Diss. ETH No. 27624

Computational Re-Forming

Computational Strategies for Robotic Fabrication of Shaping Malleable Materials

A thesis submitted to attain the degree of

Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by

Zhao Ma

Master of Engineering in Civil and Environmental Engineering
Massachusetts Institute of Technology, 2017

Master of Architecture
Massachusetts Institute of Technology, 2017

born on 27.05.1988
citizen of China

accepted on the recommendation of

Prof. F. Gramazio, examiner
Prof. M. Kohler, examiner
Prof. Dr. S. Coros, co-examiner
Dr. M. Bächer, co-examiner

2021

Zhao Ma: *Computational Re-Forming*

Computational Strategies for Robotic Fabrication of Shaping Malleable Materials © 2021

DOI: 10.3929/ethz-a-

Location:

Zürich

Time Frame:

2017-2020

Dedication

To my parents and grandma who raised me,

To my grandpa whose death taught me how to live.

To the people who suffered from the misfortune of COVID-19,

To the similar souls who still believes in the future, and moves on...

Acknowledgements

First, I would like to express my sincere gratitude to my supervisors Prof. Fabio Gramazio and Prof. Matthias Kohler for their trust and the great opportunity they provided me with. Being part of the Gramazio Kohler Research group was an extremely valuable experience for me both personally and professionally. I am particularly grateful for their extraordinary vision in the architecture field that helped me to see higher and farther.

Apart from my supervisors, I would like to thank my co-supervisor Prof. Dr. Stelian Coros for his knowledgeable advice, patient guidance and visionary decision-making that shaped me through the research. I strongly value his assistance, input and support, and his ever to-the-point feedback. I thank him especially for his join to my research at my most difficult time.

I would like to thank my co-supervisor Dr. Moritz Bächer for initializing this PhD position between ETH Zurich and Disney Research, and provided the opportunity that brought me from the US to Switzerland, and for his admirable working attitude which stimulates me throughout the doctoral period.

I would like to express my sincere gratitude to Prof. em. Dr. Gerhard Tröster, the ETH Ombudsperson who provided timely mental support with me and helped to mitigate between different parties for the issues I encountered.

I would like to thank my collaborators Alex Walzer and Simon Duenser for the two projects I developed in this dissertation. Alex conducted the fabrication tasks of the project and led the hardware updates. We had a great experience together and went through all the difficult times during the first project. Simon contributes a crucial part to the second project adapting his work to my research and the collaboration experience have taught me a lot.

I would like to thank two people, in particular, my colleague Dr. Christian Schumacher at Disney Research and Gonzalo Casas at Gramazio Kohler Research, who have had an important part in making this dissertation possible. I am very grateful to have met Christian at the beginning of my PhD. Christian provided tremendous help in various aspect including invaluable advice on programming, research, and writing throughout the past three years, language support and life guidance when I first came to Switzerland, contributions extremely supporting code guidance for my project. Gonzalo, a specially gentle and generous human being who never authored any of my papers, spent a large portion of his time helping to set up the robot environment and answering my endless technical questions.

Thank Dr. Romana Rust at Gramazio Kohler Research and Dr. Espen Knoop at Disney Research for their feedback and comments on my presentation and writings. Romana relieves my stress in many coordination and communication tasks for the past three years, and Espen provided much technical support by manufacturing hardware I need for my research.

I would like to specially thank Prof. Dr. Philippe Block, who accepted me as a summer researcher at the Block Research Group back in 2015. That experience was my first encounter with ETH Zurich and Switzerland, and fertilized my decision to move to Europe. Philippe also generously allowed me to use the *KUKA iiwa* robot arm from his group, though the robot was not chosen in the end.

I just cannot imagine how I would have survived these three years at ETH Zurich, being so far from home. I am grateful to my family and my friends for supporting me spiritually throughout conducting my research and writing this thesis.

Abstract

While the utilization of robot arms has increased since the construction industry began to deploy robotic technologies for digital fabrication processes, a pipeline is missing for fabrication-aware design as the abstraction of complex, contradictory constraints for the designer is not evident. Additional geometric complexity, material properties, etc. also contribute to the overall difficulties for fabricating the designated piece successfully without any collisions or structural failure.

Through the development of two projects focusing on different aspects of robotic fabrication, this dissertation identifies various limitations related to the overall design-to-fabrication process and categorizes them into different types of constraints. It is observed that many of the constraints occurred within one fabrication task are usually intertwined and cannot be decoupled, which requires integrated computational strategies to resolve.

By adopting available methods in the computer graphics field that address geometry and material, this dissertation presents a series of optimization-based strategies in the context of two specific research projects, targeting geometry processing and path planning for robotic fabrication. Its aim is to demonstrate the potential of using optimization methods to obtain achievable robotic fabrication solutions under sophisticated requirements. Focusing on geometry processing and path planning, respectively, this dissertation employs optimization

approaches to assist with design aims, and develops a conceptual framework for solving fabrication-aware robotic fabrication tasks.

The formulation of the optimization problems in this dissertation empowers the design processes to be fabrication-aware so as to be compatible with the selected fabrication technology. It provides a more mathematical and holistic perspective for looking at robotic fabrication technologies in the architectural domain.

Zusammenfassung

Deutsche Zusammenfassung hier.

Seit die Bauindustrie begonnen hat Technologien für digitale Fabrikationsprozesse einzusetzen, konnte auch ein Anstieg der Anwendungen von industriellen Roboterarmen verzeichnet werden. Heutzutage fehlt jedoch ein integrierter Arbeitsablauf für fertigungsgerechten Entwurf, da die Abstraktion komplexer, widersprüchlicher Einschränkungen für die Entwerfenden oft nicht offenkundig ist.

Zusätzlich tragen geometrische Komplexität, Materialeigenschaften usw. ebenfalls zu den allgemeinen Schwierigkeiten bei, das gewünschte Objekt oder Bauteilerfolgreich und mit hoher technischer Machbarkeit oder strukturelles Versagen herzustellen.

Durch die Entwicklung von zwei Projekten, die sich auf verschiedene Aspekte der robotischen Fertigung konzentrieren, werden in dieser Dissertation verschiedene Einschränkungen im Zusammenhang mit dem gesamten Prozess vom Entwurf bis zur Fertigung identifiziert und kategorisiert. Es wird festgestellt, dass viele der Beschränkungen, die innerhalb einer Fertigungsaufgabe auftreten, normalerweise miteinander verflochten sind und nicht entkoppelt werden können, was integrierte Lösungsstrategien erfordert.

Durch die Übernahme verfügbarer Methoden aus dem Bereich der Computergrafik, die sich mit Geometrie und Material befassen, werden in dieser Dissertation eine Reihe von Optimierungsstrategien im Rahmen von zwei spezifischen Forschungsprojekten vorgestellt, die auf Geometrieverarbeitung und robotische Pfadplanung abzielen. Ziel ist es, das Potenzial von Optimierungsmethoden zu demonstrieren um realisierbare Lösungen für die robotische Fertigung unter anspruchsvollen Anforderungen zu erhalten. Diese Dissertation konzentriert sich auf die Geometriebearbeitung und die robotische Pfadplanung, verwendet Optimierungsansätze zur Unterstützung von Entwurfszielen und entwickelt einen konzeptionellen Rahmen für die Lösung von robotischen Fabrikationsaufgaben.

Die Formulierung der Optimierungsprobleme ermöglicht es, die Entwurfsprozesse so zu gestalten, dass sie mit der gewählten Fertigungstechnologie kompatibel sind. Sie bietet eine mathematische und ganzheitliche Perspektive für die Betrachtung von robotischer Fabrikation im Bereich der Architektur.

Contents

Chapter 1

Introduction

It was the best of times, it was the worst of times.

— *Charles Dickens*, A Tale of Two Cities

1.1 Background and Motivation

The utilization of robots plays one of the utmost important roles in modern engineering industries as automation has become a core competence and more or less the standard in many classical manufacturing industries for products ranging from small scale (*e. g.* food, cellphones) to large scale (cars, aeroplanes) (Figure 1.1). However, the architectural construction field is left behind the trend due to the complexity of the building systems and the non-standard characteristics of the building products (Aste, Manfren, and Marenzi 2017; García de Soto et al. 2018). In recent years, however, robot arms have been utilized in the architectural academia and industries (Hamid, Tolba, and El Antably 2018; Potstada et al. 2016) in a growing trend for novel fabrication processes: exploring the possibilities of increasing the magnitude of automation to reduce the amount of human labour in the fabrication process, developing novel material systems or sustainable fabrication systems, and even expending the range of building components that we can design and fabricate.

FIGURE 1.1: **Robot arms in industrialized environment.** A. Tesla assembly line; B. Food assembly line; C. Human-Robot collaborative assembly line.

The efforts for facilitating the manufacturing process in architectural fields using robot arms started around 2006, marked with a series of architectural research projects conducted by Gramazio Kohler Research (GKR) at ETH Zurich (Gramazio, Kohler, and Willmann 2014). Throughout the years, various research and applications along this direction have proven great viability for fabrication and construction processes worldwide, and brought novel utilization in construction with various materials and of unconventional geometries into the fields (Figure 1.2). Amongst these projects, one of the highlights is the collaborative demonstrator *DFAB*

House (Graser et al. 2020) of ETH Zurich, Empa, and the Swiss National Centre of Competence in Research (NCCR) Digital Fabrication, where a variety of digital fabrication technologies have been employed and tested on a real building site, exemplifying how architecture can change along the way it is designed and built (Figure 1.3).

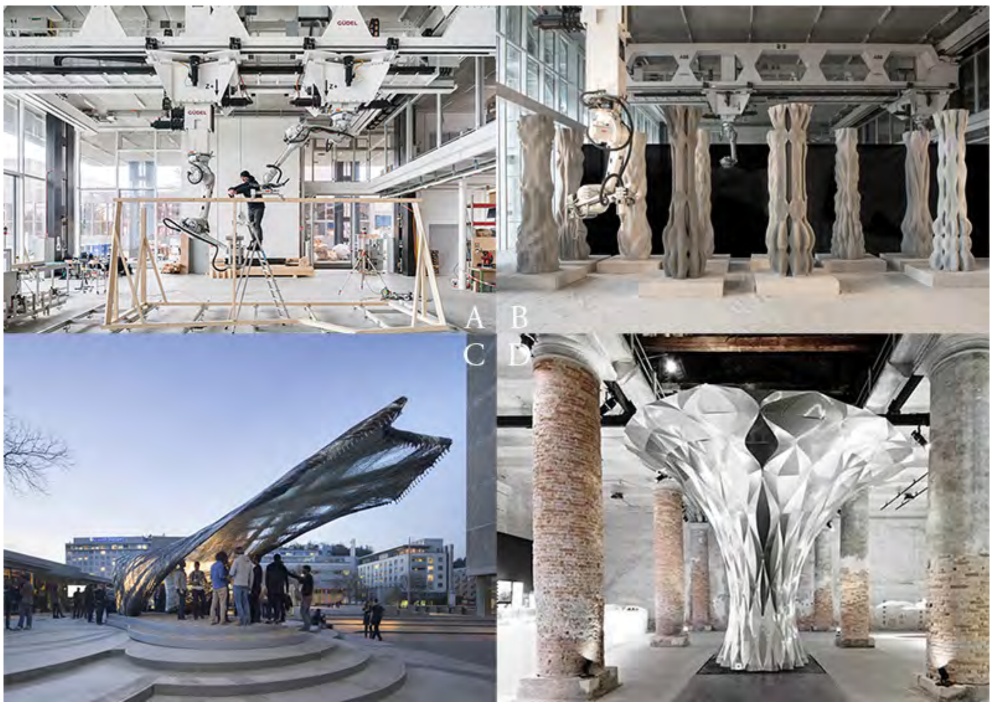

FIGURE 1.2: **Various researches and applications of digital fabrication in the architecture field**: A. *Spatial Timber Assemblies*, timber frames assembled by robot arms (Gramazio Kohler Research, ETH Zurich); B. Concrete 3D Printed Columns (Digital Building Technologies, ETH Zurich) ; C. 2017 Institute for Computational Design and Construction (ICD) Research Pavilion (ICD, ITKE University of Stuttgart), ; D. *ARUM*, robotically folded steel sheet structure (RoboFold & Zaha Hadid Architects).

Nonetheless, utilizing robot arms for research and applications in the Architecture, Engineering and Construction (AEC) fields poses special challenges due to the characteristics of the fields. One of the most prominent differences of the AEC field is the need for adaptable fabrication processes that suit the variation of the manufactured objects. Unlike robot arms used in industrial assembly lines where they repeatedly execute the same set of movements in a fixed and programmed procedure, the ones used in the architecture industries and re-

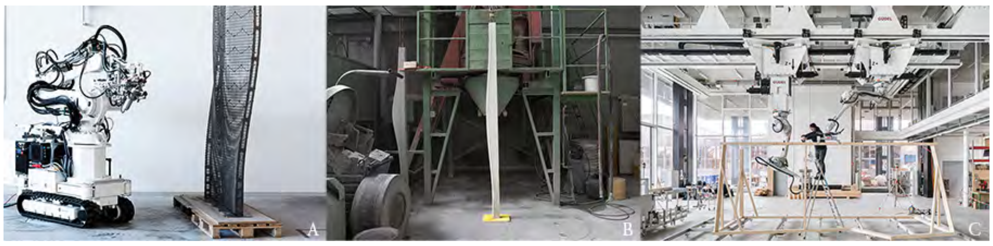

FIGURE 1.3: **Three examples of digital fabrication technologies developed and applied in the *DFAB House* project**: A. In Situ Fabricator; B. Smart Dynamic Casting; C. Spatial Timber Assemblies.

search environments face different scenarios—architectural elements are usually different per component [1], and thus require flexible and adaptable robot control and task management. Consequently, how to control the robot arms efficiently and smartly during each specific task with less unnecessary movements, no undesirable collisions, and minimum effort of human intervention becomes a nontrivial question.

On the other hand, The construction of geometrically complex shapes also challenges the approaches that robot arms are conventionally used. As one of the core topics in architectural design, complex geometries has always been one of the main drivers for the development of architectural construction technologies. However, the employment of robotic fabrication in the industry was not common until the recent ten years, and related projects mostly adopted computer-aided fabrication technologies for subjects like surface ruling or component modularization (Bermano, Funkhouser, and Rusinkiewicz 2017). While robotically fabricating general geometries seems to be a long-term goal, the gap between the types of existing architectural geometries and the types that can be processed and robotically fabricated is far from filled. This presents a more urgent need for technologies that facilitate the processing and fabrication of complex geometries with robots.

Besides the two challenges mentioned above, numerous constraints should also be managed during a fabrication process, such as material viscosity and structural failure, which are usually

[1]Since one of the goals of architectural digital fabrication is to expand the fabrication capability of non-standard units with minimum increased cost, the widely used standard modular system is excluded.

raised by the different selection of the interacted materials and the objects' scale. While different projects may share specific needs, a collection of the common interests across different robotic fabrication tasks may includes, but is not limited to: 1) *structural stability*—how well the fabricated artefact stands, 2) *topology*—the way how the member components are organized, 3) *geometric proximity*—divergence between the digital model and the physical fabricated artefact, 4) *motion planning*—whether available robot movements exist, and 5) *user-defined constraints*—additional information required by the user as fabrication constraints according to the project needs.

In general, these constraints that are related to the fabrication of non-standard and geometrically complex architectural components using robot arms could be addressed in three different stages during a design-to-fabrication process: during the design stage, between the design stage and fabrication stage, and during the fabrication stage.

To address the constraints during the design stage, researchers and designers have been exploring the possibilities of using design constraints as design drivers throughout the years (Eckert and Stacey 2014; A. Kilian 2006; Peters 2010). While design constraints may include various types of design requirements (for instance, structural stability, acoustic performance, assembly sequence, etc.), integrating those that are related to fabrication at this stage will benefit the design-to-fabrication process by eliminating the common iterations between design and fabrication caused by non-fabricable design. However, it is not realistic for most projects to beware of all the fabrication constraints, and addressing them in the following two stages is thus substantial.

To address the constraints between the design and fabrication stages, *i. e.* transform a design to a fabricable design, the target geometries are usually adjusted or modified to suit specific technologies. This usually happens when the fabrication approach is not decided or fully exposed to the designers during the design stage. While it is not the main focus of this dissertation, this process is usually named as *rationalization* and has been widely applied in many built projects. Austern, Capeluto, and Grobman (2018) has provided a thorough review in this direction.

The address of the constraints in the fabrication stage is not commonly employed—it is usually not practical to reverse the rationalization process and suit the fabrication technologies to the designed targets by developing new fabrication tools or processes. However, the rise of robotic fabrication using robot arms in the architecture field provides exactly such an opportunity. As the complexity of the architectural components increases, the diversification of these components also demands a wide range of specified tools—the end effectors attached to the robot arms—to conduct the fabrication procedure. While robot arms provide more freedom in motion and accuracy in precision for fabrication processes, end effectors provide the essential operations for the fabrication of the end products. Depending on the specific fabrication process, the two aspects are usually weighted differently according to the specific characteristics of the project. For instance, weaving carbon fibres in Figure 1.2.B requires careful sophisticated control of the robot motions, while the complexity of the fabrication for the *Mesh Mould* project in Figure 1.3.A is mainly embedded in the robot end effector where additional mechanisms accomplish the process of bending, inserting and welding rebars.

It is obvious to notice that some of the limitations are related to robotic fabrication processes or robot arm manoeuvre directly, and some are not. For robotic fabrication processes, two types of fabrication constraints can be categorized:

1. *Local fabrication constraints*: introduced by the end effectors at the interface between itself and the target object;
2. *Global fabrication constraints*: introduced by robot arms, target objects, environments, etc. during a fabrication process.

Besides the fabrication constraints, non-fabrication constraints introduced by other factors, such as material strength, structural stability of the building parts, geometric proximity to the designed targets, etc. should also be considered. Without loss of generality, these three types of constraints usually co-exist in a design-to-fabrication process, differently weighted depending on different aspects (scale, material, etc.) of the specific projects.

When these constraints can be decoupled as independent problems, existing techniques are usually applicable to solve these problems individually. For instance, global fabrication constraints, such as the robot-target collisions, may be resolved by carefully planning the robot motions, but the local fabrication constraints may not, especially when the end effector contains additional degrees of freedom (DOF) (motors, mechanisms, etc.) or cannot be included into the planning algorithms. Once the local fabrication constraints cannot be resolved by planning approaches, the target geometry will thus need to be modified, or rationalized, in order to successfully conduct a fabrication task. Additionally, non-fabrication constraints may also result in a need to modify the geometry by employing a rationalization step.

