of foam layer (b). Image credit: Steven Keating. Image (a) originally published in [3].

> Thankfully, this failure was easily remedied by removing the unbonded layer; manually drying the top layer of the structure; and restarting the print. However, this is an important concern for outdoor application of the Print-in-Place process, where changes in humidity or rain are common occurrences. Modifying the chemistry of the foam used could potentially mitigate this problem, as could implementing some type of drying system (for example, a hot air stream directed immediately ahead of the extruder).

Overall, the fabrication of the Print-in-Place dome was a major success for the project. The structure was fabricated rapidly and effectively, with relatively few process errors. Despite not being filled with structural material, he completed dome stood for nearly a month outdoors before demolition without suffering any noticeable damage. The structure remains among the largest monolithically-fabricated additively manufactured structures ever built, and is a testament not just to the capabilities of the DCP, but to the viability of additive foam-based processes for automated construction.

2.3 Review of DCP v.2 Architecture 2015-2016

Development of the DCP v.2 system between 2015 and 2016 made substantial progress toward the project's core goal of providing a platform for experimentation in the automated construction space. The Altec AT40GW hydraulic lift that serves as the DCP v.2 macro manipulator was instrumented, and a number of important system performance metrics were characterized. Three separate means of interfacing with the KUKA micro-manipulator – RSI UDP communications, direct input from sensors, and triggering of KRL programs – were implemented. A MATLAB-based software architecture, including toolpath generation, planning and control – for the complete system was developed and optimized. Finally, the DCP v.2 system was used in a series of implementations, including large-scale light paintings; system performance characterization using the ISO 9283 pose repeatability test; and successful fabrication of an architectural-scale structure using the Print-in-Place process.

2.3.1 The DCP v.2. in Context

With the DCP v.2's development at a point where large-scale fabrication with the system has been successfully implemented, we return to the comparative performance analysis proposed in Chapter 0, to examine the DCP in the context of other contemporary automated construction systems. Figure 43 and Figure 44 show the plots originally presented in Section 1.1.3, with the DCP v.2 system incorporated.

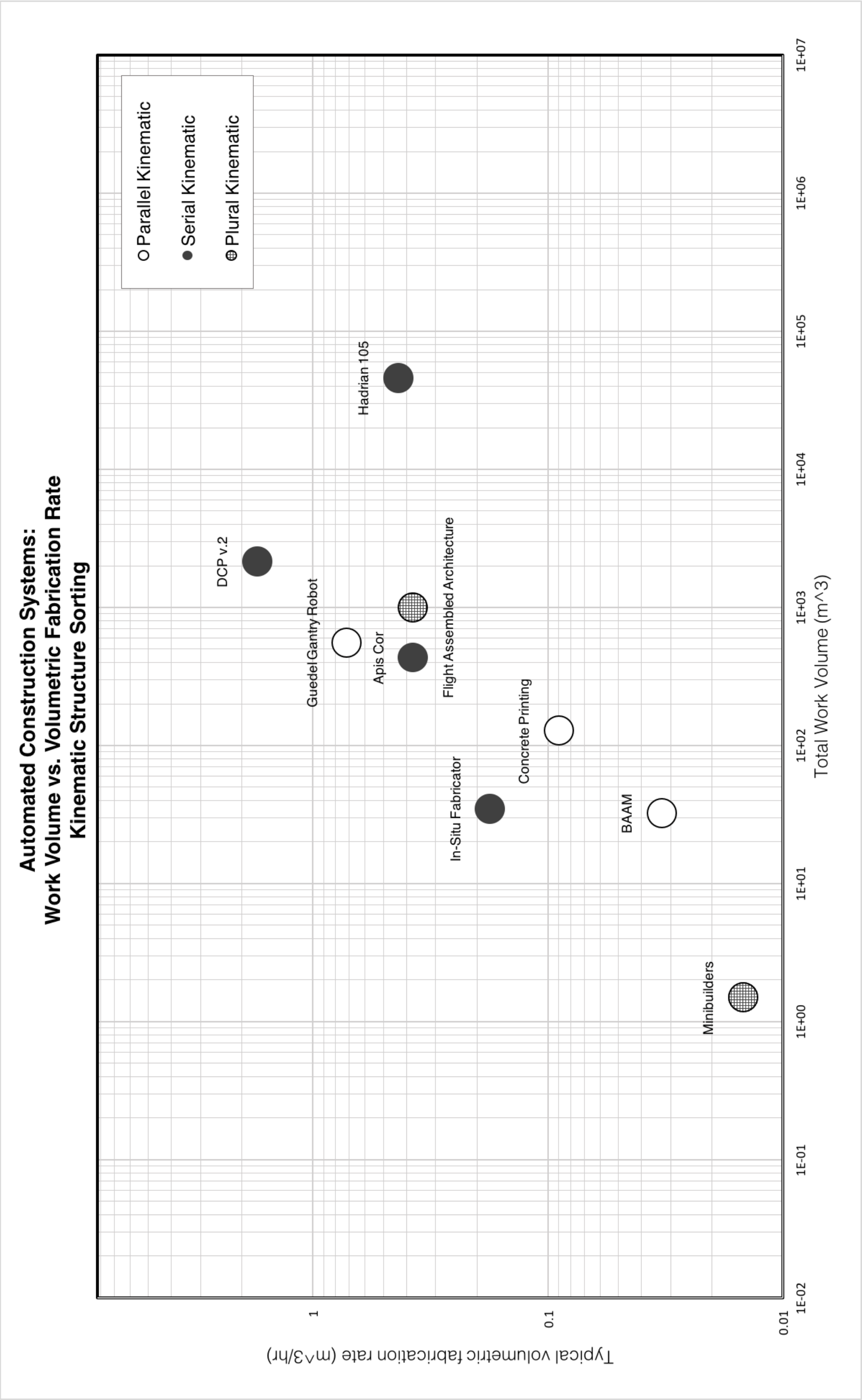


Figure 43: ACS Comparison Mapping by Kinematic Type – Including DCP

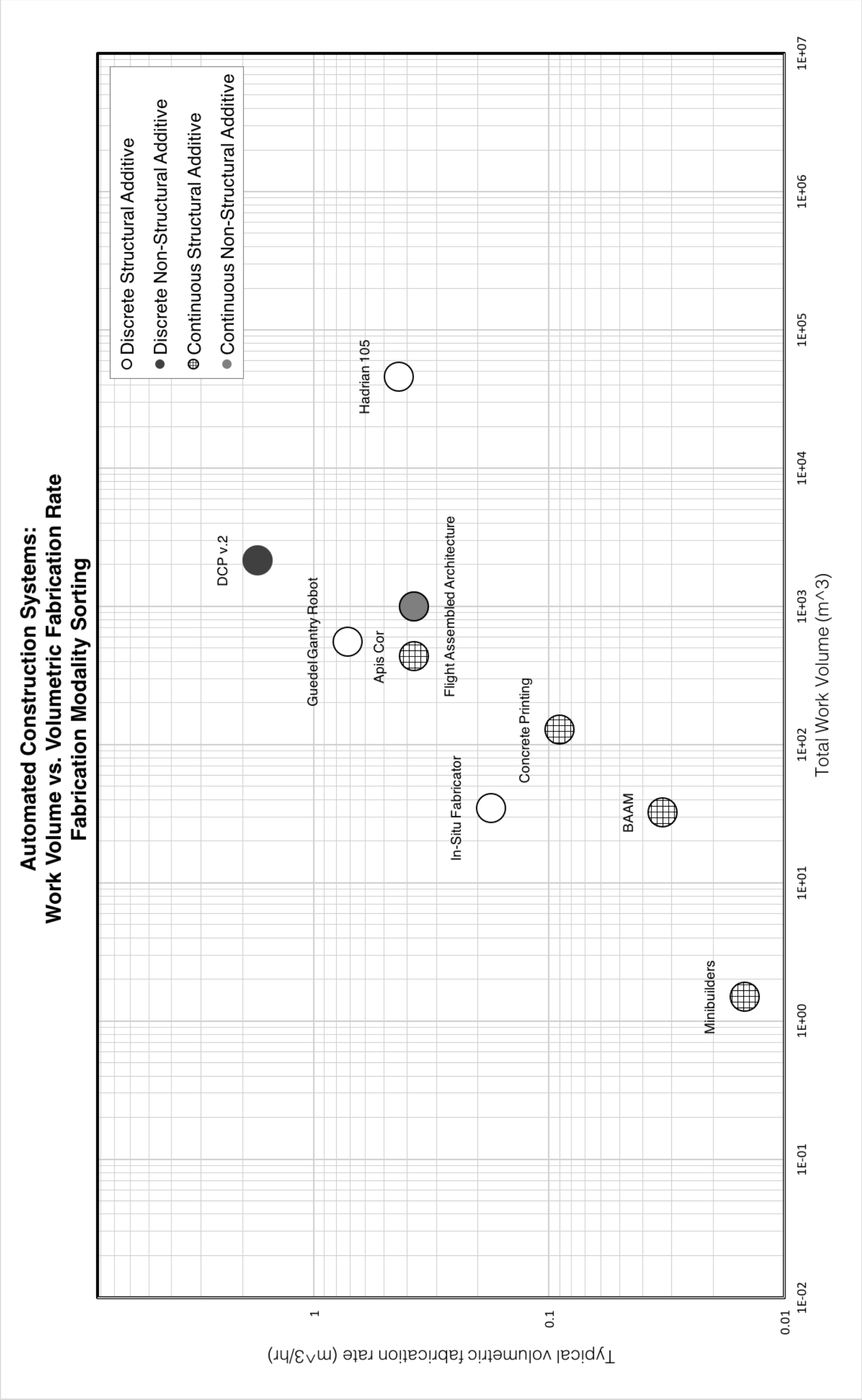


Figure 44: ACS Comparison Mapping by Fabrication Modality – Including DCP

As these plots show, the DCP v.2 system occupies an interesting position in this landscape of automated construction systems. First, the work volume of arm-based systems like the DCP generally exceed that of static gantry-based systems, and even the practical workspace of swarm-based systems. It is important of course to remember that a system's work volume is not necessarily immutable, and that it is relatively straightforward to expand the size of a system within reasonable limits – for example, by lengthening the axes of a gantry. However, especially for fixed, parallel-kinematic systems like gantries, these limits do exist. A gantry can only be made so large before it becomes logistically infeasible to move to a site or assemble while there – or even build in the first place. Particularly as arm-based systems become increasingly mobile – as the In-Situ Fabricator project has demonstrated – the work volumes of arm-based systems will continue to grow.

Furthermore, this mapping shows that the DCP's primary fabrication system – Print-In-Place Construction – substantially outperforms other fabrication techniques in terms of volumetric fabrication rate, largely thanks to the extremely high expansion rate of the PU foam. The volumetric fabrication rate measurement here only captures the time required to produce the polyurethane foam formworks, and excludes secondary operations such as pouring concrete/applying other structural materials; or external finishing. However, Print-in-Place's dramatic speed advantage over other automated construction techniques – more than an order of magnitude, in some cases – suggests that even with the additional time required to incorporate additional operations, the process will still perform substantially better. This validates the Print-In-Place concept, and strongly suggests that fabrication techniques which can leverage high-volume elements like spray polyurethane foam are well-adapted to automated construction tasks.

2.3.2 DCP v.2 Architecture Weaknesses

While the work described in the previous three sections validated the DCP concept in a number of important ways, it also exposed a number of major weaknesses in the DCP v.2 hardware and software architecture.

- Sensor Performance: First, the sensors used on the DCP v.2 system during this phase of development – Balluff magnetostrictive analog absolute position sensors on J2-J4, and a YUMO quadrature rotary encoder at the hydraulic motor on J1 – were extremely limited in their capabilities. As mentioned previously, the DCP v.2 produces substantial EMI during operation, from both the electrohydraulic pump assembly as well as the KUKA's drive cables, which introduced noise in the absolute position sensor signals and made their derivative too noisy to be meaningful. Consequently, position control performance with the analog sensors alone was challenging, and any sort of control on velocity was impossible.

  EMI also occasionally caused the J1 quadrature encoder to report substantial false increases in count, although the root cause of this unusual error was not found. The more substantial limitations of the J1 encoder were a) its inability to measure the actual output position of the joint (and the backlash in the joint) thanks to its connection at the motor; and b) its accompanying lack of a "home" reference position relative to the J1 axis.

- Control Rate & Jitter: Second, the MATLAB-based DCP system controller used in this phase is not a real-time application, and is fundamentally a poor choice for real-time control. As described in Section 2.1.2, the controller was generally limited to roughly a 50Hz sample rate, with high jitter and the tendency for sample rate to decrease over time. While this controller yielded impressive results given its limitations, it was not going to be adaptable to tasks requiring higher performance, such as active micro-manipulator position error compensation.

- System safety: Beyond the emergency stops available on the AT40GW itself and on the KUKA control pendant, no provision was made for safe operation of the DCP system. These emergency stops were located at the base of the DCP, requiring the user to be within the system's work volume and near the moving arm to actuate them. There were no means provided for connecting other external safety devices, such as light curtains or external emergency stops, and there was no means for the controller to trigger the emergency stops. This posed substantial hazard during operation, and made the system highly inappropriate for operation by a broader population of users.

- System modeling: The kinematic model of the complete DCP system used during this phase of development was incomplete, with only joints 1-4 captured in the DCP DH parameter set, and the joints and kinematic offsets of the bucket leveling bracket & KUKA neglected. This introduced substantial kinematic error, and made the system incapable of executing motion commands accurately.

- KUKA-AT40GW Integration: The methods developed to integrate the KUKA with the AT40GW were also unreliable or quite limited in their functionality. RSI communications provided were highly unreliable, thanks partially to the low frequency of the MATLAB system controller, but also the sensitivity of the C++ server to changes in configuration and operating load on the control computer. While direct sensor integration and external KRL program triggering were highly robust, the limited functionality they made available, and the time required to reconfigure these methods made them poor choices for a system designed to provide a platform for rapid experimentation.

- System adaptability: Finally, the DCP's control architecture was generally limited in its adaptability to new tasks. Because of the structure of the system's controller, implementing a new task to be executed in real-time required substantial, manual re-writing of the system's controller, and re-structuring the trajectory input format. Furthermore, since MATLAB is not generally designed for real-time control, the amount of support provided for interfacing with outside hardware – particularly at high rates – was limited. Again, this made the system poorly adapted as a platform for rapid experimentation.

In order for the DCP to meaningfully realize its potential as a platform for explorations in automated construction – capable of implementing a wide range of operations, and accessible to stakeholders from robotics, engineering, architecture, design and more – it was clear that substantial improvements would need to be made to address these shortcomings. This was the focus of work between 2016 and 2017, and is described in the third chapter of this thesis.

3 DCP v.2 System Refinement

This chapter describes the most recent phase of the Digital Construction Platform's development, where we attempt to transition the DCP v.2 system from "proof-of-concept" to "minimum-viable-prototype" stage, through improvements to the reliability, accessibility and adaptability of the platform.

This chapter opens by describing the high-level functional requirements that guided this stage of the DCP's development. We then describe the major hardware and software improvements made during this stage, including the transition to a hard-real-time control architecture, and the addition of new sensing capabilities to the DCP. The value of these improvements is examined in Section 3.3, through repetition of the characterization tests described in Chapter 2; and the creation of new large-scale light paintings that leverage the DCP's new capabilities. Finally, we discuss the many avenues for future work with the DCP v.2 system, including further development of the Print-in-Place process; exploration of more sophisticated control techniques, including input shaping and active micro-macro vibration control; and the need for a standard reference artifact for characterizing automated construction systems.

The work described in this chapter was conducted between August 2016 and August 2017, at MIT and at the Autodesk BUILD Space facility in Boston, MA. This phase of the DCP's development was led by the author, and conducted in collaboration with Selam Gano, John Zhang, Barrak Darweesh, Owen Trueblood, and Damien Martin.

3.1 Guiding Functional Requirements

At the end of the 2015-2016 development phase, the DCP v.2 system had provided an impressive proof-of-concept for both the DCP concept as well as the Print-in-Place fabrication process. However, the system also suffered from a number of substantial limitations (described in Section 2.3.2), which hampered its ability to serve as an open, adaptable platform for experimentation in automated construction.

The primary goal of the 2016-2017 development phase was to address these limitations, and to move the DCP system closer towards this goal of effectively serving as an effective experimental platform. Three high-level design parameters were established to guide this development:

- Reliability: The DCP v.2 system must operate reliably for a broad base of users to be able to comfortably interact with the system. This reliability parameter motivates a wide range of improvements to the system, including transition to a new control architecture to reduce control jitter; development of a robust RSI interface for the KUKA; addition of extra sensors to provide absolute position referencing for the entire AT40GW; and streamlining and standardization of the toolpath generation and trajectory calculation architecture for the DCP. Critically, this parameter also encompasses the development of a more complete safety architecture for the DCP v.2 system, including both expanded E-STOP functionality, and other safety interlocks such as enabling switches.

- Accessibility: The DCP v.2 system should be accessible to a wide range of users, with varying technical backgrounds – not just control engineers and roboticists, but architects, structural/civil engineers, designers, and other users in the AEC field. At the end of the 2015-2016 development phase, using the DCP to execute a simple task, such as creating a light painting from an STL file generated using the HomePrint slicer (described in Keating, [3]), required the creation of a new input parsing function in MATLAB to process data from HomePrint; implementation of a new variable set to control the light end-effector in the trajectory generation and control code; and manual tuning of control gains (and potentially controller architecture) to provide adequate performance. This degree of tuning and development was onerous for the DCP development team, and could provide an insurmountable barrier to adoption for less technical users. The primary goal of this parameter is to provide a complete, robust, basic toolchain for operating and controlling the DCP, so that users need only apply development effort to the specific problem they wish to investigate.

- Adaptability: Finally, to support the needs of the users the DCP v.2 system is intended to serve, the system needs to be easily and quickly adapted to a wide range of potential tasks. This could include the integration of additional sensor hardware or external subsystems; the development and implementation of new control architectures to address specific problems such as backlash in the J1 joint, or lateral oscillations in the AT40GW boom structure; the creation of trajectories for novel fabrication operations, such as assembly operations; or even major revisions to the DCP's operation method, such as through the creation of a ROS node for the DCP. Here, making adaptation as simple as possible, and facilitating the use of preexisting code and other resources, are of primary importance.

3.2 System Improvements

During the 2016-2017 development phase, improvements were made to the DCP system's hardware, mechatronic and control systems, and software architecture. Here, we describe the four most important:

1) The transition to the Simulink Desktop Real-Time control environment, and the corresponding re-development of the DCP's control architecture;
2) The creation of updated safety systems for the DCP;
3) The installation of new joint sensor hardware on the AT40GW to address some of the shortcomings of the system's existing sensor architecture, and the creation and tuning of controllers to take advantage of these new sensors;
4) The standardization and streamlining of the DCP's toolpath generation architecture, including the creation of a standardized import format for toolpaths generated by external programs.

3.2.1 Simulink Desktop Real-Time Control Architecture

The most substantial improvement made during the 2016-2017 development phase was the transition from the non-real-time, MATLAB function-based control architecture of the previous phase to a hard-real-time, Simulink-based control architecture. This improvement addressed all three of the guiding functional requirements listed in Section 3.1, improving system reliability by guaranteeing consistent real-time operation, and moreover by enabling high-performance real-time control of the KUKA to be established; accessibility, by simplifying (in most cases) complex programming tasks through leveraging the Simulink graphical programming interface; and most dramatically, adaptability, by enabling the entire DCP control architecture to be rapidly expanded and reconfigured to meet the needs of new tasks and experiments. Most importantly, transition to this new architecture has enabled all system components to be controlled simultaneously, in real-time, through a common architecture.

Simulink Desktop Real-Time

To implement this Simulink-based real-time control architecture, we used the Simulink Desktop Real-Time module [67]. Simulink Desktop Real-Time is an add-on to the existing Simulink dynamic systems modeling and simulation graphical programming environment that provides hardware-in-loop (HIL) real-time control functionality, similar to the capabilities offered by measurement automation and control packages like National Instruments' LabVIEW. While other Simulink packages exist that allow controllers developed in Simulink to be used on real-time systems, Desktop Real-Time is unique in that it does not require that code be compiled and transferred to a dedicated real-time controller. This is accomplished by installing a real-time kernel alongside the existing operating system on a conventional laptop or desktop PC, running either OS X or Windows, and then interfacing with physical hardware through a range of data acquisition interfaces. Desktop Real-Time offers up to 1 kHz control rates by default, and up to 20 kHz rates with additional packages installed.

Simulink Desktop Real-Time's primary advantages are that it enables use of existing Simulink functionality – simple graphical programming, rapid code re-use, and much of the extensive existing Simulink block library – to create real-time control systems. Particularly for the DCP v.2 project, it also simplified integration with our existing MATLAB-based toolpath and trajectory generation architecture, and made translating our MATLAB-based control architecture to Simulink much simpler. In many cases, MATLAB functions from the original dcpctrl_v1 library – such as functions for translating raw joint sensor readings to joint positions – could be directly reused in Simulink-based controllers, through programmatic function blocks like the MATLAB Function block. Desktop Real-Time is not without its limitations, however. There are a handful of Simulink block types and operations, such as the Simulink ROS toolkit, which are not supported under Desktop Real-Time. Furthermore, Desktop Real-Time's reliability is far from perfect. On a few occasions, kernel panics have occurred on the host PC during model execution, causing the controller to crash, and (in one notable case) the DCP to continue moving without feedback control. While these failures were caught by existing safety systems, the DCP controller is not at a point where it can be left to run without supervision: a more robust control interface, and better integration between the controller and safety systems, are needed before the DCP is ready to run in "lights-out" conditions.

dcpController

Many different controllers, examples, and test harnesses have been created in Simulink Desktop Real-Time during development 2016-2017. Out of these, the most fully-developed is the block diagram dcpController (Figure 45), a complete, basic control system for the entire DCP platform, including the AT40GW, KUKA, and external end-of-arm tooling.

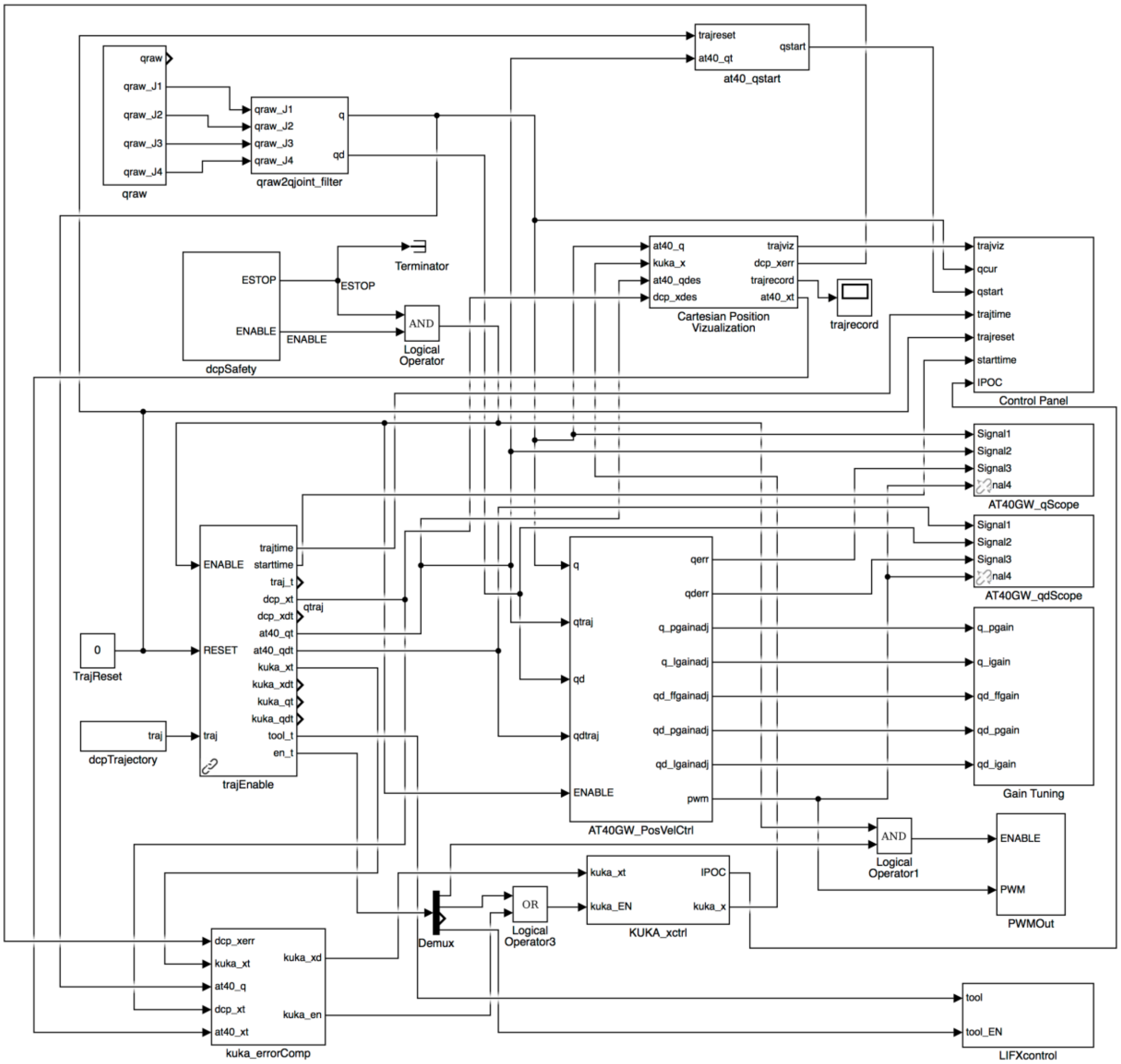


Figure 45: dcpController Block Diagram Overview

In the next section, we present each major component of the new Simulink-based control architecture using this block diagram as a reference point. As mentioned previously, a wide range of examples, test harnesses and tools beyond dcpController have been built using the core dcpctrl block library along with other Simulink blocks. dcpController, along with all of these auxiliary systems, may be seen in the dcpctrl_v2 Git repository, available at: https://github.com/mitmedialab/dcpctrl_v2.

KUKA Control & Feedback

Development of the dcpctrl_v2 system architecture in fact started with addressing the problem of how best to interface with the KUKA in real-time over RSI. As described in Chapter 2, implementing a robust means of controlling the KUKA had presented major problems during platform development 2015-2016. While Simulink Desktop Real-Time was already a candidate for DCP system control at the beginning of the 2016-2017 development period, it was critical to determine whether it could successfully be used to control the KUKA before it could be fully adopted.

There are a wide range of solutions developed previously for interfacing with KUKA robots, including the following:

- KUKA-KRL-Toolbox [68], developed by Maletzki et. al. at Wismar University. KUKA-KRL-Toolbox was one of the earliest known attempts at developing an open interface for KUKA controllers. It is reported by Chinello et. al. in [69] to use a serial interface to trigger specific functions within KRL programs. Unfortunately, the Toolbox is no longer available online, and we were not able to evaluate it.

- KUKA Control Toolbox (KCT) [69], developed by Chinello et. al. at the University of Siena. KCT improves upon KUKA-KRL-Toolbox by implementing control of the KUKA over RSI, using a C++ server that runs on the KUKA Controller. A series of MATLAB functions are then provided to enable communication with this server and control of the robot, including forward/inverse kinematics, motion control and basic trajectory generation functions. KCT is notable in that it does not require a hard-real-time kernel to successfully communicate with the C++ server. KCT is still publicly available, but has not been updated since 2013.

- The KUKA RSI Blockset for the QUARC real-time control software toolbox [70], developed by Quanser. This blockset provides a complete interface to a KUKA robot running on a KRC 2 controller, via the QUARC real-time control add-on for

MATLAB/Simulink. It provides the ability to send task-space or joint space position correction or velocity commands, and read a wide range of data from the KUKA, at 12 ms.

- KUKA|prc [71], developed by the Association for Robots in Architecture. KUKA|prc provides a toolset for programming KUKA robots through the Rhino/Grasshopper visual programming system. Notably, it is not designed for real-time control: instead, it performs definition and simulation of a robotic task, and then generates KRL code to be uploaded onto the KUKA for execution

- Finally, there are a variety of other open-source RSI servers similar to the one developed by Mediated Matter available, including KUKA RSI-3 Communicator by Eren Sezener [72]; Marionette [73], LightServer [74], and others by sjaakjules; and the KUKA Matlab Connector by Matthias Seehauser [75].

Many of these systems were evaluated during early development of the new control architecture for possible inclusion in a Desktop Real-Time-based controller. The QUARC KUKA RSI blockset was the most logical choice for this application, as it provides low-level control of the KUKA over RSI in a variety of different modes, and leverages the real-time control capabilities of QUARC. Unfortunately, the cost of this toolset was prohibitively high, particularly because the blockset had not yet been adapted for KRC 4 controllers and would require specialized development work from Quanser to implement. Other open-source RSI servers were also considered, although none were found to be substantially more robust than the existing Mediated Matter C++ server. Finally, the KUKA Control Toolbox provided a compelling option for controlling the KUKA, but it was unclear a) how much development work would be required to implement the KCT MATLAB functions in Simulink; and b) whether the KCT server program was compatible with the KRC 4 controller.

Thankfully, an alternative system was discovered: a simple Simulink Desktop Real-Time interface for KRC 2/RSI 2 developed by a researcher at the University of Emden/Leer, published on the German KUKA Roboter-Forum (Robot Forum) [76]. Using this interface as a starting point – and with assistance from the researcher – an updated version of the interface supporting KRC 4/RSI 3 was developed, along with accompanying RSI configuration files. This library, called kukaslxctrl, was released under the MIT License in February 2017, and is available at: https://github.com/mitmedialab/kukaslxctrl.

The kukaslxctrl library is rudimentary, but quite robust. Currently, it contains one primary block, called KUKASimulinkControl. This block, in combination with the RSI configuration files included in the library, provide the ability to command relative task-space position corrections – essentially, task space velocities. It is relatively straightforward to modify the library to command joint-space velocities, or provide absolute position commands, although this has not been implemented. The library is currently configured to return task-space position information, along with a range of diagnostic information about the state of the RSI connection and the internal state of the Simulink controller. It is configured to operate at the 4ms (250 Hz) RSI command rate. It has successfully run at this rate for extended durations (>1 hr), although this may be dependent on the capacity of the host PC. The most notable shortcoming of kukaslxctrl is that it is dependent on Simulink Desktop Real-Time, limiting its accessibility to users who do not have access to this toolbox. It also still requires that the user manually start/restart the KRL function that runs on the KUKA, and does not provide any error monitoring or torque/speed limiting. While this is acceptable for experimental applications, it is not sufficient for more industrial use.

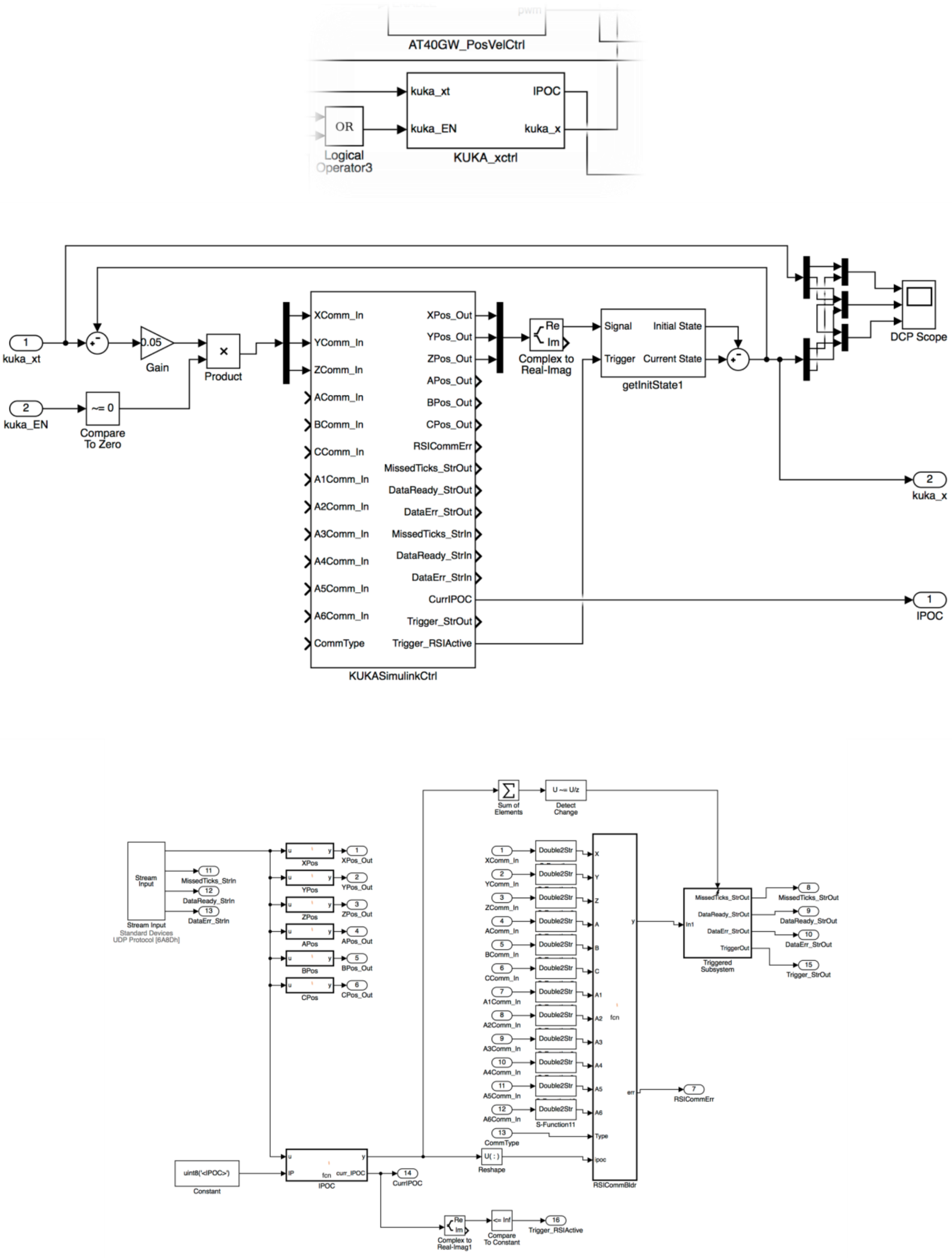


Figure 46: KUKA control system – kuka_xctrl and KUKA_SimulinkCtrl

In dcpController, the kukaslxctrl library is used to provide basic XYZ position control of the KUKA. A simple proportional control loop on endpoint position is used to generate velocity inputs for the KUKA, as shown in Figure 46. An ENABLE input is also provided to disable KUKA motion. Inside the KUKASimulinkCtrl block, these inputs are combined into an appropriately-formatted UDP packet, and transmitted whenever the block detects a change in the most recently read IPOC value (a timestamp value that the RSI protocol sends, and requires be returned in the subsequent UDP command packet). A combination of MATLAB Function blocks and MATLAB S-Function blocks are used to implement string operations, which Simulink does not support well natively.

Although relatively limited in its functionality, kukaslxctrl has more than met the needs of the DCP project, providing robust real-time control of the KUKA at a high control rate. It has already been used to implement a range of interesting control modalities, including reacting in real-time to AT40GW position to maintain constant endpoint location, and following trajectories calculated with the dcpctrl_v2 toolchain, as described in Section 3.3.2. It mitigates one of the major prior limitations of the DCP platform, and provides a valuable tool for future researchers interested in controlling KUKA robots from within Simulink.