However, in many cases, these constraints are difficult to decouple and cause the complexity of the problem to increase tremendously. The effort cost for manually tuning the right set of parameters for planning robot motions, or controlling the fabrication process in general, will thus increase as the constraints increase in number or difficulty—it may take an enormous amount of time or sometimes even be impossible to find the right set of parameters. Furthermore, the non-standard characteristics of architectural components make the situation worse—one set of parameters for one of the components may not suit the other, as the robot motion and atomic fabrication procedure[2] differs.

Hence, integrated strategies, systematically designed and easily applicable, are needed to resolve related constraints in a simultaneous way for a successful robotic fabrication process. This is the reason why optimization-based strategies are introduced in this dissertation for handling complex robotic fabrication tasks.

The successful abstraction and transformation of a complex robotic fabrication task into the corresponding mathematical formulation of an optimization problem relies on a thorough understanding of the fabrication process in order to describe the physical reality in

[2]The most basic robotic movement in a complex fabrication process composed of a serious procedures.

mathematical sense. By virtue of the natural connection between the Computer Graphics (CG) field and problems related to geometry and material simulation and modelling, there exists great potential in adopting methods from the CG field to facilitate the development of robotic fabrication in the architecture context. The CG field may also benefit by testing these methods in physical reality and at an architectural scale, since they are originally developed for animation, computational modelling, rendering, etc. in the virtual world.

Such interaction with the physical world may pose new challenges, as the results need to be not only visually plausible, but also physically realisable and hardware compatible (Bermano, Funkhouser, and Rusinkiewicz 2017). Figure 1.4 shows several selected projects in this context, in which CNC milling accessibility (Figure 1.4.A), material properties (Figure 1.4.B), assembly (Figure 1.4.C), and geometric control of light transportation (Figure 1.4.B) are addressed respectively.

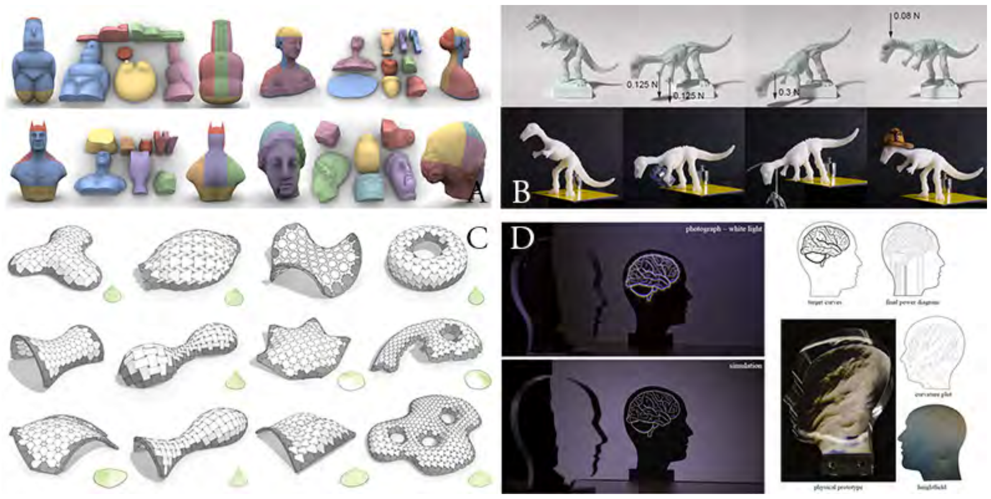

FIGURE 1.4: **Selected digital fabrication projects from the CG field**: A. Decomposition of 3D objects for 3-axis CNC milling (Muntoni et al. 2018); B. Inverse design of elastic shapes under given loading conditions (Chen et al. 2014); C. Design and structural optimization of topological interlocking assemblies (Wang et al. 2019); D. Caustic design by geometry optimization (Schwartzburg et al. 2014).

Different from the desktop-level projects mentioned in Figure 1.4, structural stability and material properties affect the results more severely, as they do not scale in the same order as the sizes of the building components scale [3]. This echoes with the challenges mentioned above in the robotic fabrication for building components context, where the fabrication of shapes is closely intertwined with its geometric complexity, structural stability and robotic motions and collisions. Besides, other types of constraints may also apply according to the specification of the projects, such as a required assembly sequence of individual members, or limitations of the robot reach.

Facing these challenges, novel methods need to be developed to seek better computational design methods without breaking the integrity of the problems. Optimization-based strategies, in this context, are well suited, as their mathematical properties theoretically allow multiple objectives and constraints to be addressed at the same time, if they can be abstracted mathematically.

1.2 Dissertation Statement

From the previous discussion around the fabrication and non-fabrication constraints, two key problems can be distilled for architectural robotic fabrication. There is a keen need for:

- rationalization tools for design that adapt the target to a robotic fabrication process by fulfilling the relevant constraints, and
- path planning tools that integrate relevant constraints regarding the fabrication process.

Additionally, some circumstances require integration of the two types of tools according to the complexity of the problem. For instance, when the robot motion generated by the path planning tools is feasible, the target may still fail due to other types of constraints (*e. g.* structural stability) and thus needs to be modified by the rationalization tools. In such cases,

[3]In his *Two New Sciences* (Galilei 1914), Galileo Galilei for the first time discussed the argument that material does not scale, as the relationship between volume and surface area relationship follows the "square-cube law".

the global and local fabrication constraints mentioned in the above section (Section 1.1) co-exist in the same project and pose additional questions.

To respond to these questions, this dissertation presents a series of optimization-based strategies targeting the geometry processing and path planning for robotic fabrication in the architecture domain. It identifies key open problems in the context of specific project-based research including, but not limited to structural stability, topology, geometric proximity, and path planning.

The research objectives are contextualized in a design-to-fabrication workflow using optimization-based methods: 1) computationally modelling the target material using either material-based or geometry-based methods[4], 2) abstracting the parameters related to the design process and fabrication constraints, 3) formulating optimization with multiple objectives according to the fabrication processes and design preferences, and 4) finding executable robotic motions such that the fabricated results match the simulated ones.

This thesis attempts to provide a more mathematical and holistic perspective of looking at robotic fabrication in the architecture domain: a) how optimization-based strategies will help the formation and solving of emerging fabrication problems in the Digital Age, as previous independent problems become more interdependent and intertwined, and b) how optimization-based strategies will facilitate the design process to be fabrication-aware by looking at it as a problem-solving process with the integration of fabrication-related constraints. Expectedly, these challenges will call for novel solutions with more systematic and computational thinking.

1.3 Methodology

This dissertation is constructed upon two complimentary projects that exemplify the overall research question of utilizing optimization-based methods targeting *geometry processing* and *path planning* problems for robotic fabrication tasks. For ease of structuring, the two

[4]material-based methods model the material property (*Project I*) while geometry-based methods model the geometry (*Project II*).

projects address their main focus at each of the two needs stated in the Dissertation Statement (Section 1.2), which relates to the *Local fabrication constraints* and *Global fabrication constraints* mentioned in Section 1.1, with a secondary focus on the other. The two projects are independently structured but contribute to the same goal in the global context as a whole.

Both projects are organized around the following areas of investigation:

- modelling and simulation of the interaction between the tool and the workpiece,
- direct or indirect modelling of the fabrication process with a robot arm[5],
- abstraction of the design parameters and fabrication constraints, and
- optimization formulation to solve the conjugated fabrication problems.

Within each of the projects, a combined approach of physical experiments and developments of computational methods is conducted to guarantee the correspondence between the simulation results and physical results. The associated physical experiments serve to not only validate the proposed strategies but also provide inspiration for the algorithm development. Each of the individual projects utilizes a different hardware setup and interact with a different material:

- *Project I*: ABB-IRB 4600 (robot arm), steel rebar (material);
- *Project II*: Universal Robot (robot arm), water-based clay [6] (material).

1.3.1 Projects

The two projects developed in this dissertation serve as demonstration cases for the two situations described in Section 1.2, respectively.

[5]Here, "direct" or "indirect" refers to whether the fabrication process is modelled directly (*e. g.* simulation of the fabrication process, as in Project II), or indirectly (*e. g.* optimization constraints using the extracted corresponding parameters, as in Project I).

[6]The "water-based clay" used in this dissertation refers to the most common type of clay soil used for sculpting, which are soft in room temperature and will turn solid quickly as water evaporates. It differs from the oil-based clay that can be directly milled used in car industries or polymer-based clay that have persistence viscosity. For simplicity, it will be referred to using the word "clay" in the following text.

Project I: FrameForm—Robotic Fabrication of Structural Metal Frames. This project aims to develop an automated pipeline for a pre-defined robotic fabrication process to fabricate structurally sound and topologically efficient metal frame structures that can support additional loads under the designed loading conditions. It inherits the legacy hardware from the *Mesh Mould* project (Hack, Wangler, et al. 2017) and focuses on the development of an optimization-based computation system. As the legacy hardware is already a complete tool set with little space for major modification and has a fixed motion sequence for fabrication, this project will take these characteristics as existing constraints and focus on developing geometry modification methods to suit the integrated needs. The computational strategy of this project addresses structural optimization considering robotic fabrication constraints. Integrated approaches are thus needed for modelling geometric and material properties, as well as robotic fabrication processes of a specifically designed end effector to build a combined optimization that can resolve the problem as a whole.

Project II: RoboSculptor—Robotic Clay Sculpting. This project aims to provide a novel approach for integrating user-defined design expressions as sculpting styles and robotically sculpting the geometric details of a given input with clay materials. In contrast to *Project I*, where legacy hardware is given, new hardware is built during this project, and co-evolves with the software development. The project furthermore addresses adequate physical experiments to understand the material properties and abstract essential parameters to support the formulation of the optimization problem involving the specific fabrication process. This project will focus on geometry processing methods for the design stage, and integrate robot path planning for the fabrication stage. The integration of design preference of the user also belongs to the objectives, so as to establish a feature-rich system that offers more than non-interactive solutions. The consideration fabrication process and utilization of fabrication constraints during the design stage through the means of optimization establish a fabrication-aware design process that smoothly facilitates the materialization of the design intention.

1.3.2 Interdisciplinary approach

This dissertation involves research methodologies and tools closely related to both the architecture domain and the Computer Graphics (CG) domain and is funded by Disney Research. By evaluating and examining the available methods in an interdisciplinary context, the dissertation will benefit from both domains, and hopefully originate valuable outcomes in return. The dissertation was developed collaboratively and in tight exchange with the chairs of Gramazio Kohler Research (GKR), Computational Robotics Lab (CRL), and Disney Research.

The legacy hardware for *Project I* was developed by the groups of GKR at the Institute of Technology in Architecture and Agile and Dexterous Robotics Lab (ADRL) at the Institute of Robotics and Intelligent Systems, embedded within the NCCR Digital Fabrication. In the context of the research of this dissertation, this hardware set serves as the fabrication tool and constraints for developing the optimization system that utilizes various methodologies from the CG field.

The research conducted in this dissertation enjoys a close collaboration between the author and Alexander Walzer (for *Project I*) and the Ph.D. candidate Simon Dünser (for *Project II*) from the CRL, and with additional technical support from the Computational Design and Manufacturing Group at Disney Research. Mr. Walzer conducted all the fabrication tasks and the upgrade of the hardware in *Project I*. Mr. Dünser led the formulation of the optimization objectives based on the optimization platform he developed in Duenser et al. (2020), which is also partially used in *Project II*. Dr. Christian Schumacher and Dr. Espen Knoop provided coding guidance for *Project I* and developed the turntable and manufactured the end effectors for *Project II*, respectively.

More details for the two projects can be found in Section 4.8 and Section 5.9, respectively.

1.4 Dissertation Outline

This thesis is structured into six chapters. Following this introduction chapter, Chapter 2 covers the state of art research for optimization augmented fabrication in architecture and the CG domain. Specifically, it provides an overview of the geometry representation and processing in architectural fabrication, material fabrication and modelling for simulation, structure optimization for discrete structures (*i. e.* truss and frame), and path planning for standard robot arms.

Chapter 3 provides the mathematical fundamentals that help to understand the following chapters better due to the use of mathematical-heavy methods that are not usually employed for architecture research. The chapter includes a brief introduction and summary of optimization formulation and solving strategies, and some basics of vector field mathematics.

Chapter 4 presents *Project I: FrameForm*, an interactive design system that utilizes computational techniques to aid the design of structurally-sound metal frames. The system is tailored for robotic fabrication using an existing fabrication process that integrates automated bar bending, welding, and cutting. The project minimizes combinations of functional and aesthetic objectives under strict fabrication constraints that model the assembly of discrete sets of bent metal bars.

Chapter 5 presents *Project II: RobotSculptor*, an interactive design system that allows users to create sculpting styles and fabricate clay models using a 6-axis robot arm. The project allows the user to interactively select sub-areas of the mesh through decomposition and embeds the design expressions into an initial set of toolpaths by modifying key parameters that affect the visual appearance of the sculpted finish. The toolpaths are then optimized to find the robotic sculpting motions that match the target surface, maintaining the design expression, and resolve collisions and reachability issues so that a robot can fabricate the clay model physically with a customized loop tool.

Chapter 6 concludes the dissertation with summarized remarks and discusses possible directions for the future research.

The Appendix provides supporting materials for the dissertation, including the design iterations of the tool development of *Project II*, implementation details of the software developed during the whole dissertation, etc.

Chapter 2

Context

Seldom do more than a few of nature's secrets give way at one time.

— Claude Shannon

The objective of this chapter is to contextualise the research with both historical and contemporary references and examples. As stated in the first chapter, for the optimization-based methods targeted in this dissertation, the simulation of the fabrication processes is needed. Thus, three key elements are required: 1) the geometric representation of the involved objects, such as parts of the robot, the target object to be fabricated, essential collision objects, etc., 2) the simulation of the fabrication process, in which both the interaction between the tool and the target and the robotic process is modelled, and 3) the mathematical formulation and computation of the optimization. The structure of this chapter will follow these requirements, contextualizing the related topics based on the projects, and conclude by introducing the focus areas of the research.

2.1 Geometry Representation and Processing

Geometry is the core for almost all the modern field that involves Computer-Aided Design (CAD) process to aid modelling. While different types of representations and modelling techniques, such as Constructive Solid Geometry (CSG), Boundary Representation (BREP), Non-Uniform Rational B-Splines (NURBS), etc., enjoy different benefits and are used for different modelling purposes, the research in this dissertation chooses the general triangle mesh data, due to the vast amount of available tools existing in the CG field for low-level operation of the model data. Since this type of representation only stores vertices by coordinates and the edges and facets by vertex relationships, it also enjoys a more efficient access to the shape information compared to other parametric modelling representations, and benefits computation cost in the optimization process.

While designers and researchers in the architecture field rarely use the triangle representation for modelling, this section mainly focuses on the CG field for mathematical methods and tools on this topic, in which triangle mesh models have been studied and used as a major type of geometric representations for decades. As both of *Project I* and *Project II* employ a similar set of tools for geometry processing, the following paragraphs provide a general overview of the ones related to this dissertation:

2.1.1 Vector Integration and Field Interpolation

Geometry processing for digital fabrication often requires consideration of additional information such as loading conditions, structural strength, material properties and fabricability due to either the specification of the projects or the physical laws of the real world. The information usually exists in or can be transformed to a form of *vector field*, which requires it to be integrated into the respective representation of the input geometry. Additionally, certain information also needs to be extracted or interpolated to obtain the required data format for further processing. Among those used in the CG field, the *Lapacian-Beltrami operator* and the *Poisson's equation* are probably the most widely applied ones.

Generally known as a mathematical differential operator on scalar fields, the Laplace operator is most widely researched and used in its discrete version in the CG field, due to the discrete nature of computer science. It has become an essential tool in geometry processing. Many interpretations and flavors of the Laplace and Laplace-Beltrami operator exist, metaphorically known as the "swiss army knife" of geometry processing (Solomon, Crane, and Vouga 2014). The discrete version of Laplacian operator for triangles elements has been researched intensively for geometry processing (Sorkine, Cohen-Or, et al. 2004; Sorkine 2005) and re-meshing (Nealen et al. 2006), and recent development also extends to general polygon mesh (Alexa and Wardetzky 2011; Bunge et al. 2020).

The *Poisson's equation* on the other hand, is a generalization of the *Laplace's equation* and relates non-zero information on a manifold. There is a wide range of applications in various fields including image editing (Pérez, Gangnet, and Blake 2003), geodesic distance computing (Crane, Weischedel, and Wardetzky 2013), light routing (Pereira, Rusinkiewicz, and Matusik 2014), surface reconstruction (Kazhdan, Bolitho, and Hoppe 2006; Kazhdan and Hoppe 2013), etc.

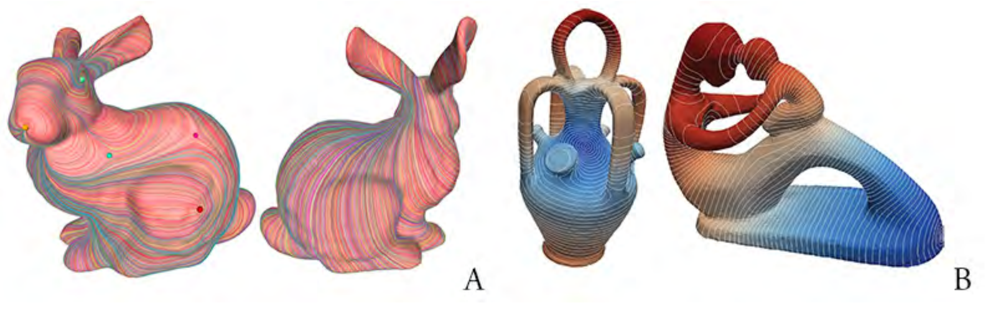

FIGURE 2.1: **Applications of Laplacian-Beltrami operator and the Poisson's equation**: A. Mesh Parameterization (Laplacian) (Solomon, Crane, and Vouga 2014); B. Vector Field Design (Crane, Desbrun, and Schröder 2010).

2.1.2 Shape Abstraction

Austern, Capeluto, and Grobman (2018) provided a great overview of the rationalization methods used in computer aided fabrication. The main aim of the rationalization is to connect the virtual design space with the physical world through the language of geometry. The different geometric representation and the translation in between are the keys in this connection and evolve a wide range of tools for various purposes. For digital fabrication, the rationalization process is determined by the material and fabrication methods used in the specific project. Shape abstraction, as the crucial translation from the input geometry to the fabrication geometry, differs in the two projects of this dissertation, due to the different fabrication tools and methods used.

Project I employs the rebar assembly as the final physical format. Methods for translating surface 3D models to abstracted wire models [1] (Figure 2.2) are thus required. A rationalization process similar to Knöppel et al. (2015) is necessary, where equal-distance stripes are extracted from a given manifold. Early work (Mueller et al. 2014) introduced a method to create 3D

[1]The term *wire structures* will be used in the following context.

printed wireframe structures for fast prototyping of shapes. Wu et al. (2016) extended this approach to a 5-DOF system, increasing the complexity of shapes that can be fabricated.

Apart from fast prototyping, there are also other applications of wire structures in shape abstraction. Lira, Fu, and Zhang (2018) proposed a hybrid meta-heuristic model to create such abstractions with as few wires as possible. On the basis of Chebyshev nets, Garg et al. (2014) introduced a method to design wire mesh structures. Focusing on aesthetic aspects of wire-based surface design, Zehnder, Coros, and Thomaszewski (2016) described a technique for the design of visually-pleasing, structurally-sound ornamental curve networks.

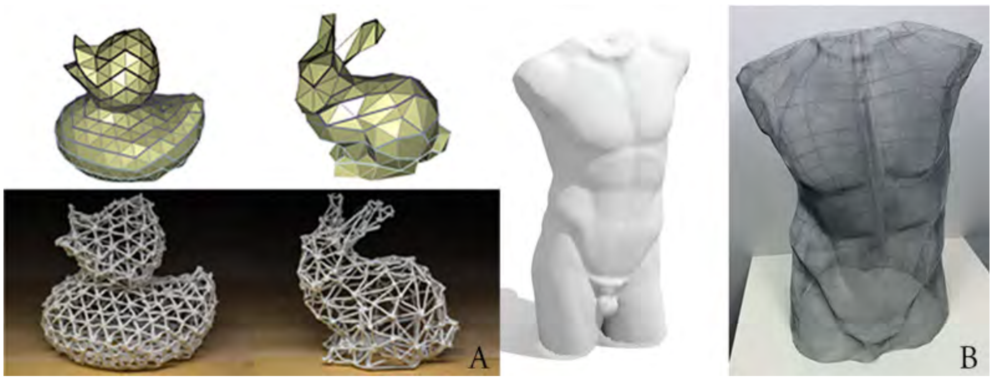

FIGURE 2.2: **Various examples of shape abstraction**: A. Abstraction from mesh models to 3D printed wire structures (Wu et al. 2016); B. Abstractions with regular wire mesh sheet (Garg et al. 2014).

While the fabrication process in *Project I* is conducted in an "additive" style where the final structures are assembled from rebars, *Project II* employs a subtractive process for the fabrication—the final clay model is sculpted from an initial volume of the material. The shape abstraction process happens in the generation of toolpaths, where curves like the isolines (YiHong and YongJiang 2010) are extracted for additional processing. As the toolpaths have a closer relationship to the motion planning of the robot, the related work is discussed in Section 2.3.2.

2.2 Material Fabrication and Modelling for Simulation

As stated in the first chapter, the aim of simulating the fabrication process at design stage is to serve the optimization—finding the right set of parameters desired for the fabrication by conducting virtual tryouts. Thus, the behaviour of the material and the interaction between the robot and the material is crucial to the simulation accuracy of the fabrication.

2.2.1 Steel Rebar

As steel has so many forms of applications, the literature review of the steel fabrication for *Project I* in this chapter is limited to rebar free-form fabrication. Cortsen et al. (2014) introduced a robotic system to create double-curved rebar meshes from a digital model. Hack and Lauer (2014) presented a technique to create 3D printed in-place formwork for concrete. They later extended their approach to enable the design of structures that serve as molds *and* reinforcement (Hack, Wangler, et al. 2017).

From a slight different perspective, the fabrication of desktop scale bar assemblies in different material has also emerged in CG field. Huang et al. (2016) introduced a robotically 3D printed rod assemblies where they assumed a volumetric frame structure as input, and their output is tailored for robotic thermoplastic extrusion. Though *Project I* inherits the tool from the *Mesh Mould* project (Hack and Lauer 2014) and operates on steel rebars where the fabrication process dictates a particular fabrication sequence, the sequencing of instructions could be referenced for other robotic processes (Huang et al. 2016; Wu et al. 2016).

However, from the above review in this section, the design and fabrication of both the rebar structures in the architecture fields and the extrusion-based 3D printing of plastic structures in the CG field rarely consider the material behaviour. While research in the CG field may each concerns different aspects of fabrication, one common cause exists for the architecture fields—architectural design is usually separated from the structural engineering. Structure stability and material behaviour are considered by structural engineers. This separation requires a complex interaction that is prone to human errors and thus leads to a limited amount of possible iterations during a design-to-fabrication process. It may also result in unexpected failure of the fabricated structures or an unnecessarily redundant use of the materials as an inclusion of safety margins to mitigate risks.

Since the material property of steel has been well studied, the simulation of rebar structure under small deformation should be conducted with ease. A common linear elastic model (Slaughter 2012) with steel property should suffice for the simulation, as long as the structure is deformed within a relatively small range under the designated load case. Thus, integration of such material simulation should not take much additional efforts, and provides extra guarantee to the structural stability of the fabricated structures, as well as more sustainable use of the materials.

Note that once the rebar is combined with other materials in a composite manner, say reinforced concrete, the behaviour of composite is more complex and needs to be researched according to the specific contact interface (Beliaev et al. 2016; Chiriatti et al. 2019; Cosenza, Manfredi, and Realfonzo 1997).

2.2.2 Clay

The *Project II* employs a different material medium, clay, as the operational material. As a representative material for geometry forming, clay has been widely used in arts and crafts fields as a hand modelling process (P. Faraut and C. Faraut 2013; P. Faraut and C. Faraut 2009). Recently, its economical and malleable characteristics also gained increasing popularity in 3D printing and multi-axis robotic applications. These applications typically employ a customized tool attached to a common 3-axis CNC machine (Nan, Patterson, and Pedreschi 2016) or a 6-axis industrial robot arm (Bechthold 2016) and manufacture artefacts through additive, subtractive, or formative processes.

The additive process usually deploys clay either in layers to create sealed surface geometries

(Rael and Fratello 2017; Tryfonos 2018) or in a woven style (Friedman, Kim, and Mesa 2014; Rosenwasser 2017) to create patterns. Taking advantage of the material's malleability, digitally controlled throwing of the clay was also studied in order to erect large-scale building structures over distance (Dörfler et al. 2016).Subtractive and formative processes, however, generally start with an initial block of clay which is shaped by either placing pressure to deform the material (Tan 2016), or cutting off the material (Schwartz and Prasad 2013). Weichel et al. (2015) combined the additive and subtractive processes, *i. e.* milling, using two distinct tools.

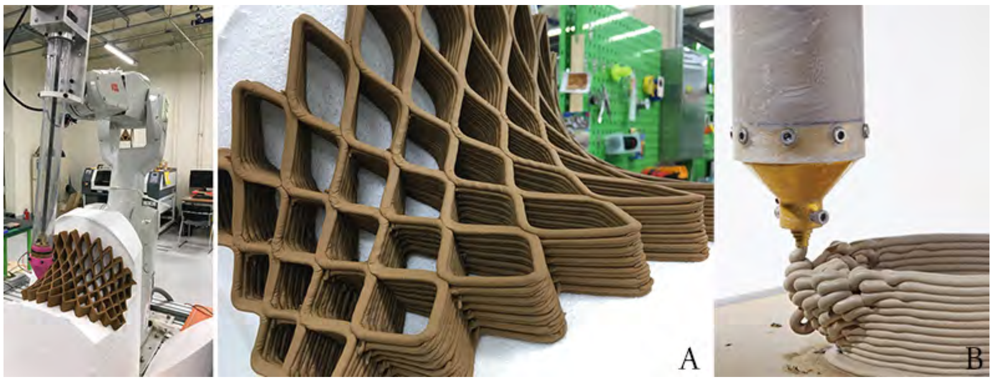

FIGURE 2.3: **Clay Fabrication and Failure**: A. 3D printed clay object on a customized doubly-curved base (Ko et al. 2019); B. 3D printing clay collapses during the fabrication. (Rael and Fratello 2017).

However, the application of material simulation is even worse for the fabrication with the clay material, and pure toolpath-based 3D printing of clay will cause fabrication failure due to either geometric or material reasons (Figure 2.3.B). Some research for simulating clay material exist in a molecule level (Marry, Rotenberg, and Turq 2008; Skipper, Chang, and Sposito 1995; Young and D. E. Smith 2000), but little practical work has been found for simulating clay in a larger scale that related to the fabrication process in *Project II*. The constant evaporation of water inside the clay also increases the complexity of simulation.

On the other hand, fabrication experiments in Section 5.4.1 showed that for the fabrication process with the tools the author developed, clay behaves almost like a quasi-plastic material—a pure geometry-based method is thus chosen to ease the simulation process. Thus, the simulated fabrication only depends on the Boolean process between the base geometry and cutting geometry where deformation can be ignored.

2.3 Optimization

While optimization-based methods include too large a body of work, the projects in this dissertation exploit specific optimization-based methods to facilitate robotic fabrication processes. This section will review the related methods separately for the two projects.

2.3.1 Truss and Frame Optimization

The fabrication method for *Project I* sets hard constraints on the representation of the target, and consequently determines its structural type—trusses and frames[2].

The analysis of trusses and frames is well-understood, and has been a central topic in structural engineering (Logan 2011). The optimal design of such structures is an active area of research, with the seminal work on *Michell trusses* (Michell 1904) serving as the source of inspiration for more recent work. Truss optimization (Achtziger 1999; Achtziger 2007), where combinations of shape, topology, and sizing parameters are optimized, has been addressed with a variety of methods (Stolpe 2016).

Targeting the CG field, J. Smith et al. (2002) proposed a system for the synthesis of truss structures. Tang et al. (2014) explored the use of polyhedral meshes for truss optimization, and target applications in architecture, and Jiang et al. (2017) introduced a method to find material-efficient space structures.

Because forces *and* moments are transferred at welded connections, *Project I* relies on *frames* in physical modeling. In contrast to standard synthesis and optimization of frames,

[2]The terms "truss" and "frame" used in this dissertation refer to their definition in structural engineering: joints in a *truss* are *pin* joints that can only take axial force loads (constrained in transition, unconstrained in rotation) and joints in a *frame* can be *moment* joints that can take both axial force loads and moment loads (constrained in both transition and rotation).

robotic processes conducted by the hardware in *Project I* severely restrict the types of feasible structures. Hence, they require a different modeling. However, both the process-aware, structurally-informed initialization of globally-coherent frames (global step), and the fabrication-aware co-optimization of strength-to-weight ratio, target matching, artistic, and regularity objectives (local step) are applicable and can be easily transferred to other robotic processes.

While the initialization and optimization developed here are tailored to robotic construction processes, this dissertation shares goals with work on material-minimizing forms and structures (M. Kilian et al. 2017), and align bars with principal stress lines (Pellis and Pottmann 2018; Pietroni et al. 2015).

2.3.2 Motion Planning

Project II seeks for a set of motion parameters that controls the robot to conduct the corresponding sculpting behaviours, which, in essence, is a motion planning problem.

The general problem of robotic path planning has been studied extensively in the past, and respective software is readily available. For example, the Open Motion Planning Library (Şucan, Moll, and Kavraki 2012) provides a collection of sampling-based algorithms to plan a feasible path between two points, subject to optimality conditions. The *Descartes* package of the ROS-Industrial project (Edwards and Lewis 2012) implements a tree search to find a robot trajectory that matches a suite of prescribed tool positions.

Unlike these applications, however, the planning for *Project II* calls for long trajectories with highly dense sampling, in order to accurately follow the fine-scaled details of the target shape. Further, the entire length of the cut path is heavily restricted by collision constraints and computationally expensive optimization criteria. Such conditions are inherently challenging for sampling-based approaches. Notably, De Maeyer, Moyaers, and Demeester (2017) reported that already for 50 trajectory points, memory usage starts to become a matter of concern for the tree search used by *Descartes*. In *Project II*, this number routinely exceeds ten-fold. Therefore, iterative optimization is chosen, namely Newton's method[3], to handle the large number of parameters—albeit at the expense of global optimality.

While motion planning plays an essential part of *Project II*, the embedding of the design expression requires additional information to be incorporated into the planning process. The information is embedded into an initial set of toolpaths that are used as the starting point of the planning. A large body of work about toolpath generation is available in CNC machining (Chiou and Lee 2002; Feng and Li 2002; Jun, Cha, and Lee 2003; Sullivan et al. 2012; Tournier and Duc 2005; Zhu and Lee 2004). Dragomatz and Mann (1997) surveyed general path generation methods used in the CNC milling field, and Elber and Cohen (1994) summarized two main approaches, *isocurves* and *contours*, and their strengths and weaknesses.

Robotically manipulated wire-like tools have recently been studied in Duenser et al. (2020), where an elastically deformable, heated rod cuts through blocks of polystyrene foam. This work focuses on trajectory optimization for a small number of individual cuts using a comparably large tool, rather than on a global cutting strategy. In contrast, the tools employed in this dissertation are much smaller in size, such that a global strategy for path generation is necessary. Nevertheless, inspiration is drawn from this work for the path planning component in this dissertation, and feasible robot trajectories are optimized in a similar fashion.

2.4 Conclusion

The previous sections provide a review of the state-of-the-art computational tools and methods available in both architectural and CG fields, specifying the three key elements for developing an optimization-based system. As a result, some research areas that show potential for expanding the present methods have been identified:

[3]A common root-finding algorithm used in numerical analysis, also known as the Newton–Raphson method.

For *geometry representation and processing*, both of the two projects mainly exploit the use of existing tools to facilitate the translation and integration of geometric information between the input data and the targeted fabrication data. However, research into how to design and develop customized Graphic User Interface (GUI) to facilitate the ease-to-use intervention of the geometric data transformation pipeline may bring new opportunities to advance the geometry processing strategies for robotic fabrication.

For *Material Modelling and Fabrication Simulation*, *Project I* investigates the material-based model by using an elastic model of the steel material in the simulation components, while *Project II* employs the geometry-based model due to the complex material behaviour of the clay material. Neither of the investigations has been conducted for robotic fabrication research in the architectural domain. Developing methods that integrate such simulation will have the potential to resolve many concerns emerged from previous research, and will expand existing ways of managing robot-target intervention.

While *optimization* is a grand topic to explore, the approaches required in this dissertation mainly serve for robotic fabrication processes, and depend largely on the flexibility of the fabrication methods. For *Project I*, research into the specific optimization for the frame structures with additional objectives of topology and geometric proximity will expand the existing pool of frame optimization methods, and explore the possibility of combining tools available in both engineering field and CG field into an integrated system that serves the need of combined optimization of interests, including structure, fabricability, topology, shape proximity, and other additional constraints.

On the other hand, while many existing path planning algorithms are available, little research has been done for planning paths based on specifically initialized toolpaths or how to initialize toolpaths based on specific needs. The research in *Project II* will examine these areas. The research of the novel toolpath generation approach will allow for embedding additional user-defined information with less emphasis on the machining time or toolpath length (the main consideration in CNC machining). This collection of research will provide a different perspective to look at path planning problems where machine accuracy and time efficiency are no longer the main planning goals. As a result, the toolpath generation and planning approaches will be expanded to serve the diverse needs in different design tasks.

Chapter 3

Fundamentals

Almost all processes that are not obviously simple can be viewed as computations of equivalent sophistication.
— *Stephen Wolfram* , A New Kind of Science

As research conducted in *Project I* and *Project II* contains a large amount of material related to the CG field, and employs methods that are mathematically heavy and not often used for research in the architectural field, this chapter aims to provide a general high-level explanation to some of the important concepts used in the following chapters, aiming to assist the reader with a better understanding of the whole dissertation.

Section 3.1.1 provides some basic explanations of optimization problems without any mathematical terms, and are aimed to help readers with limited mathematical background. Section 3.1.2 tries to connect the context described in Section 3.1.1 with the methods used in the following project chapters with more mathematical contents. The readers are free to skip this section without affecting the understanding of the rest of this chapter.

Notice that this chapter by no means serves as a tutorial or a beginner's guide to the corresponding topics. Please refer to other more complete materials for the details.

3.1 Optimization in a Nutshell

3.1.1 Optimization in Layman's Terms

What is Optimization? Optimization is the process of choosing the optimal solution from all the feasible solutions. Optimization is something that people do all times in their routine life while making decisions: a customer may check the date labels on the milk bottles and select the one that was produced latest; a clerk may choose the transport route that takes the least time or the minimal number of transits to work (with the help of the map software on his phone); one may pick the most suitable clothes after checking the weather report to keep warm or stay cool.

In the architecture field, optimization processes have also been applied in various contexts, such as optimizing the architectural plan layout for more sunlight facing windows, optimizing the position of the load-bearing wall for targeted spatial requirements, etc.

While the optimization processes can be conducted by human beings, as shown in the above examples, the computation speed of human brains is much slower than that of modern computers. However, in order to ask computers to conduct these tasks, it is necessary to define the problem in a way that computers can understand, and develop algorithms for the computers to find optimal solutions following procedural commands.

How to define the optimization problem? An optimization problem will have an objective function, as known as the target, which is meant to be either minimized or maximized mathematically. The objective is a function of the decision variables (the available choices). For instance, in the above milk example, the objective function is the number of days from the manufacturing date to today, and the decision variables may be the number of bottles the customer may check.

Additionally, the objective function is usually maximized or minimized subject to some constraints. Constraints are restrictions like the total time the above customer has to buy milk, or the oldest date that the customer can accept. Similar to objective, constraints are also a function of decision variables. An optimal value is a feasible solution that generates the minimum or maximum objective function value. An optimal solution must satisfy the constraints and give us the largest or smallest value of the objective function, depending upon whether it's a maximization or minimization problem. The milk example is a minimization problem since the less the day difference from the manufacturing date to today, the fresher the milk is.