AT40GW Control & Feedback

As described in Chapter 2, control of the AT40GW requires measurement of joint position and other state variables from various types of sensors; and creating PWM signals to control the hydraulic valve drivers. To provide these functions in the Desktop Real-Time environment, a specially designed I/O interface – the Humusoft MF-644 – was identified and implemented to replace the LabJack T7 used previously. Like the LabJack, the MF-644 offers a variety of input and output sources, including eight 14-bit ±10V analog inputs, eight 14-bit ±10V analog outputs, four quadrature inputs, and four internal timers that can be configured to produce PWM signals. It communicates with a user's PC through a Thunderbolt 2 interface, and like

Desktop Real-Time is compatible with both Macintosh[12] and Windows systems, making it easy for multiple users to work with the DCP simultaneously.

To carry the MF-644, protect accompanying power supplies and other electronics, and house interconnections to the DCP's various cables, a custom enclosure was fabricated. The control enclosure is designed to meet EIA rack-mount standards, enabling it to be installed in any standard 19" server rack enclosure. Internal power supplies provide both +12V and +5V DC power to support sensors and other system components. In addition to four analog sensors and three quadrature encoders, the MF-644 also monitors the enabling switch and E-STOP circuitry described below in Section 3.2.2. While all of the MF-644's timers are occupied for PWM signal generation, there are still a number of analog and digital I/O ports available for future expansion: the control enclosure facilitates this by providing removable front panels for rapid reconfiguration.

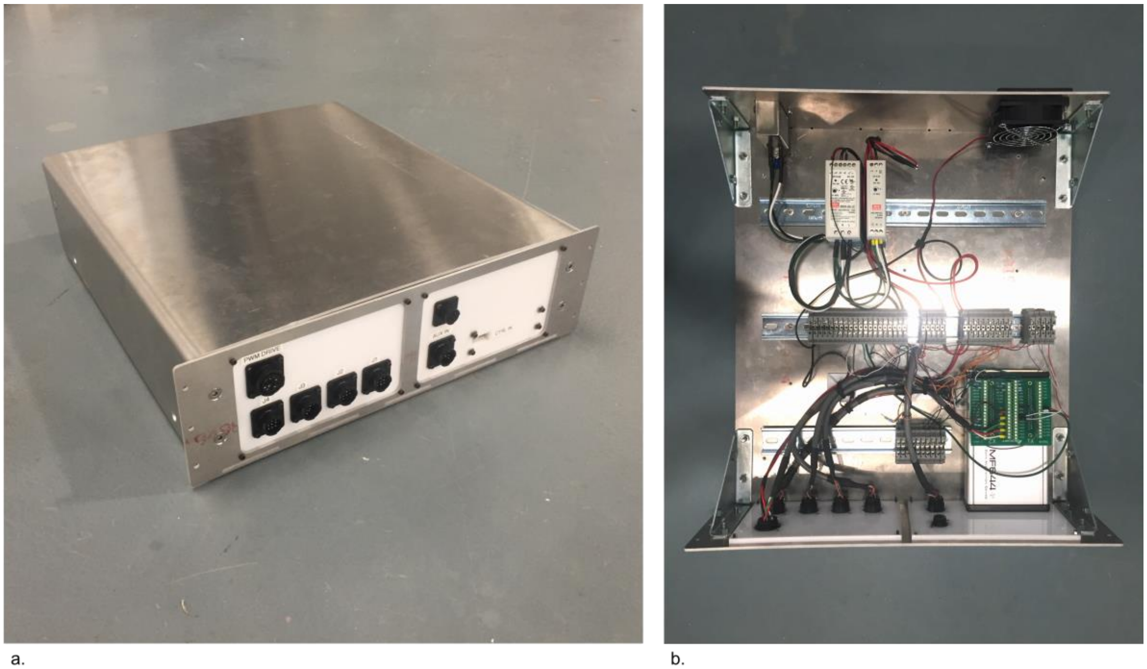


Figure 47: DCP control interface – cabinet (a) and interior wiring (b)

[12] Notably, the MF-644 has a bug when operating on OS X under MATLAB R2016a and b, where use of the frequency output block (for example, to generate a PWM signal) will cause a kernel panic. A patch is available from Humusoft, and also in the dcpctrl_v2 repository.

Communications between the DCP controller and the AT40GW are implemented through two primary blocks: the qraw block (Figure 48.a) and the PWMOut block (Figure 48.b).

- qraw: In the qraw block, joint sensor signals are brought in to the Simulink workspace using Analog Input and Encoder Input blocks, where they are multiplexed together on a per-joint basis. At this point, the signals are still in sensor space, as either analog voltages or quadrature encoder counts. Conversion to joint space and absolute referencing (where appropriate) is handled in the qraw2qjoint_filter block, described below in Control Implementation.

- PWMOut: The PWMOut block takes in PWM commands as percent duty cycles, and outputs these to the AT40GW using frequency output blocks. The PWMOut block also provides trajectory enable (play/pause) functionality, by reading the state of the system ENABLE signal. If the ENABLE signal is high, then the desired PWM duty cycles are commanded to the system; otherwise, a 50% duty cycle – corresponding to no motion – is commanded.

Actual control of the AT40GW is implemented using a variety of different controller blocks, such as AT40GW_PosVelCtrl. These are described further below, in Section 3.2.3.

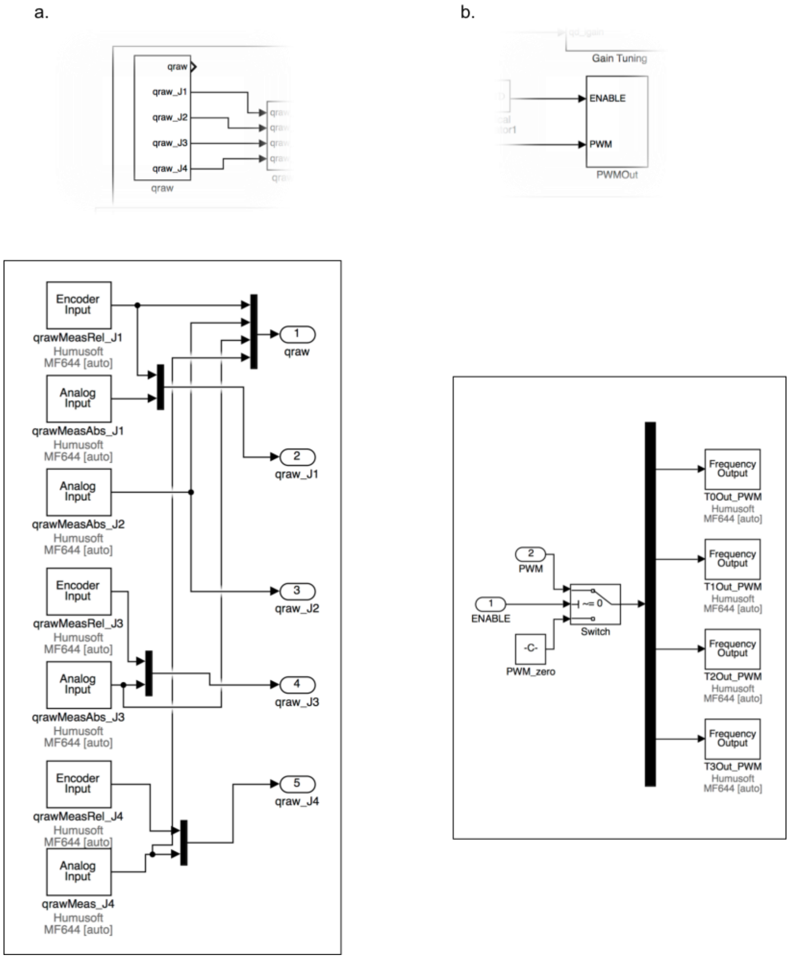


Figure 48: Communications blocks – qraw (a) and PWMMap (b)

The most significant challenge faced while developing the control interface for the AT40GW was management of signal noise. When the MF-644 was initially integrated with the AT40GW, measured sensor noise was substantially worse than had been observed with the LabJack T7: for example, the J4 joint would routinely measure spikes in measured position of more than 600 mm. This is believed to have been due to a few factors, including inappropriate grounding within the sensor system, the lower sampling rate of the T7, and also the fact that the T7 implements some low-pass pre-filtering at the ADC level [77]. Eventually, noise was largely mitigated through a combination of appropriate grounding (specifically, tying all sensor 0V references to the MF-644 analog ground; tying all power grounds and cable shields together; and joining the two ground sets at one location in a star-grounding configuration), along with tuning of the complementary filters described below in Section 3.2.3.

Tool Control

One of the major advantages of the dcpctrl_v2 architecture is the simplicity it affords to integrating new systems, such as additional sensing systems and tools. The integration of the "tool" used in the majority of experiments conducted during this phase of development – a LIFX Color 1000 Wi-Fi color LED bulb, used primarily for light painting – provides an instructive example of how simple this sort of development project can be[13].

The LIFX Color 1000 bulb is a Wi-Fi connected, "smart" multi-color LED bulb. It provides high-intensity light at up to 1055 lumens, without excessive power draw or heat generation, and can produce 16 million different colors [78]. It is controlled over Wi-Fi, through either a LAN or HTTP protocol. The LAN protocol is particularly well-suited for use with the DCP, as it uses UDP communications with a simple packet format that is well-documented by LIFX [79].

[13] The development of this interface was assigned to Selam Gano, a undergraduate researcher assisting the DCP project. While the author provided mentorship and technical advising, she led development and is the primary author of the lifxctrl library.

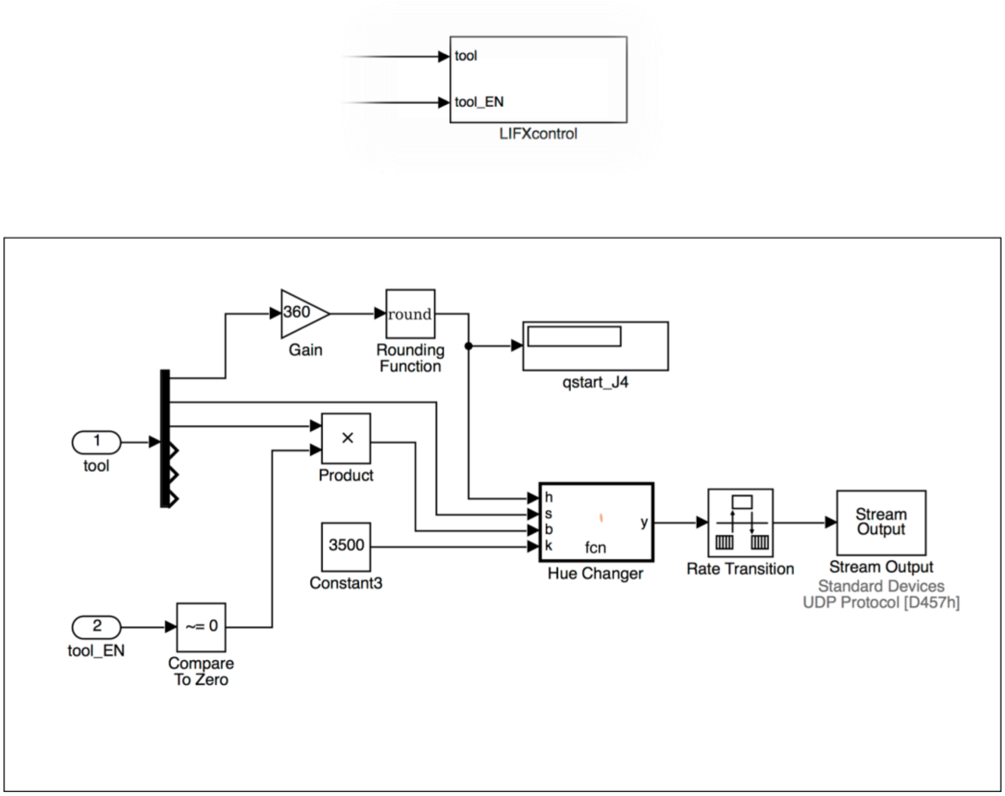


Figure 49: Tool control block - lifxctrl

Although the LIFX LAN protocol exposes a wide range of functionality from the LIFX bulbs, for use with the DCP, enabling real-time control of the color and brightness of the light were the only development objectives of concern. As shown in Figure 49, a Simulink MATLAB Function block takes in hue, saturation, brightness and temperature values, which in dcpController are provided in the system trajectory (see below for further description). These are then used to modify the example LIFX packet provided in the LIFX LAN Protocol documentation to produce the desired lightbulb behavior. The resulting packet is then sent to a Desktop Real-Time UDP Stream Output block, which broadcasts it to the lightbulb at 10 Hz. This block, along with example block diagrams and supporting code, form the lifxctrl library, and are available at https://github.com/mitmedialab/lifxctrl.

This implementation has proved quite robust, and is more than sufficiently responsive for light painting applications. By enabling continuous control of light color and brightness, it has enabled complex, striking light paintings to be created, as demonstrated in Section 3.3.1.

Moreover, it demonstrates how easily useful external systems can be integrated into the dcpctrl_v2 architecture, without requiring in-depth knowledge of Simulink, or even the dcpctrl architecture as a whole.

System Infrastructure

While the previous three sections have detailed the components of dcpController that provide critical functionality – control of the KUKA, AT40GW and tool – to the system, these components are actually a relatively small part of the infrastructure that makes up the DCP control system. There are a number of other important subsystems that enable control of the DCP, including the following:

- dcpSafety: dcpSafety monitors the state of the E-STOP and enabling switch circuitry, and feeds these signals to the rest of the control system. The E-STOP is largely not used in dcpController, except as a block on the ENABLE circuit (if the E-STOP is triggered, then the enabling switch is ignored). The ENABLE circuit is used for a wide variety of applications, however, including playing and pausing trajectory execution; enabling PWM signals to be commanded; and providing reset functionality to control loops (for instance, to reset integrators).

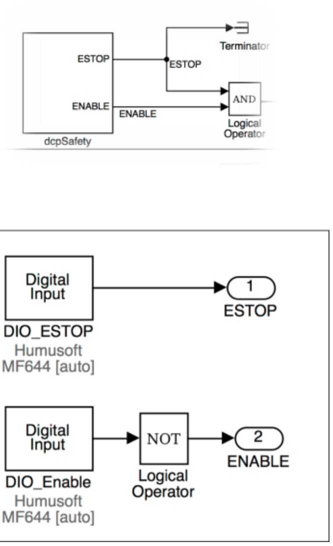


Figure 50: Safety management block - dcpSafety

- kuka_errorComp: The kuka_errorComp block enables real-time compensation for errors in the AT40GW's motion using the KUKA. Because of the AT40GW's kinematic arrangement, transforming coordinates between the AT40GW's base frame and the KUKA's internal reference frame is quite straightforward: the J5 leveling joint on the AT40GW ensures that the KUKA and AT40GW Z axes remain aligned at all times, so a simple 2D coordinate rotation by the current J1 rotation angle is sufficient to re-align the two reference frames. The kuka_errorComp block takes advantage of this, transforming the error between the AT40GW's measured endpoint position (which incorporates the offset provided by the KUKA when it is in its home position) and the desired endpoint position for the entire DCP system into the KUKA's coordinate system, and commanding the KUKA to compensate for this error, as shown in Figure 51. It is important to note that this error compensation only controls for errors that are measurable by the AT40GW's joint sensors: other errors, such as AT40GW link deflection/vibration, cannot be detected or compensated for. However, particularly when control gains for the AT40GW have been softened (typically to prevent jerking), this error compensation method may be able to provide some improvement in positioning and tracking performance. This function may be activated or deactivated during operation using a toggle switch on the Control Panel. The variant of the kuka_errorComp block shown here can also be used to manually adjust the KUKA's task-space position using the sliders; or perform compensation for differences between the AT40GW's commanded trajectory and the "true" DCP trajectory, as part of a toolpath segmentation technique described below in Section 3.3.2.

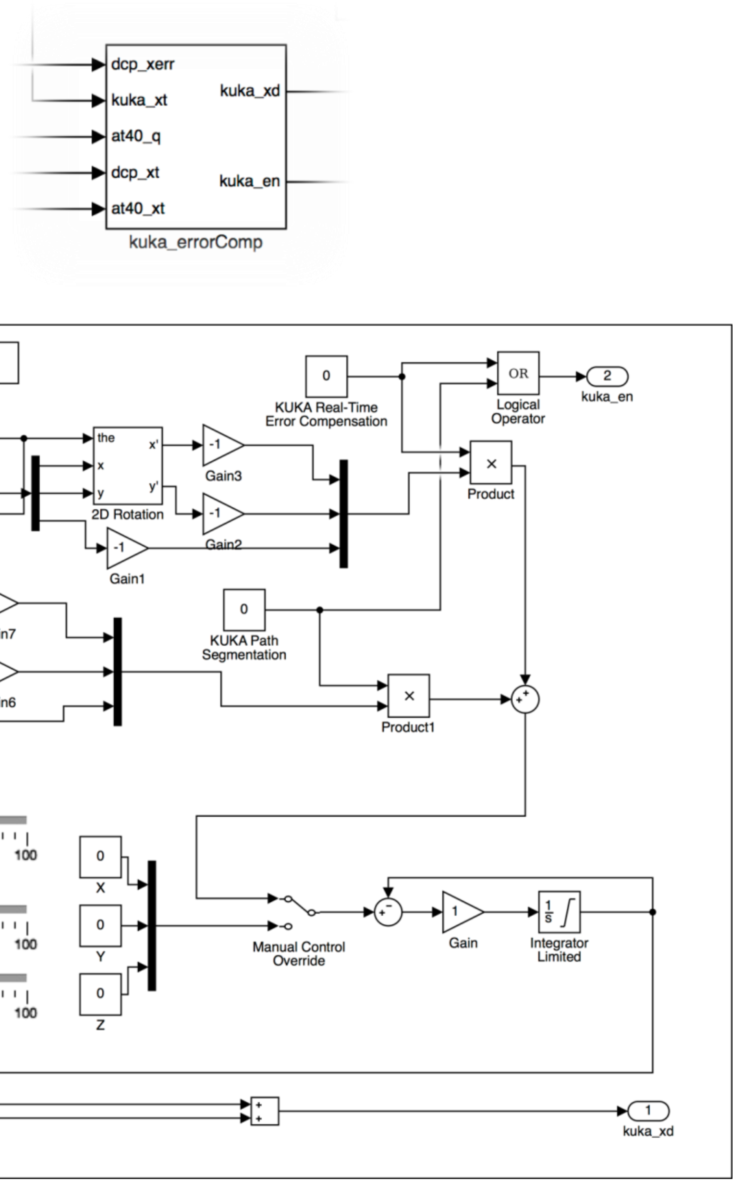


Figure 51: KUKA error compensation block - kuka_errorComp

- Trajectory Control: Trajectory control is arguably a major subsystem of the same importance as AT40GW or KUKA control, and is one of the primary points of improvement between the dcpctrl_v1 and dcpctrl_v2 architectures. In dcpController, trajectory control consists of two blocks: dcpTrajectory, and trajEnable.

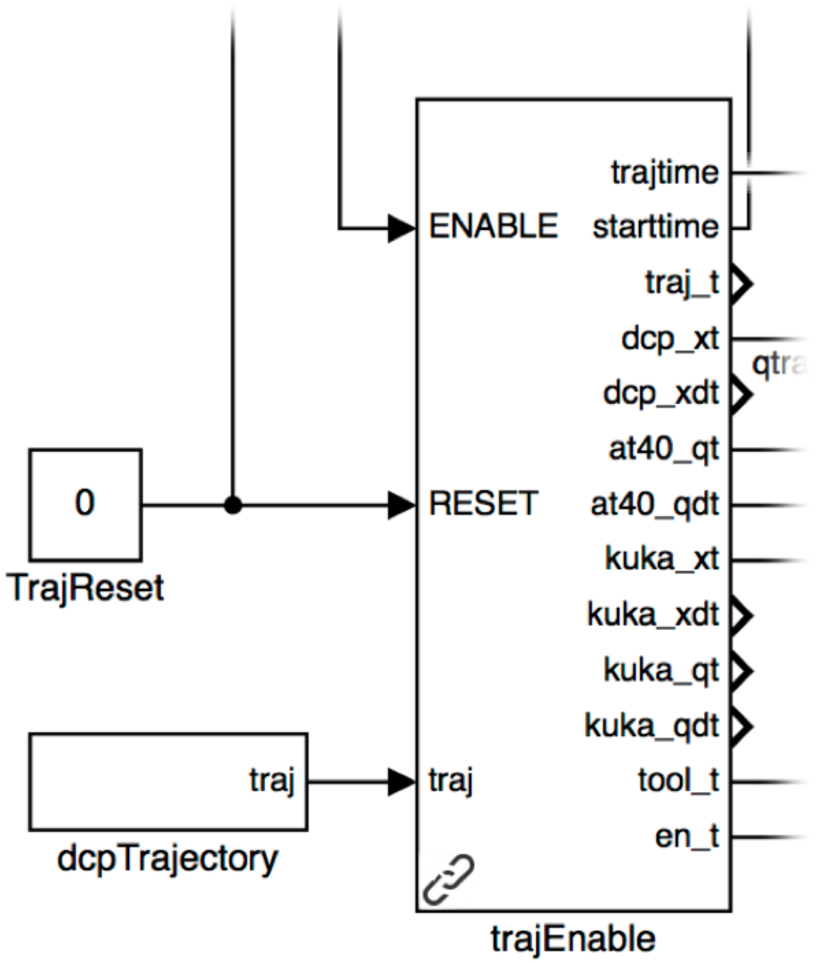


Figure 52: Trajectory control blocks - dcpTrajectory and trajEnable

As described below in Section 3.2.4, trajectories for the DCP are generated using a MATLAB-based toolchain, which relies on some of the same core functionality from the dcpctrl_v1 architecture. To bring these trajectories into Simulink for control, they are converted to MATLAB timeseries data format, which provides explicit timestamps to vector signals. One timeseries element is created per trajectory: for example, the DCP task-space trajectory (dcp_x), or the tool trajectory (tool). The dcpTrajectory block then accesses each of these timeseries elements, as shown in Figure 53.

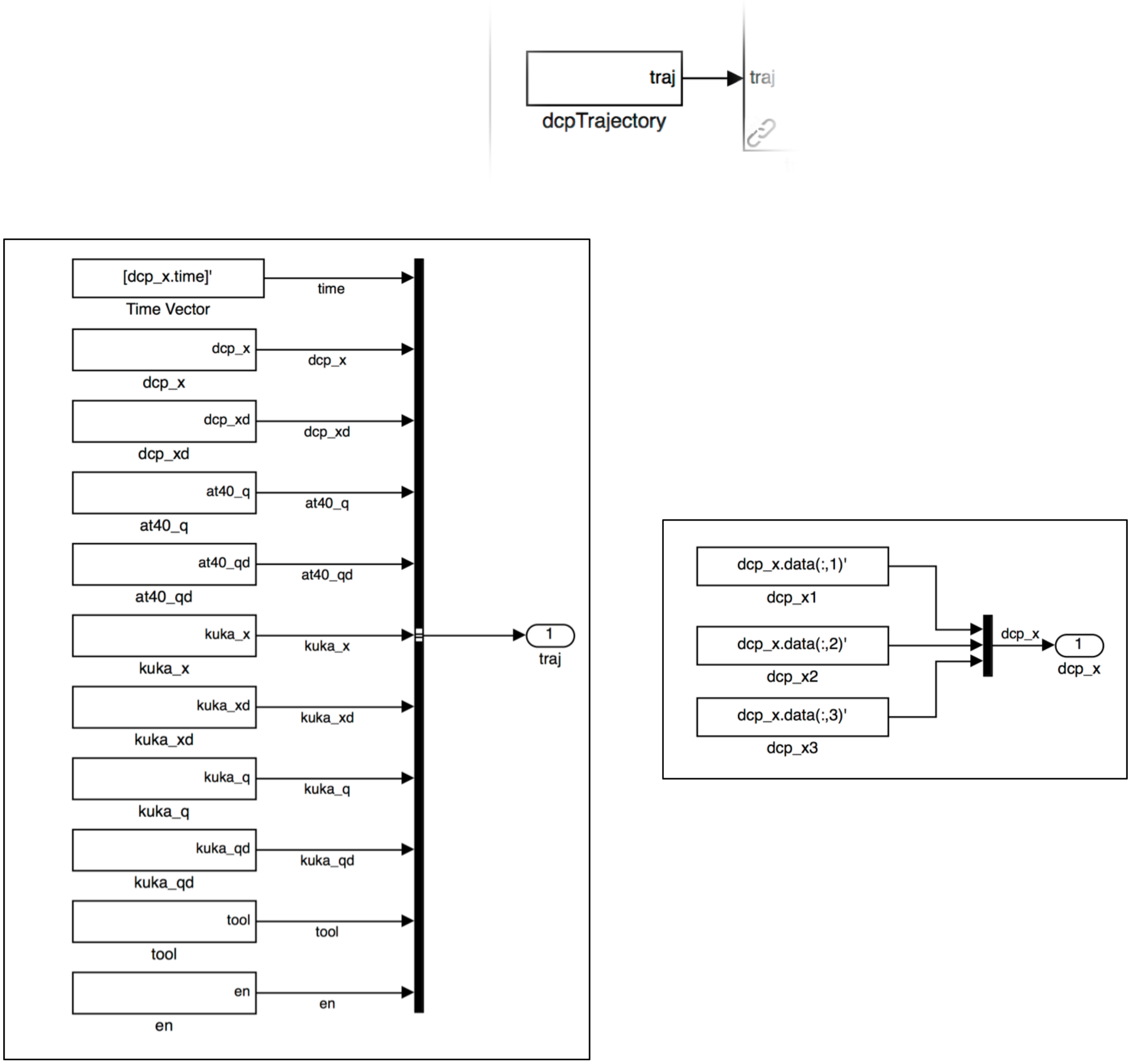


Figure 53: Trajectory import, detail view. dcpTrajectory block (left) and dcp_x channel block (right)

These signals are then combined into a single bus signal containing the entire system trajectory. From here, the signals are passed to the trajEnable block, which enables the trajectories to be played, paused and restarted at different positions. The trajEnable block is shown in Figure 54.

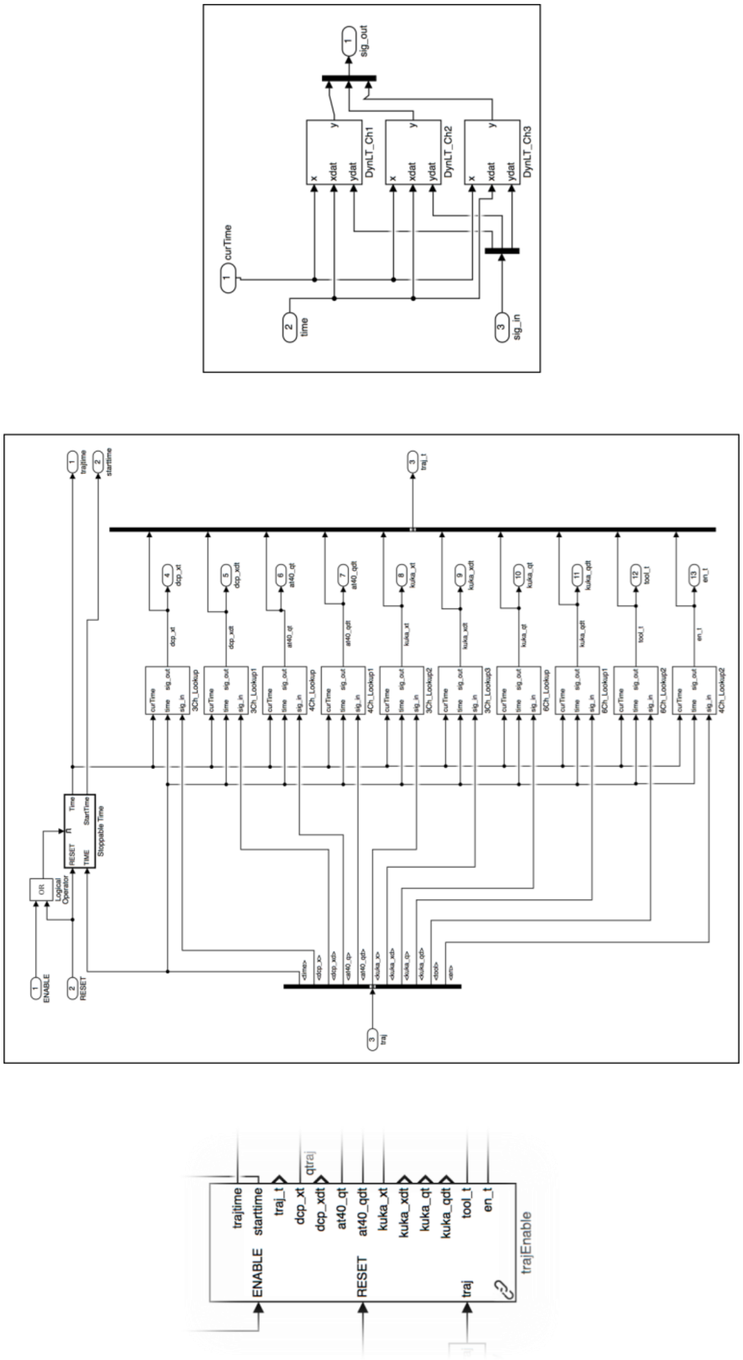


Figure 54: Trajectory control, detail view. trajEnable block (middle) and 3Ch_Lookup block (right)

One of the major challenges Desktop Real-Time/Simulink presents for combined command-control systems like dcpController is that time within the program is generally linear. Signals are referenced to the Simulink solver's internal time reference, which advances forward and cannot be changed (except by stopping the simulation). To get around this, trajEnable uses two components: a block called Stoppable Time, and a series of dynamic lookup table blocks. The Stoppable Time block uses a resettable integrator to provide a dynamically re-startable “time” signal. This time signal is then added to a time offset value, which can be tuned from the Control Panel. This enables users to “seek” through trajectories to their desired start point.

This dynamically adjustable time signal is then used to provide a reference input to a series of Dynamic Lookup Table blocks. These lookup table blocks each take a channel of the input trajectory timeseries, and use the time signal provided from the Stoppable Time block as an X reference to determine the correct value to output from that trajectory channel, for that instant in time. In addition to enabling trajectories to be paused and restarted at different points, these blocks also effectively increase the resolution of the input trajectory by performing linear interpolation. This enables trajectories with much lower time resolutions than are used in the controller (generally, 0.1 s timesteps as opposed to the 1 ms timestep used in the controller) without creating large step discontinuities in the input signal, reducing the size of trajectory signals.

- Control Panel: Finally, the Control Panel block provides a basic user interface for the DCP controller.

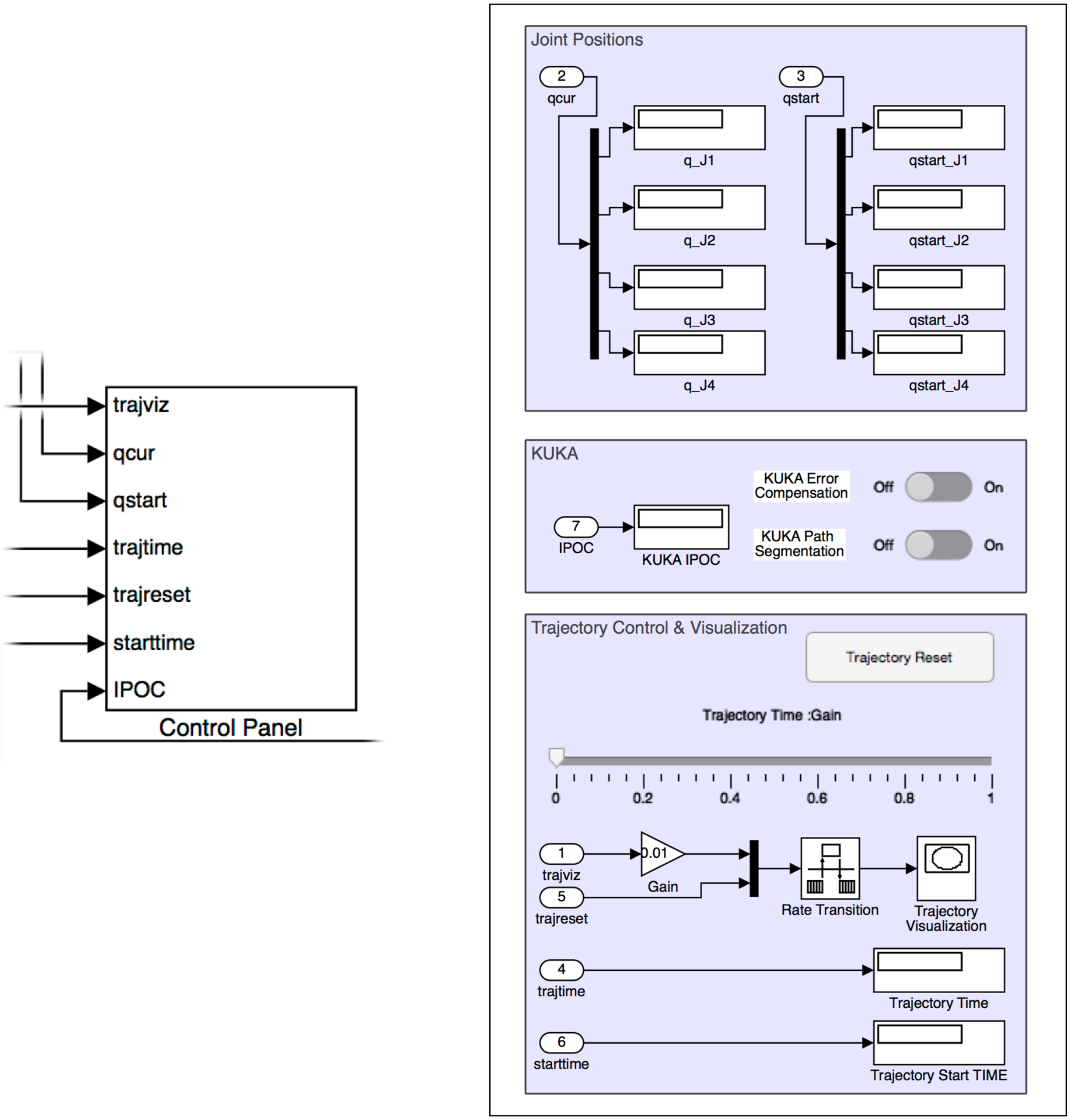


Figure 55: Run-time control panel block – Control Panel

When the controller is running, this interface reports current system joint position, as well as the joint position expected at the current location in the trajectory. This is provided so that users can start a trajectory away from the DCP's current location, and then manually move the system to the required starting pose. The control interface also provides a slider to jog through trajectories; a 3D plot showing the DCP's desired and actual trajectories (although this block's functionality is limited); and a trajectory reset button to clear previously recorded 3D trajectories and reset the Stoppable Time block integrator. Finally, the control interface also monitors the current IPOC value reported by the KUKA, and allows the user to enable or disable KUKA real-time error compensation.

While this section has provided a basic overview of the primary subsystems that make up dcpController, there are many other blocks and tools that have been developed in the course of this project. These can be seen in the dcpctrl Simulink block library, which is included as part of dcpctrl_v2.

3.2.2 Safety Systems

A second major priority in refining the DCP v.2 was the improvement of the system's safety architecture. Conventional industrial robots are subject to strict safety requirements, usually defined by national or international standards such as ISO 10218 [80]. These standards define important safety factors such as quantity, type and location of safety interlocks relative to the robot's work volume; best practices for system redundancy; acceptable reaction times between an emergency stop event triggering and the robotic system stopping; and acceptable operation modes for the robot. In the United States, the primary robotic safety standard is ANSI/RIA 16.06[14] [81]. This standard is targeted at – and widely adopted in – traditional industrial applications. It has also been adopted at the Autodesk BUILD Space as a reference for developing internal best practices for robot safety. The standard is not written to support "reconfigurable robotic" implementations, where a robotic system may be used for a wide range of tasks: for example, the projects that the BUILD Space robotic arms are used for, or the tasks the DCP v.2 is intended to perform. However, it provides a starting point for creating safe reconfigurable robotic implementations.