How to formulate optimization problems? To abstract a problem in reality into the mathematical form that a computer can understand, four steps are usually conducted to help formulate an optimization problem:

1. *Define the problem*—first, one starts with the definition of a problem and see what one is trying to solve, for example, minimizing the cost, reducing the time, etc;

2. *Define the decision variable(s)*—next, one should define the decision variables, usually by looking into the problems and selecting the variables that are under control and are likely to influence the solution to the given problem;
3. *Formulate the objective function*—afterwards, one should formulate the objective function as an equation that consists of how decision variables relate to the solution in terms of minimizing or maximizing. The selection of minimization or maximization depends largely on the characteristics of the objective, such as minimizing the cost, or maximizing the profit margin;
4. *Define the constraints*—the last step, sometimes optional, is to define all the constraints in terms of the number of variables that can be produced, or in other words, what are the restrictions to the solution?

After following these four steps, an initial mathematical formulation will be obtained so that some algorithm(s) can be used for the maximization or minimization process. In reality, the final formulation is usually built up through an iterative process by starting simple and moving forward by adding more general types of restrictions to the problem.

3.1.2 A Brief Intro to Mathematical Optimization Problem

This section will provide a brief introduction of the mathematical formulation of optimization problems[1], in order to assist readers with limited mathematics backgrounds to get a glimpse of the mathematical form of an optimization problem, and how it relates to the context introduced in Section 3.1.1. It introduces the simplest form of an optimization problem, from which the formulations in later chapters derive.

[1]This introduction is simplified for ease of understanding, and may not be precise in the mathematical sense. For a more detailed definition of numerical optimization problems and approaches to solve them, please refer to Nocedal and Wright (2006).

As mentioned above, optimization refers to the class of problems that consists in choosing the best among a set of alternatives. Even in this simple, imprecise statement, one can identify the two fundamental elements of an optimization problem:

- *best choice*, that conveys establishing a criterion to choose the solution; this is usually expressed by means of an objective function to be minimized (or maximized);
- *alternatives*, that refer to the set of possible solutions, usually determined by a set of constraints—-equalities and/or inequalities that must be satisfied by any candidate solution.

Therefore, an optimization problem may be posed as follows:

$$\min_{x \in S} \quad f(x) \tag{3.1}$$

where S is the set of possible solutions and $f\colon S \mapsto \mathbb{R}$ is the objective function. Equation (3.1) is restricted to minimization problems since any maximization problem can be converted into a minimization formulation by just replacing the sign.

Additionally, restrictions, if apply, are usually expressed in the form of equality or inequality constraints in the mathematical formulation of the problem:

$$\begin{aligned} \min_{x \in S} \quad & f(x) && \text{(3.2a)} \\ \text{s.t.} \quad & g_i(x) = 0, \quad i = 0, \ldots, m, && \text{(3.2b)} \\ & h_i(x) > 0, \quad i = 0, \ldots, n && \text{(3.2c)} \end{aligned}$$

where Equation (3.2b) is a set of equality constraints and Equation (3.2c) is a set of inequality ones. The algorithms of solving these optimization problems are omitted here for ease of understanding, since it is not the purpose of this chapter.

Problems with the above formulation are called *constrained optimization* where a successful solution to these problems will guarantee all the constraints are met, besides obtaining the minimum objective. Comparatively, problems formulated in the form of equation (3.1) are called *unconstrained optimization*.

Scientists have developed various types of optimization solvers to solve problems formulated into the corresponding type. In general, a *constrained* optimization problem usually takes longer than an *unconstrained* version with similar formulation due to the approaches by which they are solved.

In many engineering scenarios, many constraints are "soft" rather than "hard", allowing the restrictions in reality to be met as much as possible, but not strictly met. Thus, some of the restrictions can be formulated as *constraint objectives*, namely *regularizers*. Hence, an alternative formulation of a *constrained* problem is given as:

$$\min_{x \in S} \quad f(x) + \alpha g(x) \tag{3.3}$$

where $g(x)$ is the *constraint objective* and α is the weight to control the magnitude that the constraints are met.

For the optimization problems defined within this dissertation, only the continuous types are considered, where S is "continuous". In other words, S is a subset of $\mathbb{R}^n$ determined by a finite (possibly empty) set of equalities and/or inequalities.

Besides, *Project I:@ FrameForm* employs a *constrained* formulation as some of the restrictions abstracted from the hardware need to be strictly met, while *Project II:@ RobotSculptor* employs a *unconstrained* formulation.

3.1.3 Optimization Involving Fabrication Process

From the discussion in Section 3.1.2, an optimization problem has been illustrated simply as the following diagram Figure 3.1, where the optimization process will modify the initial data to the optimized data by fulfilling the restrictions and the optimality criteria.

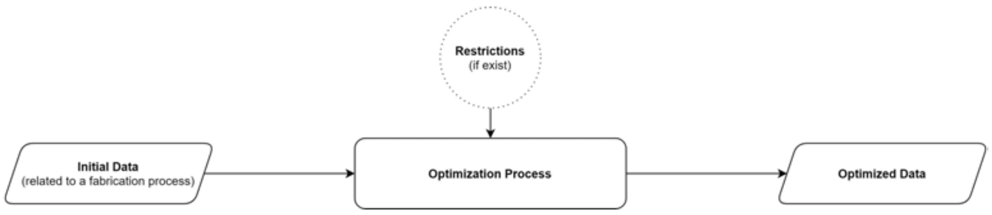

FIGURE 3.1: **A simple illustration of an optimization process.**

Fabrication-aware optimization has no major difference from other types of optimization mathematically, except for the mechanism that the optimization process works inside the iteration loop.

To obtain the optimized data, the optimization simulates how the initial data changes during the fabrication process, and modifies it based on extracted information from the simulated results. For continuous optimization problems, gradient-based iterative methods are usually employed, including the *gradient descent* method, the *Newton's* method, the *Quasi-Newton* method, the *Broyden–Fletcher–Goldfarb–Shanno (BFGS)* algorithm, etc.

The stopping criterion is usually related to the gradient of the objective function $f'(x)$ that when the objective function reaches its minimum (either local or global), the gradient will usually be zero. In practice, a small value ϵ is usually used as the stopping condition, due to numerical noises.

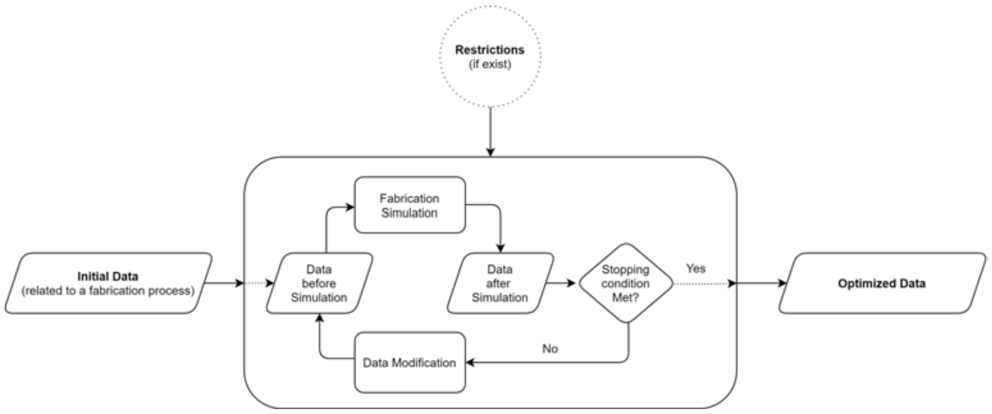

FIGURE 3.2: **How the required data is optimized in a fabrication-aware optimization process.**

Thus, a more detailed version of Figure 3.1 related to a fabrication process is shown in Figure 3.2, where the optimization iteratively modifies the initial data until the process stops and outputs the optimized data.

3.1.4 Data Initialization for Optimization

One question may be raised from Figure 3.1 and Figure 3.2 what the "Initial Data" actually is. In the context of robotic fabrication, or the context of this dissertation, to be more specific, the "Initial Data" refers to the data directly usable for the optimization process.

As the input data is usually geometric data, *i. e.* geometry models, vector fields, etc., some geometry processing steps need to be conducted to process the input data and extract the information desired into the form that the optimization can take. The input data for *Project I* is the surface geometry whose shape the final frame structure should match, and the input data for *Project II* is the base model that the system would like to sculpt with customized design styles.

Additionally, both of the projects (see Chapter 4 & 5) utilize a "divide-and-conquer" strategy, or "global-to-local" strategy in other words, in the processing stage: the input geometric model is first decomposed into smaller sub-elements and various methods are then applied to these sub-elements individually. These two conceptual steps are referred to as *Decomposition* and *Initialization* respectively, and the initialized data of each sub-element is then directly usable for the optimization process.

This "divide-and-conquer" strategy benefits the research in two ways that 1) it allows finer level of control and adjustment to the sub-area of the input model without losing generality (the methods applied to different sub-element can be either different or same), and 2) it helps to control the scale of the optimization problems, especially when the variable-time relationship exceeds quadratic ones.

Indubitably, the combination of individually optimized sub-elements is usually not the global optimal solution (as in *Project I*), and decisions need to be weighted with other factors as well. For more details about the specific tools and techniques in each of the projects, please refer to the corresponding sections (Section 4.4, Section 5.5).

3.2 Vector Field Mathematics

The content in this section aims to provide a high-level overview on some of the methods and tools about vector field processing used in the two projects in the following chapters. The discussion here focuses specifically on the tangent vector fields for triangle meshes. Much of the content is extracted from the SIGGRAPH 2016 course notes *Vector Field Processing on Triangle Meshes* (de Goes, Desbrun, and Tong 2016) and the tutorial of geometry processing library *libigl* (Jacobson, Panozzo, et al. 2018), and reorganized or re-summarized based on the topics of this dissertation.

The vector fields used in this section are limited to face-based vector fields, where each triangle corresponds to one vector. As a triangle is flat, it is particularly convenient to define a tangent vector on its supporting plane. It is also not difficult to define discrete notions of divergence, curl, or even Laplacian of these vector fields with this setting so that the mathematical tools for processing continuous vector fields can be translated to discrete triangle mesh processing with ease.

The vector fields used in this dissertation are mostly related to the stress information of a materialized shape under given loads, or the curvature information embedded in the geometry.

3.2.1 Differential operators of Vector Calculus

For vector calculus, there are three basic differential operators defined on scalar or vector fields, and, in a way, “transform” one type to the other.

Gradient The gradient is defined on scalar fields and measures the rate and direction of the fastest change in that scalar field.

For a function $f(x, y, z)$ defined in the three-dimensional Cartesian coordinate, the gradient of it is the vector field:

$$grad(f) = \nabla f = \frac{\partial f}{\partial x}\boldsymbol{i} + \frac{\partial f}{\partial y}\boldsymbol{j} + \frac{\partial f}{\partial z}\boldsymbol{k}$$

where $\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{k}$ are the standard unit vectors for the x, y, z-axes, respectively.

The gradient operator transforms a scalar field into a corresponding vector field.

Divergence The divergence is defined on a vector field and measures the volume density of the outward flux of that vector field from an infinitesimal volume around a given point.

For a continuously differentiable vector field $\boldsymbol{F} = F_x\boldsymbol{i} + F_y\boldsymbol{j} + F_z\boldsymbol{k}$, the divergence of it is the scalar-valued function:

$$div\boldsymbol{F} = \nabla \cdot \boldsymbol{F} = \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y} + \frac{\partial F_z}{\partial z}$$

Figure 3.3 illustrated the relationship between a vector field and its divergence.

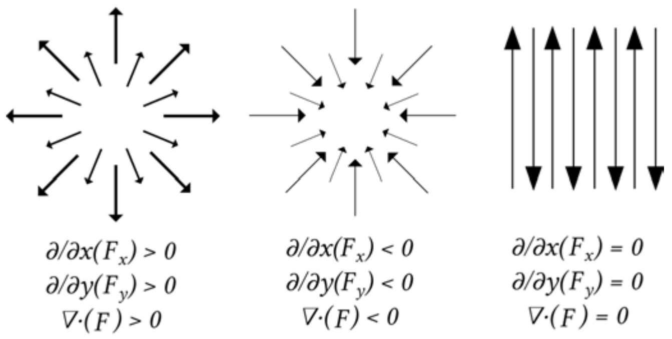

FIGURE 3.3: **The divergence of different vector fields.** Image by Bfoshizzle1; license: Attribution-Share Alike 4.0 International.

The divergent operator transforms a vector field into a corresponding scalar field.

Curl The curl is defined on a vector field and measures the infinitesimal rotation, *i. e.* tendency to rotate, about a point in that vector field. At every point in the field, the curl of that point is represented by a vector, whose attributes, *i. e.* length and direction, characterize the rotation at that point.

For a continuously differentiable vector field $\boldsymbol{F} = F_x\boldsymbol{i} + F_y\boldsymbol{j} + F_z\boldsymbol{k}$, the curl of it is the vector field:

$$curl\boldsymbol{F} = \nabla \times \boldsymbol{F} = (\frac{\partial F_z}{\partial y} - \frac{\partial F_y}{\partial z})\boldsymbol{i} + (\frac{\partial F_x}{\partial z} - \frac{\partial F_z}{\partial x})\boldsymbol{j} + (\frac{\partial F_y}{\partial x} - \frac{\partial F_x}{\partial z})\boldsymbol{k}$$

where $\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{k}$ are the standard unit vectors for the x, y, z-axes, respectively. The direction of the curl is based on the *right-hand rule.*

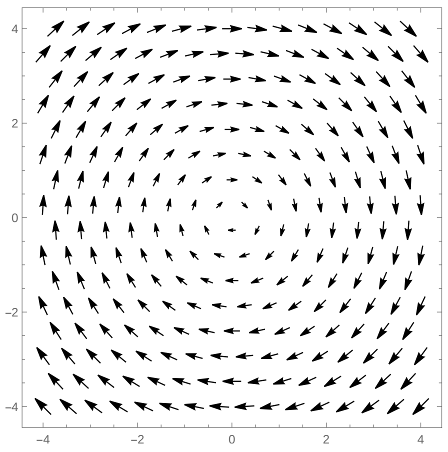

FIGURE 3.4: **The vector field of** $F(x, y, z) = y\boldsymbol{i} - x\boldsymbol{j}$**.** The curl of any point inside this field except for the original point is non-zero, and pointing inwards the paper plane.

The curl operator transforms a vector field into another corresponding vector field.

Laplacian The Laplacian is defined on a scalar field and measures the difference between the value of that scalar field with its average on an infinitesimal volume around a given point. It equals the divergence of the gradient of a scalar field, and represents the flux density of the gradient flow of a function.

For a function $f(x, y, z)$ defined in the three-dimensional Cartesian coordinate, the Laplacian of it is the scalar field:

$$\Delta f = \nabla^2 f = (\nabla \cdot \nabla)f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$$

When the Laplacian of a function is equal to 0, the function is called a Harmonic Function. That is,

$$\Delta f = 0 \tag{3.4}$$

which are called *Laplace's equation*.

When the value is not 0 but a pre-defined function or a non-zero constant, the function is called *Poisson's equation*,

$$\Delta f = g \tag{3.5}$$

where g is often given and f is sought. The *Poisson's equation* is often used to solve a smooth field, that is f, based on its pre-defined boundary conditions, that is g.

The Laplacian operator gives the corresponding property of a scalar field in the form of a scalar field.

3.2.2 Discrete Version of the Operators

For most of the geometries used in CAD, Computer-Aided Manufacturing (CAM) or CG, unless specifically required, only the shape described by the geometric surface matters during processes like modelling, animation, transforming, etc. More precisely, the models are mostly, if not all, topological 2-manifold (with or without boundary) and can be represented in triangle meshes. While the mesh data is stored in a vertex-based format where the coordinates of the vertices are stored, the vales themselves can be treated as a scalar field on the mesh surface—the above-mentioned operators are theoretically applicable.

In practice, discrete versions of the above operators are thus needed for the vertex-based mesh data and hshould ave been intensively researched in the CG field over the past twenty years. For ease of understanding, the discrete construction of these operators will only be briefly introduced here and will not be expanded, and please refer to the *libigl* tutorial (Jacobson, Panozzo, et al. 2018) for more details.

Gradient For a general triangle mesh whose vertices' coordinates are stored as matrix $\boldsymbol{f}$ (which can be seen as the corresponding piecewise linear function defined on the mesh), its gradient can be expressed in the form:

$$\nabla f \approx \boldsymbol{G}\boldsymbol{f}$$

where $\boldsymbol{G}$ is a sparse matrix and can be derived geometrically.

Divergence and Curl As divergence and curl account for the flux and circulation of a tangent vector field around infinitesimal loops in the continuous context, the finite or discrete version of these two operators can be built upon this definition.

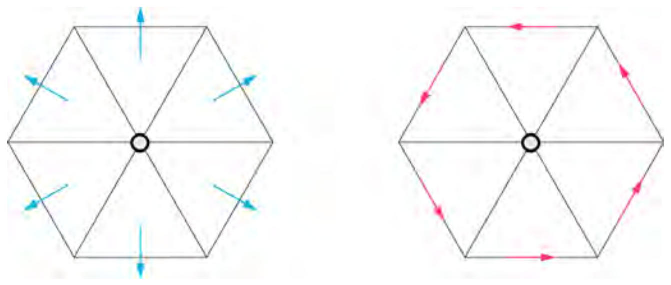

FIGURE 3.5: **Discrete divergence and curl operators can be computed by integrating the total flux and circulation, respectively, of vertex-based vector field around the one-ring neighbourhood of a vertex**.

Figure 3.5 provides one definition of the flux and circulation graphically, based on which the discrete divergence operator can be implemented (de Goes, Desbrun, and Tong 2016). In practice, the divergence operators are often used implicitly, while the curl operators are more often used in fluid dynamics. Please refer to the corresponding subject for more details.

Laplacian For the discrete version of the Laplacian over piecewise-linear functions on the triangle mesh, the most popular one is the so-called cotangent Laplacian $\boldsymbol{L}$, applying divergence theorem to vertex one-rings. As a linear operator taking vertex values to vertex values, the Laplacian $\boldsymbol{L}$ is a $n \times n$ matrix with elements:

$$L_{ij} = \begin{cases} j \in N(i) & cot\ \alpha_{ij} + cot\ \beta_{ij}, \\ j \notin N(i) & 0, \\ i = j & -\sum\limits_{k \neq i} L_{ik}, \end{cases}$$

where $N(i)$ are the vertices adjacent to (neighbouring) vertex i, and α_{ij}, β_{ij} are the angles opposite to the edge e_{ij}. Hence, the discrete version of Equation (3.4) is

$$\boldsymbol{L}\boldsymbol{f} = \boldsymbol{0} \tag{3.6}$$

3.2.3 Applications

Additionally, several applications directly related to the projects in this dissertation will be introduced as follows, aiming to provide a high-level understanding of some of the methods.