[14] This standard simply adopts ISO 10218.

At the end of the 2015-2016 development phase, the safety interlocks onboard the DCP consisted of the following:

- KUKA: Emergency stop button on the KUKA pendant controller.

- AT40GW: Emergency stop buttons located on a) the electrohydraulic drive battery pack, which disables the electrohydraulic drive and separates the battery pack cells; b) the PWM control interface, which disables PWM input; c) the AT40GW main operation panel, which disables the diesel engine; and d) the AT40GW track drive pendant, which disables all hydraulic functionality. All of these E-STOP buttons were located at or near the base of the AT40GW, within the DCP's work volume. A brief attempt was made during 2015-2016 at integrating a tethered E-STOP button into the AT40GW E-STOP circuitry. However, the tethered E-STOP reportedly caused intermittent engine shut-offs (believed to be related to EMI), and was abandoned.

There was no connection provided between the KUKA and AT40GW E-STOP circuits. In addition to these limited systems, there was no interface provided for in-process system safety control (such as through an enabling switch) or the incorporation of other emergency stop triggering devices.

To address these concerns, the modular safety interface shown in Figure 56 was developed, in collaboration with Autodesk BUILD Space staff.

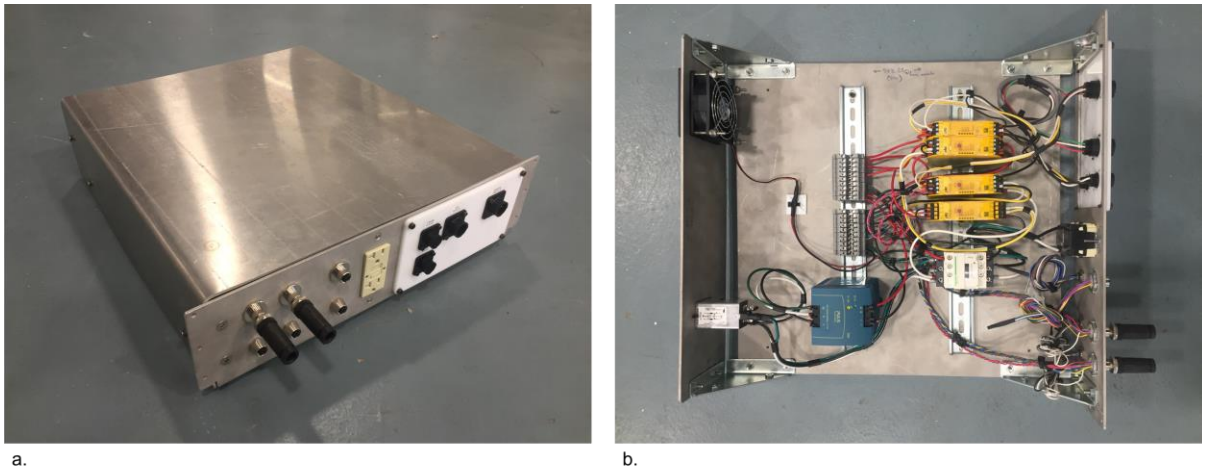


Figure 56: DCP safety interface – cabinet (a) and interior wiring (b)

This interface was developed to service the needs of both the DCP as well as the BUILD Space robotics systems, and was designed with adaptability first in mind. It provides hard-wired monitoring of up to two separate PILZ light curtain circuits, and can provide E-STOP triggering to both the ABB industrial robots used at BUILD, as well as a 120V 20A 1PH power circuit built into the interface. The intention of this is to enable BUILD Space robotic systems – particularly the small, mobile-bench mounted ABB IRB 140 robots – to be rapidly set up for safe operation inside safeguarded zones using light curtains, with E-STOP functionality applied to user-developed end-of-arm tooling through the E-STOP controlled power circuit, without any modification of the interface required. Finally, like the controller interface, an EIA-compliant enclosure was fabricated for the safety interface as well, as shown in Figure 56.

To enable additional E-STOP sources to be integrated, or new external systems to be triggered, the interface uses expandable PILZ PNOZ S4-series safety relay modules. As implemented, the interface provides up to 3 E-STOP source inputs (two light curtains, and one E-STOP button), and can control up to 6 circuits[15]. A user-modifiable plastic front panel for the interface is also provided to allow additional external connections.

[15] It is important to note that 8 separate emergency stops are unlikely to be controllable. Some of the available circuits are normally closed, which is often not appropriate for controlling emergency stop equipment. Additionally, industrial emergency stops usually require two separate circuits to function.

For integration with the DCP, the following systems were added to the safety interface:

- Tethered emergency stop button: An industrial control pendant with an E-STOP button and enabling switch was integrated into the system. The pendant was placed on a 15 m tether, to enable the operator to carry the pendant outside of the DCP's range of motion. The E-STOP button was added as an emergency stop input source to the safety interface, and was placed in series with the light curtain monitoring relays as shown in Figure 57 below.

- Enabling switch: Most industrial robotic systems can be controlled in a range of control modes, which provide different balances between robotic performance (maximum allowable speed and torque) and safety restrictions (required safety interlocks, safeguarded space dimensions, etc.). These include manual modes, which allow the robot to be operated manually and with reduced safety interlocks. These modes are typically used when configuring a robotic system, or experimenting with a new process.

  Critical to these manual modes is the incorporation of a three-position enabling device (historically known as a deadman's switch), defined in ANSI/RIA 15.06 Section 5.8. To support this component of the standard, the three-position switch included in the E-STOP/enable pendant was connected through the safety interface directly to the DCP control interface described in Section 3.2.1, where it is monitored by the Simulink controller and used to activate toolpath execution.

- Connection for KUKA and AT40GW battery E-STOP circuits: Finally, the E-STOP circuits for both the KUKA and the AT40GW's electrohydraulic drive battery pack were connected to the safety interface. The KUKA E-STOP circuit was integrated through the X11 interface on the KUKA control cabinet. The AT40GW E-STOP circuit was integrated by placing a relay circuit in the safety interface in series with the existing E-STOP button on the AT40GW battery pack. Notably, this circuit only uses one wire. This is not typical for emergency stop systems in industrial systems, and is considered insufficiently robust to shorting/welding failures. Additionally, this circuit only disables the electrohydraulic drive: it cannot stop the AT40GW when operating using the diesel engine.

The wiring diagram used to support integration with the DCP may be seen in Figure 57.

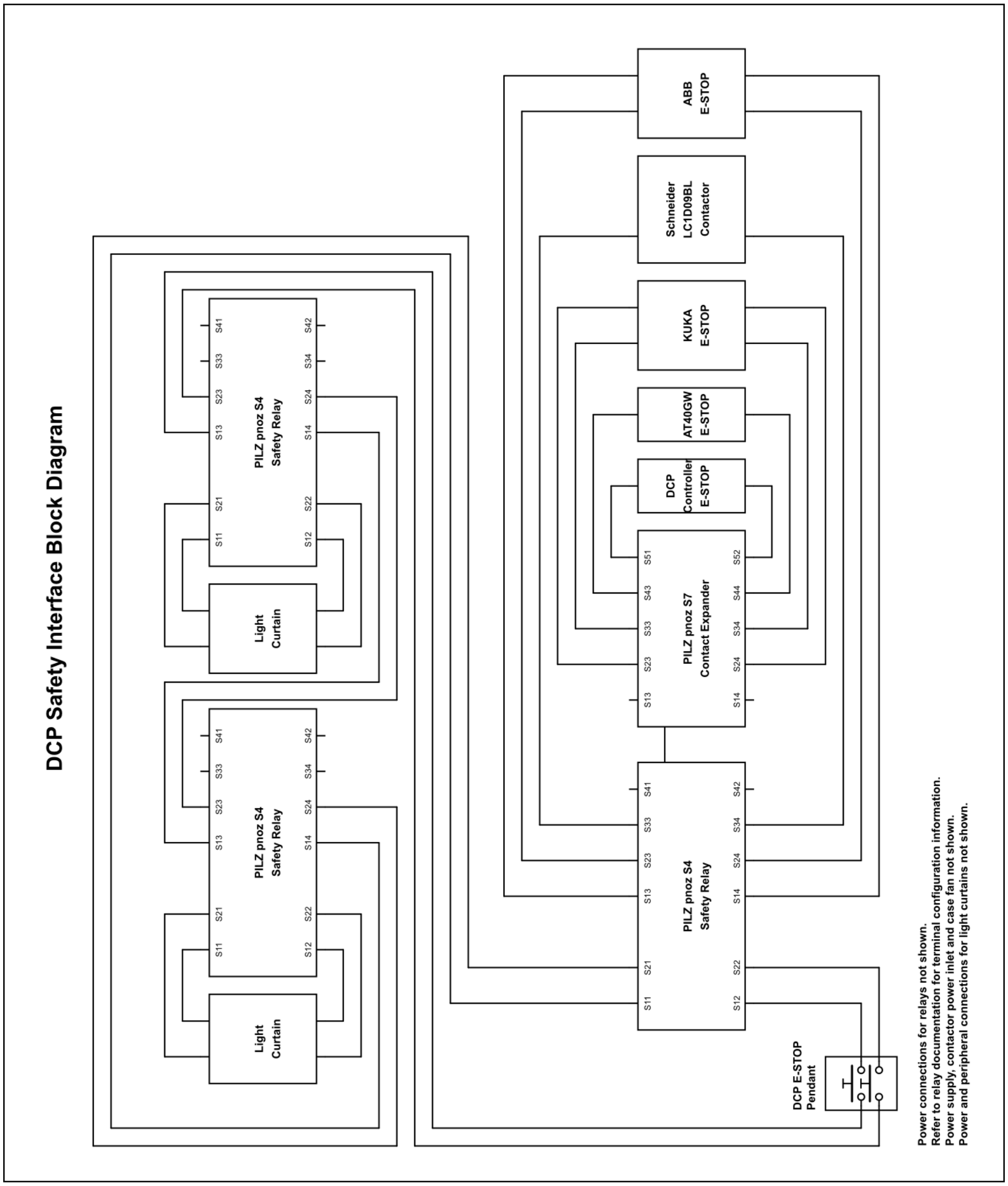

DCP Safety Interface Block Diagram
ABB E-STOP
Schneider LC1D09BL Contactor
KUKA E-STOP
AT40GW E-STOP
DCP Controller E-STOP
PILZ pnoz S7 Contact Expander
S51
S43
S33
S23
S13
S52
S44
S34
S24
S14
PILZ pnoz S4 Safety Relay
S41
S33
S23
S13
S21
S11
S42
S34
S24
S14
S22
S12
Light Curtain
DCP E-STOP Pendant
Power connections for relays not shown.
Refer to relay documentation for terminal configuration information.
Power supply, contactor power inlet and case fan not shown.
Power and peripheral connections for light curtains not shown.

Figure 57: DCP safety interface wiring block diagram

It is critical to note that this interface is not certified compliant with ANSI/RIA 15.06. The configuration of emergency stop sources (e.g. the daisy-chaining of light curtain and manual E-STOP sources) is believed to be compliant, but this has not been confirmed with a certified professional. Furthermore, there are a range of standards defining the performance characteristics of an emergency stop circuit – such as emergency stop type, stop triggering time – which are not observed here. However, the interface provides substantial improvements over pre-existing safety systems on the DCP; opens up the possibility of integrating additional safety systems in the future, such as LIDAR proximity sensors or other access control interlocks; and most importantly, provides a robust, easily expanded starting point for safety systems in future reconfigurable robotics projects.

3.2.3 Sensing & Control Improvements

The second major area of improvement in development 2016-2017 was to the sensing systems installed on the AT40GW, and to the accompanying control systems implemented using the dcpctrl_v2 architecture.

Sensor Systems Review

As described in Chapter 2, the AT40GW sensor architecture used in development 2015-2016 suffered from a number of significant limitations, including high noise and accompanying non-differentiability of sensor signals; EMI sensitivity; and, for the J1 joint, lack of a persistent absolute reference, and no measurement of backlash in the J1 transmission. While they were sufficiently functional for the experiments conducted during this period, they limited the performance and usability of the DCP.

Consequently, a new sensor architecture was developed to mitigate this, using the following design parameters:

- Integrate with existing sensors: The new sensor architecture should use the existing sensors on the AT40GW if at all possible, and integrate into existing toolpath creation and trajectory generation routines as seamlessly as possible.

- Clean velocity signal: The new sensors should be able to provide some sort of velocity signal; either directly or through differentiation.

- J1 – Absolute Measurement at Output: The new J1 sensor harness should measure at the output of the J1 joint (rather than the input), and should provide an absolute-referenced position measurement.

- J3 – Measurement at Joint Axis: As mentioned in Chapter 2, the configuration of the J3 joint and sensor in the AT40GW produced a nonlinear relationship between sensor readout and joint angular position/velocity. The new J3 sensor harness should measure directly at the J3 joint to mitigate this.

- Velocity resolution: To provide a performance metric to guide component selection and design across joints, a task-space *velocity resolution* – the smallest Cartesian velocity a sensor can resolve without additional filtering – was specified. Based largely on the performance of the existing J1 encoder (described further below), the minimum velocity resolution was defined to be 60 mm/s in task-space, when the J4 joint is fully extended.[16]

Electromechanical System Design

With these design parameters in mind, new sensor harnesses were developed for all three joints. Harnesses for Joints 3 and 4 were developed by the author. The harness used in Joint 1

[16] It is important to recognize that this specification does not actually define joint sensor performance in appropriate units. For example, the J3 joint sensor needs to be able to resolve better than 0.0085 rad/s angular velocities in order to resolve 60 mm/s Cartesian velocities at the AT40GW endpoint when the boom (~8 m long) is fully extended. Although coarse, the specification was deliberately defined in this way to focus on the quantity of interest, task-space velocity control.

was developed collaboratively by the author and John Zhang, an undergraduate researcher with the DCP project. Damien Martin, another undergraduate researcher on the project, also provided assistance in sourcing the absolute sensor used in the J1 joint. Each harness is described briefly below.

J4 Sensor Harness: Draw-Wire Quadrature Encoder

The existing J4 sensor harness on the DCP v.2 consisted of a 3350 mm Balluff BTL-6 magnetostrictive absolute position sensor, with a ±10V analog output. This sensor provided an absolute position reference for the axis, but was highly susceptible to noise. The sensor consequently could not provide a usable velocity signal, and because of the sensor's substantial length, noise spikes translated to large errors in reported position.

To augment this, a Waycon SX120-4000-6.3-L-SR12 draw-wire quadrature encoder was selected. This encoder provides 6.3 pulse/mm resolution (increased to 25.2 pulse/mm at the DCP control interface through quadruple edge detection), and at the DCP controller's typical control frequency of 1 kHz, can resolve a minimum velocity of >40 mm/s. A bracket was developed to mount the new encoder coaxially with the existing J4 encoder system, as shown in Figure 58:

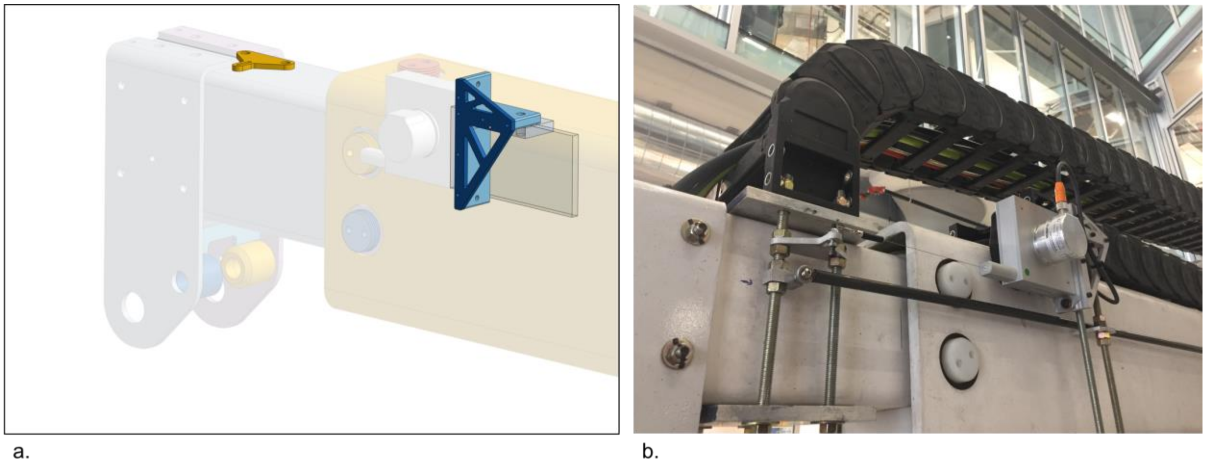


Figure 58: J4 sensor harness – CAD model (a) and installed harness (b)

J3 Sensor Harness: Quadrature Encoder with Timing Belt Transmission

The J3 joint presented a substantially more interesting design problem. The existing J3 sensor harness – a Balluff magnetostrictive sensor, mounted coaxially with the J3 hydraulic cylinder – suffered from the same limitations as the J4 sensor harness, and additionally the nonlinearity issues described above and in Chapter 2. While the existing J3 sensor was still useful for providing a position signal and an absolute position reference, the new sensor harness would ideally need to measure at the axis of joint rotation to simplify joint velocity measurement. At this location, the sensor would need to be able to resolve better than 0.0085 rad/s angular velocities. We selected a YUMO 1024 pulse/rev encoder (identical to the J1 motor encoder), and determined that for this encoder at our default 1 kHz control frequency, we would need to provide a 180:1 reduction between the joint and the encoder.

However, implementing this sensor harness directly at the J3 joint was not trivial. As shown in Figure 59, the only cylindrical component that rotates about the J3 axis are the J3 joint hubs (highlighted). This hub cannot be accessed without detaching the AT40GW boom completely from the rest of the DCP. Finally, because of the high loads carried in the AT40GW's structure, we did not want to add mounting holes to any part of the AT40GW frame, for fear of creating stress concentrations.

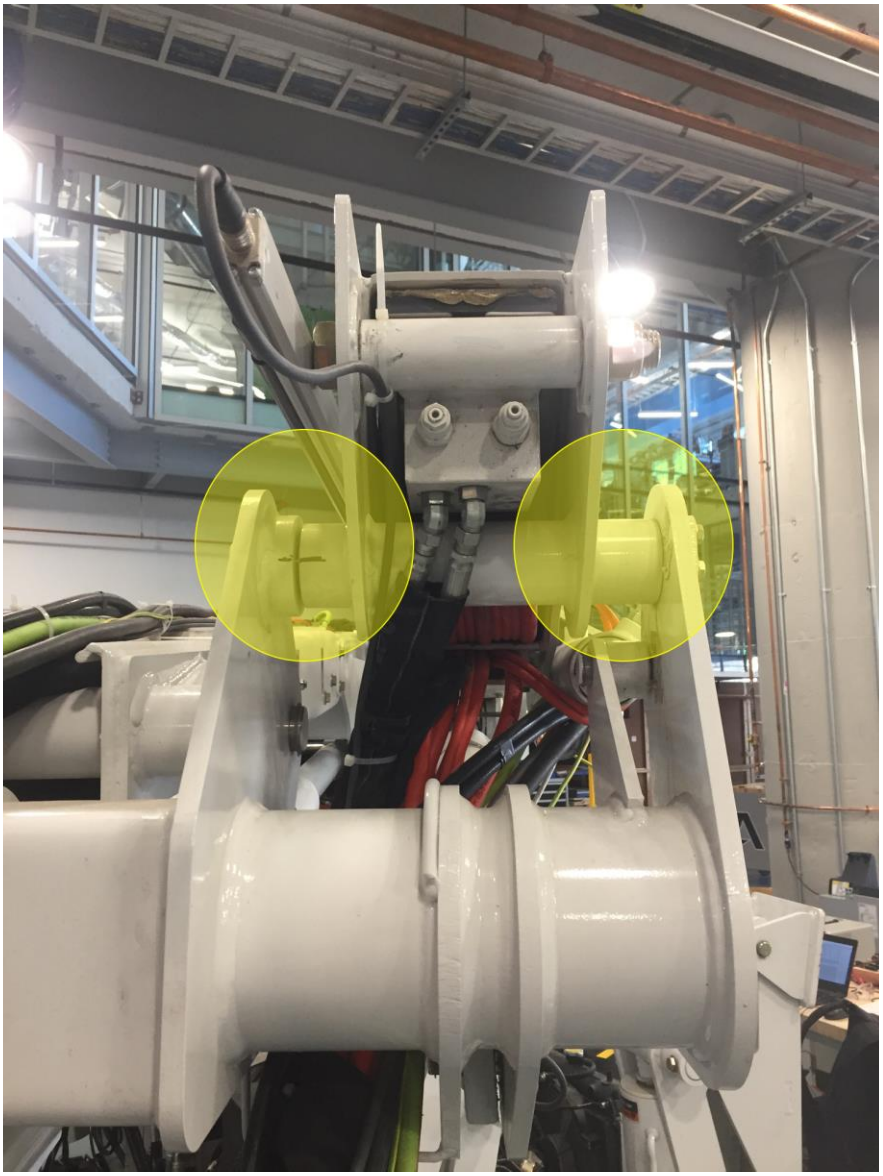


Figure 59: J3 joint, before sensor harness installation

To address these limitations, a design based around a GT2-profile timing belt pulley transmission with a custom-machined, split output pulley was developed. Timing belts can provide high-precision motion control and registration if correctly designed. They also are relatively tolerant of angular misalignment between input and output, and can accept slight imperfections in tooth form. Finally, if only a limited range of motion is required (as is the case with the DCP J1 and J3 axes), clamps like that shown in Figure 60 can be used to attach the ends of a split timing belt, enabling the belt to be installed around an axle that cannot be disassembled.

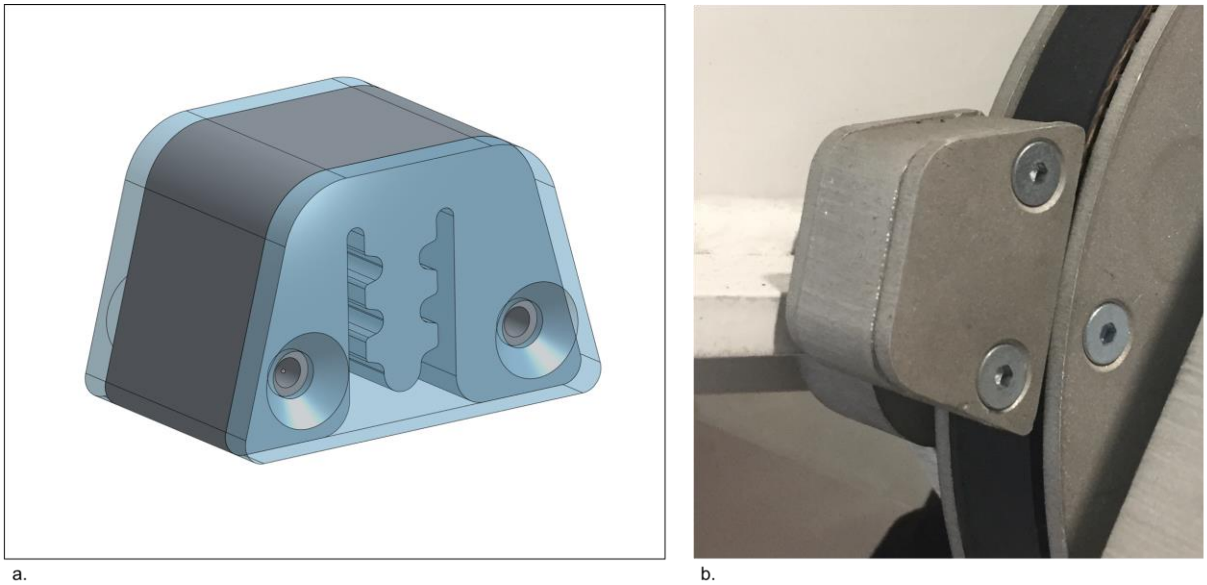


Figure 60: Timing belt clamp – CAD model (a) and installed on J3 joint (b)

A fixture was designed to clamp to the AT40GW J3 joint hub without requiring modification of the AT40GW boom structure. An adjustment plate, designed to allow positional adjustment perpendicular to the J3 axis and orientational adjustment in all three directions, connected the clamp fixture to the split timing belt pulley. To make the timing belt pulley, a two-part square blank was machined and assembled. The pulley outer profile, inner diameter, and mounting features were then cut from the assembled blank using a waterjet.

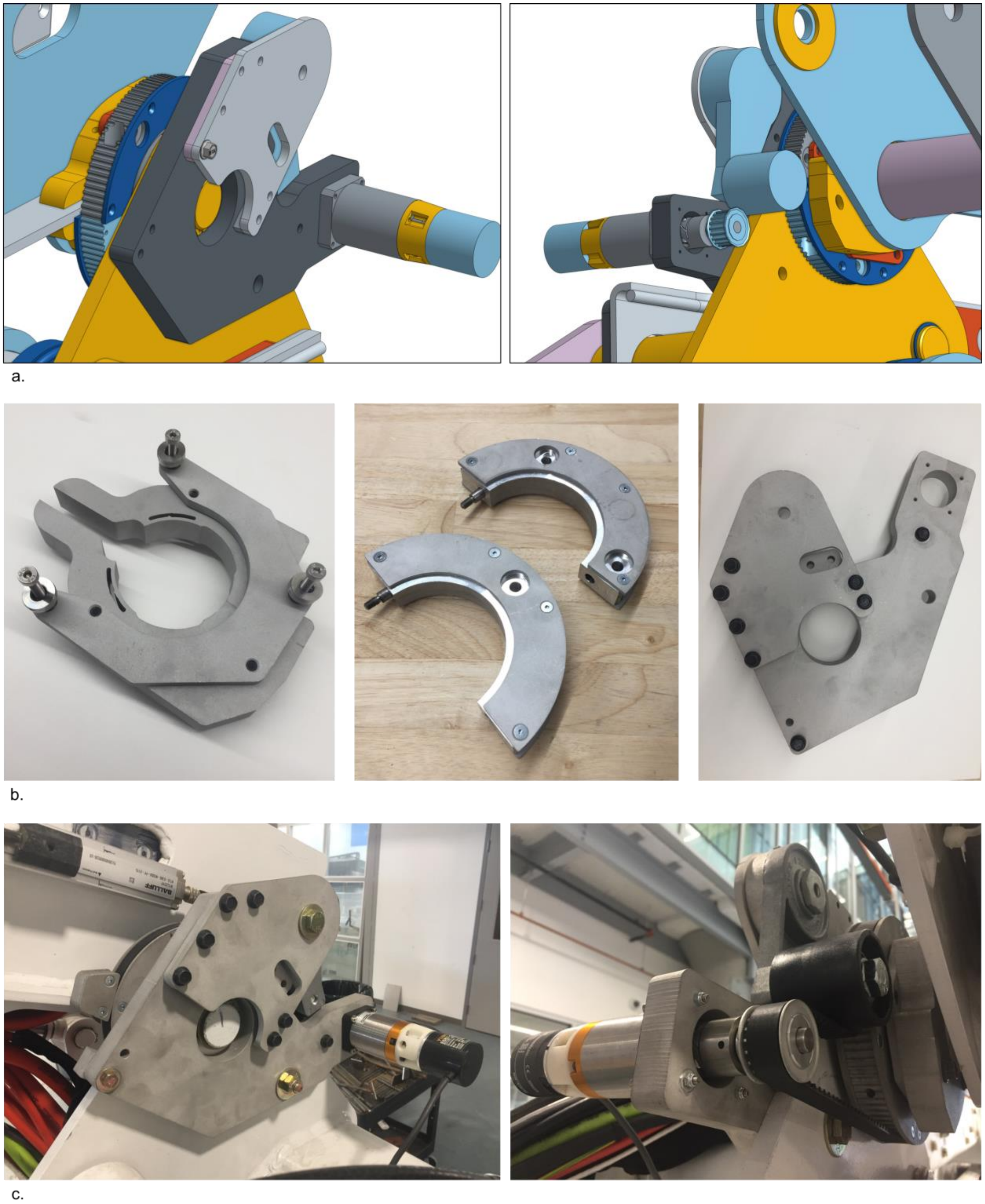


Figure 61: J3 sensor harness – CAD model (a), harness components (b) and installed harness (c)

As shown in Figure 61 above, this allowed the timing belt pulley to be installed around, and aligned to, the J3 axis of rotation without disassembling the J3 joint.

To complete the system, a Parker PV40FB-035 planetary gearbox was installed between the encoder and the output of the pulley system, to provide an additional 35:1 reduction ratio. While this gearbox is not a precision gearbox, and does introduce some backlash into the system (specified as <18 arc-min), this backlash is relatively negligible (<0.04 deg motion at the output), and is mitigated by the control scheme used to integrate these sensors, as described below. The gearbox and encoder, along with a belt tensioning mechanism, were mounted on a carrier plate which was clamped to the AT40GW elbow frame component.

The J3 encoder harness has worked well, and provides impressive sensitivity to changes in joint position, as the following figures show:

- Joint deflection under J4 extension: Figure 62 shows the position reported by the J3 rotary encoder as J4 is extended. Approximately 0.12 degrees of deflection are measured as the J4 joint moves from fully contracted to fully extended, and some hysteresis is observed on retraction.

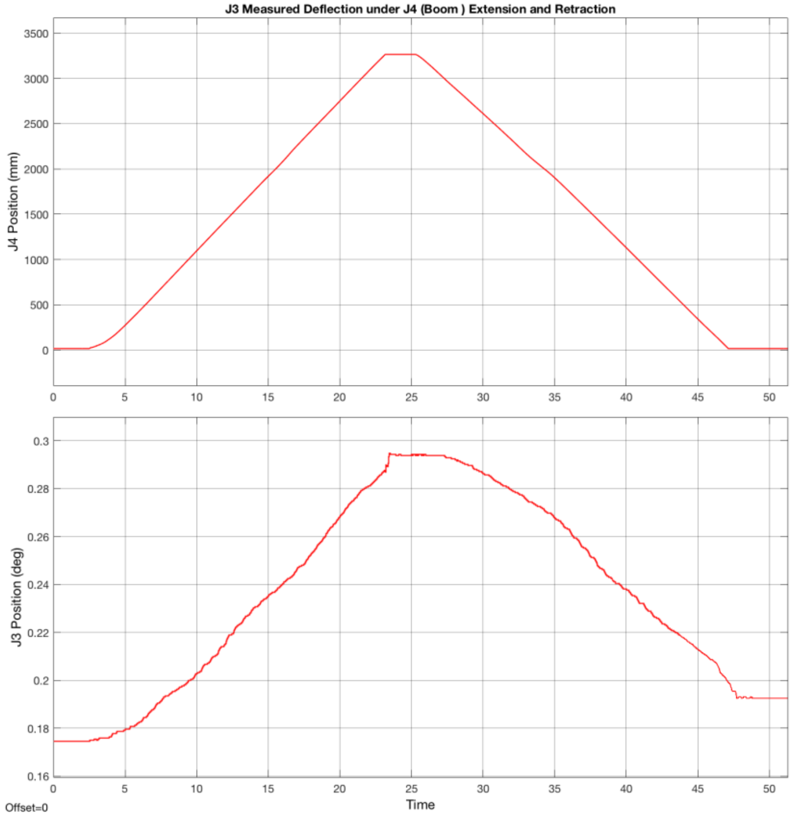


Figure 62: Joint deflection measured by J3 joint sensor as J4 joint (boom) extends

- <u>Detection of boom oscillations</u>: Figure 63 shows decaying oscillations in the boom detected by the J3 encoder after the boom is pulled down and released.

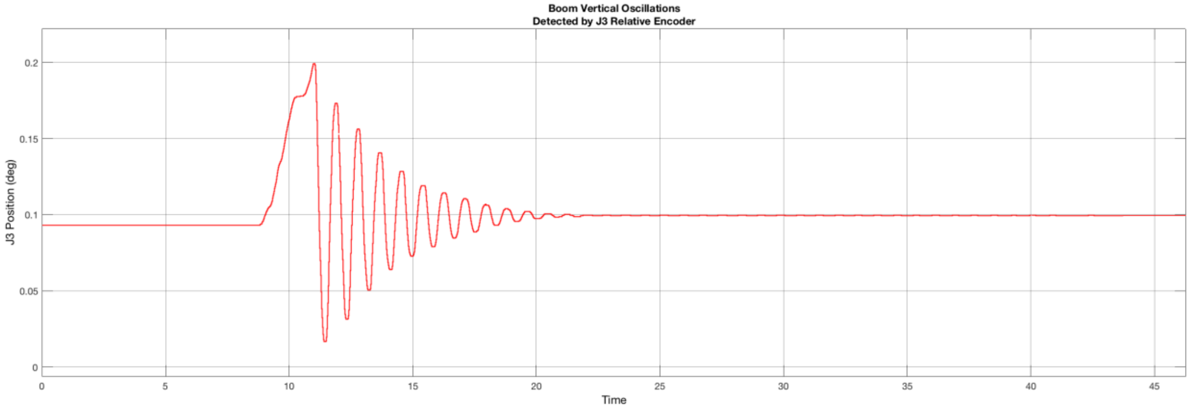


Figure 63: Boom vertical oscillation, as detected by J3 joint sensor

J1 Sensor Harness: Multi-Turn Absolute Encoder with Timing Belt Transmission

The J1 joint presented many of the same challenges as the J3 joint, except at a substantially larger scale. As described earlier, the primary requirement for the J1 sensor harness was to provide an absolute reference for the J1 joint position, and also measure the output of the joint directly. In this case, the sensor would be mounted to the rotating J1 frame, while the measurement would be taken relative to the stationary base of the AT40GW.

Like in J3, a design based around a GT2 timing belt pulley transmission, with a split input pulley, was developed. The split pulley was attached to the DCP base "pier" (visible in Figure 64) using a three-part circular clamp, held in place with large-diameter strap clamps. Because of its substantial size (nearly 500 mm diameter), the pulley was cut as two separate pieces, and only joined after the tooth profile had been cut. While the profile has not been inspected (beyond simply verifying that the timing belt can travel in and out of the pulley), it has been operated with no evidence of poor tracking or belt ejection, and suggests that this simpler method of manufacture may be viable.

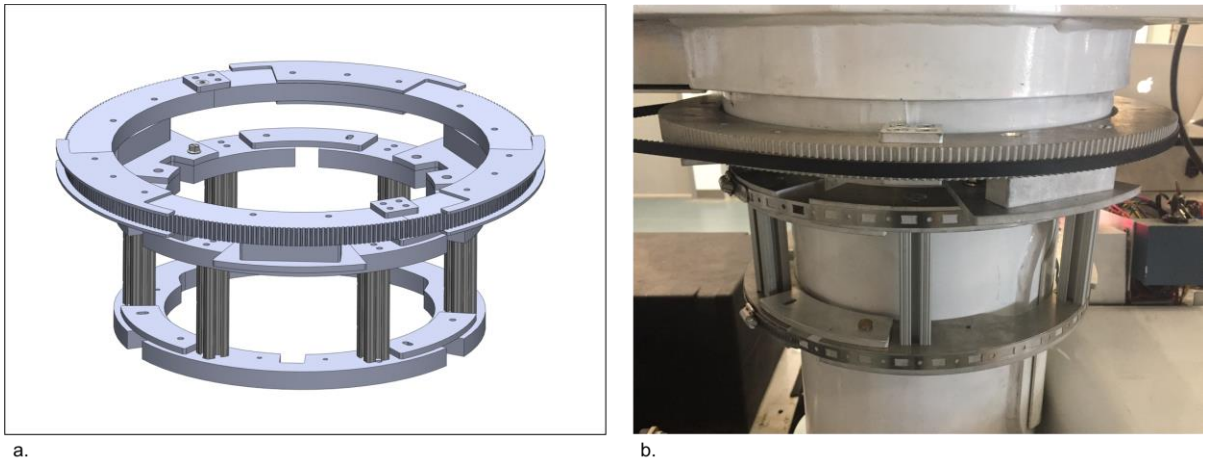


Figure 64: J1 pulley and clamp assembly – CAD model (a) and installed on AT40GW (b)

A number of different sensor technologies were considered initially for the J1 harness, including optical sensors, magnetic tape encoders and draw wire encoders. Ultimately, the Pepperl+Fuchs UV36M multiturn absolute encoder was selected for implementation. This encoder can measure rotation over up to 16 turns, while maintaining an absolute position reference. Our encoder outputs a 0-10V DC signal to report position, and was easily integrated into the DCP control architecture. A mounting frame was developed to carry this encoder and provide tension to the drive belt, as shown in Figure 65.