Laplacian Equation for Field Interpolation The Laplacian equation in Equation (3.6) is a common linear system in geometry processing, subject to some boundary conditions. One common type is the *Dirichlet boundary condition*, which only fixed some of the values:

$$\begin{cases} \mathbf{Lf}(x) = \mathbf{0}, & x \in \quad \Omega \\ \mathbf{f}(x) = \mathbf{f}_0(x), & x \in \partial\Omega \end{cases} \tag{3.7}$$

where x is the vertex of the mesh, and Ω is the set containing fixed values. Figure 3.6 provides three different scenarios to interpolate scalar fields using this technique.

Poisson's Equation for Field Interpolation In many circumstances, some additional information is required to be incorporated into the field interpolation process mentioned in Section 3.2.3. In such cases, the additional information needs to be extracted as a function and be incorporated with the Poisson's equation.

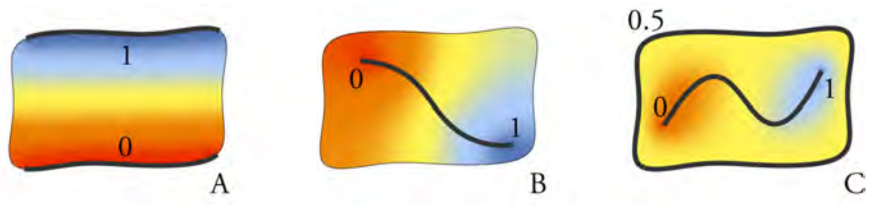

FIGURE 3.6: **Different scenarios for using the Laplacian equation with Dirichlet boundary conditions to interpolate scalar field**: A. Values fixed on top and bottom borders; C. Values fixed along a supporting curve (value interpolated along the curve between 0 and 1); C. Values fixed both on the mesh boundary and the supporting curve inside.

For instance, to interpolate the same scalar field but with the stress field under a given load, the following method can be applied to embed how the stress field is affecting the interpolated scalar field through its divergence, and at the same time keeps the assigned Dirichlet boundary conditions:

$$\begin{cases} \mathbf{Lf}(x) = \nabla \cdot g(x), & x \in \quad \Omega \\ \mathbf{f}(x) = \mathbf{f}_0(x), & x \in \partial\Omega \end{cases} \tag{3.8}$$

where $\nabla \cdot g(x)$ is the divergence of the vector field, defined on the mesh.

The next chapter provides a realistic application of the above scenario.

Poisson Equation and Geodesic Distance Approximation While exact geodesic distance computation (Mitchell, Mount, and Papadimitriou 1987) is expensive and cost $\mathcal{O}(n^2 \log(n))$, approximate geodesic distance can be computed in a much faster way (Crane, Weischedel, and Wardetzky 2013) by solving a heat equation on the mesh surface, filtering the result and then reconstructing a smooth solution by solving a Poisson equation.

Libigl (Jacobson, Panozzo, et al. 2018) provides an implementation of this method, which can be applied repetitively for different heat sources to collectively compute the distance field of a mesh (Figure 3.7).

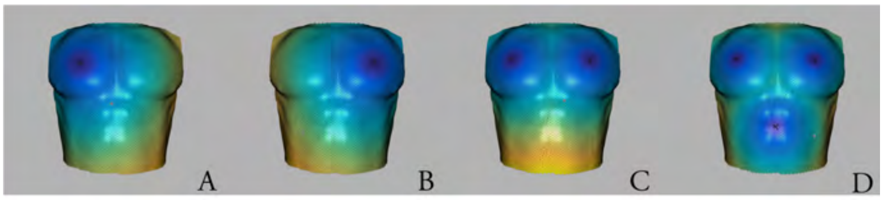

FIGURE 3.7: **Distance Fields created with individual geodesic distance maps from different sources using the heat method**: A,B. Distance maps with different sources; C. The distance fields created with the above two maps; D. The field with an additional source.

3.3 Conclusion

This chapter provides a brief introduction of the mathematical concepts and methods used in this dissertation which are not commonly seen in the architectural fields. While some of the concepts may be simplified and re-interpreted for the fabrication context, the core information should be easily understood.

The following two chapters in this dissertation will show the real world applications of these methods, and help the readers to better understand the concepts introduced in this chapter.

Chapter 4

Project I: FrameForm
Robotic Fabrication of Structural Metal Frames

There are some natures too noble to curb, and too lofty to bend.

— Little Women

4.1 Introduction[1]

Frame structures made of bent and welded steel bars or wires are omnipresent as a great form of abstraction art. Applications of this type include furniture design, concrete reinforcement, sculpture, and architectural components (Figure 4.1). However, while the structures being flexible and powerful expressions of forms, the fabrication of these structures is often time consuming and labor intensive—the bars need to be deformed first and manually welded or fastened node by node to form a structural object that can take its own weight as well as any additional designed load (as for the chair example in Figure 4.1). This challenge applies to the AEC industry as well, and limits the mass application of this type of structures, even though one of the most widespread types of frame structures is the rebar structure for reinforcement concrete.

With the development of robotic fabrication in the AEC fields, robotic construction processes have become more mature and have been proposed for fabricating such structures. For instance, Brugnaro (2016) described a robotic system for weaving soft linear elements using the on-the-fly scanning system, and Hack, Wangler, et al. (2017) suggested a robotic system that seamlessly integrates CNC bar bending, cutting, and welding, requiring only limited intervention by a human (Figure 4.2 left).

Such processes usually encounter unseen levels of complexity in two scales:

1) how the elements are arranged to form the global shape, and
2) how the elements are connected locally at the node level.

Hack, Wangler, et al. (2017) used a regular grid structure with welding connections for simplicity, and put severe restrictions to reduce the complexity of fabricable surfaces. Notwithstanding the success of developing this sophisticated robotic fabrication system, the advantage of the 6-axis robot arm was not fully explored and the surface constructed was quite limited.

[1]Some of the contents of this chapter have been published in Ma, Walzer, et al. (2020)

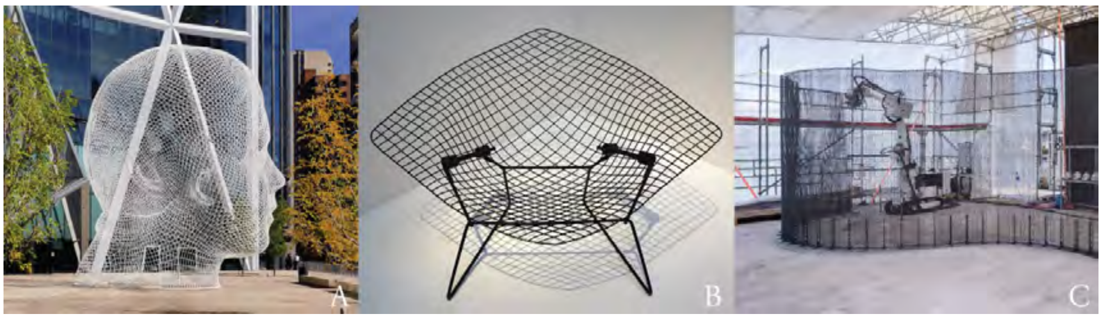

FIGURE 4.1: **Applications of frame structures**: A. The 12-meter tall metal frame sculpture "Wonderland" by Spanish sculptor Jaume Plensa; B. The *Diamond Chair* design by Harry Bertoia; (Image by Sandstein; license: Attribution-ShareAlike 3.0 Unported.) C. Mesh Mould Metal (Hack, Wangler, et al. 2017).

The following three challenges yield potential for future investigations:

- As illustrated in Figure 4.2-right, with a similar regular grid structure, a wider class of surfaces can be approximated by differentiating the space between nodes and adding "inbetween" bars whenever neighboring bars branch too far apart, or removed whenever they become too close, but there lacks methods for systematically conducting this process;
- To approximate large and complex shapes similar to the head in Figure 4.1-A, especially when the scale of the target is larger than the size of the machine, the surface has to be decomposed into smaller fabricable units, and be reassembled after the fabrication of the components. It is still a question if there exists a universal decomposition strategies for different types of geometries;
- It is also unclear how to place and orient bars at the local unit level to achieve an aesthetically pleasing and structurally-sound metal frame at the global scale.

Using the legacy hardware from the *Mesh Mould* project (Hack, Wangler, et al. 2017) and the content discussed above, this chapter describes a fabrication-aware system that aids with the design of robotically-constructed metal frame structures of high geometric complexity. Identifying both the geometric limitations and fabrication limitations imposed by the design

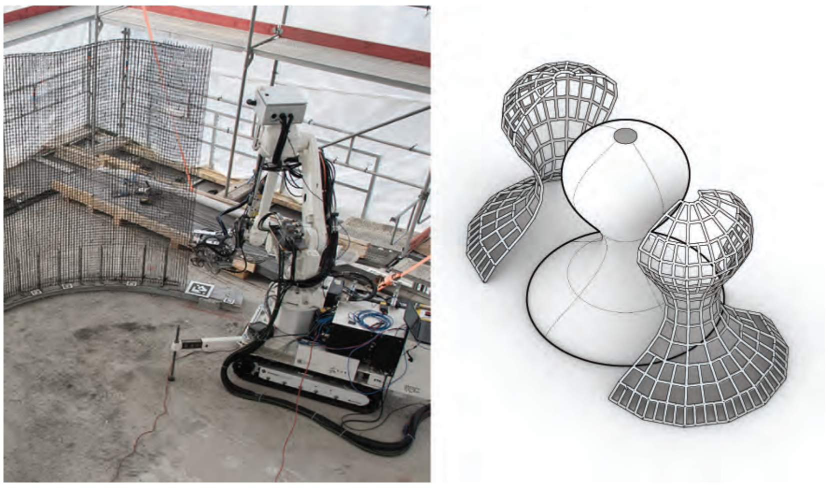

FIGURE 4.2: **The conceptual abstraction of the rebar Frame compared to the *Mesh Mould* project.** Left: the *Mesh Mould*, robotically-constructed rebar for concrete applications (Hack, Wangler, et al. 2017). Right: regular structures are insufficient to achieve high curvature detail, and it is unclear how to decompose complex shapes into fabricable units with globally aligned bar structures.

of the end effector, the system formulates the whole fabrication process as an optimization problem and incorporates these limitations as constraints. By solving this optimization problem, the system successfully provides fabricable results that can be directly fabricated with the robot arm.

Recalling the classification of the constraints described in Chapter 1, this project places more emphasis on the *local fab-constraints* due to the employment of existing parts of the end effector which limits the desired robot motion (the fabrication constraints described in Section 4.5.3) and prevents the execution of the desired movements. *Non-fabrication constraints are also addressed* such that the integrated problem requires the target to be modified (in the minimum amount for preserving geometry proximity) for compensating the successful fabrication of an alike version of the same target.

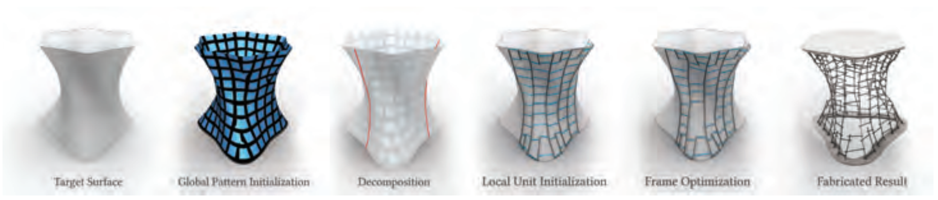

FIGURE 4.3: **Overview.** Given a target surface as the input (Target Surface), an approximate frame structure with a continuous global parameterization (Global Pattern Initialization) is first created to guide an interactive decomposition of the model into smaller units (Decomposition). For each unit, a dense frame structure is initialized (Local Unit Initialization), and the positions and topology of bars are optimized to meet weight, structural, fabricability, and aesthetic targets (Frame Optimization). Individual units are then constructed with a robotic process, and assembled to a monolithic structure (Fabricated Result).

4.2 Overview

As described in Chapter 2, this chapter follows the design-fabrication process that takes a geometry model as the *input model* and processes and optimizes the abstracted objectives while considering both the fabrication and non-fabrication constraints. The geometry processing stage decomposes the model into fabricable pieces and initializes a frame structure with the topology that the hardware can manage, with information obtained from the designed load.

A more detailed overview of the pipeline is illustrated in Figure 4.3. Specifically, given an input shape, an approximate stress field under a designed load case is computed first. This field is then used to guide the initialization of a globally-aligned pattern with which an initial frame structure is generated. To assist the user with the decomposition of the model into fabricable units, the system then provides interactive feedback on where to best place cuts by taking structural and aesthetic considerations into account. Afterwards, each unit patch is optimized by modeling both the fabricability and non-fabricability constraints of the robotic construction process to fulfill structural and aesthetic requirements while remaining as close as possible to the desired target surface. To this end, both the position of welded connections and the inclusions of bar segments are optimized. The efficacy of the global-to-local technique is demonstrated through several examples in this chapter, targeting furniture design or applications in art and architecture.

As this project aims to explore how optimization-based methods can help robotic fabrication when the motion is limited (by the legacy hardware), the inherited end effector, the fabrication setup, and the corresponding hardware modification for this project will be introduced first in the following section to provide the readers with a better understanding of the starting point. Afterwards, the "global-to-local" strategy used in this chapter, namely the *Mesh Decomposition and Frame Initialization* components, will be expanded in detail as a general framework for fabricating targeted structures of different scales. Then, the *Frame Optimization* section follows, with a thorough description of how the objectives are formulated, and how the constraints, either fabricability ones extracted from the hardware and fabrication process or non-fabricability ones related to structural stability and geometry proximity, are extracted and incorporated. The *Demonstrations and Results* section will be presented at the end of this chapter, showing the successfully fabricated examples and the structure stability affected by the decomposition process.

From a general perspective, though this project targets fabrication on a specific robotic process that has already been defined [2], the decomposition, global alignment, and local co-optimization have applications in related processes where a discrete set of entities are assembled into a large-scale structure.

4.3 Fabrication Setup

This section provides an in-depth look at the legacy end effector from the *Mesh Mould* project ((Hack, Wangler, et al. 2017; Kumar et al. 2017)), the update of hardware parts that expands the end effector's range of operations, and the final operations allowed in this project. Mr. Alexander Walzer conducted all the fabrication tasks and the upgrade of the hardware (The decisions of the modification were made based on the author's feedback from the developed

[2]The hardware has actually been slightly modified during the development of the project, allowing movement of one more degrees of freedom to the robot end effector.

computation system and the discussion between Mr. Walzer and the author throughout the initial period of the project).

For ease of understanding, the technical details for the robot arm and rebars are summarized at the end of this section in Section 4.3.4 for reference.

4.3.1 End Effector Legacy

Hardware Before diving into the methodologies and technical details, it is important to gain a better understanding of the robotic fabrication process. As mentioned in Section 4.1, this project inherits the robot end effector of the *Mesh Mould* project as the main fabrication tool. This end effector, or tool heads, as referred in the following context, is mounted on a 6-axis industrial robot arm (Figure 4.4).

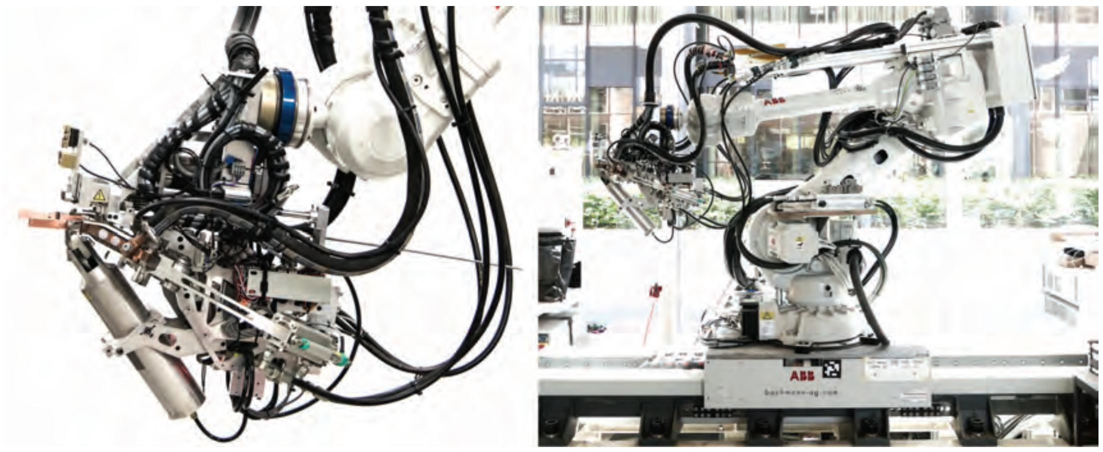

FIGURE 4.4: **A close-up of the end effector and the 6-axis robotic arm with the tool head mounted.**

A descriptive illustration of the functionality of different mechanical parts in the tool head is given in Figure 4.5. The tool head is designed to be multi-functional, and combines different mechanical units to clamp and bend bars, and to insert, weld, and cut bars.

Original Fabrication Actions Developed in the *Mesh Mould* project (Hack, Wangler, et al. 2017), the legacy hardware allows a series of fabrication operations with industrial-level rebars.

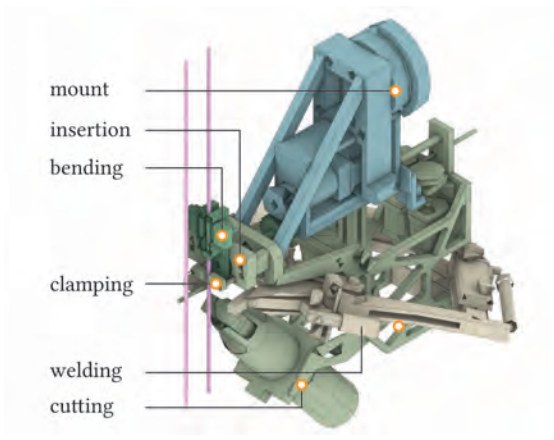

Figure 4.5: **An illustration of different mechanical parts of the end effector.**

The fabrication differentiates between two sets of bars, the long *continuous bars* (magenta in Figure 4.6) that are bent at a discrete set of locations and short *straight bars* (blue in Figure 4.6) that connect pairs of *continuous bars*.

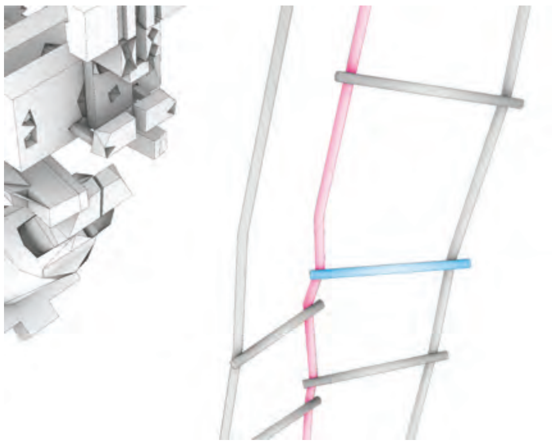

Figure 4.6: **Two types of bars**: the magenta *continuous bars* and the blue *straight bars*.

As illustrated in Figure 4.7, the inherited end effector allows for a series of four actions, namely 1) translating up or down along a *continuous bar*, 2) bending the *continuous bar* with a given angle to the target position, 3) inserting and clamping another straight rebar to two consecutive *continuous bars*, 4) cutting and electric welding the *straight bar* to the two *continuous bars*, Theoretically, the 6-DOF robot arm should allow the end effector to conduct the fabrication process in various orientations, but the actual fabrication task conducted by the end effector in Hack, Wangler, et al. (2017) is fixed in one orientation relative to the fabrication plane (the tangent plane of the surface at the current fabrication location), where the *continuous bars* are aligned towards the vertical direction, for ease of construction, rebar feeding, robot reach, etc.

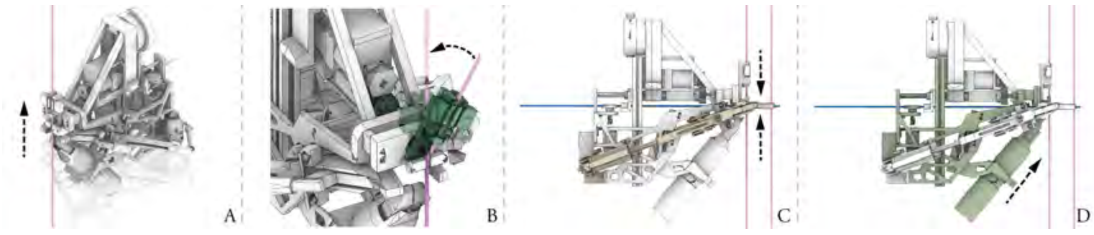

FIGURE 4.7: **A close-up of the end effector and the 6-axis robotic arm with the tool head mounted.**

Conducting these operations, the end effector, as part of the *Mesh Mould* technology, successfully supports the fabrication of a doubly curved reinforced concrete wall (Hack, Wangler, et al. 2017) for the *DFAB House* project (Graser et al. 2020), but also reveals two limitations that hinder the scope of geometry and stability of the fabricated piece:

- the *straight bars* can only be inserted perpendicular to the *continuous bar* where the robot is attached to, and
- the orientation of the end effector is limited to the tangent plane of the target surface, *i. e.* the plane defined by the inserted *straight bar* and the *continuous bar* the robot is attached to, to avoid potential collisions between the end effector and the fabricated part.

The two limitations prevent the end effector from either rotating along the *continuous bar*, or tilting up or down towards the *continuous bar*. Specifically, the first limitation is constrained by the previous hardware, where a transition sleeve guide is attached to the end effector and limits the tilting, while the second limitation is a deliberate choice of the researchers to guarantee a collision-free fabrication process. As a result, the end effector is always oriented in the tangent plane of the target surface at its current location, as shown in Figure 4.7.C/D.

However, these limitations also over-restrict the 6-axis robot arm to a quite limited set of motions, and restrain the geometric complexity of the piece that the end effector can fabricate as well. Hence, certain hardware modifications are needed to eliminate barriers and unleash the capacity of the robot arm[3].

4.3.2 Hardware Modification and Prototype Fabrication

Continuous Bar Release Mechanism To transit along the *continuous bar*, the tool head needs to be related to a reference point on the bar. This was achieved by using a short plastic sleeve of the *continuous bar* as the guide such that the tool head, to which the sleeve is attached, can transit in the direction of the *continuous bar* (Figure 4.8).

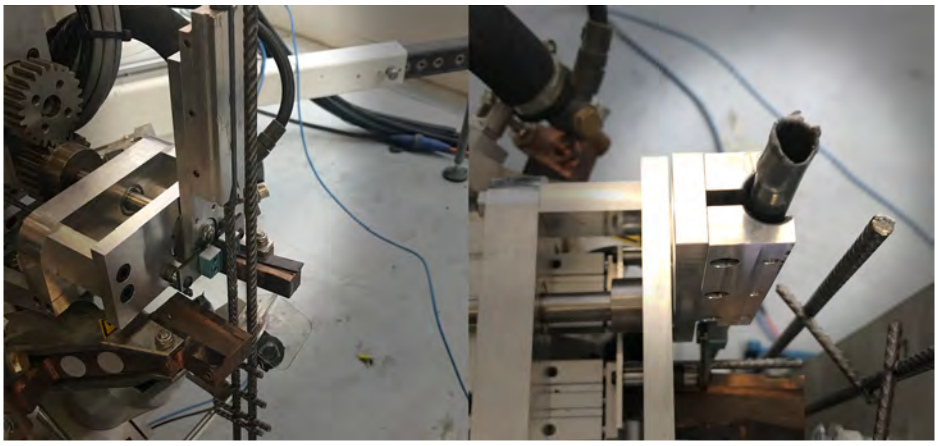

FIGURE 4.8: **The transition sleeve guide from the legacy tool head.**

However, since the sleeve guide is fixed with the tool head, the tool head can only orient to the position defined by the curvature of the target geometry. This constraint is indeed not necessary, as the bending of the rebar and the movement of the robot should be decoupled—the shape of the geometry should be only defined by the bending, and the robot should have more

[3]The hardware upgrade can only eliminate the first limitation. The second one, as a deliberate choice of the previous researchers but not a hardware constraint, requires better computational tools to overcome (Section 4.5).

freedom for manoeuvre. Thus, the upgraded version of this guiding system, illustrated in Figure 4.9, contains two metal pieces which can be closed to form a sleeve-shape guide when transiting, or opened when the robot needs to move without affecting the *continuous bar*.

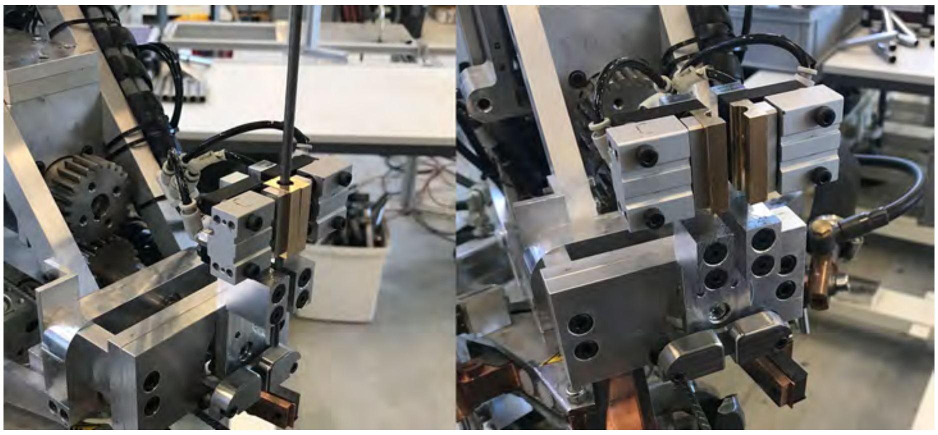

FIGURE 4.9: **The new sleeve guide in its open and closed states.**

Clamp Shape The other mechanism that needs to be modified is the two bronze metal parts attached to the inside of the clamp. These two metal pieces touch the rebar directly during the electric welding, and were originally designed with a cross-sectional shape of a rectangle. Since the welding quality largely depends on the clamping quality, a stable contacting point under certain clamping force is thus important. While the clamp works fine when the tool head is aligned with the target surface and the inserted bar perpendicular to the two *continuous bars*, the metal part on the *straight bar* side will often contact both the *straight bar* and the *continuous bars*, causing a short circuit or weak welding quality.

This issue was resolved by modifying the cross-sectional area of the metal parts into a semicircle, as illustrated in Figure 4.10. Whatever the robot orientation is, the clamp will always contact the centerline of the short *straight bar*, given the tool head is positioned correctly.

Minimum Insertion Distance Previously in the *Mesh Mould* project, the inserted *straight bars* are evenly spaced along the *continuous bars* (Figure 4.11-left), limiting the spacing between

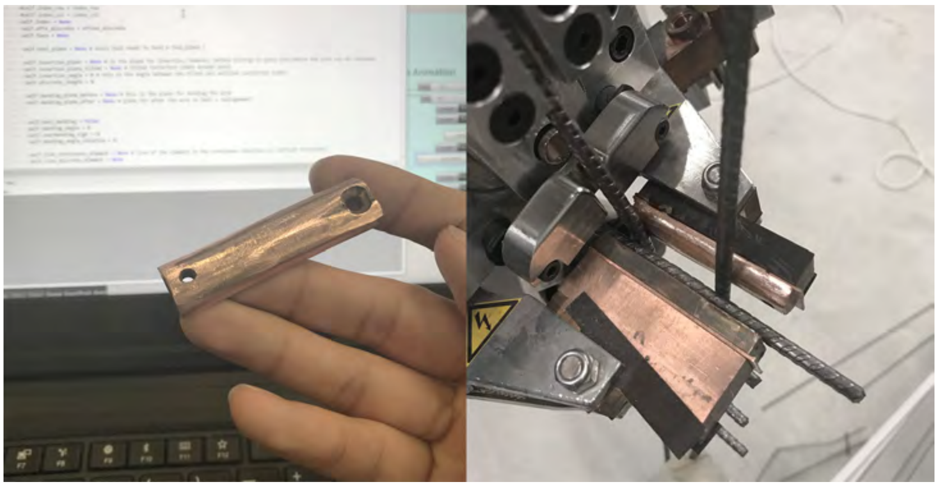

Figure 4.10: **The modified bronze metal part (left) and a work-in-progress clamp with one side metal part replaced (right).**

two neighbouring *continuous bars*. To achieve maximum geometric capacity, dynamic spacing is aimed. Fabrication experiments are thus needed to first obtain the data related to this fabrication action, including

1) the maximum or minimum distance between the two end points of the *straight bars* (constrained by the length of the clamp), and
2) the minimum spacing distance between two *straight bars* along a *continuous bar*.

Figure 4.11-right shows some of the fabrication test results, and the data obtained are given in Section 4.3.4.

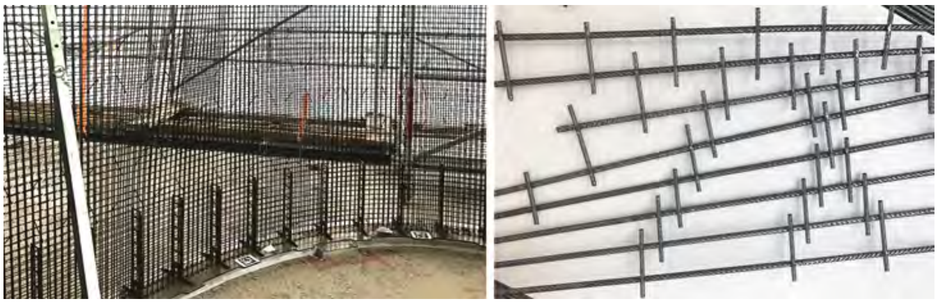

Figure 4.11: **Insertion Experiments.** Left: the regular-spaced *straight bars* in the *Mesh Mould* project; Right: the fabricated experiments to obtain data for the computational system.

Prototype To test the completeness and capacity of the upgraded hardware, a prototype of doubly curved surface was fabricated with the upgraded tool (Figure 4.12). This prototype incorporates all the hardware modifications made above and highlights the completeness of the hardware development stage[4].

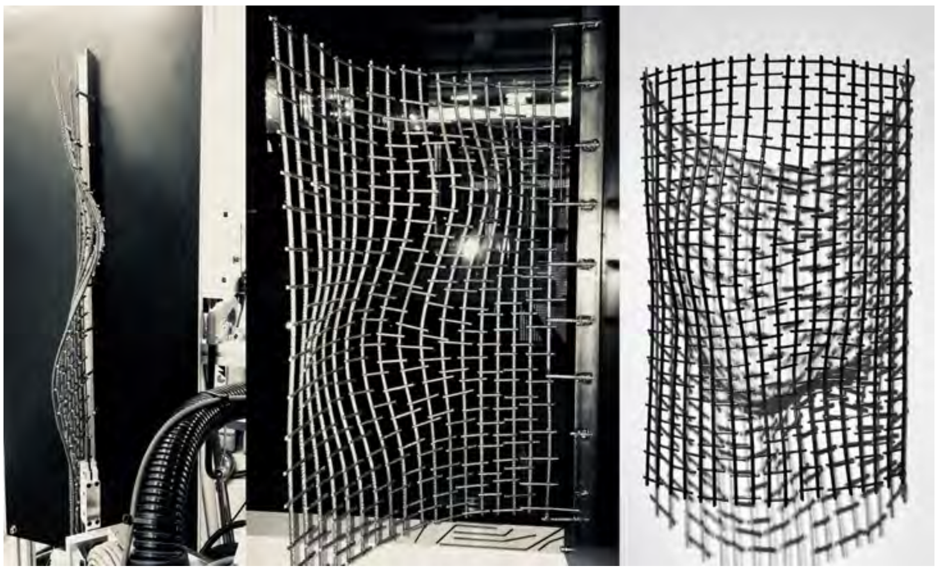

FIGURE 4.12: **The fabricated prototype for testing the upgraded hardware.**

4.3.3 Updated Fabrication Operations

The upgraded hardware now allows for several additional actions and expands the range of some other original actions. For ease of understanding, all the actions are grouped into two atomic operation sets, which are repeatedly executed until a frame unit is built (Figure 4.13):

1. Utilizing the 6 DOF of the robot arm, the head is positioned and oriented along the leftmost or rightmost *continuous bar*, and the structure held fixed with the clamp unit while a discrete bend is introduced with the bending unit.

[4]This piece of artefact has also been exhibited in Museum für Gestaltung, Zürich, as a part of the exhibition *DESIGNLABOR: MATERIAL + TECHNIK*.

2. The arm is then positioned and oriented to connect the newly added bend with a location previously introduced on the neighboring *continuous bar*. To this end, if in position, the insertion unit is activated, and introduces a shorter bar segment that is then welded in place, and cut to length.

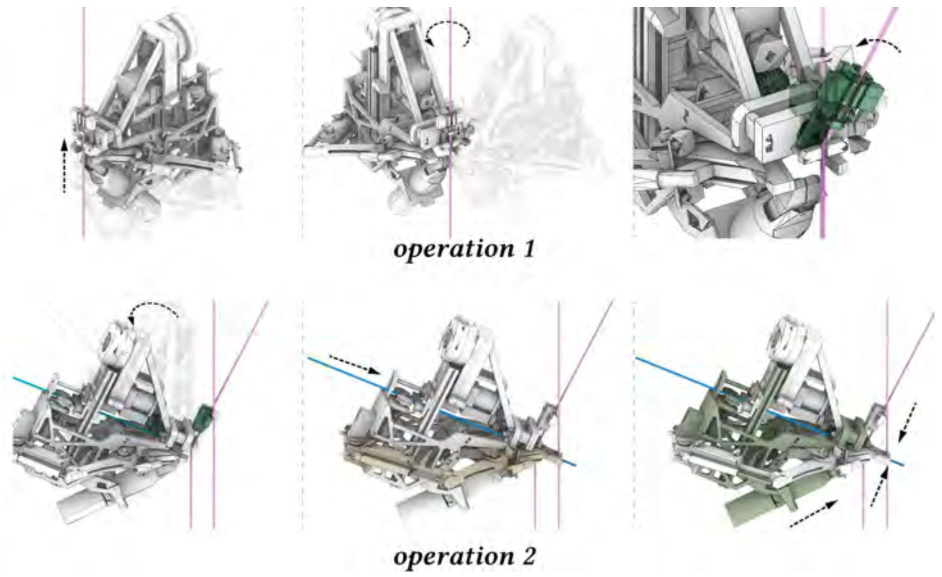

FIGURE 4.13: **Two fabrication operation sets.** Mechanical clamping, bending, bar insertion, welding, and cutting units are used to implement operations to (1) bend longer *continuous bars* and (2) insert, weld, and cut shorter *straight bars*.

More specifically, the fabrication starts with manual pre-bending of the first *continuous bar*, and attaching it to a temporary fixture. Subsequent *continuous bars* are then successively added to the existing structure. Each *continuous bar* is attached to the previous bar through discrete bars, starting with the bottommost one of a given layer. For each repeatable fabrication unit, the robotic tool head will: 1) automatically move to the insertion point, 2) spatially align the tool head to the bending location, 3) bend the *continuous bar*, 4) align the tool for the insertion sequence, 5) insert a bar orthogonally, 6) clamp the inserted bar to the two *continuous bars*, and cut and double cross-welded to create a robust connection, and 7) moving to the next insertion point as the starting step of the next bend.

There are some additional considerations regarding how these two atomic operations can be combined. Since a discrete bending in *operation 1* can be skipped, a shorter bar segment in the middle of a straight section of a *continuous bar* can be introduced (Figure 4.14, case 1). Similarly, a shorter bar segment can be ignored during *operation 2*, allowing frame structures with discrete bends that are not connected to neighboring *continuous bars* or only connected on one side to be fabricated (case 2). Moreover, *continuous bars* can end, or can be introduced during manufacturing (case 3).

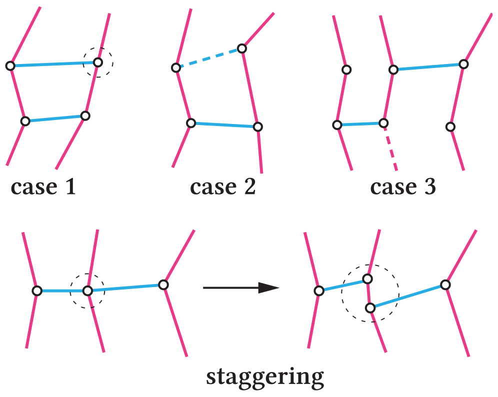

FIGURE 4.14: **Three allowed special fabrication cases and how staggering works at nodes where two short segments meet.**

There is one exception: shorter segments can only be welded to *continuous bars* but not to one another, and not at the same location. Hence, the short segments that exist at such nodes are separated with a small offset in between (staggering). Summarily, while the robotic fabrication process is specific, it enables the fabrication of frame structures of almost arbitrary network *topology*.