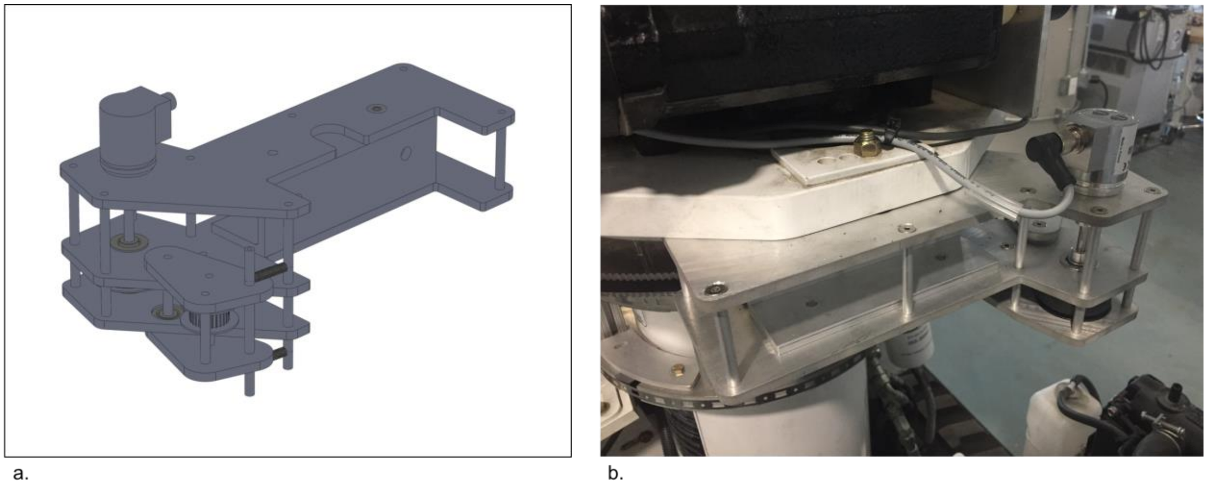


Figure 65: J1 sensor carrier assembly – CAD model (a) and installed on AT40GW (b)

The J1 sensor harness successfully provides an absolute position reference within a ±180-degree range about the DCP's typical home position. It has also enabled the backlash in the J1 transmission to be captured for the first time. As shown below, the backlash in the J1 joint has been measured to be on the order of 0.45 degrees.

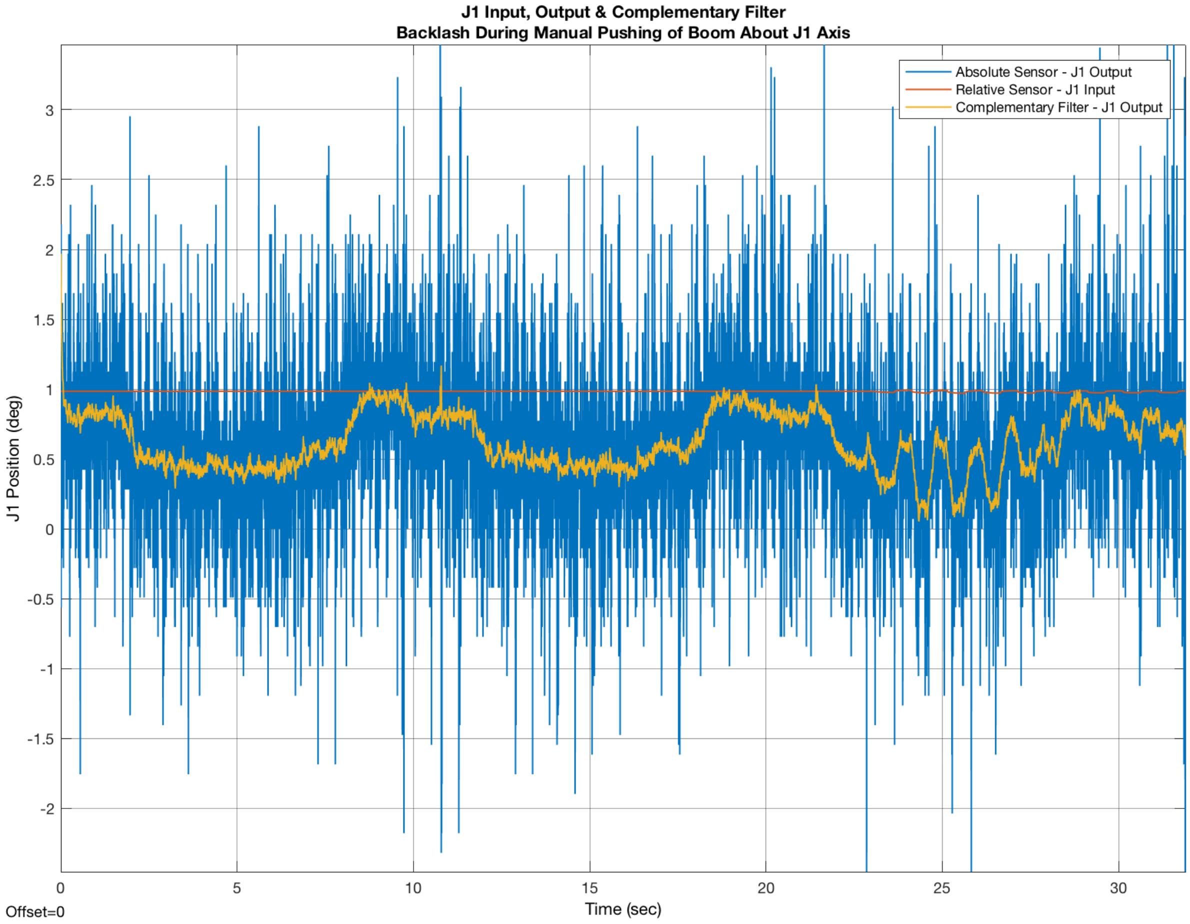


Figure 66: J1 joint backlash measured with new J1 sensor harness. Note that no motion is registered at the J1 input, as expected.

As mentioned previously, backlash in the J1 joint is expected to be variable, both as a function of J1 joint position as well as system pose. Further work is required to fully characterize this backlash, but the new J1 sensor harness is a major step towards this, and towards enabling the use of backlash compensation techniques like those described below.

Control Implementation

While the major focus of this phase of the DCP project was on system architecture development, ensuring stable system control is still a substantial challenge for the DCP, and imperative to making the DCP accessible to new users/for new tasks. Particularly with the new sensor systems described above installed, the DCP's control architecture needed to be updated to take advantage of them, and ensure stable performance across a range of tasks.

Signal Processing & Sensor Fusion

The first major step in updating the controls implementation in the dcpctrl_v2 architecture was to address the way that signals from the AT40GW were pre-processed through the qraw2qjoint_filter block, shown below.

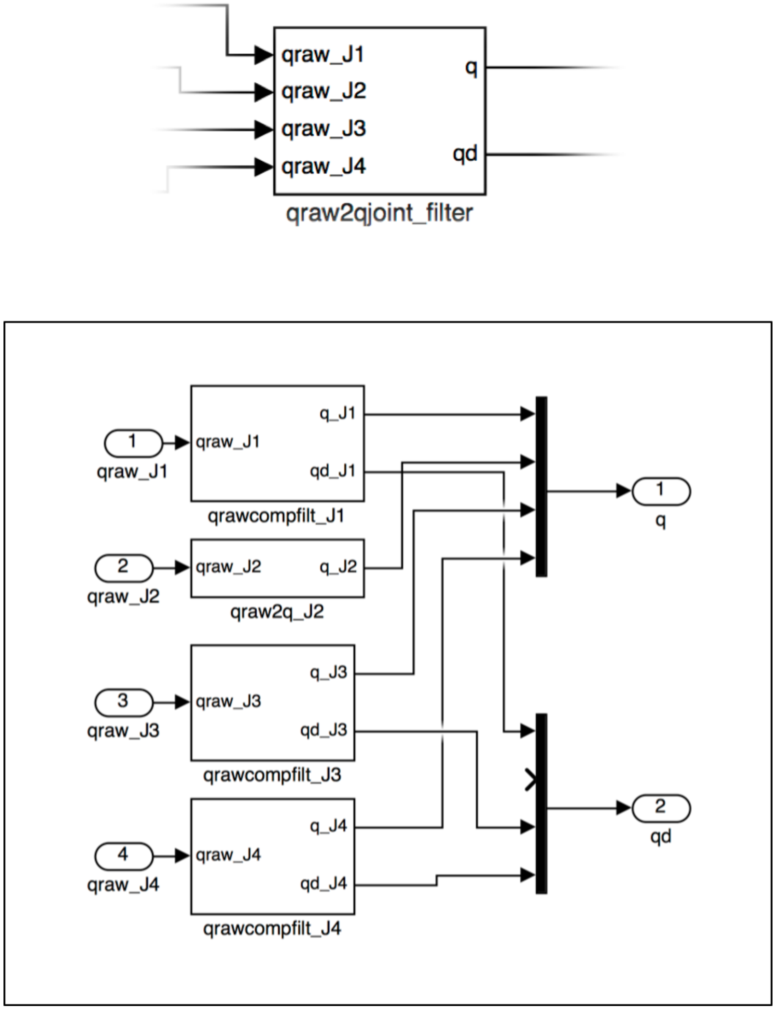


Figure 67: Signal processing block – qraw2qjoint_filter

The first major improvement made was to transition from performing control in “sensor” space – e.g. on raw sensor values, in units of volts or counts – to performing control in joint space, in appropriate units of degrees, mm, deg/s or mm/s. This was done in the qraw2q blocks, shown below. For simpler joints like J1, this was implemented using Simulink blocks. For more complex joint sensors – particularly the J2 and J3 absolute position sensors, where sensor position and joint angle are related through trigonometric identities – mapping functions developed previously in dcpctrl_v1 were reimplemented using Simulink function blocks. These blocks also calculate derivatives of quadrature encoder signals to produce a clean velocity signal; and use the absolute encoder as a reference to offset the position signal taken from the relative encoder. Generally, each block outputs position signals for both the absolute (analog) encoder as well as the relative (quadrature) encoder at each joint; as well as an output joint velocity signal.

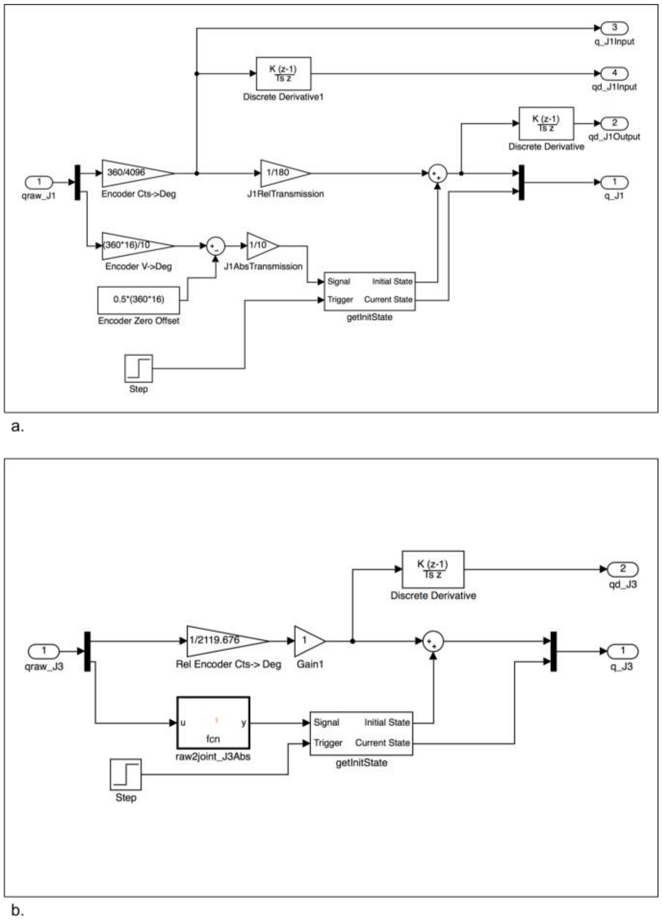


Figure 68: Conversion from sensor-space to joint-space, for J1 joint (a) and J3 joint (b)

There are many techniques for leveraging these new sensor signals to provide joint position information. The simplest method is simply to rely on the signal from the relative encoder, using the absolute encoder only as a starting reference to determine the joint's absolute position. However, this technique is still highly susceptible to both high-frequency noise in the analog sensor (causing spikes that may lead to erroneous absolute position measurements at startup), and to low-frequency noise in the quadrature encoder in the form of missed or accumulated counts, as was observed during the foam dome print (described previously in Chapter 2). A more robust approach combines both sensor streams to create a composite signal, using a variety of different sensor fusion techniques such as Kalman filtering. One particularly straightforward, but still effective, sensor fusion technique for combining signals with the noise characteristics described above (low-frequency noise in one signal, high-frequency in the other) is the complementary filter.

The complementary filter can be configured in a variety of ways to solve different sensor fusion problems [82]. For use in the DCP, we adopt the "signal-plus-derivative" configuration described by Colton in [83], where the complementary filter is used to generate a drift-free estimate of angular position from combined accelerometer (signal) and gyroscope (signal derivative) datastreams. We use the following form of the complementary filter:

$$\hat{\theta}_{t+1} = \theta_{A_t}(1-\alpha) + \left(\dot{\theta}_{R_t} \cdot \mathit{\Delta} t + \hat{\theta}_t\right)\alpha,$$

where

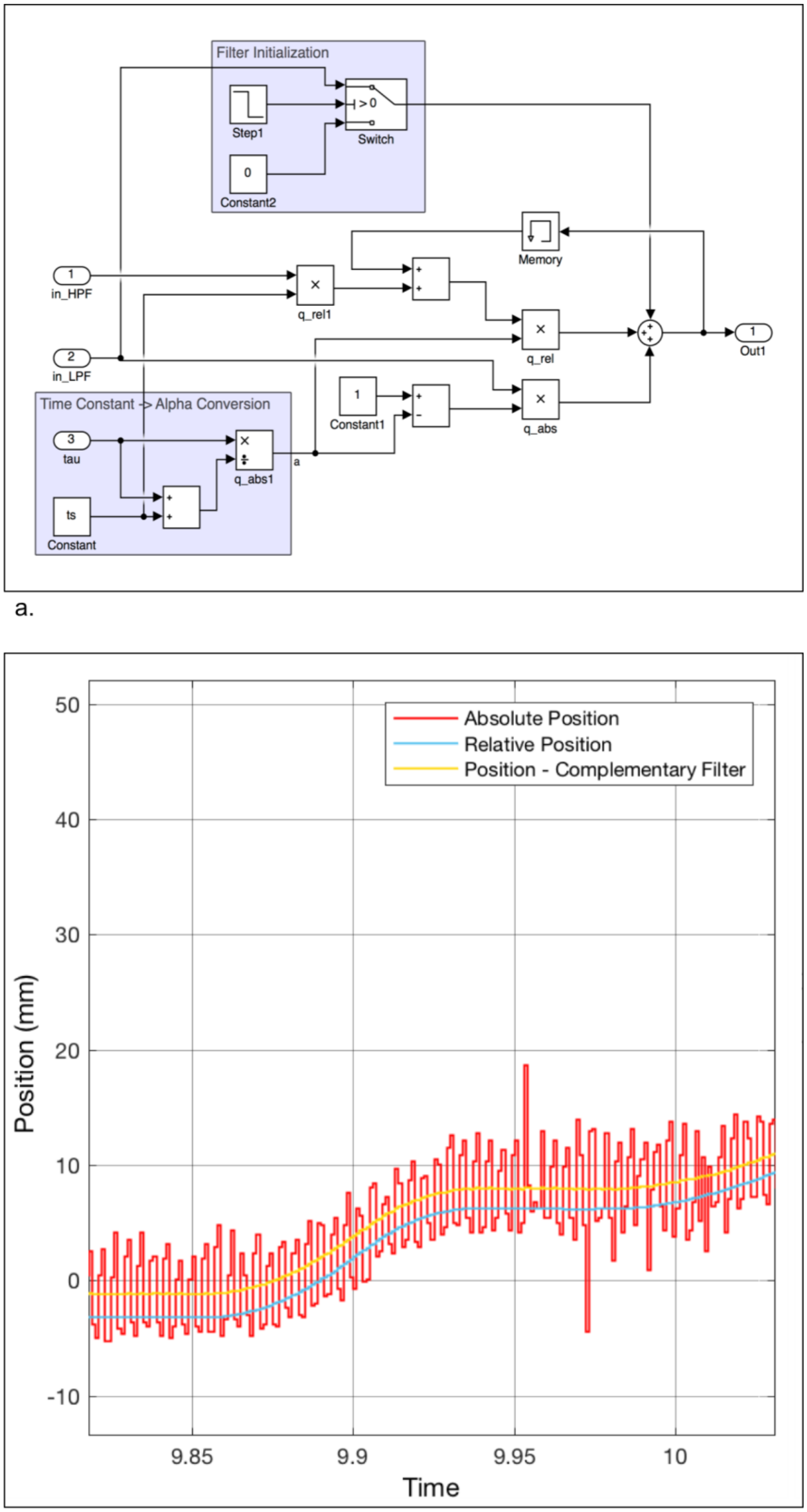
$\hat{\theta}_t =$ current position estimate;

$\hat{\theta}_{t+1} =$ new position estimate;

$\theta_{A_t} =$ current position measurement from absolute sensor;

$\dot{\theta}_{R_t} =$ current velocity measurement from relative sensor.

The new position estimate is generated from a combination of the absolute sensor's current position estimate, and an integration of the relative sensor's current velocity measurement with the past position. The impact of these two estimates is combined through complementary low-pass and high-pass filters, such that the net gain of the filter is 1. This function is implemented in Simulink as shown in Figure 69.a.



a.

b.

Figure 69: Complementary filter – Simulink implementation (a), and resulting signal (b)

As shown in Figure 69.b, the complementary filter performs well in this configuration, rejecting high-frequency noise in the absolute position sensor's signal. It is also insensitive to large steps – for example, produced by the sudden accumulation of missed counts – in the relative position sensor's signal. It is not used in the J1 joint, because of the backlash present between the absolute and relative sensors[17], but has been successfully implemented at the J3 and J4 joints.

During the development of this filter, an interesting phenomenon was observed in the velocity signals produced by the relative encoders. As described earlier, the new joint encoder harnesses were designed to provide minimum joint velocity resolution specifications. These velocity resolutions were calculated as the velocity that would be produced if one count of the quadrature encoder was observed during one control cycle: in the case of the DCP's controller, 1 ms. However, this number not only defines the minimum velocity the sensor can resolve, but also the velocity increments that the sensor can resolve. This is a common problem encountered in low-speed control with digital encoders. In [84], Petrella et. al. list a number of different approaches can be adopted to address it, including measuring the period between individual pulses instead of the number of pulses within a time window; a variety of filtering strategies; and observer-based strategies such as Kalman filtering. However, they also note that simple moving-average filtering can provide adequate results with frequency-measured data, as long as the steady-state measurement error and lag introduced by the average are acceptable.

[17] Currently, the J1 joint is controlled using only state information from the relative encoder at the motor, with an absolute reference provided at startup by the absolute encoder at the joint output. Brief experiments after installation of the J1 absolute encoder demonstrated that J1 backlash, though small, still produced limit cycle behavior. As described below, some bench-level experiments with backlash control have been conducted, but not yet implemented on the DCP. This, along with more effective combination of the J1 relative and absolute sensor streams, are topics for future research.

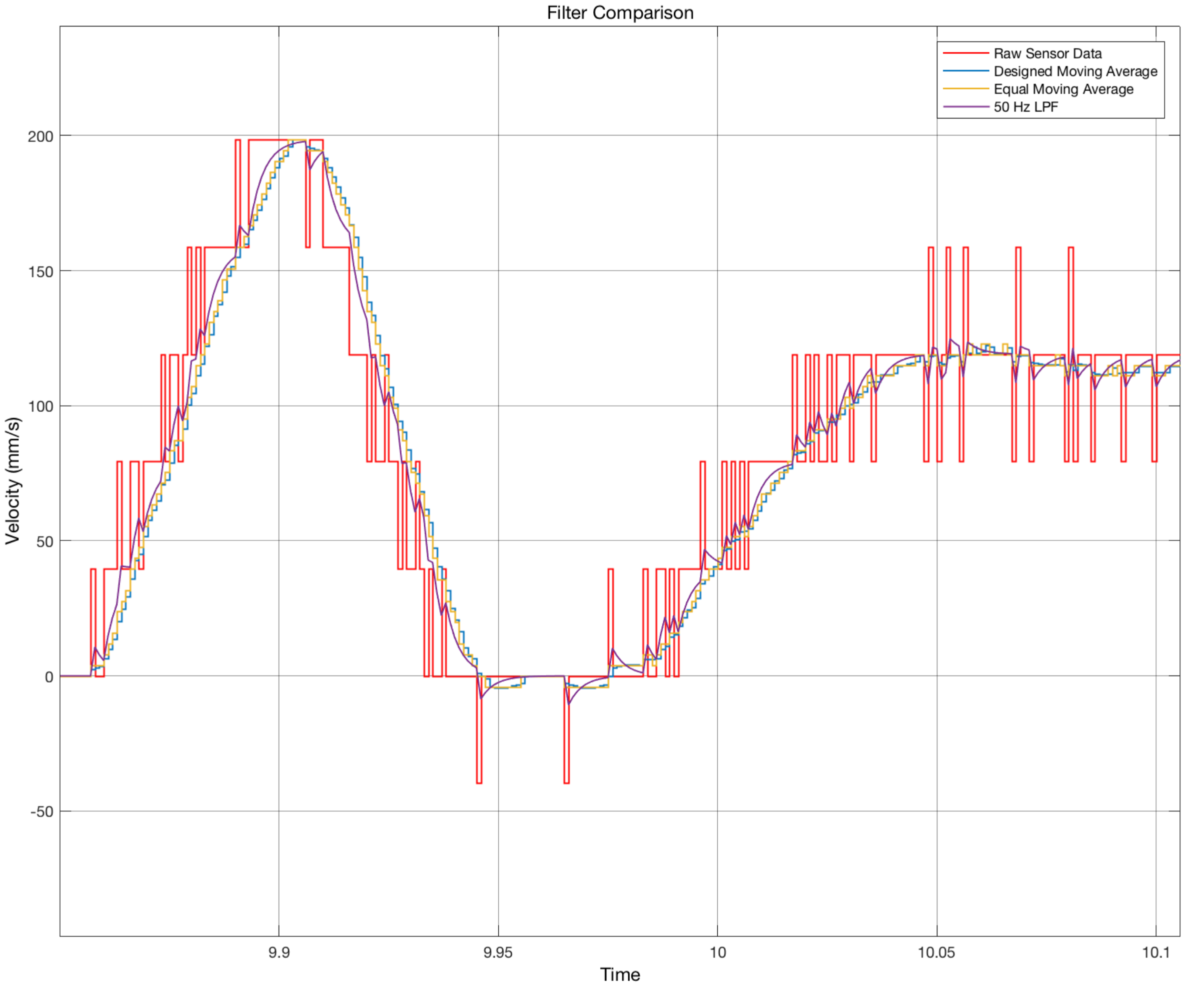
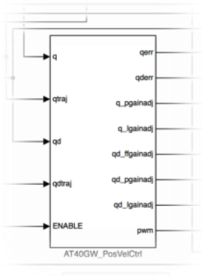


Figure 70: Velocity signal with different filters applied to improve velocity resolution

As shown in Figure 70, a number of different filters – including a continuous low-pass filter with a 50 Hz cutoff; a designed moving average; and an equal moving average were tested. Ultimately, a 50-sample equal-weighted moving average filter was applied to all joints. This filter provides reasonable velocity resolution, and introduces only 24.5 ms of lag to the velocity signal. There is still substantial room for improvement of this filtering scheme, particularly through more careful design of the filters with respect to the rest of the joints (ensuring equal worst-case task-space velocity resolution across joints) and in the context of the rest of the control system (examining interactions between the filter and the rest of the joint controller).

Joint Controller Development & Tuning

With high-resolution, low-noise joint position and velocity signals now available, a much wider array of controller forms can be implemented on the DCP. During work 2016-2017, a number of different controllers were prototyped at different points during the DCP's development, becoming more complex as additional sensors and other components were brought online. The most fully developed of these is a nested position-velocity controller, implemented in the AT40GW_PosVelCtrl block, which we use here as an example system.



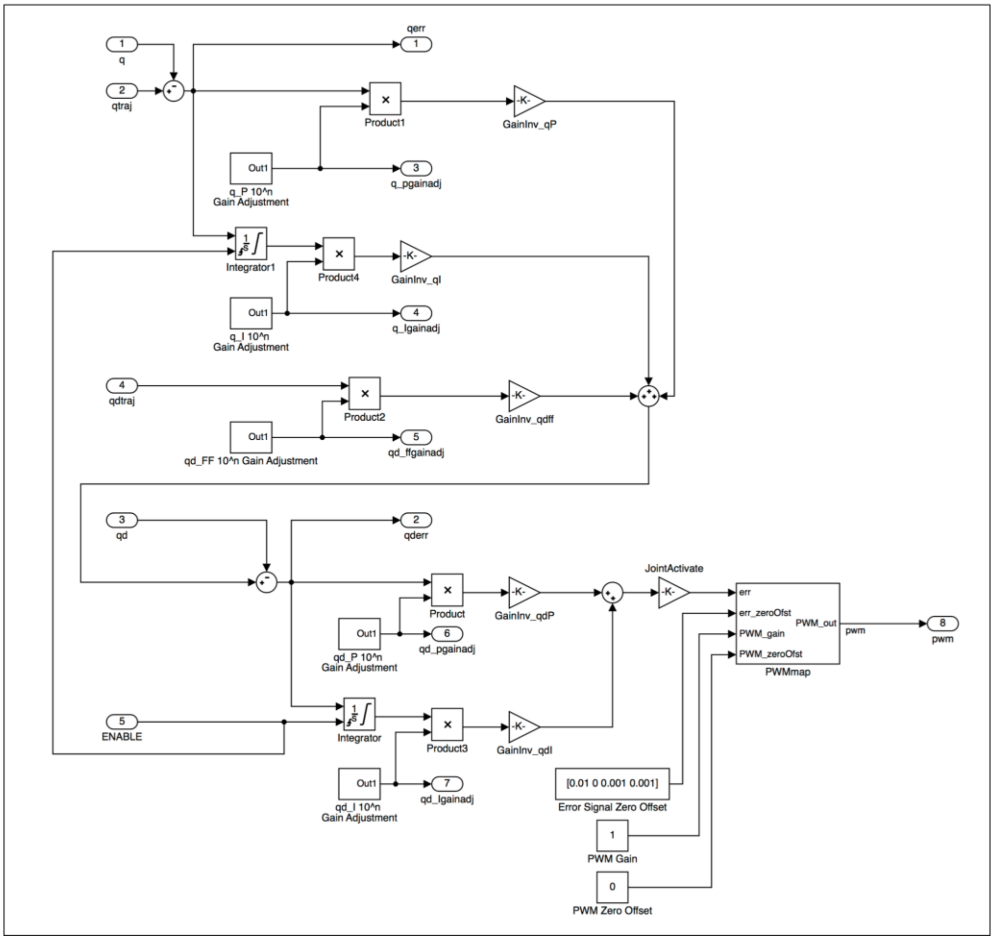
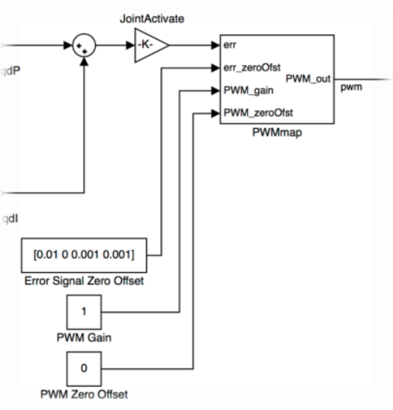


Figure 71: Controller block – AT40GW_PosVelCtrl

This controller implements a basic nested position-velocity control loop architecture, modeled after the example given in [85] by Corke. Both loops implement proportional-integral control. The integrator blocks are limited to prevent integrator windup; they additionally implement an integrator reset that is triggered whenever a rising edge is detected on the

ENABLE line (for example, whenever the DCP resumes executing a trajectory after being paused). In the position loop, error is calculated between measured joint position and desired (trajectory) joint position. In the velocity loop, error is calculated between measured joint velocity and the sum of the position controller's output plus a feedforward velocity term, provided by the joint velocity trajectory that the DCP's trajectory generation architecture produces.



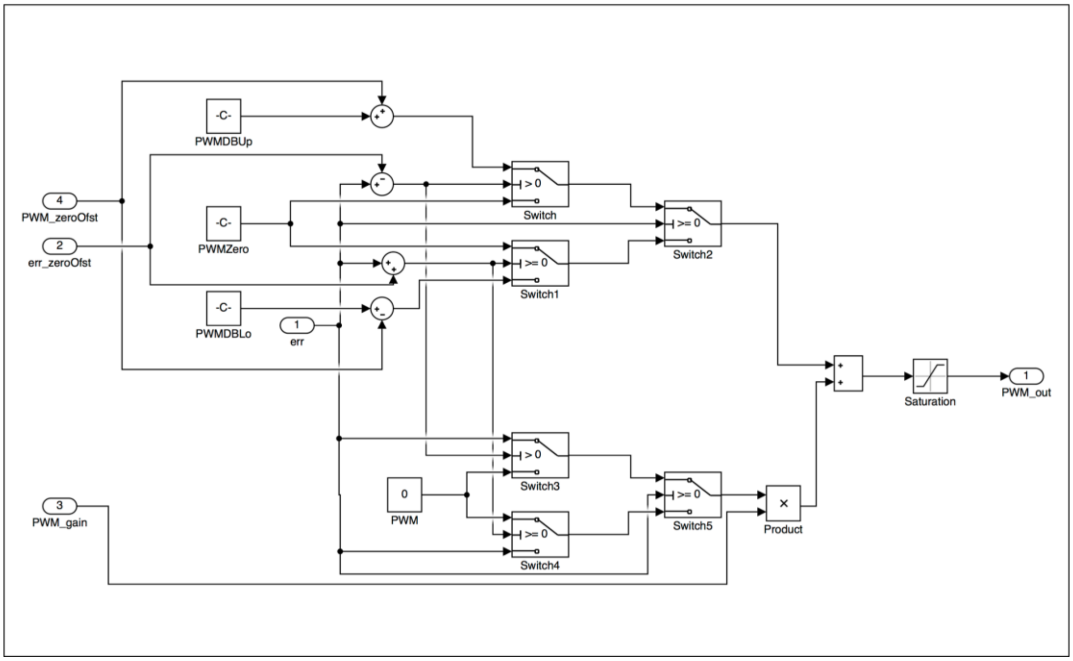


Figure 72: PWM mapping block – PWMMap

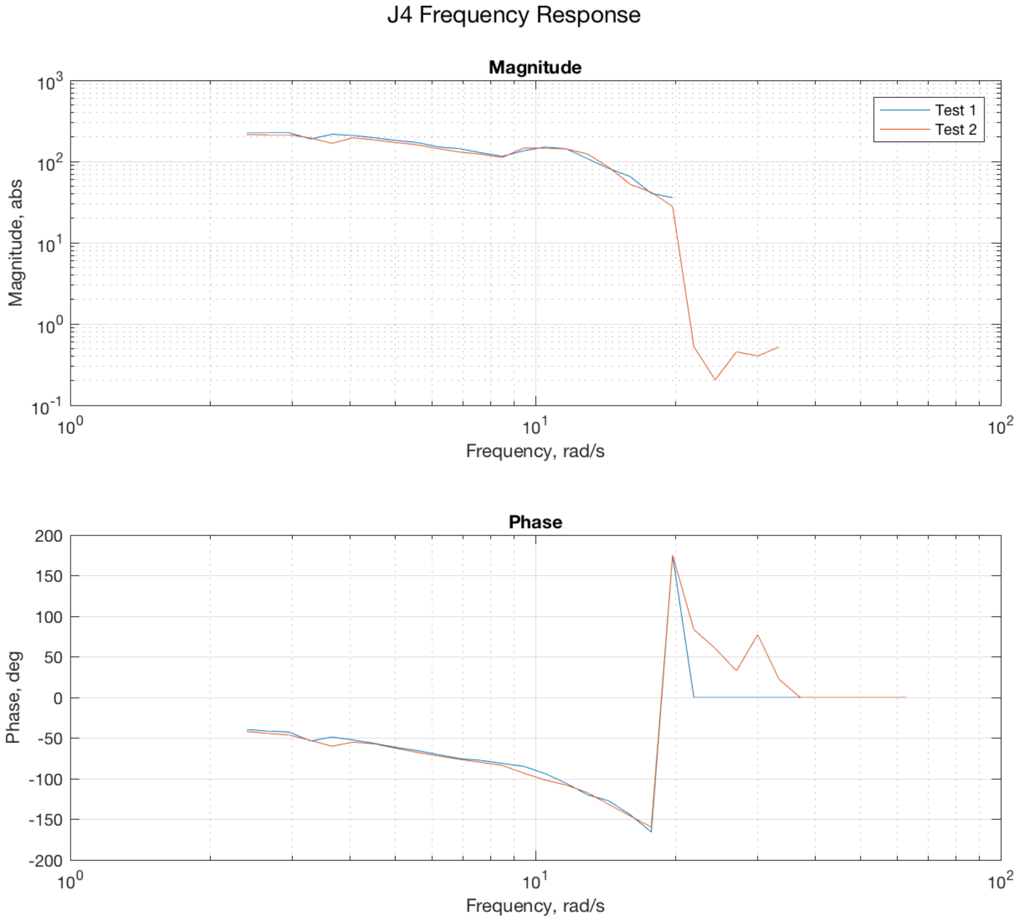
Finally, the output of the velocity controller is fed to the PWMMap block, shown above in Figure 72. The PWMMap block provides two functions:

1) Mapping the velocity controller output from a zero-centered range to a 0.5-centered range, appropriate for commanding PWM duty cycles, and;

2) Attempting to compensate for the valve deadband nonlinearity by creating an "inverse deadband" function using measured deadband limits. This inverse deadband technique improves system performance, but is still far from perfect. First, the deadband limits are not consistent: they have been qualitatively observed to vary over time, and as a function of system load. Second, the system response at the deadband limits is not linear. There can be considerable delay between actuation and response from the joint at the deadband limit. Furthermore, the response is generally smooth in one direction (e.g., the joint achieves its steady state velocity), but stepped in the other direction (the joint moves briefly at one velocity, and then accelerates to a higher steady-state velocity). This is believed to be related to the system's counterbalance valves, as this behavior is not observed on J1.

The PWMMap block also provides inputs that allow the user to modify the "zero range" for the input (the band of input signals considered = 0); the gain applied to the PWM duty cycle output; and the offset applied to the PWM deadband limits, expanding or contracting the deadband symmetrically. While functional (and critical to the controller's functioning), the PWMMap block is one of the most significant weaknesses of this controller. It is a highly nonlinear element, and its impact on the dynamics of the controller as a whole have not been carefully analyzed. Future controls development will need to revisit this block to better understand its impact on the system, and develop improved methods of dealing with hydraulic valve deadband.