However, the mechanical units, as well as the dimensions and a finite number of DOFs of head and arm, constrain the feasibility of the *shapes* of the frame structures. For instance, bar segments have to fulfil minimal length constraints, and angle constraints limit the range of allowable extreme curvature.

4.3.4 Technical Specifications

While the procedure of the fabrication has been described in Section 4.3.3, technical specifications are provided here for a more in-depth description.

Robotic Setup The fabrication setup consists of a 6-axis *ABB IRB 4600* robot arm with a $40\,\mathrm{kg}$ payload, a reach of $2.55\,\mathrm{m}$, and a position repeatability of less than $0.01\,\mathrm{mm}$ (Figure 4.5). The arm is mounted on an *ABB IRBT 2005* linear track and equipped with a process-specific, custom-built end effector (Kumar et al. 2017) that is able to

- plastically cold-form continuous rebar with a diameter of up to $6\,\mathrm{mm}$ (Grade B500A or B500B) through a bending mechanism,
- pneumatically cut continuous rebar into straight segments of appropriate length,
- hydraulically clamp a pair of welding electrodes that are actuated by a four-bar linkage, onto pairs of bars, reaching a maximum of $6\,\mathrm{kN}$ pressure, and
- cross-weld shorter bar segments to *continuous bars* with an industrial-grade resistance welding setup with a peak output current of $21\,\mathrm{kA}$.

Constraint Bounds From the experiments conducted in Section 4.3, bounds for our fabrication constraints were determined: During insertion of shorter bar segments, the tool head can tilt backward by 8° ($\alpha_{\text{back}} = 98^\circ$), and forward by 15° ($\alpha_{\text{forw}} = 105^\circ$). The minimum and maximum lengths of the bar segments are $l^s_{\text{min}} = 28\,\mathrm{mm}$ and $l^s_{\text{max}} = 105\,\mathrm{mm}$, respectively, while the minimum distance between nodes (the minimum length of each segment on the *continuous bar*) is $l^l_{\text{min}} = 15\,\mathrm{mm}$. The bending tool can bend a bar by up to $\alpha_{\text{bend}} = 60^\circ$ in either direction, taking the spring-back angle of 12° into account. The bending plane constraint is $\epsilon_{\text{plane}} = \cos(20^\circ)$.

Steel Bar For fabrication, the rebar used in this project follows the European standard BS EN 10080: B500A rebar with a diameter of $4.5\,\mathrm{mm}$ and a material strength of $\sigma_c = 500\,\mathrm{MPa}$.

The Young's modulus of the material is $E = 210\,\mathrm{GPa}$, and its shear modulus is $G = 150\,\mathrm{GPa}$. For the *continuous bars* in the *Letter Wall Elements* in Section 4.6, the $6\,\mathrm{mm}$ B500B rebar with the same material properties is used for structural reasons, because of the scale of the artefacts.

4.4 Mesh Decomposition and Frame Initialization

For the design of a globally coherent, structurally-sound, and visually-pleasing monolithic frame structure, a good initialization is the key. The project in this chapter adopts a recent stripe pattern approach (Knöppel et al. 2015), feeding it with the field information derived from an analysis of stresses. Additionally, the method is also used as the decomposition guide for targets that are in reality beyond the robot reach. Overviews of the decomposition and initialization pipeline are illustrated in Figure 4.15 and Figure 4.16 respectively.

To allow the fabrication of models beyond scale of the robot reach, or with local curvature too high, a decomposition strategy is developed to cut the input mesh into smaller units. Theoretically, the user can feed in the smaller unit pre-decomposed from any CAD software and only ask the system to initialize each of the units with a regular frame. However, a frame structure consisting of units with randomly-aligned elements would be 1) difficult to assemble, 2) visually-displeasing, and 3) cut into units across structurally-relevant stress lines.

Thus, in order to decompose the input models in a structurally-informed manner, the frames are related to the stripe pattern discussed in Knöppel et al. (2015). By using the principal stresses to guide the stripe alignment, and their magnitude to guide the stripe density, the system generates globally continuous patterns with branching points to maintain regular spacing.

Moreover, the system bases the pattern generation on a stress field to generate stripes that align with global stress lines (Pellis and Pottmann 2018). Aligning *continuous bars* with stress lines and assisting the user with the informed decomposition along these lines, the structure's load-bearing capacity can be increased significantly because stress-aligned bars absorb external loads with axial stresses instead of transverse stresses and moments.

4.4.1 Stripe Pattern and Mesh Decomposition

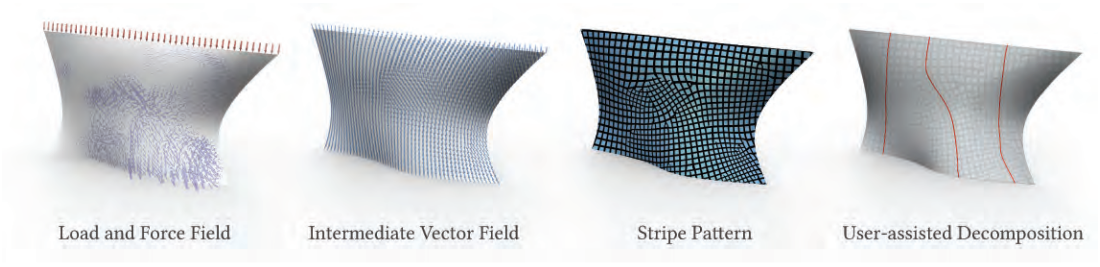

FIGURE 4.15: **Mesh Decomposition.** To obtain an initial stress field, the FEM simulation is run on the input surface with a standard shell model (Load and Force Field). The field is then smoothed and weighted as an intermediated field for the next step (Intermediate Field). To globally align frame structures, a stress-aligned stripe pattern is generated where stripes align with a principal stress field (Stripe Pattern). The user then is allowed to select decomposition locations, using the stripe as a guidance (User-assisted Decomposition).

Principal Stress Aligned Stripe Pattern For the frame structures discussed here, the *continuous bars* form most of the load-bearing capacity of the structure, while the discrete bars only partially help to absorb and transfer loads. To create a decomposition and subsequent initialization for a structurally-sound frame, the designing stage therefore aims to include this knowledge of stress lines and the orientation of the *continuous bars* into the whole pipeline.

The initial surface structure is imported as the *input model* and treated as a standard shell (Preisinger and Heimrath 2014) for structure analysis. For realistic reasons, only small deflection is allowed and linear elastic material is thus applied globally. The *input model* is then loaded with a user-specified load and analyzed (Figure 4.15 Load and Force Field). For every point on the parameterized surface, the Cauchy stress and the principal stress can be computed, as well as its direction and magnitude in the tangent plane. Since small stresses tend to be noisy for non-trivial and tessellated input, and the alignment of bars with principal stresses is most important in high-stress regions, the original principal stress field is smoothed and weighted. To this end, the *Globally Optimal Direction Fields* approach by Knöppel et al. (2013) is employed with the field of vectors whose magnitude is set to the squared principal stress, pointing in the direction of the principal stress.

To generate an aligned stripe pattern, the resulting smoothed vector field (Figure 4.15, Intermediate Vector Field) is then provided as the input to the pattern generator proposed in Knöppel et al. (2015). Besides a vector field, the generator takes a scalar field that controls the desired density of stripes as an additional input. For regions with zero principal stress, this second parameter is set to the lowest bar density supported by our fabrication process. Similarly, the maximum supported bar density is assigned to the parameter at where the principal stresses peak. Linear interpolation of the density value is conducted for the stresses between the two extremes.

The stripe pattern generator is then run twice, to align longer *continuous bars* and shorter *straight bars* with the principal stresses and the principal stresses in orthogonal directions, specifically. The vectors in orthogonal are obtained by rotating the smoothed vector field by 90 degrees. The result is a pair of parameterizations that can be plugged into a symmetric periodic function like the cosine to visualize the stripes (Figure 4.15, Stress-Aligned Stripe Pattern).

Structurally-Informed Decomposition To assist the user with the decomposition, suitable locations are chosen by the user himself on the input model with visualized stripe pattern. The system will trace out a cut by following the isolines in either direction until the cut reaches the boundaries or an already introduced cut (see red lines in Figure 4.15, User-assisted Decomposition). Interactively adding and removing cuts, the user can quickly converge to a structurally-informed decomposition.

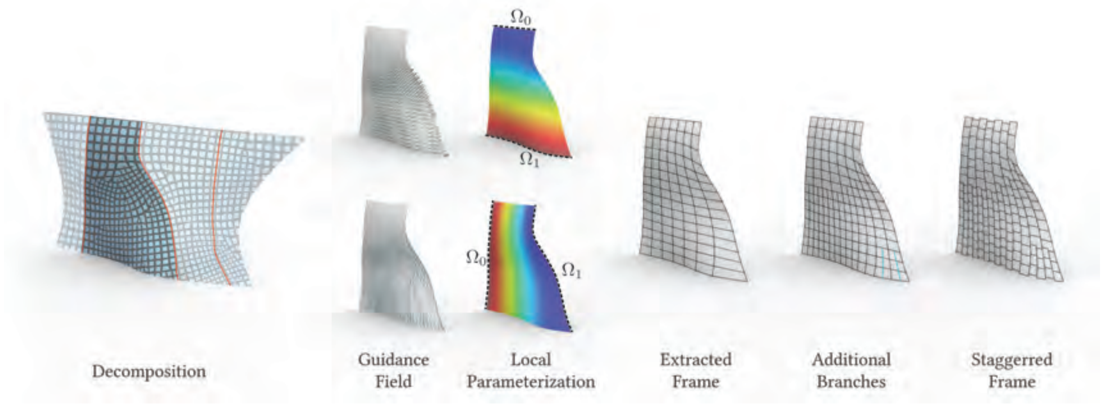

FIGURE 4.16: **Frame Initialization.** Assisting the user with the decomposition of the model along stress lines (Decomposition, cuts in red), the pipeline initializes parameterizations (Local Parameterizations) to generate longer *continuous bars*, and shorter *straight bar* segments (Extracted Frame). Branches are added accordingly to maintain a general density (Additional branches). Shorter bars are then staggered for fabricability (Staggered Frame).

4.4.2 Local Frame Initialization

The decomposition only provides surface patches of valid size, and a frame structure is initialized from the patches after decomposition for every fabricable unit. To achieve the best possible result with the developed topology and shape optimization, the initial frame structure should be as close as possible to fulfilling our fabricability constraints. The stripe pattern helps with this task as stripes can be regularly spaced within the range of densities that the robotic construction system supports. However, a hard constraint when initializing frames is that longer *continuous bars* can neither cross nor overlap. The same situation holds for shorter *straight bar* segments.

To extract frames that meet these *desiderata*, a similar method to the routing approach proposed by Pereira, Rusinkiewicz, and Matusik (2014) is used. Instead of routing in 3D, the parameterization is solved so that it enables the extraction of *continuous bars* along stress lines from one boundary to its opposite, and shorter bars in orthogonal directions, respectively.

For this reason, the boundary of a unit patch is split into four segments, and two non-consecutive segments are assigned with the values 0 and 1. Then, a scalar field s defined at every point $\mathbf{x} \in \Omega$ on the patch surface is needed so that its isolines of constant value are in the interval $[0, 1]$ and align with the stripe pattern, and that the start and end points of these isolines are on the other two non-consecutive boundary segments (Figure 4.16 Local Parameterizations). Inspired by the energy formulation used for Poisson Image Editing (Pérez, Gangnet, and Blake 2003), the above scalar field whose gradient aligns with the guidance field $\mathbf{g}(x)$ is computed and set to a scaled version of the stripe pattern field, rotated by 90 degrees:

$$\min_{s(\mathbf{x})} \int_{\Omega} \|\nabla s(\mathbf{x}) - \mathbf{g}(\mathbf{x})\|^2 d\mathbf{x} \quad \text{s.t.} \quad s(\mathbf{x}) = \begin{cases} 0 \;\text{ for }\; \mathbf{x} \in \partial\Omega_0 \\ 1 \;\text{ for }\; \mathbf{x} \in \partial\Omega_1. \end{cases}$$

The unique minimizer of this energy is found by solving the corresponding Poisson problem $\Delta s(\mathbf{x}) = \nabla \cdot \mathbf{g}(\mathbf{x})$ for the scalar field. This process is conducted twice for the two corresponding directions by alternating the boundary segments and rotating the guidance field by another 90 degrees.

The initialization of a frame structure starts by first tracing *continuous bars* that are sufficiently far apart in the first parameterization and adding additional *continuous bars* that end and start at branching points if neighboring bars are not sufficiently dense. Then the second parameterization is used to trace along orthogonal directions (Figure 4.16 Extracted Frame). The *staggering* process of the shorter *straight bar* segments is conducted afterwards as weld connections between shorter and longer bars cannot be introduced at the same location (Figure 4.16 Staggered Frame).

While experiments with multiple load cases are not employed, a smooth maximum of the principal stress fields should be sufficient when initializing the structures.

4.5 Frame Optimization

After initialization, the input model is decomposed into units consisting of a dense set of bars, “zippering” well at the unit-unit interfacing boundaries. The latter is achieved properly by aligning unit boundaries with global stress lines, and by using $0\,\mathrm{b}$ to 1 boundary conditions for the generation of the two local parameterizations.

However, while bars are regularly spaced according to densities supported by the robotic process, there are various other fabrication constraints that limit the shapes of a fabricable unit. For instance, the maximum bending angle, and potential collisions between the tool and the fabricated object, limit the magnitude of the local curvature that the system can achieve. Moreover, bars can likely be removed if the stresses in other bars do not exceed their limits under the designed loads.

Thus, an optimization process is required to overcome these issues. This project parameterizes the locations of weld connections and bends, and assigns continuous inclusion variables to the shorter *straight bar* segments that range between the maximum cross-sectional area (keep the bar) and zero (remove the bar). Co-optimizing the strength-to-weight ratio together with a target matching objective under strict fabricability constraint, the optimization generates construction-ready unit frames that can be welded together to form a sound monolithic structure. Furthermore, with a regularizer favoring regularly-spaced bars, and an objective enabling the embedding of artistic targets, the system provides the user with mechanisms to control artistic aspects of the optimization outcome.

4.5.1 Representation and Analysis

As the topology of the resulting structures is a graph $\mathcal{G} = \mathcal{E}, \mathcal{V}$ that consists of a network of *straight bar* segments $\mathcal{E}$, and deformations are expected to remain within the small strain, elastic domain, standard frame modeling lends itself for analysis ((Logan 2011)). For frames, in contrast to the more widely used truss modeling, both forces and moments are transferred at connections $\mathcal{V}$.

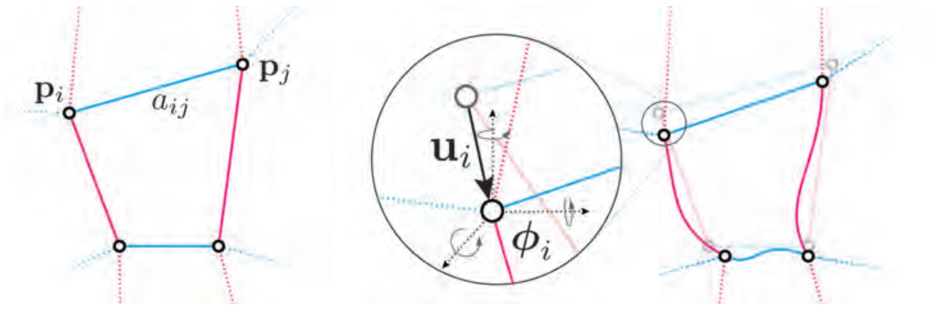

FIGURE 4.17: **Representation and Analysis.** Parameterized rest configuration (left), and its deformed configuration (right).

As illustrated in Figure 4.17, nodes $\mathbf{p}_i$ is represented as 3D points that displace to locations $\mathbf{p}_i + \mathbf{u}_i$ when loads or moments are applied to the structure. To quantify local orientation changes at node i, rotations around the axes in the global coordinates are quantified with a set of three angles $\phi_i \in \mathbb{R}^3$. When defining inclusion variables, A differentiation between shorter straight segments $\mathcal{E}_s$ and segments $\mathcal{E}_l$ that belong to longer *continuous bars* exists. As longer *continuous bars* need to always exist and cannot be removed, area parameterization only happens to the shorter segments with scalar variables a_{ij}.

When optimizing the strength-to-weight ratio of a frame while keeping stresses within limits, it is important to be able to quantify how changes to shape and topology affect the compliance of the overall structure, and the stresses in individual bars. To this end, we minimize the total potential energy

$$f_{\text{sim}}(\mathbf{u}, \phi) = \sum_{(i,j)\in\mathcal{E}} E_{ij}^{\text{int}}(a_{ij}, \mathbf{p}_i, \mathbf{p}_j, \mathbf{u}_i, \mathbf{u}_j, \phi_i, \phi_j) - \sum_{i\in\mathcal{V}} E_i^{\text{ext}}(\mathbf{u}_i, \phi_i)$$

to first-order optimality. The internal energy E_{ij}^{int} stored in every edge (i, j) depends on the cross-sectional area of the bar, and the positions, displacements and orientations of adjacent nodes, and can be represented as

$$E_{ij}^{\text{int}} = \frac{1}{2}\left[(\mathbf{u}_i - \mathbf{u}_j)^T \mathbf{A}_{ij}(\mathbf{u}_i - \mathbf{u}_j) + (\mathbf{u}_i - \mathbf{u}_j)^T \mathbf{B}_{ij}(\boldsymbol{\phi}_i + \boldsymbol{\phi}_j)\right.$$
$$\left. + (\boldsymbol{\phi}_i - \boldsymbol{\phi}_j)\mathbf{C}_{ij}(\boldsymbol{\phi}_i - \boldsymbol{\phi}_j) + \boldsymbol{\phi}_i \mathbf{D}_{ij}\boldsymbol{\phi}_j\right]$$

where the elemental stiffness matrices (Logan 2011)

$$\mathbf{A}_{ij} = \mathbf{R}_{ij}^T \begin{bmatrix} \frac{Ea_{ij}}{l_{ij}} & 0 & 0 \\ 0 & \frac{3Ea_{ij}^2}{\pi l_{ij}^3} & 0 \\ 0 & 0 & \frac{3Ea_{ij}^2}{\pi l_{ij}^3} \end{bmatrix} \mathbf{R}_{ij}, \quad \mathbf{B}_{ij} = 2\mathbf{R}_{ij}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{3Ea_{ij}^2}{2\pi l_{ij}^2} \\ 0 & -\frac{3Ea_{ij}^2}{2\pi l_{ij}^2} & 0 \end{bmatrix} \mathbf{R}_{ij},$$
$$\mathbf{C}_{ij} = \mathbf{R}_{ij}^T \begin{bmatrix} \frac{Ga_{ij}^2}{2\pi l_{ij}} & 0 & 0 \\ 0 & \frac{Ea_{ij}^2}{\pi l_{ij}} & 0 \\ 0 & 0 & \frac{Ea_{ij}^2}{\pi l_{ij}} \end{bmatrix} \mathbf{R}_{ij}, \quad \mathbf{D}_{ij} = \mathbf{R}_{ij}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{3Ea_{ij}^2}{\pi L_{ij}} & 0 \\ 0 & 0 & \frac{3Ea_{ij}^2}{\pi l_{ij}} \end{bmatrix} \mathbf{R}_{ij}$$

depend on the constant rotation matrix $\mathbf{R}_{ij}$ that rotates the vector $\mathbf{e}_{ij} = \mathbf{p}_i - \mathbf{p}_j$ to the global x-axis $[0, 0, 1]^T$, the rest length $l_{ij} = \|\mathbf{e}_{ij}\|$ of the bar, its cross sectional area a_{ij}, and the Young's and shear moduli E and G, respectively.