Control gains for the nested position-velocity controller were tuned manually, on a per-joint basis.[18] A step signal source (the enabledStep block, available in the dcpctrl block library) was

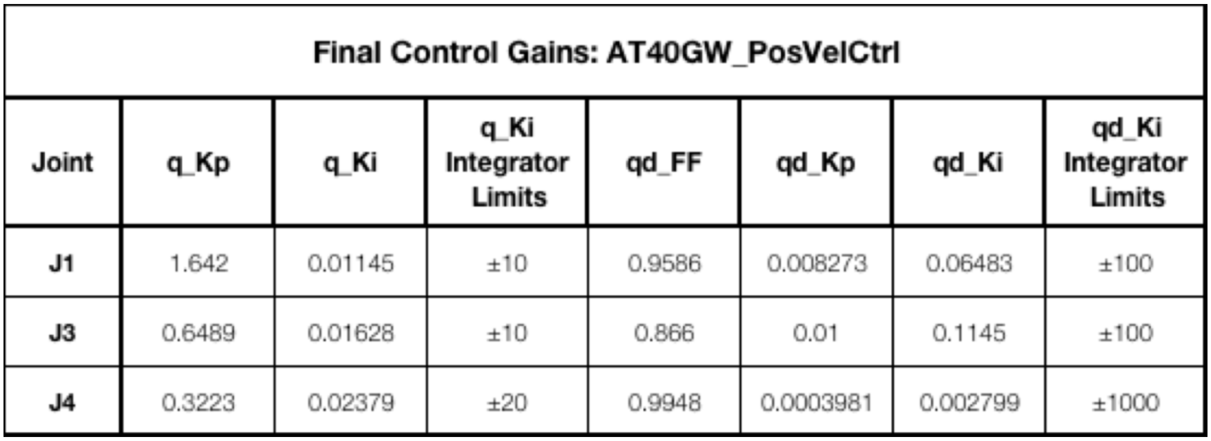


J4 Frequency Response Plot. Frequency response is measured between scaled velocity (0-1) and measured velocity (mm/s).

Despite this, sufficient data was collected during the test to generate the Bode plot shown above. The plot shows attenuation beginning to occur around 1.5 Hz, with a sharp drop-off at 3 Hz. These numbers

[18] A brief attempt was made at identifying system models for the AT40GW's joints. The Simulink dynamic system analyzer (DSA) developed by Royalty [99] was used to provide a sinusoidal PWM input to the AT40GW J4 joint, using the PWMMap block to try to compensate for valve deadband. While this system was able to successfully drive the AT40GW, asymmetry in the AT40GW's PWM/velocity relationship caused the J4 joint to extend further than it retracted on every cycle of the controller, eventually causing the joint to extend to its limit. Consequently, the test – which had been set to sweep from 10 Hz to 0.1 Hz – failed at 0.34 Hz.

used as a reference signal for tuning, with amplitude adjusted to joint-appropriate limits (generally, around 75% of joint maximum velocity, or 5% of joint total range). The velocity loop was tuned first, followed by the position loop. Generally, we sought to achieve a critically-damped or overdamped response in all joints, to try to minimize excitation of system oscillatory modes. The exception to this is the J1 position control loop, where a slightly underdamped response was observed to actually damp out oscillations, at a range of different boom extension distances. The final control gains and integrator limits identified through this process may be seen below, in Table 5.



Final Control Gains: AT40GW_PosVelCtrl

Joint	q_Kp	q_Ki	q_Ki Integrator Limits	qd_FF	qd_Kp	qd_Ki	qd_Ki Integrator Limits
J1	1.642	0.01145	±10	0.9586	0.008273	0.06483	±100
J3	0.6489	0.01628	±10	0.866	0.01	0.1145	±100
J4	0.3223	0.02379	±20	0.9948	0.0003981	0.002799	±1000

Table 5: Final control gains and limits used in AT40GW_PosVelCtrl

This controller has been used in a range of experiments, including the ISO 9283 characterization and micro-macro toolpath segmentation experiments described in Section 3.3 below. It also provides a basic template for how controllers can be implemented in the DCP. However, the wide range of states that the dcpctrl_v2 architecture enables us to measure – such as endpoint position as measured by joint sensors, or even by external measurement systems – makes a wide range of other control architectures accessible. For example, a controller could be implemented on DCP endpoint position, using an external position tracking system like a laser

have been anecdotally confirmed by Altec engineers as comparable to what they have observed with similar systems. However, the many problems with this test – particularly the asymmetry in system velocity response, and the inclusion of the nonlinear PWMMap block in the system – made these results too suspect to use in design, and this experiment was abandoned.

tracker or vision system, and producing velocity commands for the AT40GW using the system's Jacobian matrix, which can be implemented in Simulink using a MATLAB Function block.

Advanced Control Techniques

While the control architecture described above is reasonably functional, there are a wide range of specific control techniques described in the literature that could potentially help improve the performance of the DCP. A number of these techniques were explored in the course of the DCP project, but due to time limitations, none were fully implemented on the system. However, lessons learned from these explorations are listed below, in hopes that they may support future work with the DCP system.

- Input Shaping: One of the most substantial challenges faced by the DCP project – and indeed, any automated construction system built around relatively flexible systems like the AT40GW – is the problem of controlling link oscillation. A wide range of techniques have been proposed for control in the presence of flexible links, including the various micro-macro manipulator control strategies referenced in Chapter 1; feedback-based control strategies that leverage additional sensors, such as strain gages measuring link deflection directly, or accelerometers located at the end of the flexible link [86]; and strategies based on pre-shaping input commands to avoid exciting vibratory modes in the structure. Overviews of the flexible link control problem and the various solutions that have been proposed may be found in Book [40] and De Luca and Book [87].

  Of these techniques, one of the simplest and most intuitive is the Input Shaping technique developed by Singhose, Seering and Singer. In the simplest implementation of Input Shaping, a model of the system's oscillatory behavior is used to create a "filter" composed of impulses appropriately spaced in time that will cancel out the system's response to a unit impulse. In [88], Singhose and Seering provide an excellent intuitive demonstration of this, through the example of a user swinging a pendulum in their hand, and then cancelling out the oscillation of the pendulum by rapidly moving their hand sideways over the pendulum end as it reaches the peak of its swing. This filter can then be applied to non-impulse commands via convolution, providing "shaping" to these inputs.

The Input Shaping technique has been implemented in control of a wide range of flexible systems, particularly including tower cranes and large flexible robots, and has been extended to include robust and adaptive control formulations. For the DCP, it is a particularly compelling solution for addressing lateral oscillations produced by motions about the J1 joint. The deadband and backlash present in the J1 plant pose challenges for a successful implementation of an input shaper. However, in [89], Sorensen and Singhose assert that inclusion of inverse backlash and inverse deadband elements between an input shaper and a plant may provide adequate performance, even if the inverse models are imperfect. When taken in combination with adaptive inverse generation techniques like those described in Backlash Compensation and Control, below, this may provide a promising route forward for input shaping on the DCP.

- Arm-Finger Controller Architecture: A second compelling control technique that could have potential applications in the DCP comes from efforts in anthropomorphic robotics to mimic the functionality of the human arm and hand. One example of this is in [35], where Quan et. al. report on a controller architecture for a finger-arm robotic system comprising a 6-DOF macro-manipulator and a 3-DOF robotic finger (or micro-manipulator). They use the manipulability index of the micro-manipulator to determine whether to command motions to the micro- or macro-manipulator. If the manipulability index is high, the micro-manipulator executes a motion; when it drops below some threshold, the macro-manipulator moves to execute the motion and allow the micro-manipulator to recover.

Quan et. al. implement this technique in a controller, rather than as an offline path-planning technique. They demonstrate the viability of this method both on simple paths (3D sinusoidal curves) and complex shapes (a small triangle and circle, separated by a long line). Their work provides a compelling alternative to the offline trajectory segmentation techniques described in Section 3.3.2, and may similarly provide benefits for reducing system vibration by diverting small-volume, high-acceleration motions to the micro-manipulator.

- Backlash Compensation and Control: Finally, the variable backlash in the J1 joint presents another problem for precise motion control of the DCP, making tracking control challenging, and also impeding the development of other control techniques (such as input shapers). In systems like the DCP – which ideally use coarser, construction-grade mechanical components rather than more sensitive (and more expensive) precision components – this type of backlash is likely to be a common occurrence, and an important control challenge to address early. In the interests of this, a brief exploration of backlash-mitigating controllers was conducted by the author, as a final project for 2.152 (Nonlinear Control System Design).

  Because of the backlash nonlinearity's ubiquity, control engineers have explored a variety of ways to address backlash over the decades since the problem was first explored in the 1940s. In their thorough review of the existing backlash literature [90], Nordin and Gutman describe a wide range of solutions to the backlash problem that have been developed over the years, including linear controllers; state-feedback controllers; observer-based controllers; adaptive controllers; fuzzy controllers; and switched linear controllers.

  To help leverage these solutions on the DCP, a bench-scale mockup of the DCP's J1 joint was developed, as shown in Figure 73. The test bench connects a brushed DC motor through a planetary gearbox and a series of toothed belts to a custom-designed adjustable backlash coupling, with a total reduction that is roughly equal to that in the DCP. By placing the backlash at the output of the transmission, the system closely models the kinematics of the DCP's J1 joint. The adjustable backlash coupling (Figure 73.b) enables the amount of backlash in the system to be adjusted from 0° to approximately 20°, by adjusting the axial position of a ball-tipped plunger mounted to the outer hub relative to a vee machined in the inner hub. The position of the motor and output shafts are monitored using the same sensors that are implemented in the DCP.

The motor is controlled using a Pololu jrk 12v12 PWM DC motor driver module, which is run at 12 V and can supply up to 12 A.

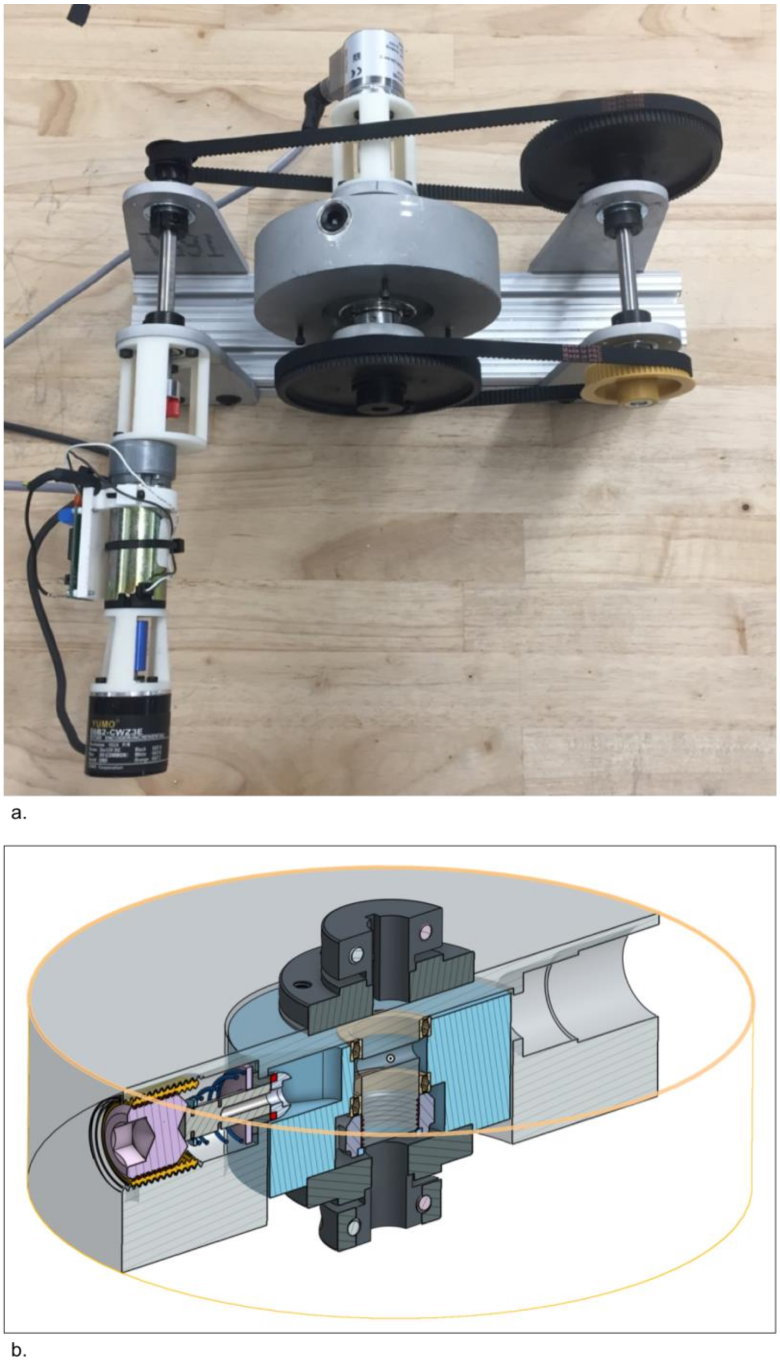


Figure 73: DCP J1 test bench (a), and detail of adjustable backlash coupling (b)

Using this bench-scale model, two separate control architectures were explored: the improved dual-loop (IDL) controller, described by Tal in [91], and the adaptive backlash inverse controller proposed by Ahmad et. al in [92]:

- The Improved Dual-Loop Controller: The IDL controller is based on the standard dual-loop control architecture, where an inner loop on motor position is driven by an outer loop on load position. Generally, the inner loop controller provides derivative action only, while the outer loop provides proportional-integral action. The advance proposed by Tal is to shift proportional control to the inner loop, causing the inner loop to act as a low-pass filter element and improving loop stability.

  This architecture was implemented in a Simulink block diagram for simulation as shown below in Figure 74.

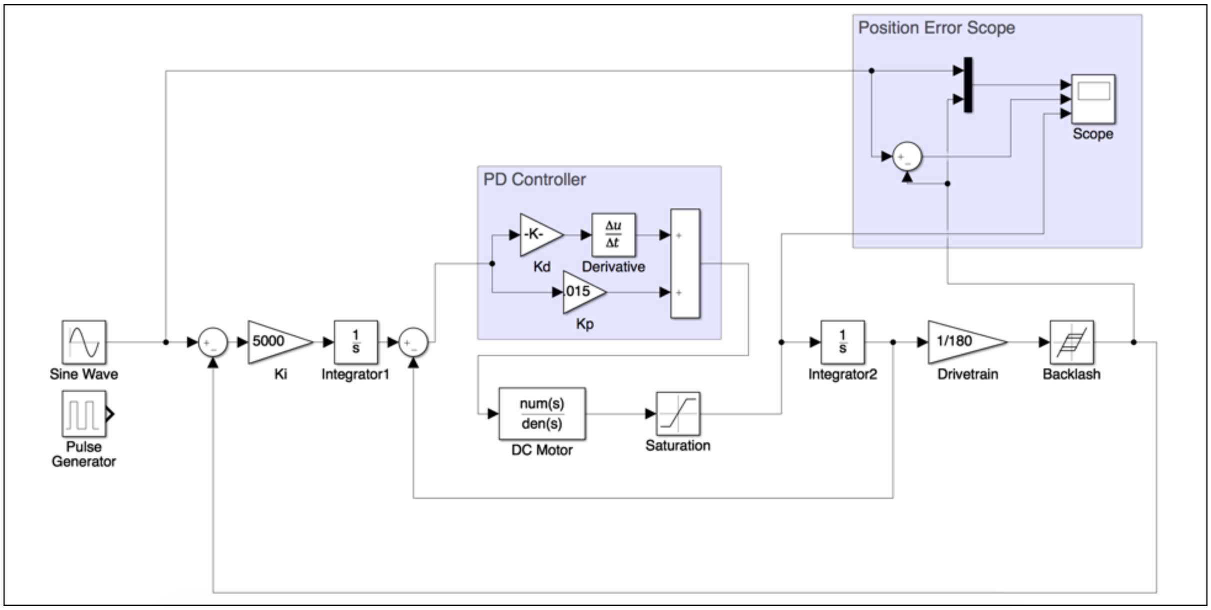


Figure 74: Improved Dual-Loop controller in Simulink

This controller can be relatively easily manually tuned. For computer simulations, the following gains were used: Ki = 5000, Kp = 0.015, Kd = 0.0005.

The system was designed for stable response to a 1 rad/s, unit amplitude sine wave. This controller was then transferred to the physical test bench, where it was used to track a using the 1 rad/s, 2 rad amplitude sine wave input. Results from this test may be seen in Figure 75.

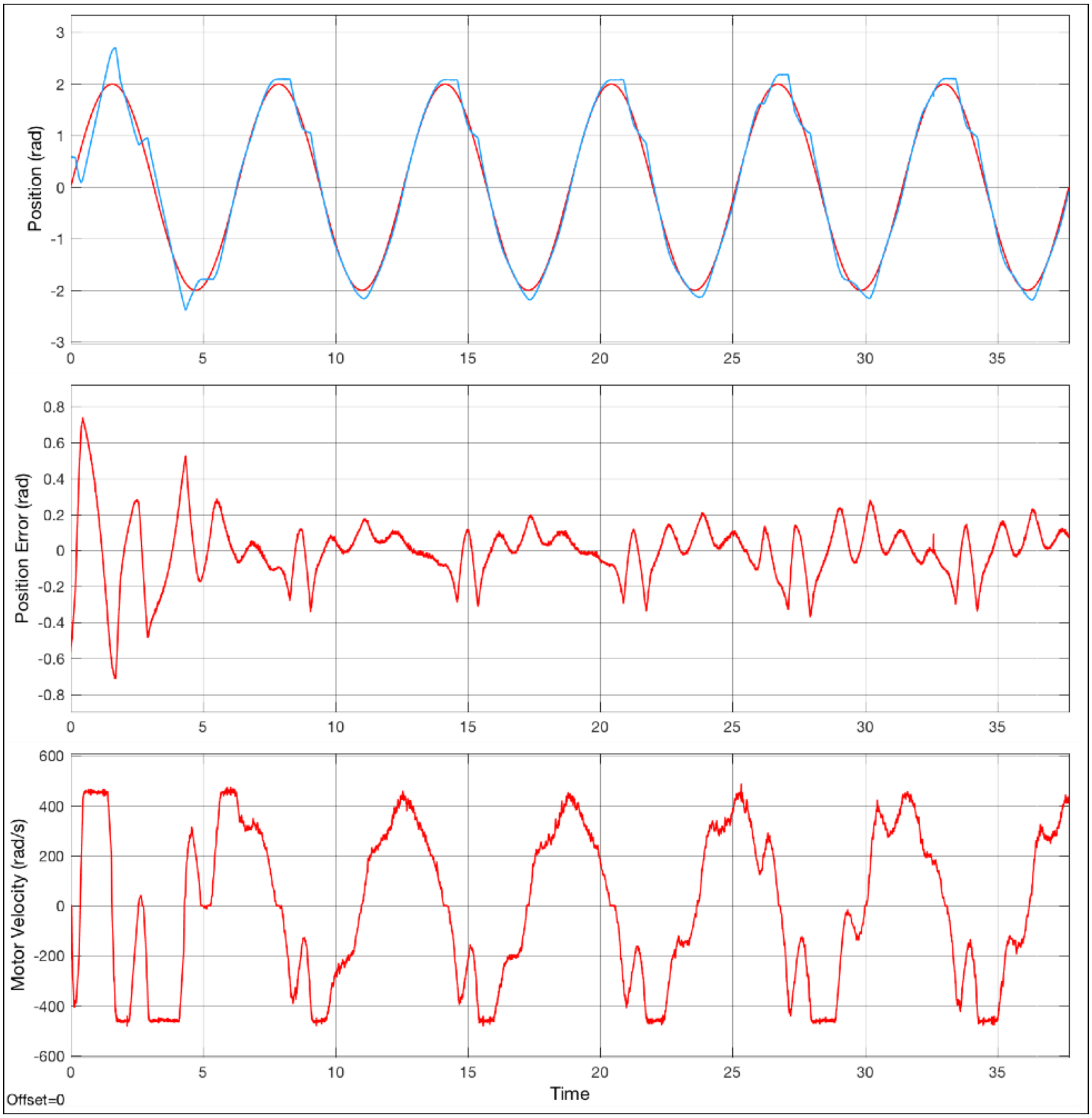


Figure 75: Improved Dual-Loop controller response on test bench

The controller provides adequate tracking performance, avoiding limit cycle behavior observed on the test bench with simple proportional controllers. However, the load velocity is not smooth, particularly when the system begins to move with negative velocity. In this region, if the motor advances quickly through the backlash gap and causes the backlash input to "strike" the backlash output, the backlash output actually springs away. This means that it is behaving as an inertia-driven hysteresis, rather than as a friction-driven hysteresis. The DCP is believed to be better represented as a friction-driven hysteresis, so one possible solution to this problem would be to apply a frictional resistance to the backlash output such that its velocity is ensured to be zero when the backlash gap is open. However, clearly defining the behavior of the real system is an important consideration in the development of backlash controller, as discussed further below.

- The Adaptive Backlash Inverse Controller: The second controller explored here is the adaptive backlash inverse controller proposed by Ahmad et. al in [92]. This controller a) implements a continuous-time approximation to the backlash inverse function to provide strong action in the backlash gap by augmenting the current motor position command, and then b) incorporates an adaptive coefficient with this action to accommodate changing or imprecisely measured backlash in the system. It also assumes that system state – load position as well as motor position – is fully known. It is conceptually quite similar to the improved dual-loop controller – even implementing a PD controller around the motor – except that the I term on load position is replaced by the adaptive element.

  To briefly summarize the presentation of this controller in [92], the desired behavior of the motor as a function of desired load velocity can be described as:

$$\theta_m(t) = \theta_{ld}(t) + f(t);\; f(t) = \begin{cases} B & \dot{\theta}_{ld} > 0 \\ -B & \dot{\theta}_{ld} < 0 \\ \dot{f}(t) = 0 & \dot{\theta}_{ld} = 0 \end{cases}$$

where B = $\frac{1}{2}$ the backlash spacing, and $f(\dot{t}) = 0$ indicates that f(t) should not change. In short, this condition specifies that the motor command should follow the desired load position, plus or minus some constant amount to accommodate the backlash present in the system.

To implement this in a continuously differentiable form, Ahmad et. al. define the following new function to replace f(t) in the backlash inverse model given above, where B_n is the nominal (or estimated) backlash distance; k_s is a positive constant used to a) make the hyperbolic tangent function operate like a signum function, and b) modulate the “speed” of the function’s transition from the forwards to the backwards state; and α is a positive constant which also affects the “speed” of transition of W.:

$$f(t) = \alpha W; \;\; \dot{W} = \tanh\left(k_s \dot{\theta}_{ld}\right) - \frac{\sigma}{B_n} W$$

Finally, to accommodate a situation where the exact backlash spacing is not known – or where the backlash spacing changes as a function of time – Ahmad et. al. introduce an adaptive coefficient of the form:

$$\theta_m^*(t) = \theta_{ld}(t) + \hat{\beta} f(t); \;\; \dot{\hat{\beta}} = -k_s \alpha W(\theta_l - \theta_{ld})$$

The purpose of this coefficient is to scale the fixed backlash inverse defined by f(t) = αW to match a changing or undefined backlash gap[19].

[19] Ahmad et. al. also provide Lyapunov-based stability arguments for W and the controller as a whole. They are not reproduced here for conciseness, but may be seen in [92].

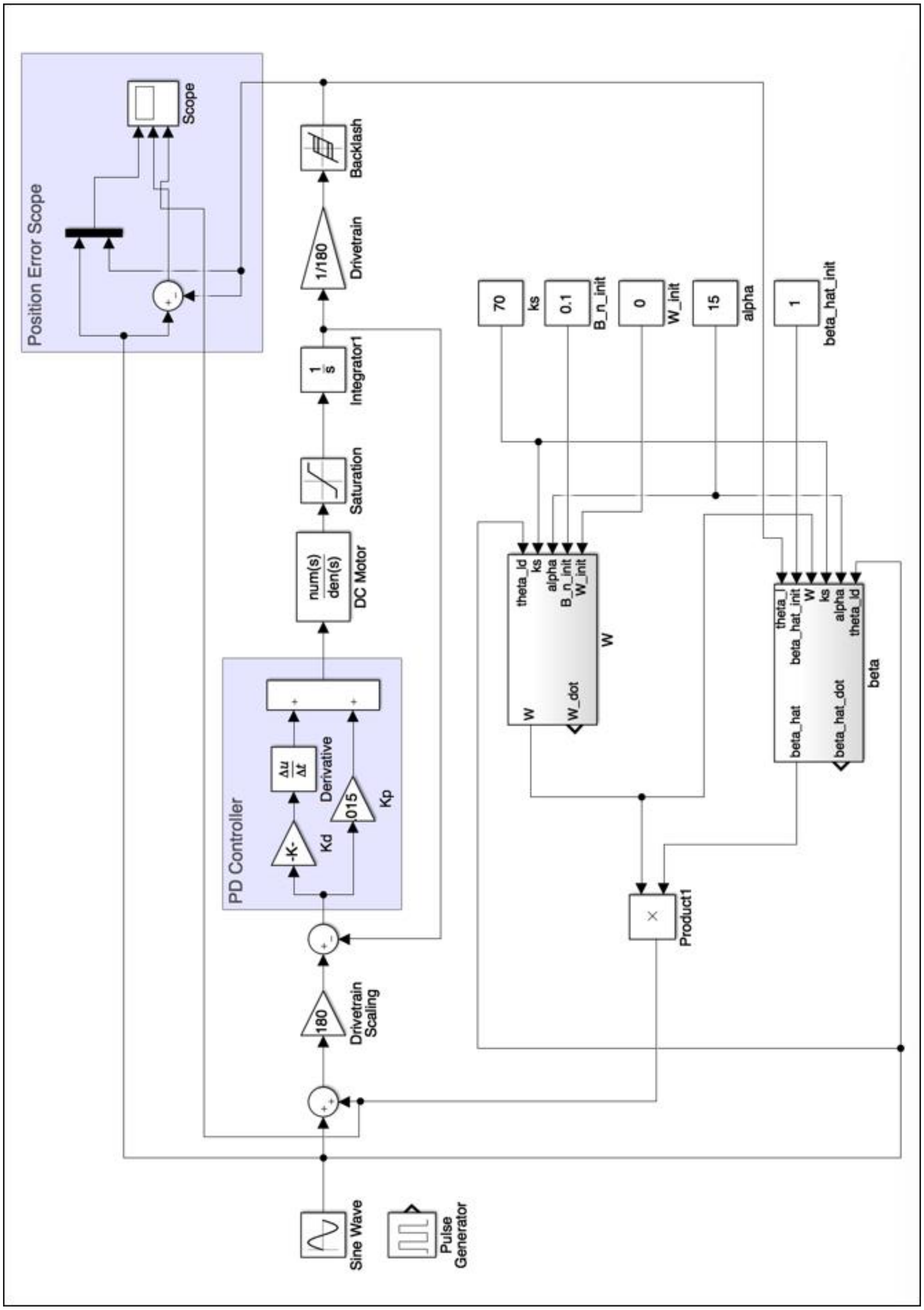


Figure 76: Adaptive Backlash Inverse controller in Simulink

This controller was also implemented in Simulink as shown in Figure 76. Once the controller was tuned to stability in simulation, it was transferred to the test bench, and driven with the same 1 rad/s, 2 rad amplitude sine wave input. Results from this test may be seen in Figure 77.

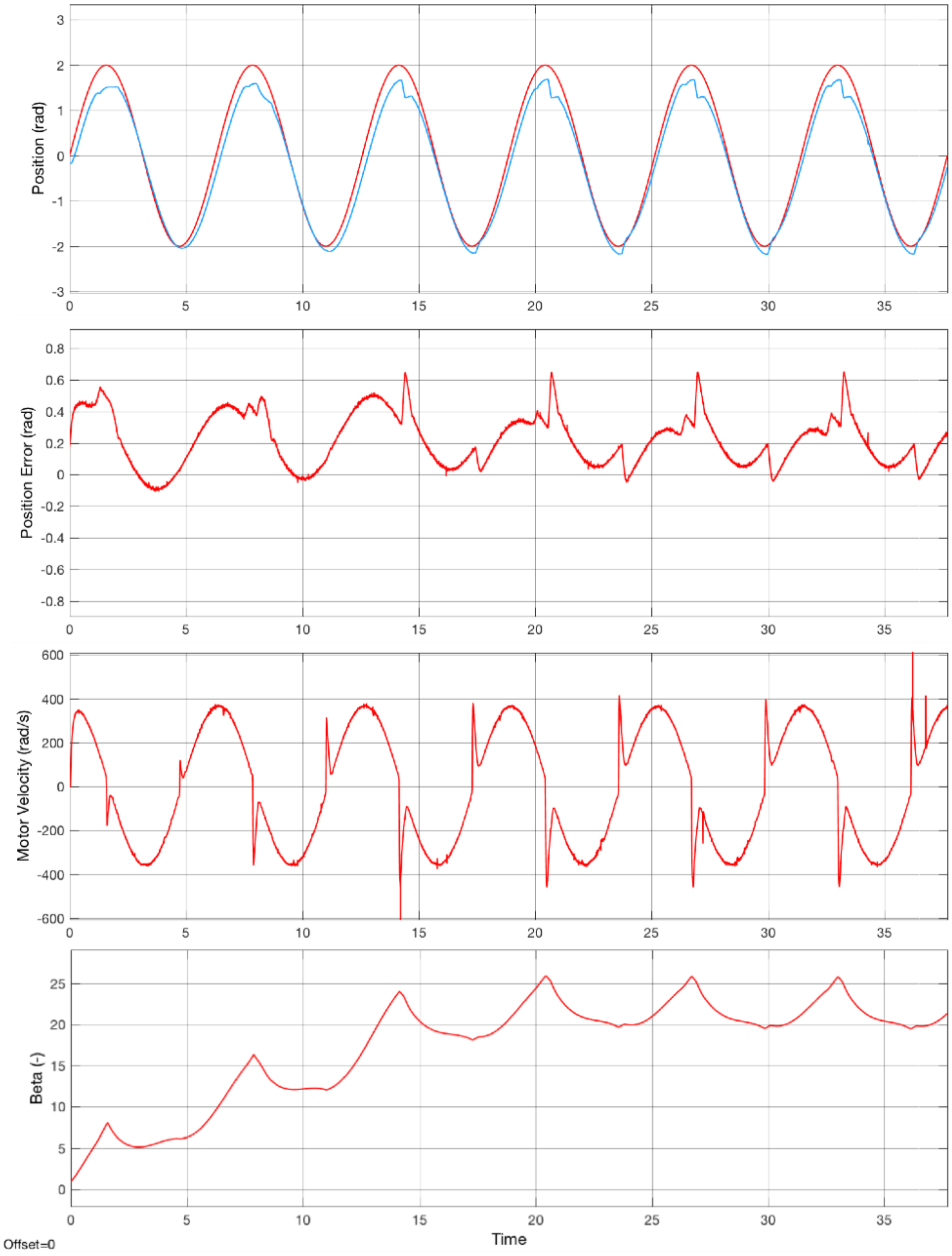


Figure 77: Adaptive Backlash Inverse controller response on test bench

While tracking performance of this controller was smoother across the majority of the trajectory (except for when the backlash gap is traversed), tracking accuracy was substantially poorer, thanks to a DC offset present at the beginning of the test. Since this controller does not implement any sort of integral action except for adjustments for backlash compensation, it is unable to eliminate this offset. This was not discovered initially during simulation because default simulation parameters always assume that the input and output positions initialize at zero. It is a serious problem with this control architecture in practical applications: it both fundamentally limits tracking performance, and limits the effectiveness of the adaptive architecture, as the adaptation law has trouble converging. Additionally, the ABI controller has a tendency to overestimate the half-backlash distance in the system, leading to overshoot.

With the knowledge gained from these experiments, a new control architecture incorporating the best features of both the IDL and ABI controllers is proposed, creatively titled the "integral-adaptive backlash inverse" (IABI) controller. The IABI controller architecture is shown in Figure 78:

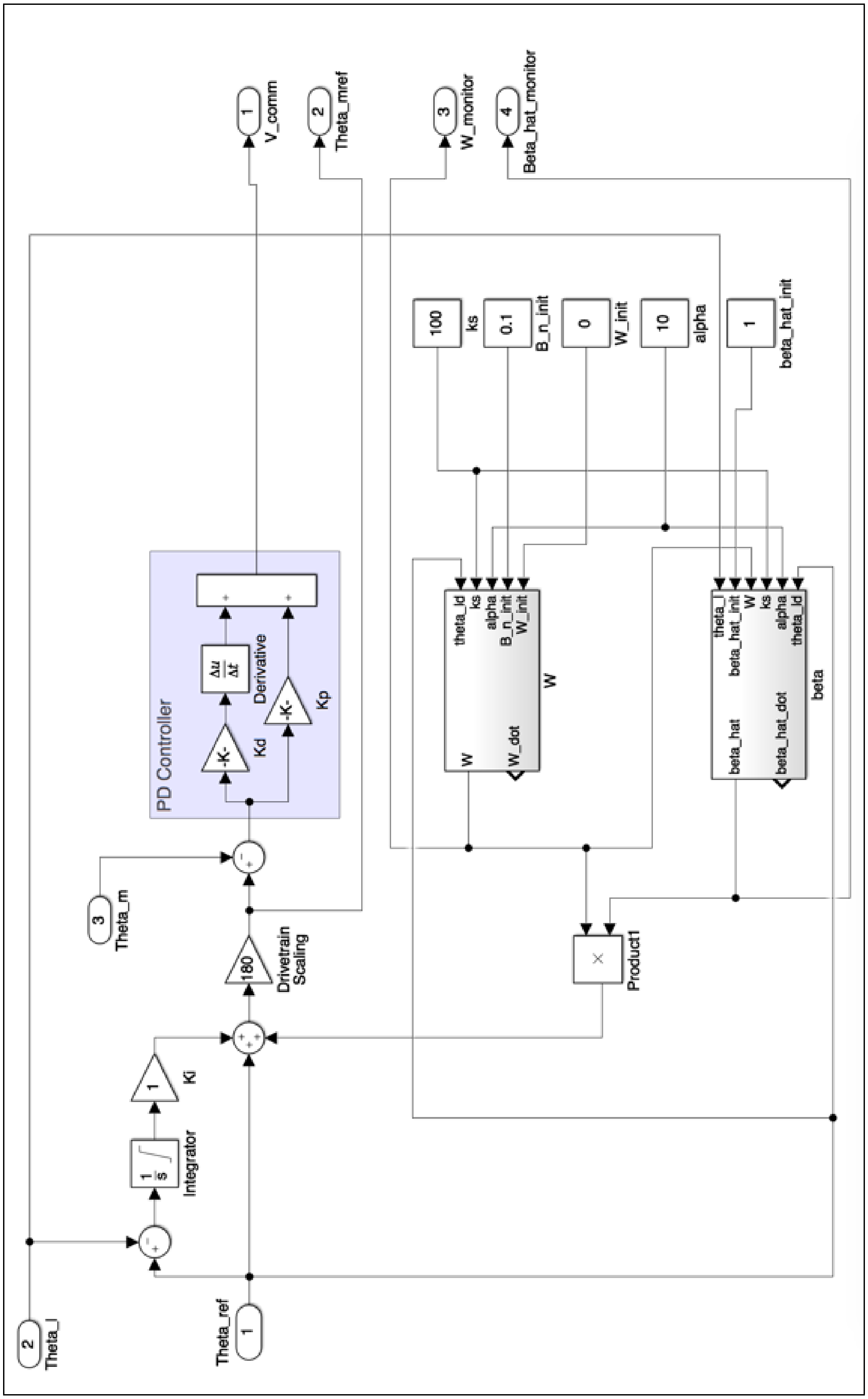


Figure 78: Integral-Adaptive Backlash Inverse controller in Simulink

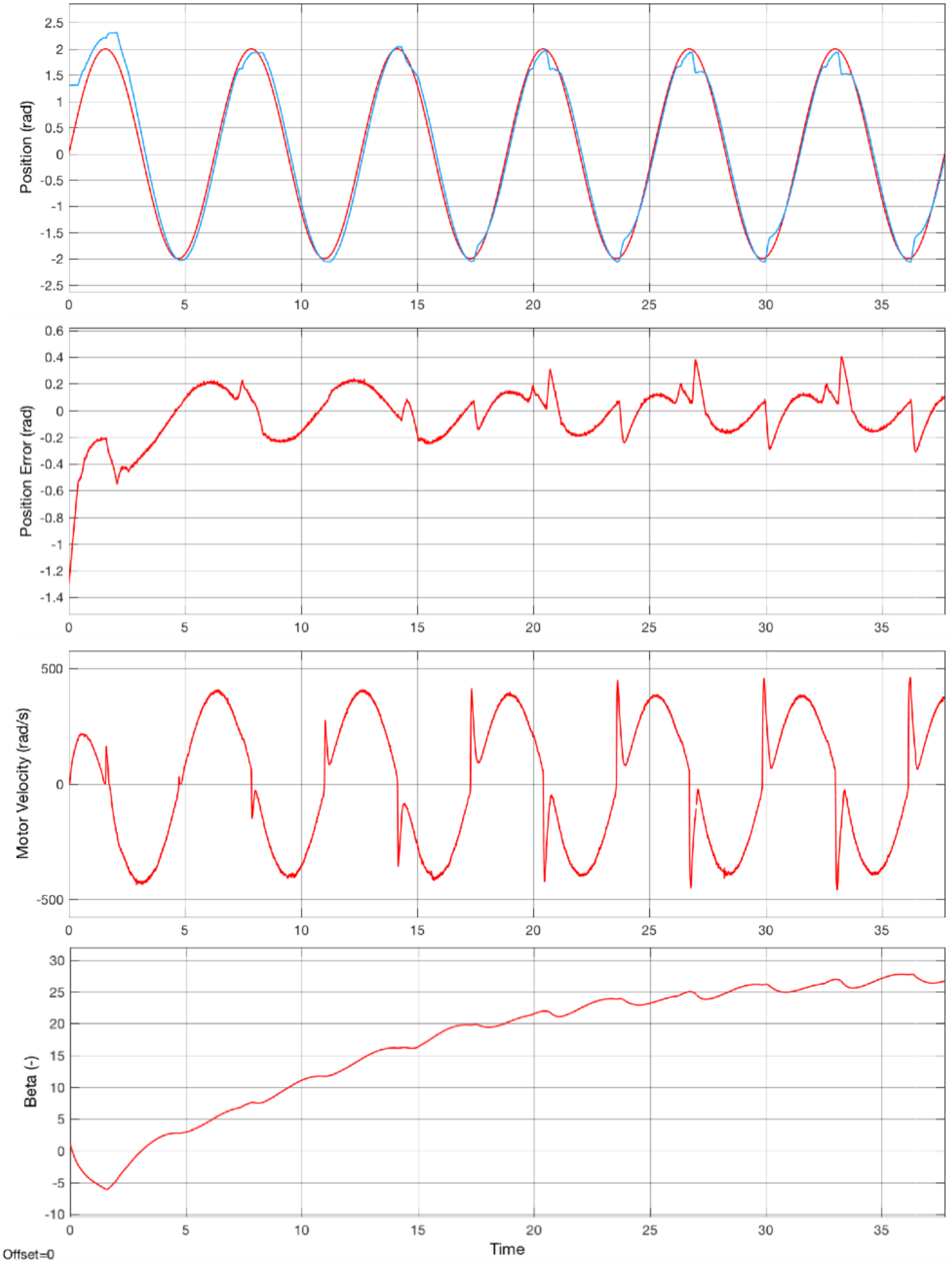
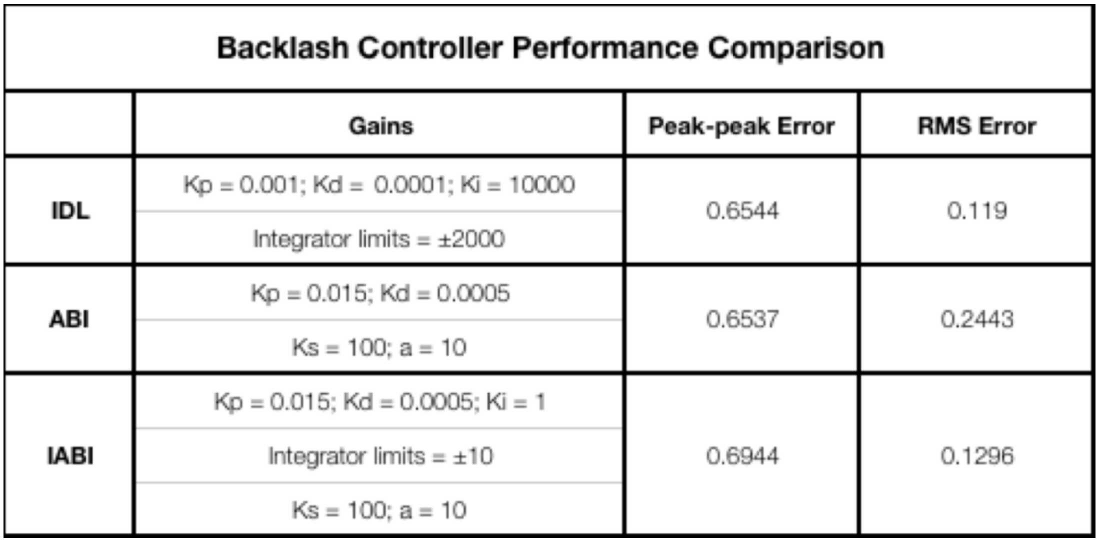


Figure 79: Integral-Adaptive Backlash Inverse controller response on test bench

The IABI controller simply incorporates a “gentle” integral on load position error in parallel with the backlash adaptation term to address position errors that occur when the system is not changing direction and the adaptive component is not active. The performance of this controller is shown in Figure 79. The new control architecture satisfactorily compensates for a large initial position offset, and position tracking is achieved. However, the adaptive controller again substantially overestimates the backlash in the system.

While all three controllers were able to track the reference sine input signal with reasonable success, a variety of unexpected behaviors were observed, including the poor ability of the adaptive controller to respond to DC offsets; and the inertia-driven hysteresis behavior observed in the adjustable backlash element. To quantitatively compare the three controllers, the peak-to-peak difference and RMS values for the error signal were calculated.



Backlash Controller Performance Comparison

	Gains	Peak-peak Error	RMS Error
IDL	Kp = 0.001; Kd = 0.0001; Ki = 10000 Integrator limits = ±2000	0.6544	0.119

ABI	Kp = 0.015; Kd = 0.0005
	Ks = 100; a = 10
		0.6537	0.2443

IABI	Kp = 0.015; Kd = 0.0005; Ki = 1
	Integrator limits = ±10
	Ks = 100; a = 10
		0.6944	0.1296

Table 6: Backlash controller performance comparison

As these results show, by these metrics, there is actually not substantial difference between the three controllers. The IABI controller provides improvement over the ABI controller in terms of RMS error thanks to its ability to address DC offset errors.

However, it performs slightly less well than the IDL controller in terms of both peak-to-peak and RMS error. It is important to recognize that these metrics do not capture important components of performance such as control smoothness/control reversal, robustness across differing trajectories, and ability to adapt rapidly to changing backlash amounts. Particularly if the IABI backlash controller's tendency to overestimate the backlash distance can be successfully addressed, we believe that this controller could prove to be a robust, easily implemented method for addressing backlash. Finally, in order to successfully implement these controllers on the DCP, it will need to be determined whether the DCP behaves as a friction-driven backlash as believed, or if it is in fact an inertia-driven backlash: if so, then alternative controller architectures may need to be considered.

While the techniques described here will require further effort to implement usefully on the DCP, they provide a roadmap for further controls engineering development with the system. Some of the techniques explored – specifically, the use of adaptive techniques to mitigate the hard nonlinearities like backlash – offer particular promise, and may be adaptable to other control challenges in the system such as changing valve deadband.

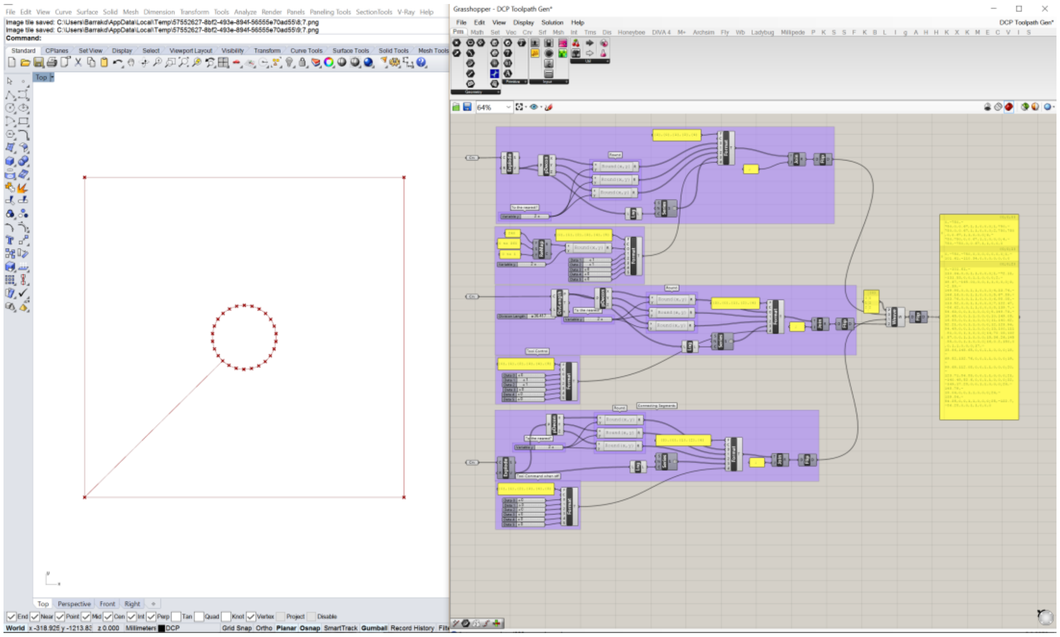
3.2.4 Toolpath Generation Architecture Improvements

The final major area of improvement during the 2016-2017 development period was in the MATLAB toolpath generation architecture the DCP system uses. The architecture developed during work 2015-2016 was relatively complete, with robust trajectory generation and forward and inverse kinematics routines. However, there was no standardized method for using these routines. New toolpath generation routines were frequently written from scratch to support new tasks, with limited standardization between routines and their outputs. Finally, the improvements made to the rest of the DCP system during this period – particularly the system's increased modularity, and the improvements to the KUKA control interface – made it even more important to provide a toolpath generation architecture that was standardized but still adaptable to new tasks.

While improvement of the toolpath generation architecture was an iterative process conducted over the entire 2016-2017 development period, it has yielded a reasonably stable final product that we believe is sufficiently adaptable to support a moderate range of new applications for the system, but also accessible enough that users with minimal MATLAB and robotics experience can use the system. We here provide a simple walkthrough demonstrating the process of importing a toolpath generated by an external source (in this case, Rhino/Grasshopper) and converting it to a dcpctrl_v2-compatible Simulink trajectory using this architecture.

Toolpath Generation & Import

The dcpctrl_v2 toolpath generation architecture is designed to allow toolpaths to be designed in a wide range of applications and imported into MATLAB. Here, we show a simple toolpath generated in the Rhino/Grasshopper parametric CAD design environment, using a Grasshopper script developed by Barrak Darweesh[20].



```
Image tile saved: C:\Users\Barrakd\AppData\Local\Temp\57552627-8bf2-493e-894f-56555e70ad55\8;7.png
Image tile saved: C:\Users\Barrakd\AppData\Local\Temp\57552627-8bf2-493e-894f-56555e70ad55\9;7.png
Command:
```

File Edit View Curve Surface Solid Mesh Dimension Transform Tools Analyze Render Panels Paneling Tools SectionTools V-Ray Help
Standard CPlanes Set View Display Select Viewport Layout Visibility Transform Curve Tools Surface Tools Solid Tools Mesh Tools
Top
Grasshopper - DCP Toolpath Gen*
File Edit View Display Solution Help
DCP Toolpath Gen*
64%
Top Perspective Front Right
End Near Point Mid Cen Int Perp Tan Quad Knot Vertex Project Disable
World Millimeters DCP Grid Snap Ortho Planar Osnap SmartTrack Gumball Record History

Figure 80: Rhino/Grasshopper toolpath generation script. Image credit: Barrak Darweesh.

[20] Tim (Yen-Ju) Tai also supported early work on a similar script.

Toolpath generation in the dcpctrl_v2 architecture uses the concepts of segments originally developed in the dcpctrl_v1 architecture to break up toolpaths. Segments generally correspond to specific elements of a toolpath: a continuous section of motion, for example, or a data-gathering operation. Because of the improvements made to the DCP control architecture, multiple subsystems (AT40GW, KUKA, tool, etc.) can be operated in parallel within a given segment, obviating the need to use segments to perform operations like turning off a tool after an AT40GW move has completed. However, segments still have some utility in command generation, providing an easy separation mechanisms for operations that should be performed with different motion parameters, for instance, or created using different trajectory generation rules.

Determination of what makes up a "segment" is the responsibility of the user. In the Rhino/Grasshopper script used here, segments are defined by individual curves in Rhino: the outer square is one curve, the circle in the middle is another, and the line that connects them is a third. Each of these segments is then broken up in to some number of waypoints (although a segment must contain at least two waypoints to be complete), in the following format:

$$\begin{aligned}
&[t_0, x_0, y_0, z_0, h_0^1, h_0^2, h_0^3, h_0^4, h_0^5, h_0^6];\\
&[t_1, x_1, y_1, z_1, h_1^1, h_1^2, h_1^3, h_1^4, h_1^5, h_1^6];\\
&\ldots\\
&[t_n, x_n, y_n, z_n, h_n^1, h_n^2, h_n^3, h_n^4, h_n^5, h_n^6];\text{ where}\\
&t = \text{ waypoint timestamp}\\
&x, y, z = \text{ desired absolute endpoint position at that time}\\
&h^1 \ldots h^6 = \text{ desired tool state at that time}
\end{aligned}$$

This toolpath can then be exported as a text file, with waypoints delimited using semicolons, and segments delimited using newline characters.

This toolpath format is intended to provide a starting point for basic operations with the DCP. It does not provide a number of important functions, including support for more than six tool channels; the ability to define endpoint orientation; or the ability to define parameters of how a toolpath should be segmented (for instance, whether it should be commanded to the AT40GW or the KUKA). However, it is sufficient for light painting and for simple Print-in-Place fabrication tasks, and provides a template for expansion to include new functionality.

Once the toolpath has been created by the user's external program, it needs to be imported into MATLAB and converted to an appropriate format. In this example, this is done with the script rhinoimporter.m (which in spite of its name, can be used to import any toolpath meeting the above formatting requirements). The waypoints are extracted from the text file and converted to a MATLAB cell format, where each cell corresponds to a segment, and contains a N-by-10 matrix of the waypoints that make up that segment. This cell is saved with the variable name *waypts* in the MATLAB workspace.

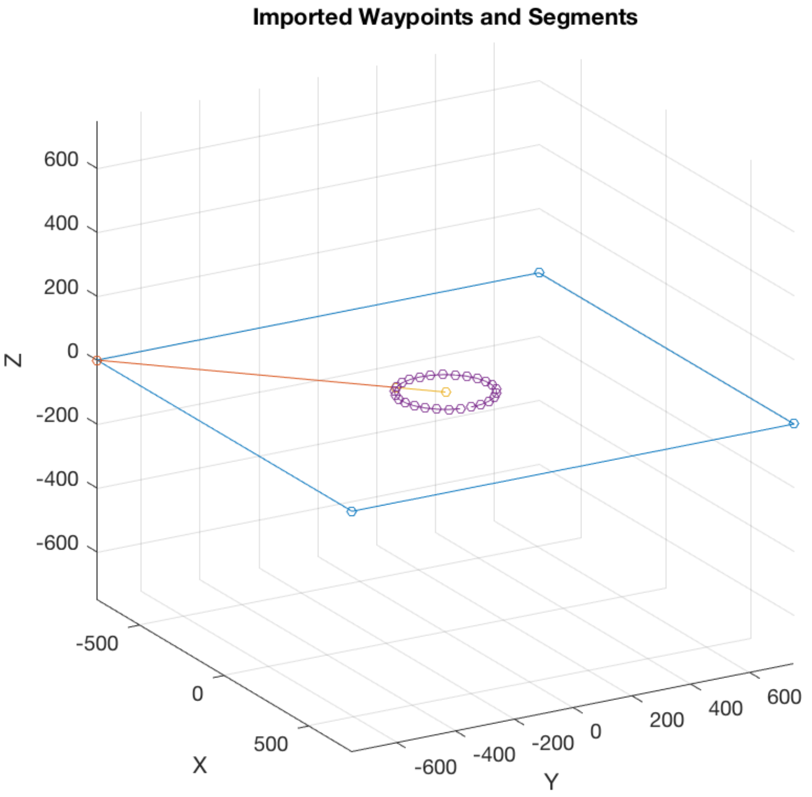
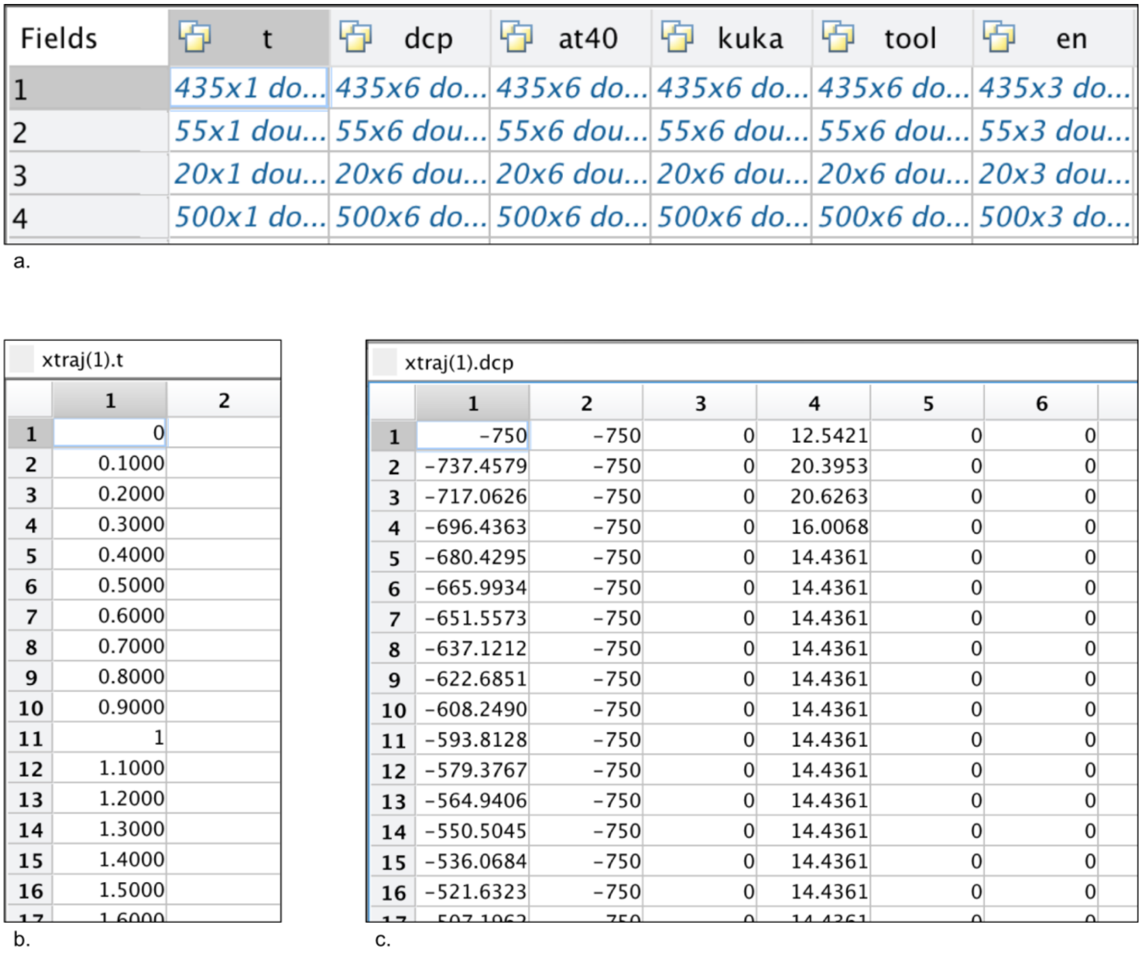


Figure 81: Waypoints imported from text file using rhinowayptimport. Line colors indicate segments.

waypt2xtraj

With toolpath segments and waypoints imported into MATLAB, the next step is to convert them to a task-space, time-parametrized trajectory, called a *xtraj*. This is done using the same fundamental trajectory generation tools developed previously by Cai (using elements from the RVC toolbox [60]), with a few notable differences:

- Toolpath format: At this stage in the toolpath generation process, the toolpath begins to take on its final format. Task-space trajectories are implemented in MATLAB using the structure datatype, with rows corresponding to segments, and named columns corresponding to specific trajectory components. At this stage, there are six trajectory components defined:
    - t: N-by-1 timestamp trajectory. Timestep resolution may be defined in waypt2xtraj, but is generally set to 0.1 s.
    - dcp: N-by-6 desired endpoint trajectory for the complete DCP system. Components are [x,y,z,dx,dy,dz].
    - at40: N-by-6 desired endpoint trajectory for the AT40GW, assuming the KUKA is in its default home position (acting as a static tool offset). The at40 and dcp trajectories are not necessarily identical, as explained below. Components are [x,y,z,dx,dy,dz].
    - kuka: N-by-6 desired endpoint trajectory for the KUKA endpoint, defined relative to its starting point. Components are [x,y,z,dx,dy,dz].
    - tool: N-by-6 desired tool trajectory.
    - en: N-by-3 ENABLE trajectory, providing ENABLE signals for the AT40GW, KUKA and tool. This channel is useful for operations where it may be desirable to turn off one subsystem while the others operate – for instance, stopping the AT40GW from moving while the KUKA performs a precision operation.



Fields	t	dcp	at40	kuka	tool	en
1	435x1 do...	435x6 do...	435x6 do...	435x6 do...	435x6 do...	435x3 do...
2	55x1 dou...	55x6 dou...	55x6 dou...	55x6 dou...	55x6 dou...	55x3 dou...
3	20x1 dou...	20x6 dou...	20x6 dou...	20x6 dou...	20x6 dou...	20x3 dou...
4	500x1 do...	500x6 do...	500x6 do...	500x6 do...	500x6 do...	500x3 do...

a.

xtraj(1).t

	1	2
1	0	
2	0.1000	
3	0.2000	
4	0.3000	
5	0.4000	
6	0.5000	
7	0.6000	
8	0.7000	
9	0.8000	
10	0.9000	
11	1	
12	1.1000	
13	1.2000	
14	1.3000	
15	1.4000	
16	1.5000	
17	1.6000	

b.

xtraj(1).dcp

	1	2	3	4	5	6
1	-750	-750	0	12.5421	0	0
2	-737.4579	-750	0	20.3953	0	0
3	-717.0626	-750	0	20.6263	0	0
4	-696.4363	-750	0	16.0068	0	0
5	-680.4295	-750	0	14.4361	0	0
6	-665.9934	-750	0	14.4361	0	0
7	-651.5573	-750	0	14.4361	0	0
8	-637.1212	-750	0	14.4361	0	0
9	-622.6851	-750	0	14.4361	0	0
10	-608.2490	-750	0	14.4361	0	0
11	-593.8128	-750	0	14.4361	0	0
12	-579.3767	-750	0	14.4361	0	0
13	-564.9406	-750	0	14.4361	0	0
14	-550.5045	-750	0	14.4361	0	0
15	-536.0684	-750	0	14.4361	0	0
16	-521.6323	-750	0	14.4361	0	0
17	-507.1962	-750	0	14.4361	0	0

c.

Figure 82: xtraj variable structure (a) – time vector (b) and example trajectory vector (c)

- <u>“True” toolpath creation</u>: In addition to creating a smooth, polynomial-blended trajectory for the AT40GW using waypts2carttraj and mstraj2, waypt2xtraj also creates a non-smooth, “true” toolpath for the DCP as a whole, which provides a time-parametrized path between the waypoints provided in the input file. This can be seen in Figure 83 below, where the colored path represents the path commanded to the AT40GW, and the black path represents the “true” path assigned to the DCP[21].

[21] In the current version of waypt2xtraj, the user is assumed to be performing a light painting process. Consequently, the color and size of trajectory points displayed in this plot are correlated to the hue and brightness values commanded in the tool trajectory at those points.

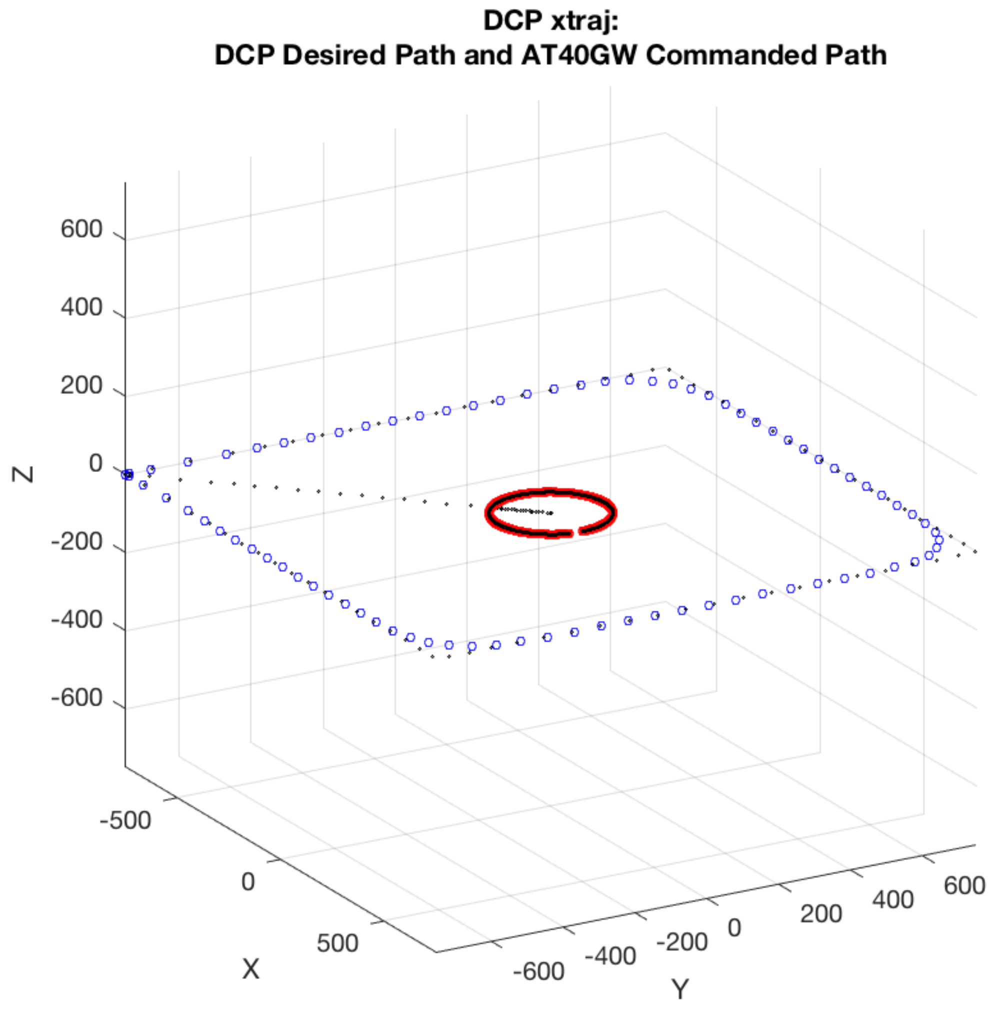


Figure 83: Completed xtraj, showing "true" DCP trajectory and calculated AT40GW trajectory

This “true” trajectory is created by mapping each point in the AT40GW trajectory onto the path defined by the waypoints, as a function of percentage distance along the complete path. While inelegant, this creates a trajectory between the original waypoints that moves through space at roughly the same rate as the AT40GW waypoints. This trajectory is then saved into the dcp element of the *xtraj* variable. It can then be used for a variety of purposes, including as a reference signal in the real-time KUKA error compensation method described in Section 3.2.1.

- Toolpath segmentation: The current version of waypt2xtraj also incorporates rudimentary toolpath segmentation, for commanding the DCP in the serial or parallel operation modes described in Chapter 2. More detail on the use of these tools is given below in Section 3.3.2.

- Constant velocity: Finally, in the current version of waypt2xtraj, the time values provided by the user in their input toolpath are ignored, and a constant velocity defined at the beginning of the waypt2xtraj script is used instead (along with user-defined timestamp and acceleration time values). The underlying tools used to create the AT40GW trajectory – specifically, mstraj2 – are able to accept time vectors as inputs when generating toolpaths between waypoints, but more development is required to enable this functionality.

xtraj2qtraj

The task-space trajectory, or *xtraj*, is then passed to the script xtraj2qtraj, where the DCP's inverse kinematics solvers are used to translate the trajectory into a joint-space *qtraj*.

- Toolpath format: The *qtraj* is also defined as a MATLAB structure variable. It is largely similar to *xtraj*, with the following exceptions:
  - The at40 trajectory is now defined in AT40GW joint space, and is N-by-8 (four joint position trajectories, and four joint velocity trajectories).
  - The kuka trajectory is transformed into kuka_x, and a new trajectory called kuka_q – a joint-space trajectory for the kuka is created. This trajectory is N-by-12, with position and velocity trajectories for each of the KUKA's six joints. While this trajectory is currently never used/not calculated, it has been implemented here to support future joint-space control tasks.
  - The en trajectory has an extra column added, corresponding to the kuka_q trajectory. This could enable the user in the future to selectively operate the KUKA in task-space and joint-space for different segments within a toolpath.

- Toolpath location: At this point, the toolpath is also located within the DCP's work volume. The toolpath may be located with the first waypoint: a) at the current location of the DCP's endpoint; b) at the location of the DCP's endpoint at a defined system pose; or c) at an arbitrary X,Y,Z location in the DCP's workspace, measured relative to the base coordinate frame.

- Updated kinematics: During work 2016-2017, the DCP DH parameter model was updated to include the AT40GW "wrist" linkage (comprising J5 and J6), as well as the KUKA (treated as a constant tool offset, with the KUKA in its default "home" position. The representation of these joints in the DCP DH parameter model may be seen in Section 2.1.1. With these new DH parameters, a new set of functions for the forward kinematics transformation and the system Jacobian were derived as described in Chapter 2, enabling this improved model to be used for toolpath generation.

  It is important to note that in this model, the KUKA is still treated as a fixed tool offset rather than as an articulated robotic system, and motion of the KUKA is not accounted for by the inverse kinematics solver. Thus, it is the user's responsibility to ensure that any motions commanded to the KUKA are defined appropriately, since the KUKA's reference frame changes orientation with respect to the DCP base reference frame as the J1 joint rotates. This is a relatively simple kinematic relationship to manage (for instance, as the kuka_errorComp block described above does), but future users may wish to incorporate this functionality directly into xtraj2qtraj.

Finally, the script also checks for exceeded joint limits in the AT40GW (although not exceeded joint limits – or RSI position correction limits – for the KUKA). Once this has been completed, the *qtraj* is complete and ready for translation into a Simulink-compatible format.

qtraj2slxtraj

The last step in toolpath import is to convert the joint-space, time-parametrized trajectory contained in the *qtraj* structure into a format that Simulink can manipulate. This is done using the MATLAB timeseries object, a data storage object specifically designed for storing regularly sampled data. While the timeseries object has a wide range of capabilities (including storing data quality information, data units, and more), its primary value in the dcpctrl_v2 architecture is that it can be imported into Simulink using the From Workspace block, with the timestamp information still attached.

This conversion process is performed by the MATLAB qtraj2slxtraj script. This script first consolidates all segments defined in each field of *qtraj* into a single segment. As part of this process, it also homogenizes the time trajectories for each segment, ensuring that the first timestep of a segment falls after the last timestep of the preceding segment. Each combined trajectory field is then converted into a timeseries object, with one timeseries created per field (dcp, at40, kuka_x, etc.) and the homogenized time trajectory used as the time reference for each object. Some basic error checking is performed to ensure that all timeseries created are the same length (although the content of those timeseries is not examined). Finally, the timeseries trajectory – now known as a *slxtraj* – may optionally be visualized. The visualization used is from the dcpctrl_v1 toolchain, and is not well-adapted to use in this manner: the behavior of the DCP model is not accurate to how the system actually performs, and the model cannot represent KUKA moves or tool operations.

At this point, the toolpath is complete and ready for execution in a dcpctrl_v2 controller, such as dcpController. While the trajectory generation and kinematic solution techniques at the core of this toolchain have not been altered from how they were implemented in dcpctrl_v1, we believe that the system described here provides substantial improvements in the accessibility and adaptability of the DCP's toolpath generation architecture. For users who are satisfied with the functionality provided by this toolchain, the straightforward toolpath input format enables almost any computational package to be used with the DCP. Meanwhile, by leaving the system open and implemented in MATLAB, users desiring more advanced functionality may still interact with the DCP trajectory at the *waypts*, *xtraj*, or *qtraj* level. We hope that this architecture provides both ease of use and adaptability, and we look forward to seeing how it is leveraged by future users.

3.3 System Implementations, 2016-2017

While the focus of work 2016-2017 has been heavily weighted towards system improvements rather than implementations with the system, a few experiments have been conducted that demonstrate how improvements in the system architecture and hardware translate into expanded capabilities. Here, we present a series of light paintings and other experiments conducted over this period that showcase some of the improvements made to system performance and capability; as well as reporting results from a ISO 9283 pose repeatability test conducted at the end of the development period.

3.3.1 Advanced Light Paintings

Throughout development 2016-2017, light paintings have continued to provide a valuable tool for examining how well the DCP performs as an architectural-scale robotic system. In addition to providing a visualization of the system's tracking performance, they also can showcase some of the new capabilities added to the DCP system.

One of the most visually impactful improvements made to the DCP system during this development period has been the incorporation of real-time tool control. For light paintings, the development of tool interfaces like the LIFX bulb controller described above in Section 3.2.1 dramatically increased the range of images and effects that the DCP can produce through real-time control of light brightness, hue and saturation.

An early example of this can be seen in Figure 84, below. Here, different light colors were assigned to each segment of the ISO 9283 trajectory. As the DCP executed this trajectory, the light changed color, indicating to the user which segment of the task the DCP controller was currently executing.

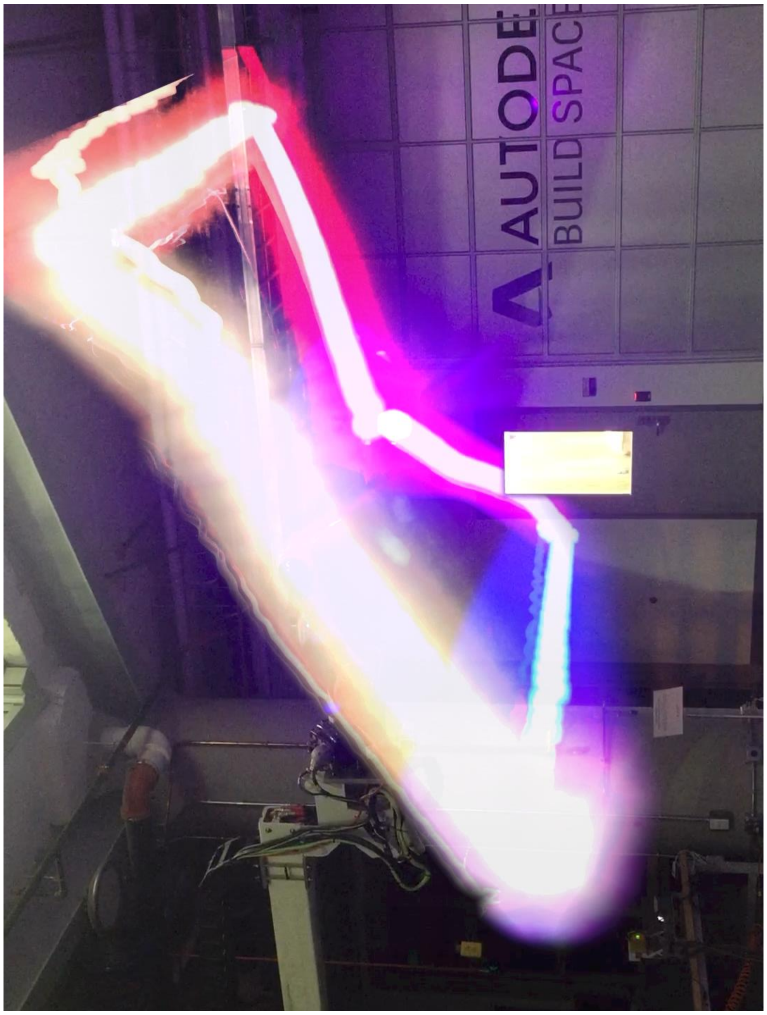


Figure 84: Light painting of ISO 9283 pose repeatability test trajectory

A substantially more impressive example of how this functionality can be used is shown in Figure 85[22]. Here, two custom trajectory generation routines – img2rasttoolpath and img2vecttoolpath, available in the dcpctrl_v2 repository – are used to create both vector and raster representations of a color input image. While the vector toolpath is monochromatic throughout the entire toolpath, the raster toolpath generator extracts color information in the HSV color space at each point along the trajectory, and commands this to the LIFX bulb.

[22] It should be noted that this image was created in March 2017, using a less developed controller than dcpController, and without the benefit of the new sensors described above, leading to the poor tracking performance observed in the vector painting.

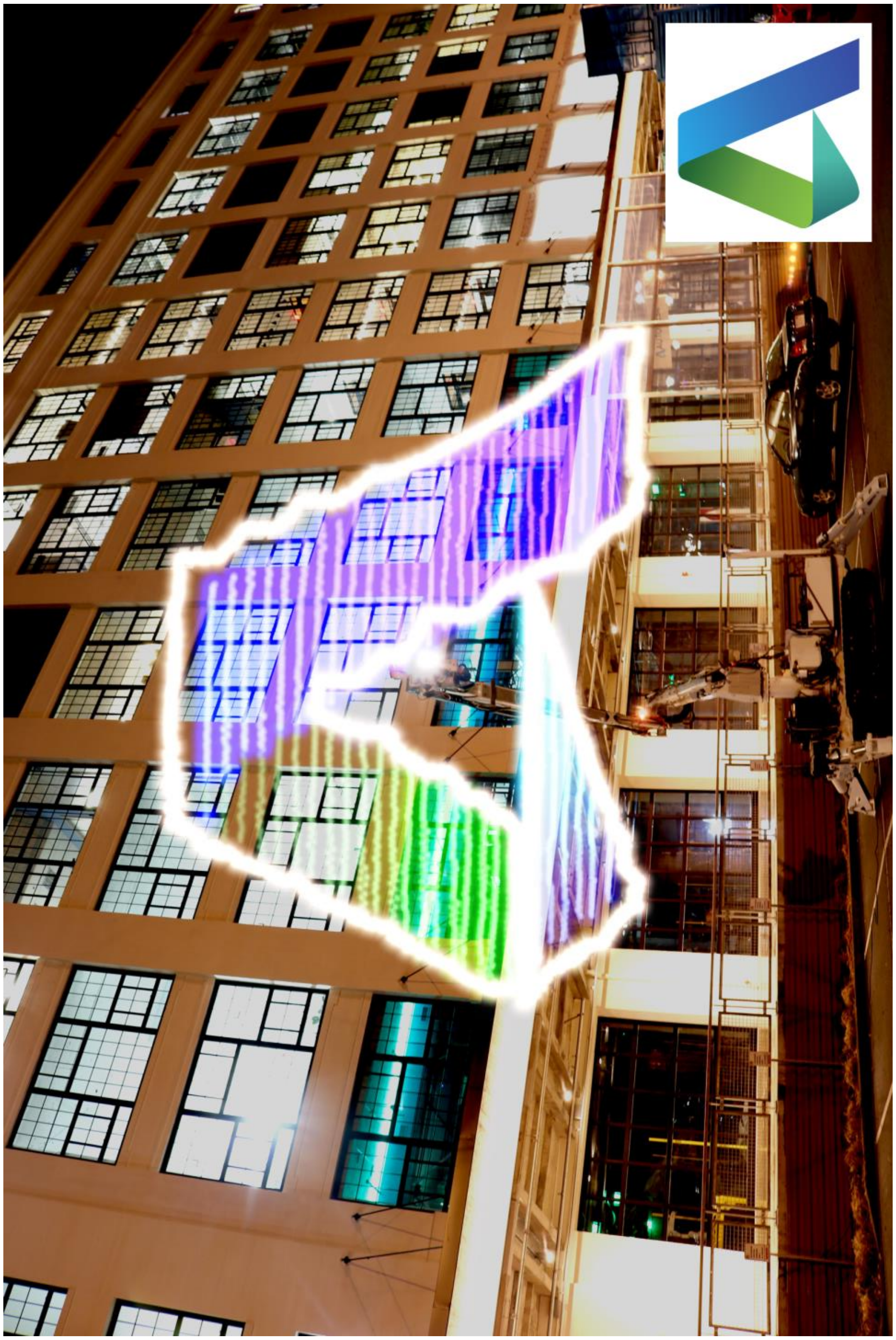


Figure 85: Light painting of Autodesk “A” logo, with original logo for color reference. Original logo: https://upload.wikimedia.org/wikipedia/commons/b/b5/Autodesk_Logo.svg. Accessed 2017-08-10.

In Figure 85, a 6 m x 6 m painting of the Autodesk “A” logo was created. The combined vector and raster trajectories took a total of 52 min to complete, and were executed sequentially. Because of the brightly lit background (the Innovation and Design Building complex in the Boston Seaport, where the Autodesk BUILD Space is located), the light painting had to be extracted from video footage and assembled using star-trail stitching, reducing the quality of the final image. However, the smooth color gradients that the dcpctrl_v2 architecture enables are clearly visible in the image, and yield a reasonably accurate reproduction of the original image. Particularly for future artistic explorations with the DCP, or environmental data visualization applications, the ability to produce these visual effects is invaluable.

3.3.2 DCP Multi-Modal Control & Toolpath Segmentation

Another valuable feature of the dcpctrl_v2 architecture is the ability to fully support the three different operation paradigms described in Section 2.1.2, both by segmenting toolpaths between the AT40GW and KUKA, and by using the KUKA to compensate in real-time for errors in AT40GW motion. As a brief review, the three operation paradigms are: 1) serial operation, where the KUKA and AT40GW execute toolpaths sequentially; 2) error compensation operation, where the AT40GW executes a toolpath while the KUKA responds only to correct errors in position tracking; and 3) parallel operation, where the AT40GW and KUKA both execute toolpaths, with the KUKA optionally also providing error compensation.

Serial, Parallel and Error Compensation Mode Implementation

The improvements made to the DCP’s toolpath generation and control architecture have made controlling the DCP in a serial operation mode substantially easier. All trajectories by default support controlling both the AT40GW and KUKA, and the inclusion of the en (enable) trajectory makes it easy to activate or deactivate the AT40GW and KUKA on a per-segment basis. Furthermore, the improved robustness of the KUKA controller, and its implementation in the same control architecture that operates the AT40GW, makes switching between the two systems much more reliable.

While serial operation has been made more robust and easier to use by the dcpctrl_v2 architecture, this operation mode was generally functional in dcpctrl_v1. However, the second and third operation modes – error compensation operation and true parallel operation – were largely unavailable under dcpctrl_v1. Error compensation operation was implemented in a limited fashion by Keating (described in [3] and in Section 2.1.2), by using a laser distance sensor to regulate KUKA Z position relative to the ground. However, this compensation operated only in a single axis, and did not interact with the rest of the DCP's control system. Likewise, parallel operation was demonstrated conceptually in light paintings by having the KUKA and AT40GW execute trajectories independently, although again there was no coordination between the two systems' controllers.

The dcpctrl_v2 architecture enables both of these operation modes in far more sophisticated and useful ways. Error compensation operation can be implemented directly in dcpController, using blocks like the kuka_errorComp block described previously, and position feedback taken from the AT40GW's joint sensors or other external references. Parallel operation is also easily implemented, using either precomputed toolpaths for the KUKA, or even just by feeding the difference between the "true" dcp trajectory and the smoothed at40gw trajectory to the KUKA in real-time. Finally, these techniques can be implemented simultaneously, with the KUKA both executing its own toolpaths and providing error compensation for the AT40GW.

Implementation Examples

Figure 86 thru Figure 89 present a series of example light paintings showing how this has been implemented on the DCP[23]. Here, a simple toolpath, consisting of a 1.5 m x 1.5 m square in the X-Y plane with a 300 mm circle inside of it, is generated using the Rhino/Grasshopper script developed by Darweesh and imported into MATLAB. A task-space trajectory is then

[23] Light paintings were used for this experiment instead of the Vive VR tracking system because the KUKA interferes with the Vive system's functioning, as described further below. While light paintings are a relatively low-resolution medium for showing these improvements, they enabled the KUKA to be used during trajectory execution, and still communicate the differences between the operation modes.

generated from this toolpath with the waypts2xtraj script. The trajectory is segmented and run in four different modes:

- AT40GW only, high acceleration (Figure 86): Here, a trajectory is generated for the AT40GW using typical trajectory settings (150 mm/s speed, 1 s acceleration). The trajectory is reasonably accurate, although the corners of the outer trajectory are imperfect, with both overshoot and round-off observed. This trajectory took 2 minutes to complete, of which 1:10 was required to complete the circle.

- AT40GW only, low acceleration (Figure 87): To make the AT40GW trajectory less aggressive and reduce vibration, the acceleration time used for trajectory generation can be increased. However, this has the effect of increasing the deviation between the desired and actual toolpath at corners. This can be seen in Figure 87, where acceleration time has been increased from 1 second to 6 seconds. Because of the reduced acceleration time, this trajectory takes 8:46 to complete, and the circle – which is still executed by the AT40GW – takes 7 minutes to complete. The endpoint never reaches its desired task-space velocity in the circle, making this method inappropriate for fabrication processes where feed rate is a critical parameter. There is also substantial vertical oscillation of the DCP endpoint as the circle is transcribed, which is not captured in this image.

- AT40GW + KUKA, low acceleration (Figure 88): The errors caused by reducing trajectory acceleration can be mitigated by commanding KUKA corrections using the difference between the dcp trajectory – the "true" trajectory – and the at40 trajectory. This can be done either during toolpath generation, or in real-time through Simulink. The example presented here uses a rudimentary form of toolpath segmentation implemented in waypt2xtraj, where segmentation is performed based on the bounding box volume of a given segment. Segments with a volume above some threshold are pathed normally, while segments with a volume below the threshold are pathed so that the AT40GW is directed to remain stationary during the segment. The KUKA is then commanded by the kuka_errorComp block during real-time execution to compensate for the difference between the "true" trajectory and the at40 trajectory, effectively implementing the segmentation described above, as well as improving tracking during AT40GW segments. The corners of the square have become dramatically sharper, and the KUKA can be seen in the smaller sub-images moving to correct at each corner point. Additionally, because the circle toolpath can be executed at a higher speed by the KUKA, this toolpath only takes 2:17 to complete, with the circle taking 1:02. However, there are still errors in the DCP's movement – visible as steps in the sides of the square – that are not effectively corrected for. The circle trajectory here is imperfect: the KUKA's move carried it to the limits of its allowable correction distance in X and Y, causing the sharp corner in its path.

- AT40GW + KUKA, low acceleration, with real-time correction (Figure 89): Finally, the KUKA can also correct in real-time for errors measured between the at40 trajectory and the AT40GW's current position. Here, the corners of the square are even sharper than in previous tests. Additionally, step errors in the DCP's movement are quickly corrected by the KUKA. Because of the limitations of the AT40GW's sensors – namely, that they cannot capture deflections of the AT40GW links – this technique can currently only respond to errors in joint position. However, the incorporation of additional on-system sensors to measure link deflection, or external sensors to provide an absolute position reference, could make this technique much more powerful in the future.

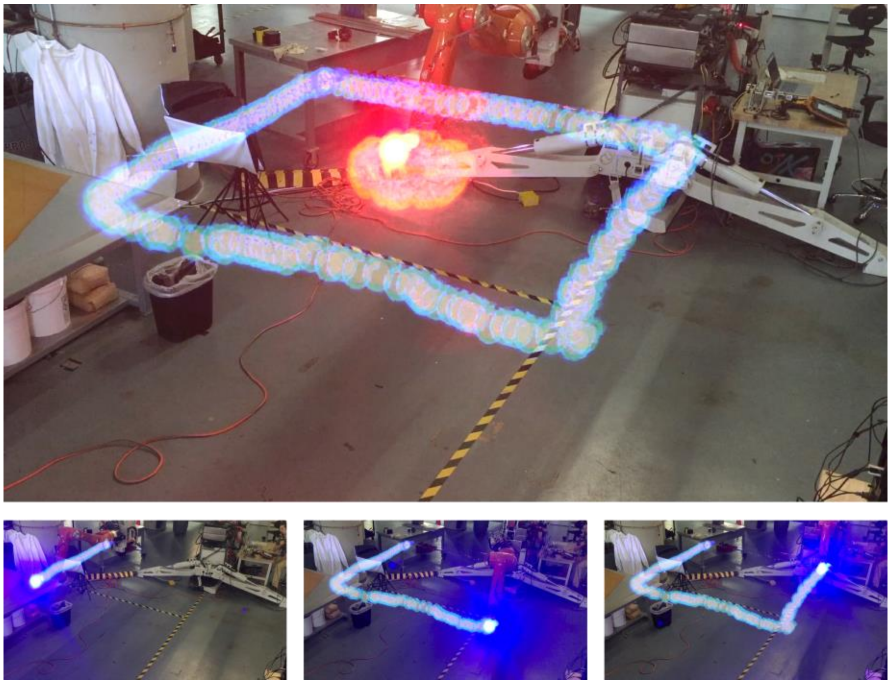


Figure 86: DCP executing square-circle trajectory: 150 mm/s feed, 1 s acceleration, no segmentation. Trajectory time: 2:00.

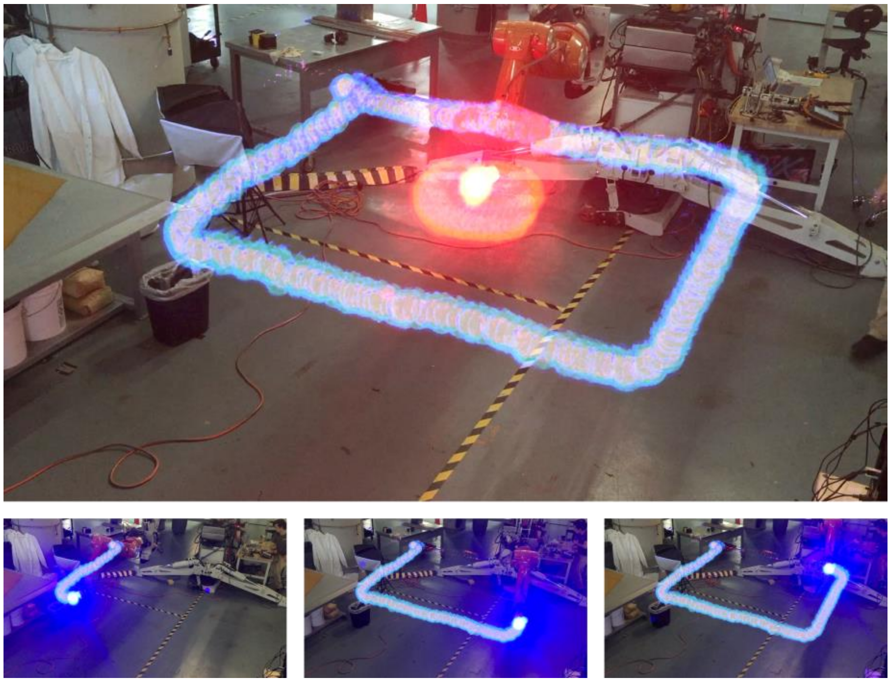


Figure 87: DCP executing square-circle trajectory: 150 mm/s feed, 7 s acceleration, no segmentation. Trajectory time: 8:46.

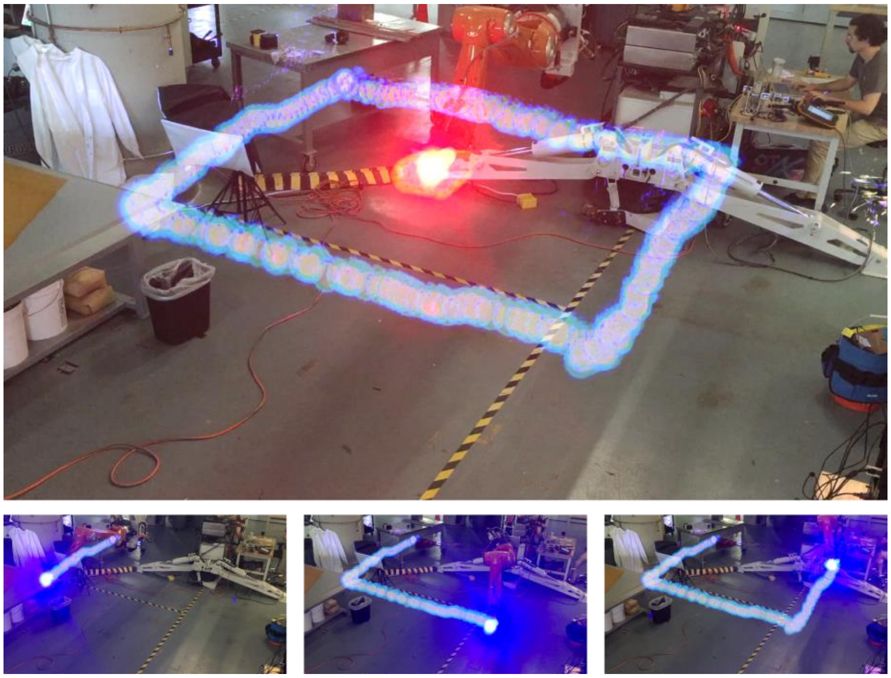


Figure 88: DCP executing square-circle trajectory: 150 mm/s feed, 7 s acceleration, segmentation to KUKA. Trajectory time: 2:17.

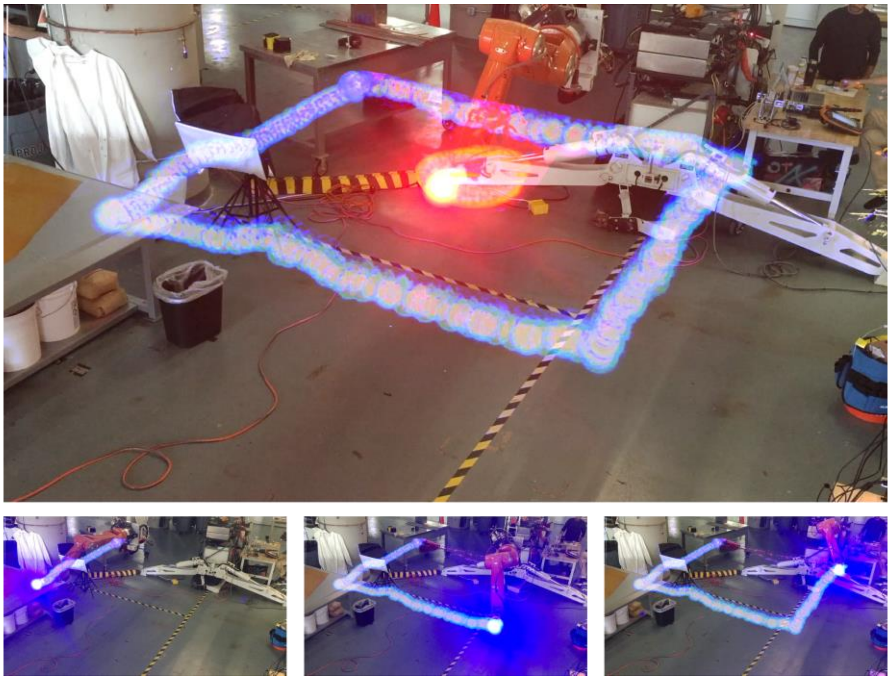


Figure 89: DCP executing square-circle trajectory: 150 mm/s feed, 7 s acceleration, segmentation to KUKA, plus active error compensation. Trajectory time: 2:17.

3.3.3 ISO 9283 Performance Characterization

The pose repeatability test conducted at the end of the 2015-2016 development period provided a valuable benchmark for the DCP's performance operating under the dcpctrl_v1 architecture, and using the 2015-2016 mechatronic configuration. To quantitatively examine the impact of improvements made during the 2016-2017 development period, we returned once again to the ISO 9283 performance characterization standard, re-conducting the pose repeatability test after all system improvements had been completed.

Because of budgetary and time constraints, a true metrology-grade tracking system like the Leica laser tracker used previously was unavailable for this test. Instead, we experimented with using the HTC Vive Tracker virtual reality tracking interface [93], an add-on component for the HTC Vive VR system, for endpoint tracking. The Vive Tracker is designed as a modular interface for incorporating real-world objects in virtual-reality environments by enabling those objects to be tracked in the real world and then drawn in VR. For example, in their developer manual [94], the Vive Tracker is shown mounted to a physical prop sword, enabling the user to use and "see" that prop element in a virtual reality environment. Like the rest of the Vive VR system, the Vive Tracker detects IR pulses from the Vive VR "Lighthouse" base stations, and uses coordinated timing information to determine its location and orientation relative to the base stations. It then reports this information to the host computer running the VR application. During intervals where the Vive Tracker cannot see the base stations, it uses internal IMU/gyroscopic dead reckoning to estimate movement, correcting accumulated error when it next receives a pulse from the base stations.

The Vive Tracker/VR system is an entertainment system, not a metrology tool, and there are no officially published specifications of its measurement performance. However, for applications with very low accuracy requirements – for example, like measuring the pose repeatability of the DCP, a quantity measured in tens of millimeters – it may be able to provide some insight. A few researchers have begun to explore the system's capabilities:

- In [95], Niehorster et. al. examine the accuracy of the Vive VR headset's position and orientation reporting, to determine whether the system is suitable for conducting experiments in human perception. Despite extensive testing, they unfortunately do not report any sort of quantitative accuracy or repeatability metric. However, they do describe a number of important characteristics of the tracking system, including its tendency to lose its orientation reference when tracking is lost; and a systematic tilt to the tracking system's ground plane.

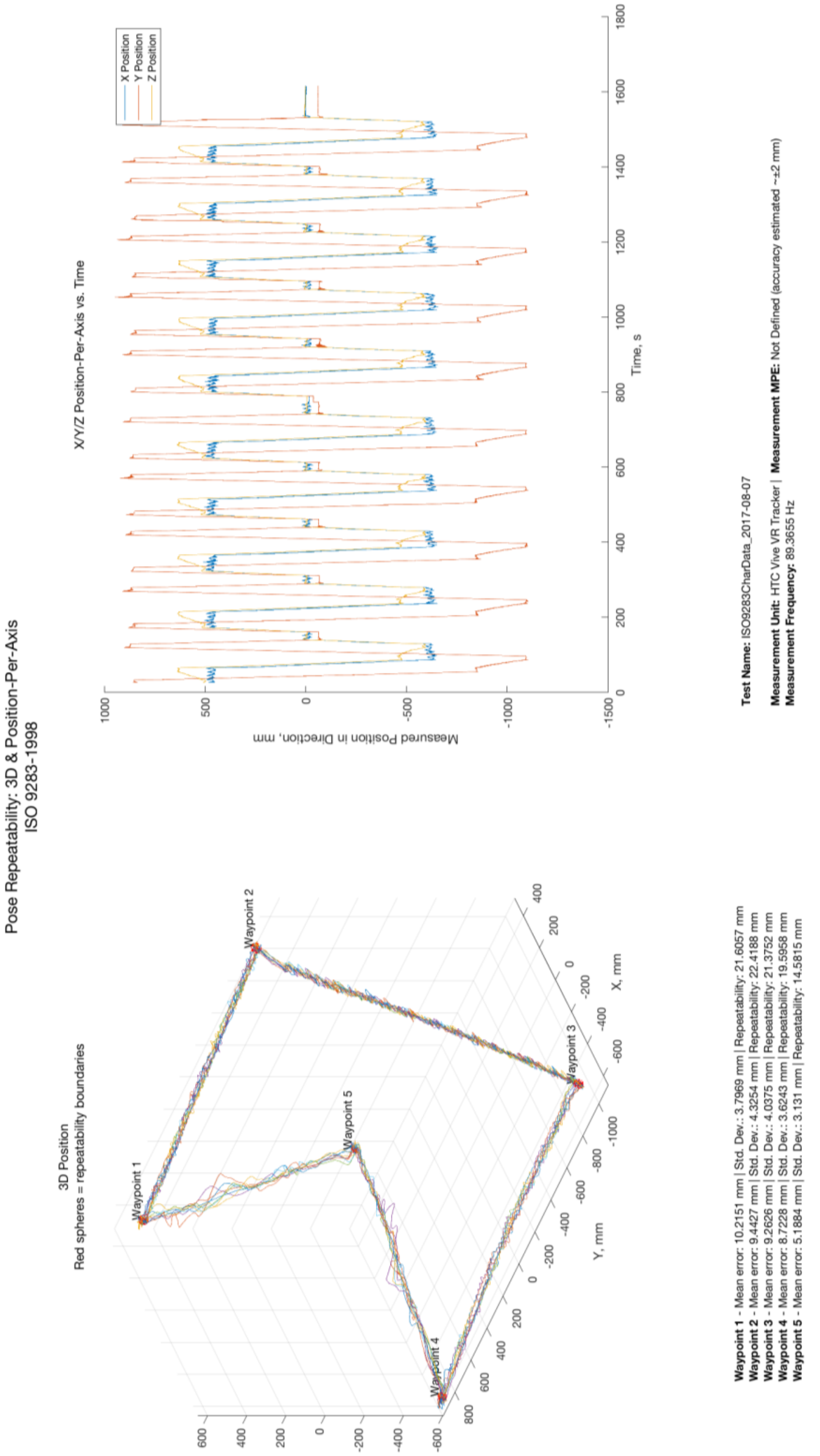
- In a widely-cited blog post [96], Kreylos examines a variety of performance parameters of the Vive VR system, including latency, tracking jitter and how drift correction operates. He also performs rough examinations of repeatability and accuracy, using a 36" ruler as a reference standard. He defines accuracy as the RMS distance between the known point set and the point set measured by the Vive tracker; and repeatability as the difference between two sequentially measured point sets. He reports an accuracy of 1.9 mm; repeatability of 1.5 mm; and a peak distance error of 2.4 mm, concluding that the expected accuracy of the Vive Tracker is on the order of 2 mm. While a starting point for understanding the Vive system's measurement capabilities, it is important to note that this is a highly limited test: it examines metrology performance in only one axis, in one area of the work volume, does not use a high-quality reference system, and does not explore the impact of the phenomena that Niehorster et. al. observed on metrology performance.

Despite the limited information available about the Vive VR system's capabilities, we elected to re-conduct the ISO 9283 pose repeatability test using the system, to examine the capabilities of both the Vive VR system and the DCP. The Vive Tracker was mounted to the KUKA endpoint, oriented with its Y axis aligned to the KUKA's X axis to maximize its field of view. A host computer running a custom Unity VR application received timestamped position updates from the Vive Tracker, and transmitted them over UDP to a Processing script, which recorded them to a text file[24].

Initially, we explored using active KUKA error compensation during the test to improve pose repeatability. However, tracking was highly unreliable, with frequent losses of connection between the Vive Tracker and the host computer, which subsequently produced position errors and major orientational shifts when tracking was recovered. After some experimentation, it became clear that electromagnetic interference from the KUKA's drives was impacting some component of the Vive VR tracking system. When the KUKA's drives were activated, any Vive component near the KUKA would abruptly lose tracking, and bringing a component within ~750 mm while the drives were active would cause it to lose tracking as well. Consequently, this experiment was abandoned, and the DCP was operated with the KUKA stationary for the pose repeatability tests reported here.

The DCP (with the KUKA stationary/inactive) was commanded to execute the ISO 9283 pose repeatability trajectory using the same test parameters used in the 2016 test (100 mm/s velocity, 2 m test cube side length). Because of the previously-observed poor robustness of the VR tracking system – particularly its tendency to change orientation when tracking was lost and recovered – we elected to conduct the ISO 9283 pose in 5-cycle sets, stopping after each set to save tracking data from the Vive Tracker and run data from dcpController. Consequently, the datasets had to be manually combined for analysis. This was done by offsetting each dataset by the difference between its mean position and the mean position of the previous dataset, essentially aligning the centroids of the gathered data. While this is likely to improve pose repeatability measurements slightly, we argue that the close agreement between the shapes of the datasets pre-combination, along with the fact that all of the points in the dataset are used to generate their centroids, make this method reasonable. Finally, since the data gathered by the Vive Tracker is considered to be of relatively low confidence for the reasons mentioned above, only 10 cycles were conducted.

[24] The Vive VR system, along with the custom Unity VR application and Processing script, were provided/developed by Owen Trueblood, who also supported set-up for the test. The author gratefully acknowledges his contribution.



Waypoint 1 - Mean error: 10.2151 mm | Std. Dev.: 3.7969 mm | Repeatability: 21.6057 mm
Waypoint 2 - Mean error: 9.4427 mm | Std. Dev.: 4.3254 mm | Repeatability: 22.4188 mm
Waypoint 3 - Mean error: 9.2626 mm | Std. Dev.: 4.0375 mm | Repeatability: 21.3752 mm
Waypoint 4 - Mean error: 8.7228 mm | Std. Dev.: 3.6243 mm | Repeatability: 19.5958 mm
Waypoint 5 - Mean error: 5.1884 mm | Std. Dev.: 3.131 mm | Repeatability: 14.5815 mm

Test Name: ISO9283CharData_2017-08-07

Measurement Unit: HTC Vive VR Tracker | **Measurement MPE:** Not Defined (accuracy estimated ~±2 mm)
Measurement Frequency: 89.3655 Hz

Figure 90: Results from ISO 9283 pose repeatability characterization, Summer 2017

The data collected was analyzed with the MATLAB scripts viveTrackerDataProcessing.m, iso9283testsegmenter.m, and iso9823testanalyzer.m, all available in the dcpctrl_v2 repository. These are updated versions of the scripts used previously for segmentation and analysis. The segmentation technique used differs slightly, but the analysis performed is identical. Figure 90 summarizes the results of this analysis.

From this analysis, pose repeatability sphere radius as defined by the ISO 9283 standard has been reduced by 59%, with all poses reporting comparable repeatability metrics – a substantial improvement. The overall trajectory is also generally quite well-defined. Because of time constraints, we did not attempt to conduct other ISO 9283 tests that could have quantified this, such as trajectory accuracy/repeatability tests. However, some insight can be taken from comparing this test trajectory to the previous pose repeatability test trajectory, as shown in Figure 91 below.

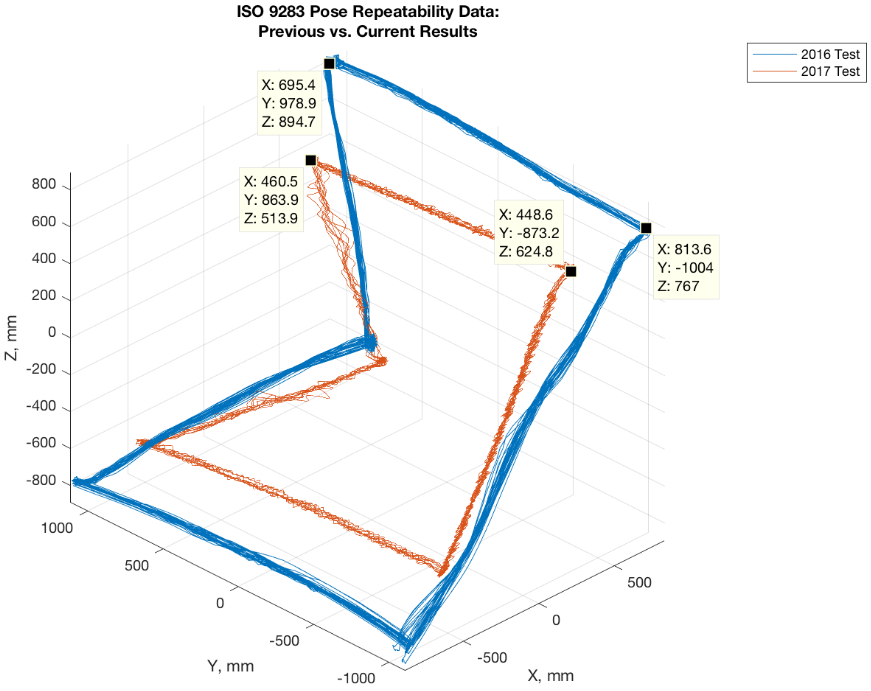


Figure 91: Comparison of ISO 9283 pose repeatability characterization, Summer 2017

As this figure shows, the trajectory from this test matches the desired trajectory considerably more accurately. The horizontal second segment of the trajectory, which should measure 1600 mm for a 2 m test cube, measures approximately 1740 mm. Additionally, the paths between waypoints are generally smoother and more linear than in the previous test[25]. This data is extremely encouraging, suggesting that the improvements to the DCP's mechatronics, controls and kinematic modeling have substantially improved the system's performance. Furthermore, by using additional techniques already available on the DCP – such as real-time error compensation by the KUKA, which could not be tested with the Vive VR system – it is conceivable that this performance could be improved even further.

However, it is critical to recognize the limitations of this experiment. Far fewer cycles were used here than in the previous test, which could substantially impact the final measured pose repeatability. Additionally, the Vive VR tracking system is largely an unknown quantity, with no meaningful data available about the accuracy or repeatability of the system. Our experience with the system suggests that when tracking is not interrupted, consistent measurements are possible. However, there is no calibrated standard to check the accuracy of these measurement against. Given the fact that the best available estimates of the Vive VR tracking system's accuracy are still within an order-of-magnitude of our desired measurement scale, further experimentation – through either validation of the Vive VR tracking system's performance, or through re-characterization of the DCP with a proper metrology-grade measurement tool – will be required to provide a truly robust pose repeatability metric for the DCP.

3.3.4 System Operation Manual

Finally, the ultimate goal of the DCP project has been to provide a platform that other researchers can use for experimentation in the automated construction space. In the service of this goal, a basic system operation manual has been created, and is included in the dcpctrl_v2

[25] There is notable relative rotation between the two trajectories: this is believed to be due to the Vive VR system's tendency to skew its internal ground plane relative to the true ground plane, as described above.

repository. This manual is targeted at a first-time user, and is intended to walk them through the process of turning on the DCP; moving the system using the tracked base; setting up the system for a robotic operation; creating a toolpath; and running a program. It also tries to provide some basic troubleshooting techniques for the DCP system. The scope of this document is necessarily limited; users wishing to use the DCP for more advanced tasks may need to refer to this thesis (and particularly this third chapter) as they work to understand and augment how the system operates.

3.4 Conclusions & Future Work

The past two years' work with the Digital Construction Platform have been highly productive, yielding true architectural-scale structures; valuable insight into the merits of foam-based additive construction techniques; and most importantly, a more robust, accessible and adaptable DCP platform that future users will be able to continue these explorations with. The major contributions to the DCP project described in this thesis, which the author either led or was substantially involved in, are:

- 2015-2016:
    - Assembled second prototype of the Digital Construction Platform concept originally developed by Keating [1], known as the DCP v.2. Tasks included retrofitting AT40GW hydraulic lift macro-manipulator for closed-loop position control with custom sensor harnesses and interface electronics; integrating KUKA micro-manipulator into DCP platform; and developing command generation and control architecture for system.

    - Performed preliminary demonstrations of multiple operational modes concept through light paintings implemented with DCP v.2 platform.

    - Characterized performance of DCP v.2 platform using ISO 9283 pose repeatability test. DCP v.2 pose repeatability, as defined by ISO 9283, was determined to be ±54.90 mm.

  - Fabricated architectural-scale dome structure using Print-in-Place additive manufacturing process originally developed by Keating [5]. The finished dome measured 14.6 m in diameter and 3.7 m high.

- 2016-2017:
  - Transitioned DCP control architecture from non-real-time, MATLAB-based controller (dcpctrl_v1) to hard-real-time, Simulink-based controller (dcpctrl_v2).

  - Developed robust Simulink-based KUKA RSI interface, enabling reliable real-time control of KUKA in dcpctrl_v2 architecture.

  - Made numerous physical improvements to DCP v.2 platform, including new controller interface; additional sensors at J1, J3 and J4 joints of AT40GW; and new safety interface providing single-point emergency stop control for KUKA and AT40GW, and enabling external stop sources to be integrated.

  - Streamlined and standardized DCP toolpath generation architecture; enabled basic toolpath segmentation operations, as well as import of toolpaths from external sources such as Rhino/Grasshopper.

  - Produced light paintings to demonstrate new DCP capabilities, including real-time tool control, toolpath segmentation between KUKA and AT40GW, and KUKA active error compensation.

  - Repeated ISO 9283 pose repeatability test after completion of improvements. Although test methodology is imperfect (as described above), preliminary results indicated that ISO 9283 pose repeatability has improved to ±22.42 mm – nearly a 60% improvement over the previous test. Substantial improvements in distance accuracy and trajectory accuracy were also qualitatively observed in the test.

In addition to these contributions, work with the DCP project has also produced many valuable insights into the future needs of the DCP v.2 platform; best practices for the design of arm-based automated construction systems in the future; and important future research questions for the field of automated construction. A brief summary of these lessons learned is provided in the following sections.

3.4.1 Lessons Learned 1: DCP v.2 Platform Improvement

As with any good engineering project, the list of improvements that still can be made to the DCP v.2 system is almost infinite. The following list captures some of the improvements that the author believes to be most important: it is not exhaustive, but should provide a starting point for future developers.

- Safety:
    - Environmental Awareness: One of the greatest risks that the DCP v.2 system currently poses during operation comes from the fact that it is not aware of its environment. For instance, the system is not aware of where the ground plane lies, or if the arm is going to collide with the AT40GW base. This could be addressed either by integration of other safety systems, such as LIDAR-based proximity sensors; or by providing the DCP control architecture with a model of the DCP's structure and the environment.

    - Controller watchdog circuitry: Another major hazard comes from the fact that the safety system is not aware of failures of the AT40GW's controller. This could be addressed by incorporating a "watchdog" circuit into the safety controller, which would monitor pulses sent periodically by the AT40GW controller, and trigger an emergency stop if a pulse was not received within the specified window.

  - Integration of diesel engine into E-STOP architecture: Currently, if the AT40GW is operating using the diesel engine, it cannot be stopped by the DCP safety system. This was not addressed in this work because the diesel engine is used so rarely, but for more extensive use, the diesel engine should be incorporated into the safety system as well.

- Hardware:
  - Link deflection measurement: Link deflection and vibration represents the most substantial unmeasured source of error in the DCP system. Results from other authors, such as Obergfell and Book [41] and Xu et. al. [97], suggest that systems based around position-sensing detectors (PSDs) are effective at measuring deflection in flexible robot links: this, or a similar technology, could be adopted for the DCP.

  - Ingress Protection: While the AT40GW is designed for outdoor use and the KUKA is IP-67 rated, the many modifications made to both systems have made the DCP unsuitable for outdoor use in inclement conditions. Before it can be used for extended durations outdoors, substantial effort will need to be expended to improve ingress protection for the DCP sensor harnesses; AT40GW control panel (which was modified during 2015-2016 to support simultaneous operation of the tracks and arm); and the KUKA controller, DCP controller and safety interface.

  - Improved system start-up interface: Currently, starting the DCP system is a tiresome, multi-step process (as described in the System Operation Manual). Streamlining this process will be greatly appreciated by future users.

- Toolpath Generation:
  - Interface improvements: Particularly as toolpath generation transitions to the Rhino/Grasshopper-MATLAB toolchain described above in Section 3.2.4, a number of exciting improvements can be made to this process. Because of its

highly visual nature, Rhino/Grasshopper makes incorporating a 3D model of the DCP into the toolpath generation process easy, and could even support limited joint limit checking or collision detection functionality. Doing this would also allow toolpaths to be rotated and moved relative to the DCP. Other Rhino/Grasshopper tools could support import of environmental data, such as 3D scans of existing structures in the DCP's environment, or measurement of ambient conditions such as temperature or windspeed.

- Command & Control:
    - Command/Control Separation: Possibly the most important improvement that can be made to the DCP v.2 system is separation of the current control architecture into command generation/sending and system monitoring (non-real-time); and system control (real-time). This type of separation would reduce the computational load on the system controller, by offloading visualization and command signal creation to the non-real-time system. It would also facilitate transitioning the DCP controller to a more robust host interface, such as a Speedgoat embedded control PC. Finally, it would also enable the DCP to be controlled from new interfaces which could provide substantial extra functionality. For instance, creating a ROS node for the DCP controller would allow the DCP to be integrated into a ROS-based system and take advantage of the many tools that ROS makes available (including system collision monitoring). Particularly as the DCP is used for more advanced robotics experiments, such as autonomous navigation and re-localization, or collaborative operation with other systems, this segmentation will become imperative.

    - Improved Simulation: A notable shortcoming of the dcpctrl_v2 architecture is that it lacks any sort of robust simulation or visualization system. Simulink provides a number of tools that could be used to create a controllable 3D model of the DCP, either for offline simulation of toolpaths, or for online monitoring of system state. Additionally, this simulator could incorporate models of the DCP's dynamics, ranging from the very simple (velocity response of the hydraulics) to complex (beam deflection and micro-macro manipulator dynamic coupling). This would make task planning substantially easier, and could also provide a useful platform for model-based control design.

- Hydraulic Circuit Characterization: While there are a wide range of dynamic behaviors present in the DCP that need to be characterized, the most pressing – and valuable to capture – is the PWM/velocity response of the hydraulic circuits. As described above in Section 3.2.3, a brief attempt was made to perform this characterization, but it was not successfully completed. However, the tools to perform this characterization exist, and with a relatively modest amount of additional work, a useful model of the system could be developed.

- kukaslxctrl Development: The kukaslxctrl library has been minimally developed, but has substantial potential. Useful improvements could include enabling joint-space control (and particularly real-time switching between task- and joint-space control); developing more robust controller examples to include in the library; enabling automatic initiation of the kukaslxctrl KRL program by the DCP controller; and determining optimal system pose to maximize range of motion.

3.4.2 Lessons Learned 2: Arm-Based Automated Construction Systems

The DCP v.2 also provides valuable lessons for the development of future arm-based construction systems, including the following:

- Macro Manipulator Selection and Actuation: We believe that the original assertion at the heart of the DCP project – that developing arm-based ACS around existing commercial equipment provides advantages in cost, system scale and speed of development – has been borne out by this project. The AT40GW, which an Altec engineer once described to the author as being the "...equivalent of a hammer and box wrench..." in terms of the sophistication of its hydraulics and appropriateness for precision control like we have implemented here, has produced impressive performance with relatively minimal additional equipment added, and no changes to its core actuation system. Furthermore, the extensive work done by other authors on the control of hydraulically-actuated arms with flexible links suggests that the compliance in the mechanical structure of these systems is not an insurmountable obstacle. By adopting even slightly higher-performance hydraulic components, we believe that existing systems like aerial lifts, excavators or concrete pumpers could be readily adapted to high-precision automated construction tasks.

- Kinematic Configuration: One of the major shortcomings of the AT40GW – and an important lesson for the design of future systems – is the kinematic arrangement used in the arm. As described earlier in this thesis, the AT40GW's kinematics have proved challenging for kinematic modeling, without providing any substantial benefit (as the J2 joint is almost never used in operation). We believe that for future arm-based ACS, a R-R-R articulated arm kinematics like that found in the DCP v.1 platform is preferable, simplifying kinematic modeling and providing improved ability to reach over obstacles. Moving beyond this, a highly redundant configuration like that commonly used in concrete pumpers could provide further benefits, although planning for these arms will require increased integration between the ACS, the structure it is fabricating, and the environment.

- Micro-macro architecture: The use of the micro-macro manipulator architecture in the DCP also seems to have been borne out, with the KUKA augmenting the AT40GW's capabilities in valuable ways. Particularly, the ability to perform tasks at two distinct scales, with corresponding degrees of precision at those scales, seems to be of value, especially in the serial operation mode described in Chapter 1.

Of course, the question still remains whether a system that is as large as the KUKA (and thus is able to execute useful trajectories on its own) actually functions as a micro-manipulator to the AT40GW's macro-manipulator. We have not examined the presence of dynamic coupling between the two systems in detail, and this could pose a substantial barrier to using the KUKA for real-time compensation of AT40GW structural vibrations. In larger ACS, this may prove to be less of a concern, as the endpoint inertia of those systems will be correspondingly larger. However, regardless of system scale, a balance between the size, load capacity, and inertia of the micro-manipulator will need to be found.

- On-system vs. off-system sensing: One major question left unanswered by this work is whether requiring that all system state measurements be performed using on-system sensors provides any advantage. The DCP has achieved the results described here without using any external sensor architecture, and there are many other sensor systems that could be integrated to improve knowledge of system state. However, the simplicity of direct endpoint position and velocity measurement using external systems is quite attractive, particularly given our experiences with the laser tracker and Vive VR tracker systems. If a technology that is sufficiently robust to operate effectively on a worksite can be identified, it may merit adoption.

3.4.3 Lessons Learned 3: The Future of Automated Construction

Many observers believe that the next 50 years will see substantial increases in the prevalence of automated systems on construction sites. As an example, in their report *A Digital Future for the Infrastructure Industry* [98], the Balfour Beatty infrastructure consulting group predicts that "...much of the actual building and construction process will be automated by 2050." While we consider this prediction optimistic, we believe it is highly likely that within this timeframe, automated construction systems will take responsibility for at least a limited series of tasks on the construction site, such as additive fabrication of foundations or hidden structural elements; painting and non-contact finishing processes; and on-site data collection. Research into the design and control of these ACS will necessarily anticipate their implementation in industrial settings, and we look forward to a highly productive coming decade in the automated construction field.

One important takeaway of the work conducted with the DCP v.2 system that is relevant here is the need for a standardized comparison method for examining highly diverse automated construction systems. As described previously, trying to reduce the performance of an automated construction to a single parameter, such as work volume or volumetric fabrication rate, frequently masks important characteristics that differentiate ACS, such as the amount of finishing required to create a viable structure, or the mechanical properties of the finished structure. Instead, we propose using a test that directly examines the quantity of interest – how well a given ACS is adapted to constructing buildings.

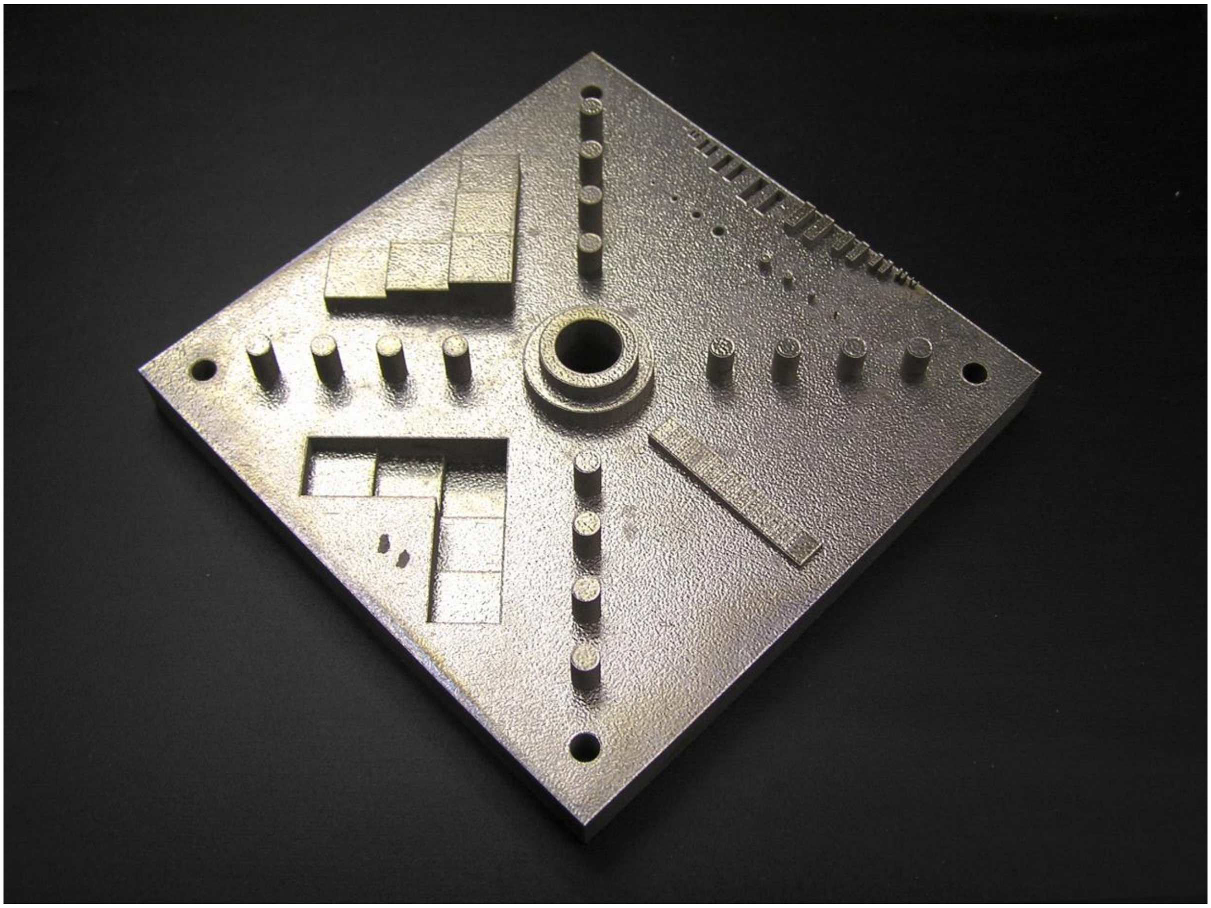


Figure 92: NIST Additive Manufacturing Test Artifact. Image courtesy of Dr. Shawn Moylan, Production Systems Group, National Institute of Standards and Technology

Figure 92 above shows one potential source of inspiration for this test. The Engineering Laboratory at the National Institute of Standards and Technology has developed a test artifact for additive manufacturing, intended to be used to characterize and compare the performance of different macro-scale additive manufacturing systems. A similar “test structure” could be envisioned for automated construction, which would examine both machine performance (repeatability, accuracy, corner round-off) as well as the performance of the complete process (achievable geometry, mechanical properties of the finished structure). The structure would need to be of reasonable scale, and incorporate a range of different common architectural forms, such as rectilinear and curved walls; arches; columns; and different types of overhang geometry, such as window lintels and roofs. Development of this artifact would need to be a collaborative process between machine builders, construction engineers, architects, and many other stakeholders in the automated construction process.

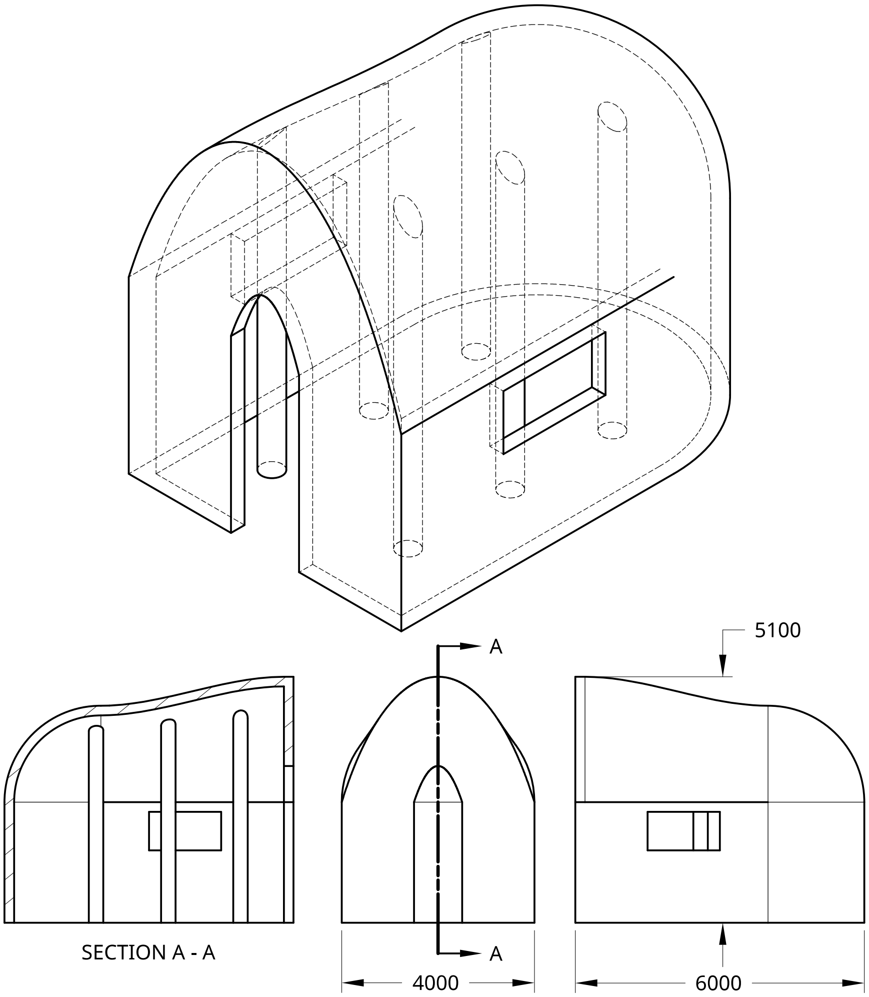


Figure 93: Concept for automated construction test artifact, incorporating architectural elements such as columns, arches, overhangs and vaults.

Regardless of its pace, we are tremendously excited to see how the field of automated construction develops over the coming years. We believe in the promise that automated construction holds for humanity, through improvements to the efficiency and safety of the construction process; enabling shelter and infrastructure to be provided in resource-constrained or dangerous environments; and creating previously impossible architectures through integration of additive manufacturing processes, environmental data gathering, and the awesome capabilities that modern robotics provides. We hope that our work with the DCP v.2 platform has made some contribution towards achieving that promise.

4 References

[1] S. J. Keating, N. A. Spielberg, J. Klein, and N. Oxman, "A Compound Arm Approach to Digital Construction," in *Robotic Fabrication in Architecture, Art and Design 2014*, 2014, pp. 99–110.

[2] S. J. Keating, J. C. Leland, L. Cai, and N. Oxman, "Toward site-specific and self-sufficient robotic fabrication on architectural scales," *Sci. Robot.*, vol. 2, no. 5, p. 15, 2017.

[3] S. J. Keating, "From Bacteria to Buildings: Additive Manufacturing Outside the Box," Massachusetts Institute of Technology, 2016.

[4] S. Nilsson, "LaTeX 7 Word Template." 2014.

[5] S. J. Keating, "Renaissance Robotics: Novel applications of Multipurpose Robotic Arms spanning Design Fabrication, Utility, and Art," Massachusetts Institute of Technology, 2012.

[6] I. Smith, S. Wamuziri, and M. Taylor, "Automated construction in Japan," *Proc. ICE - Civ. Eng.*, vol. 156, no. 1, pp. 34–41, 2003.

[7] D. A. Sangreyi and A. Warszawski, "Constraints on the development of robots for construction," in *1984 Proceedings of the 1st ISARC*, 1984, pp. 151–180.

[8] A. Scott Howe, "Designing for automated construction," *Autom. Constr.*, vol. 9, no. 3, pp. 259–276, 2000.

[9] Y. Hasegawa, "One decade of robotics in construction," in *Proc. 8th ISARC, Stuttgart*, 1991, pp. 21–34.

[10] International Association for Automation and Robotics in Construction, "Robots and Automated Machines in Construction," 1998.

[11] T. Bock and T. Linner, *Site automation: automated robotic on-site factories*. New York: Cambridge University Press, 2016.

[12] G. Pritschow, M. Dalacker, J. Kurz, and M. Gaenssle, "Technological aspects in the development of a mobile bricklaying robot," *Autom. Constr.*, vol. 5, no. 1, pp. 3–13, 1996.

[13] A. A. Apolinarska, M. Knauss, F. Gramazio, and M. Kohler, "The Sequential Roof," in *Advancing Wood Architecture: A Computational Approach*, A. Menges, T. Schwinn, and O. D. Krieg, Eds. Routledge, 2016.

[14] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative contruction with flying machines," *IEEE Control Syst.*, vol. 34, no. 4, pp. 46–64, 2014.

[15] A. Mirjan, F. Gramazio, M. Kohler, F. Augugliaro, and R. D'Andrea, "Architectural fabrication of tensile structures with flying machines," *Green Des. Mater. Manuf. Process.*, pp. 513–518, 2013.

[16] B. Khoshnevis and R. Dutton, “Innovative Rapid Prototyping Process Makes Large Sized, Smooth Surfaced Complex Shapes in a Wide Variety of Materials,” *Mater. Technol.*, vol. 13, no. 2, p. 53, Jan. 1998.

[17] B. Khoshnevis, “Automated construction by Contour Crafting – related robotics and information technologies,” *Autom. Constr.*, vol. 13, no. 1, pp. 5–19, 2004.

[18] E. Dini, R. Nannini, and M. Chiarugi, “Method and device for building automatically conglomerate structures,” WO2006100556A2, 2006.

[19] C. Gosselin, R. Duballet, P. Roux, N. Gaudillière, J. Dirrenberger, and P. Morel, “Large-scale 3D printing of ultra-high performance concrete - a new processing route for architects and builders,” *Mater. Des.*, vol. 100, pp. 102–109, 2016.

[20] B. Zareiyan and B. Khoshnevis, “Interlayer adhesion and strength of structures in Contour Crafting - Effects of aggregate size, extrusion rate, and layer thickness,” *Autom. Constr.*, vol. 81, no. June, pp. 112–121, 2017.

[21] N. Labonnote, A. Ronnquist, B. Manum, and P. Ruther, “Additive construction: State-of-the-art, challenges and opportunities,” *Autom. Constr.*, vol. 72, pp. 347–366, 2016.

[22] A. A. Apolinarska, R. Bartschi, R. Furrer, F. Gramazio, and M. Kohler, “Mastering the Sequential Roof: Computational Methods for Integrating Design, Structural Analysis, and Robotic Fabrication,” in *Advances in Architectural Geometry 2016*, 2016, pp. 240–258.

[23] R. L. Williams, J. S. Albus, and R. V. Bostelman, “Self-contained automated construction deposition system,” *Autom. Constr.*, vol. 13, no. 3, pp. 393–407, 2004.

[24] M. Giftthaler, T. Sandy, K. Dörfler, I. Brooks, M. Buckingham, G. Rey, M. Kohler, F. Gramazio, and J. Buchli, “Mobile Robotic Fabrication at 1:1 scale: the In situ Fabricator,” 2017.

[25] F. Augugliaro, A. Mirjan, F. Gramazio, M. Kohler, and R. D’Andrea, “Building tensile structures with flying machines,” in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 3487–3492.

[26] S. Jokic, P. Novikov, S. Maggs, D. Sadan, C. Nan, and J. Shihui, “Minibuilders - How to 3d Print Big Structures With Small Robots,” 2015. [Online]. Available: http://www.instructables.com/id/Minibuilders-How-to-3d-print-big-structures-with-s/?ALLSTEPS. [Accessed: 28-Mar-2017].

[27] Gramazio Kohler Research, “Robotic Fabrication Laboratory,” 2016. [Online]. Available: http://gramaziokohler.arch.ethz.ch/web/e/forschung/186.html. [Accessed: 03-Apr-2017].

[28] Putzmeister America, “Putzmeister 70Z,” *Putzmeister.com*. [Online]. Available: http://www.putzmeisteramerica.com/products/boom-pumps/boom-pumps-70z-meter-semi-trailer-mounted-concrete-boom-pump. [Accessed: 19-Jul-2017].

[29] H. B. Kuntze, U. Hirsch, A. Jacubasch, F. Eberle, and B. Göller, “On the dynamic control of a hydraulic large range robot for construction applications,” *Autom. Constr.*, vol. 4, no. 1, pp. 61–73, 1995.

[30] T. Bock and T. Linner, *Construction robots: elementary technologies and single-task construction robots*. New York: Cambridge University Press, 2016.

[31] S. Lim, R. A. Buswell, T. T. Le, S. A. Austin, A. G. F. Gibb, and T. Thorpe, “Developments in construction-scale additive manufacturing processes,” *Autom. Constr.*, vol. 21, no. 1, pp. 262–268, 2012.

[32] A. Sharon and D. Hardt, “Enhancement of robot accuracy using endpoint feedback and a micro-macro manipulator system,” *Am. Control Conf.*, pp. 1836–1845, 1984.

[33] A. Sharon, N. Hogan, and D. E. Hardt, “The macro/micro manipulator: An improved architecture for robot control,” *Robot. Comput. Integr. Manuf.*, vol. 10, no. 3, pp. 209–222, Jun. 1993.

[34] L. Luo, Y. Zhang, and Z. Sun, “Task Space Division and Trajectory Planning for a Flexible Macro-Micro Manipulator System,” *Tsinghua Sci. Technol.*, vol. 12, no. 5, pp. 607–613, 2007.

[35] B. T. Quan, J. Huang, M. Harada, and T. Yabuta, “Control of a Macro-Micro Robot System Using Manipulability of the Micro Robot,” *JSME Int. J. Ser. C*, vol. 49, no. 3, pp. 897–904, 2006.

[36] M. A. Torres, S. Dubowskytand, and A. C. Pisonis, “Path-Planning for Elastically-Mounted Space Manipulators: Experimenta Evaluation of the Coupling Map,” in *1994 IEEE International Conference on Robotics and Automation*, 1994, pp. 2227–2233.

[37] I. Sharf, “Active damping of a large flexible manipulator with a short-reach robot,” in *Proceedings of 1995 American Control Conference - ACC’95*, vol. 5, pp. 3329–3333.

[38] W. J. Book and J. C. Loper, “Inverse dynamics for commanding micromanipulator inertial forces to damp macromanipulator vibration,” in *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No.99CH36289)*, 1999, vol. 2, pp. 707–714.

[39] W. J. Book and S. H. Lee, “Vibration Control Of A Large Flexible Manipulator By A Small Robotic Arm,” *American Control Conference, 1989*. pp. 1377–1380, 1989.

[40] W. J. Book, “Modeling, Design, and Control of Flexible Manipulator Arms : A Tutorial Review,” *29-th IEEE Conf. Decis. Control , Honolulu , Hawaii*, 1990.

[41] K. Obergfell and W. J. Book, “End-Point Position Measurements of Long-Reach Flexible Manipulators,” no. 1990, 1992.

[42] Fastbrick Robotics, “Home - Fastbrick Robotics.” [Online]. Available: http://fbr.com.au/. [Accessed: 01-Apr-2017].

[43] M. Pivac and M. Wood, “Automated Brick Laying System for Constructing a Building from a Plurality of Bricks,” US8166727B2, 2012.

[44] Robohub, “Hadrian Bricklaying Robot,” 2016. [Online]. Available: http://robohub.org/robots-hadrian-bricklaying-robot/. [Accessed: 28-Mar-2017].

[45] Altec Inc., “L42M / L45M Sales Brochure.” pp. 11–12, 2012.

[46] Fanuc, “FANUC M-2000iA/1700L Datasheet.” Fanuc, pp. 6–7, 2000.

[47] Dow Chemical, “FROTH-PAK $^{\text{TM}}$ Foam Insulation Product Information Sheet,” 2017.

[48] International Standards Organization, “ISO 4287-1997.” Geneva, 1997.

[49] Digital Surf, "MountainsMap." 2017.

[50] S. Keating and N. Oxman, "Compound fabrication: a multi-functional robotic platform for digital design and fabrication," *Robot. Comput. Integr. Manuf.*, vol. 29, no. 6, pp. 439–448, 2013.

[51] B. Furet and J. Leland, "Email Correspondence 2017-04-14." 2017.

[52] E. Barnett and C. Gosselin, "Large-Scale 3D Printing With A Cable-Suspended Robot," *Addit. Manuf.*, vol. 7, pp. 27–44, 2015.

[53] G. Hunt, F. Mitzalis, T. Alhinai, P. A. Hooper, and M. Kovac, "3D printing with flying robots," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 4493–4499, 2014.

[54] S. J. Keating, J. C. Leland, and L. Cai, "dcpctrl_v1." MIT Media Lab, 2016.

[55] Altec Inc., "AT40GW Sales Brochure," 2015.

[56] J. J. Craig, "Introduction to Robotics: Mechanics and Control 3rd," *Prentice Hall*, vol. 1, no. 3, p. 408, 2004.

[57] KUKA Roboter GMBH, "KR AGILUS sixx." 2013.

[58] K. R. GmbH, "KUKA.RobotSensorInterface 3.1," no. 23.12.2010, pp. 1–83, 2010.

[59] T. Yoshikawa, K. Hosoda, and T. Doi, "Quasi-Static Trajectory Tracking Control of Flexible Manipulator by Macro-Micro Manipulator System.," *J. Robot. Soc. Japan*, vol. 11, no. 1, pp. 140–147, 1993.

[60] P. Corke, "Robotics Toolbox 9.10 for MATLAB." 2015.

[61] L. Mogas-Soldevila, J. Duro-Royo, and N. Oxman, "Water-Based Robotic Fabrication: Large-Scale Additive Manufacturing of Functionally Graded Hydrogel Composites via Multichamber Extrusion," *3D Print. Addit. Manuf.*, vol. 1, no. 3, pp. 141–151, 2014.

[62] M. Enzweiler, "StarStaX." 2014.

[63] Apple Inc., "Compressor." 2017.

[64] Apple Inc., "Motion." 2017.

[65] International Standards Organization, "ISO 9283-1998." .

[66] Leica Geosystems, "PCMM System Specifications." Leica Geosystems, 2008.

[67] Mathworks Inc., "Simulink Desktop Real-Time." 2017.

[68] G. Maletzki, T. Pawletta, S. Pawletta, and B. Lampe, "A Model-Based Robot Programming Approach in the MATLAB-Simulink Environment," in *Advances in Manufacturing Technology - 4th International Conference on Manufacturing Research*, 2006.

[69] F. Chinello, S. Scheggi, F. Morbidi, and D. Prattichizzo, "KCT: A MATLAB toolbox for motion control of KUKA robot manipulators," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 4603–4608, 2010.

[70] Quanser, "QUARC." 2016.

[71] Association for Robots in Architecture, "KUKA|prc." 2015.

[72] E. Sezener and O. Kaya, "KUKA RSI-3 Communicator." 2015.

[73] sjaakjules, "Marionette." 2015.

[74] sjaakjules, "LightServer." 2016.

[75] M. Seehauser, “KUKA Matlab Connector.” 2015.

[76] bootsman, “InputOutputStream.” 2016.

[77] LabJack Corporation, “A-3-1: Noise and Resolution,” *LabJack T7 Datasheet - Appendix A*. [Online]. Available: https://labjack.com/support/datasheets/t7/appendix-a-3-1. [Accessed: 04-Aug-2017].

[78] Lifi Labs Inc., “LIFX Color 1000 Product Specification.” 2017.

[79] Lifi Labs Inc., “LIFX LAN Protocol.” 2015.

[80] International Standards Organization, “ISO 10218.” 2011.

[81] American National Standards Institute Inc. and Robotic Industries Association, “ANSI/RIA R15.06-2012.” 2012.

[82] W. T. Higgins, “A Comparison of Complementary and Kalman Filtering,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-11, no. 3, pp. 321–325, 1975.

[83] S. Colton, “The Balance Filter A Simple Solution for Integrating Accelerometer and Gyroscope Measurements for a Balancing Platform.” p. 20, 2007.

[84] R. Petrella, M. Tursini, L. Peretti, and M. Zigliotto, “Speed measurement algorithms for low-resolution incremental encoder equipped drives: A comparative analysis,” *Int. Aegean Conf. Electr. Mach. Power Electron. Electromotion ACEMP’07 Electromotion’07 Jt. Conf.*, pp. 780–787, 2007.

[85] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, 2nd ed. Berlin, 2017.

[86] R. W. Krauss, “Sensor Fusion for Vibration Supprssion Implemented on Arduino and Raspberry Pi,” *Proc. ASME 2015 Dyn. Syst. Control Conf.*, pp. 1–10, 2015.

[87] A. De Luca and W. Book, “Robots with Flexible Elements,” in *Springer Robotics Handbook*, .

[88] W. Singhose and W. Seering, *Command Generation for Dynamic Systems*. lulu.com, 2011.

[89] K. L. Sorensen and P. W. Cross, “Analysis and Mitigation of Dead-Zone Effects on Systems Using Two-Impulse ZV Input Shaping,” *Mech. Eng.*, pp. 5539–5544, 2007.

[90] M. Nordin and P.-O. Gutman, “Controlling mechanical systems with backlash—a survey,” *Automatica*, vol. 38, no. 10, pp. 1633–1649, 2002.

[91] J. Tal, “Two feedback loops are better than one,” *Machine Design*, vol. 71, no. 7, p. 3, 1999.

[92] N. Ahmad, A. Zerai, and M. Al-Mumun, “A New Continuous Time Adaptive Backlash Inverse Controller and Applications to Output Backlash Compensation,” in *11th Mediterranean Conference on Control and Automation*, 2003, no. June.

[93] HTC Corporation, “Vive Tracker,” 2017. [Online]. Available: https://www.vive.com/us/vive-tracker/. [Accessed: 08-Aug-2017].

[94] HTC Corporation, “HTC Vive Tracker Developer Guidelines v.1.4,” 2017.

[95] D. C. Niehorster, L. Li, and M. Lappe, “The accuracy and precision of position and orientation tracking in the HTC vive virtual reality system for scientific research,”

Iperception., vol. 8, no. 3, pp. 1–23, 2017.

[96] O. Kreylos, “Lighthouse Tracking Examined,” 2016. [Online]. Available: http://doc-ok.org/?p=1478. [Accessed: 08-Aug-2017].

[97] W. L. Xu, S. K. Tso, and X. S. Wang, “Conceptual Design of an Integrated Laser-Optical Measuring System for Flexible Manipulator,” in *IEEE International Conference on Systems, Man, and Cybernetics, 1996.*, 1996, pp. 1247–1252.

[98] Balfour Beatty, “A Digital Future for the Infrastructure Industry,” 2017.

[99] J. Royalty, “Simulink Dynamic Signal Analyzer.” 2007.