The external energy at node i sums up the work $E_i^{\text{ext}} = \mathbf{f}_i^T \mathbf{u}_i + \mathbf{m}_i^T \boldsymbol{\phi}_i$ that external forces $\mathbf{f}_i$ and moments $\mathbf{m}_i$ do on the structure. Since the analysis objective is quadratic in the deformation variables for a linearly elastic material, the optimal displacements $\mathbf{u}$ and orientations $\boldsymbol{\phi}$ can be computed by solving a linear system of equations (Logan 2011).

Accordingly, the next goal is to minimize a design objective over a constraint manifold spanned by an equilibrium constraint, lower and upper bounds on cross-sectional areas, fabricability constraints, and bounds on allowable stresses

$$\min_{\mathbf{a},\mathbf{p}} \quad f_{\text{design}}(\mathbf{a}, \mathbf{p}, \mathbf{u}, \boldsymbol{\phi}) \quad \text{s.t.} \quad \begin{aligned} &\nabla_{(\mathbf{u},\boldsymbol{\phi})} f_{\text{sim}}(\mathbf{u}, \boldsymbol{\phi}) = \mathbf{0}, \\ &0 \leq a_{ij} \leq a_{\max}, \quad (i,j) \in \mathcal{E}_s, \\ &\text{(fabrication. constraints, Section 4.5.3)}, \\ &\text{(stress bounds, Section 4.5.4)}. \end{aligned}$$

Holding external loads and moments fixed, the structure's response changes if we alter the design parameters. Hence, the displacements and local orientations *implicitly* depend on the cross-sectional areas $\mathbf{a}$ and nodal displacements $\mathbf{p}$. This dependency will be further discussed towards the end of this section.

Though this minimization can be solved on the monolithic structure for smaller input, the number of variables and constraints make numerical optimizations prohibitively expensive for larger targets. To make optimizations scalable, the project employs a "divide-and-conquer" approach: 1) analyze the monolithic structure under user-specified loads and moments, 2) extract the forces and moments along the unit patch boundaries that preserve the equilibrium state at the individual unit-level, and 3) solve instances of the above design optimization on the unit level by setting the local "external" forces and moments to the extracted set.

Though for the demonstrations shown in this chapter, the magnitude of the load cases is relatively moderate and the above strategy is not necessarily needed, the strategy offers an opportunity to alternate between the global analysis and local design optimizations and to solve the latter in parallel. These global-local strategies, while heuristic in nature, have proven effective in various applications in CG field (Bouaziz et al. 2014; Sorkine and Alexa 2007). By introducing per-load-case displacements and orientations, the developed design optimization could further be used to determine optimal frame structures under a multitude of load cases.

Before introducing fabrication constraints and stress bounds, a set of objectives will be introduced which enables the generation of structurally-sound and aesthetically-pleasing frames at the furniture and architectural scale.

4.5.2 Design Objectives

Design objectives are formulated to balance structural, target matching, and artistic targets.

Strength-to-Weight Ratio To remove bars that are not needed, an objective for topology optimization is formulated to minimize a weighted sum of the overall volume and compliance

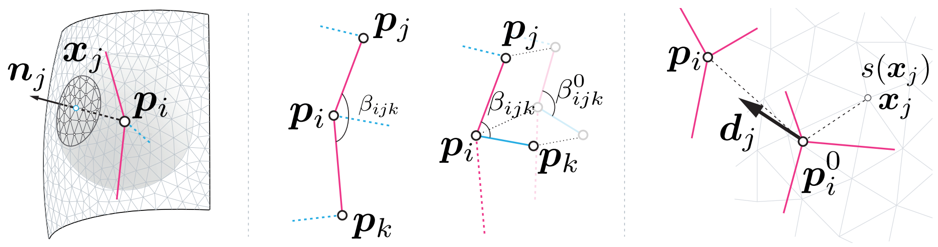

FIGURE 4.18: **Design Objectives.** The target matching objective (left) keeps frames close to the user-specified input, while the regularity objective penalizes differences between the initial and the current rest configuration (middle, left), and favors smoothness for segments belonging to *continuous bars* (middle, right). To embed artistic targets, the connector points are moved towards attraction directions that are computed from a user-specified scalar field (right).

of the structure

$$f_{\text{s-to-w}} = \frac{1}{V_0} \sum_{(i,j)\in\mathcal{E}} V_{ij} + \alpha \frac{1}{C_0} \sum_{i\in\mathcal{V}} E_i^{\text{ext}}(\mathbf{u}_i, \boldsymbol{\phi}_i). \tag{4.1}$$

The volume V_{ij} of segment (i,j) is either set to a_{ij} or a_{max} times its length $\|\mathbf{p}_i - \mathbf{p}_j\|$, depending on whether the cross-sectional area of the corresponding bar is parameterized or not. For normalization purpose, the two terms are divided by the volume V_0 and compliance C_0 of the initial structure, respectively. Their relative importance is controlled by the factor α. To favor sparse solutions, the approximate L0-regularizer (Skouras et al. 2013)

$$\frac{1}{|\mathcal{E}_s|} \sum_{(i,j)\in\mathcal{E}_s} \left(a_{ij}^2\right)^{\frac{1}{\gamma}} \tag{4.2}$$

is added to the objective. The parameter $\gamma > 2$ to 3 is set for all the demonstrations, and the term is normalized by dividing it by the number of edges with parameterized cross-sectional areas.

Target Matching Circumstance may exist that when the initial frame structure approximates the target surface well, the optimization may move nodes away to fulfill fabrication, or other objectives or constraints. Therefore, to keep frames close to the target, the signed distance of connections is penalized to an implicit representation of the input mesh (Öztireli, Guennebaud, and Gross 2009). For this reason, the signed difference is minimized between nodes i and neighboring surface points $\mathbf{x}_j$ with normals $\mathbf{n}_j$ (compare with Figure 4.18 left)

$$f_{\text{target}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \frac{1}{2} \left(\frac{\sum_j \mathbf{n}_j^T (\mathbf{p}_i - \mathbf{x}_j) w_{ij}(\mathbf{p}_i)}{\sum_j w_{ij}(\mathbf{p}_i)} \right)^2 , \tag{4.3}$$

and across sharp corners or edges (Öztireli, Guennebaud, and Gross 2009). The latter enables the preservation of sharp features, and the weights w_{ij} decay with the distance between pairs i and j.

Regularity The penalization of irregularity is conducted by minimizing the differences between the initial and current rest length of segments, and the initial and current angle between adjacent pairs of edges from the sets $\mathcal{E}_s \times \mathcal{E}_l$ and $\mathcal{E}_l \times \mathcal{E}_l$ (see Figure 4.18 middle). After defining the oriented edge $\mathbf{e}_{ij} = \mathbf{p}_j - \mathbf{p}_i$, the differences in rest lengths with per-edge terms $\frac{1}{2}(||\mathbf{e}_{ij}|| - ||\mathbf{e}_{ij}^0||)^2$ are penalized. Similarly, the differences in rest angles $\beta_{ijk} = \angle(\mathbf{e}_{ij}, \mathbf{e}_{ik})$ between pairs of edges are penalized by minimizing terms of the form $\sin^2(\beta_{ijk} - \beta_{ijk}^0)$. Note that the use of the squared sine enjoys the advantage to express the term with dot- and cross-products between the two oriented edges.

While sine-penalty is used for both types of adjacent edges, another term is needed to penalize deviations from straightness, and to discourage proximity to bending angle limits for pairs of segments belonging to a *continuous bar*. This choice tends to help with circumnavigating local minima, and positively affects fabrication time also. For this reason, the term $(1 + \cos(\beta_{ijk}))^\tau$ to measure the deviation of the angle $\beta_{ijk} = \angle(\mathbf{e}_{ij}, \mathbf{e}_{ik})$ from 180 degrees is chosen, and the penalty parameter τ is set to 5 for all our demonstrations.

Summing up the individual penalty terms, normalizing the three penalty types by dividing by the number of edges, or adjacent edge pairs in the sets $\mathcal{E}_s \times \mathcal{E}_l$ and $\mathcal{E}_l \times \mathcal{E}_l$, and weighing them, the optimization obtains the regularity objective f_{reg}.

Artistic Targets Embedding The artistic targets embedding requires an input from the user, which is defined in a scalar fields $s(\mathbf{x})$ on target surfaces.The surface points $\mathbf{x}_j$ will attract connectors if they have positive values $s(\mathbf{x}_j) > 0$, or repel them if they have negative values $s(\mathbf{x}_j) < 0$. To define the direction a connector i should move toward, the scalar field for surface points in a neighborhood of the initial position $\mathbf{p}_i^0$ (Figure 4.18 right) is evaluated by computing an average attraction direction

$$\mathbf{d}_i = \sum_j s(\mathbf{x}_j) w(\|\mathbf{p}_i^0 - \mathbf{x}_j\|) \frac{\mathbf{p}_i^0 - \mathbf{x}_j}{\|\mathbf{p}_i^0 - \mathbf{x}_j\|}, \tag{4.4}$$

where the weight of the attraction and repulsion forces for points is based on the relative distance $\mathbf{p}_i^0$, and decays with w as the distance increases. Then the measurement of whether connectors move toward these directions is conducted by minimizing

$$f_{\text{art}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{d}_i^T \left(\mathbf{p}_i - \mathbf{p}_i^0\right). \tag{4.5}$$

While the attraction directions are kept constant, they could be updated by exchanging the initial connector position in $\mathbf{d}_i$ with its current position. This is in particular important if the frame structures are expected to move significantly. For this implementation, the decay function $w(\|\mathbf{p}_i^0 - \mathbf{x}_j\|) = \frac{1}{1+\|\mathbf{p}_i^0 - \mathbf{x}_j\|}$ is chosen. While this is a good choice if the attraction directions remain constant, a function with a finite neighborhood is desirable if the attraction direction is dependent on the current positions $\mathbf{p}_i$.

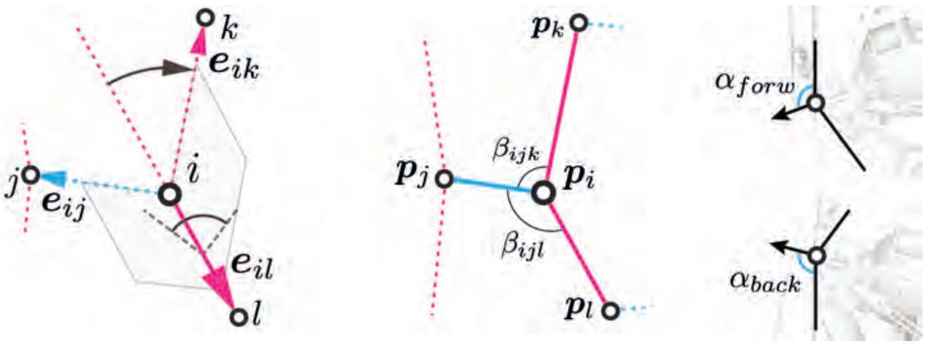

FIGURE 4.19: **Fabrication Constraints.** To avoid head-frame collisions during robotic assembly, we limit the bending angle and bending plane for segments that belong to longer *continuous bars* (left), and the insertion angle for shorter bars that connect two neighboring *continuous bars* (right).

4.5.3 Fabrication Constraints

While the design objectives trade off functional and aesthetic targets, there are several hard limits posed by the robotic fabrication process, namely: 1) the angle and construction plane for a discrete bending operation, 2) the length of *continuous or straight bar* segments, 3) the angle for an insertion operation of a shorter segment, which is constrained by the robot-artefact collisions, and 4) the limited degrees of freedom of the head *w. r. t.* the *continuous bar* it is moving along.

Bending Angle and Plane The bending tool has a fixed limit of the bending angle, *i. e.* the angle between adjacent bar segments from the set $\mathcal{E}_l \times \mathcal{E}_l$, which is constrained to be smaller than or equal to the maximum bending angle α_{bend} with constraints of the form (Figure 4.19 left)

$$\angle(-\mathbf{e}_{il}, \mathbf{e}_{ik}) \leq \alpha_{\text{bend}}. \tag{4.6}$$

Besides, the bending process applied to the *continuous bars* conducted by the bending tool is constrained in a plane perpendicular to the direction which the insertion tool is extruding shorter *straight bar* segments toward. To avoid robot-artefact collisions when introducing a bend at location i, the preference is given to the circumstance where the normal of the bending plane lies in, or close to the local tangent plane at i.

The tangent plane at i is approximated by the two vectors: the directional vector of the already fabricated segment $\mathbf{e}_{il}$, and that of the yet-to-be-inserted, shorter segment $\mathbf{e}_{ij}$. Then, the constraint will prefer the dot product between the normal of the bending plane, $\mathbf{e}_{il} \times \mathbf{e}_{ik}$, and the normal of the tangent plane, $\mathbf{e}_{il} \times \mathbf{e}_{ik}$, to not deviate too far from orthogonality

$$(\mathbf{e}_{ij} \times \mathbf{e}_{il}) \cdot (\mathbf{e}_{ij} \times \mathbf{e}_{ik}) \leq \varepsilon_{\text{plane}}. \tag{4.7}$$

Bar Length The discrete bars that are inserted between *continuous bars* are welded using an electronic welding clamp whose size limits the length that these bars can have. Additionally, the tool head has to move along the *continuous bar* to bend and insert sequentially. These processes pose a requirement that the shorter discrete bars should be short enough so that the clamp can fix its both ends simultaneously for a successful weld, and long enough so that the space between the two end points, *i. e.* distance between two neighbouring *continuous bars*, will not block the movement of the robot. Together, these constraints introduce a lower and upper bound, l^s_{min} and l^s_{max}, on the length of shorter, *straight bar* segments:

$$l^s_{\text{min}} \leq \|\mathbf{e}_{ij}\| \leq l^s_{\text{max}}, \quad (i,j) \in \mathcal{E}_s. \tag{4.8}$$

Similarly, the size and functionality of the tool head impose a minimal distance constraint between consecutive bends along *continuous bars*

$$l^l_{\text{min}} \leq \|\mathbf{e}_{ij}\|, \quad (i,j) \in \mathcal{E}_l \tag{4.9}$$

where l^l_{min} is the lower length bound.

Insertion Angle To insert a shorter bar segment, the tool head has to tilt forward or backward [5] (Figure 4.19 right), and the amount the head can tilt before colliding with the structure is limiting the two angles between the three oriented bars connecting i to j, k, and l

$$\angle(\mathbf{e}_{ij}, \mathbf{e}_{il}) \leq \alpha_{\text{back}} \quad \text{and} \quad \angle(\mathbf{e}_{ij}, \mathbf{e}_{ik}) \leq \alpha_{\text{forw}}. \tag{4.10}$$

4.5.4 Stress Bounds

Apart from ensuring fabricability, the structural instabilities must be avoided to guarantee the validity of a physical artefact. For this reason, the axial stress

$$\sigma_{ij} = E\epsilon_{ij} = E\frac{\|(\mathbf{p}_j + \mathbf{u}_j) - (\mathbf{p}_i + \mathbf{u}_i)\| - \|\mathbf{e}_{ij}\|}{\|\mathbf{e}_{ij}\|}. \tag{4.11}$$

in segments (i,j) is constrained to be within the yielding and buckling stress limit

$$|\sigma_{ij}| < \sigma_{\text{yield}}, \quad \text{and} \quad \sigma_{ij} > -\frac{1}{a_{\text{max}}}\frac{\pi^2 EI}{(K\|\mathbf{e}_{ij}\|)^2}, \tag{4.12}$$

[5]this is one of the main improvements achieved from the hardware modification.

respectively (Logan 2011). The column effective length factor K in Euler's critical load criterion is set to 1.2, the Young's modulus E to the tabulated value for the steel in use, and the area moment of inertia to $I = \frac{1}{4\pi} a_{\max}^2$. Additionally, $a_{\max}$ instead of a_{ij} is deliberately chosen in the criterion. Albeit this choice overestimates the buckling resistance for intermediate states, it guarantees that the constraints will be fulfilled if non-zero cross-sectional areas are set to $a_{\max}$ after optimizations.

4.5.5 Implementation Detail

Using the KNITRO (Byrd, Nocedal, and Waltz 2006) solver, the optimization problem is solved with KNITRO's implementation of the interior point method with BFGS. All constraints are treated explicitly, with the exception of the equilibrium constraint, which is implicitly enforced by updating the deformation variables $\mathbf{u}$ and ϕ whenever the design variables are changed. This requires the determination of the derivatives of the deformed configuration *w. r. t.* design variables, which can be computed by applying the *implicit function theorem.*

The GUI is developed in the C++ programming language (C++) language, using the *Qt* library as the main graphic library. All the initialization and optimization components are integrated into the same interface, providing the user with an interactive all-in-one solution the whole pipeline. A screenshot of the interface is provided in Figure 4.20. A list of open-source libraries used for the implementation can be found in the AppendixB.3.

4.6 Demonstrations and Results

The computational technique developed above has been verified to optimize and fabricate a *stool* and *table* design (Figure 4.22, Figure 4.24), as well as two wall elements with engraved letters (Figure 4.26), as demonstrations of the successful use in furniture design and architectural ornamentation. In addition, several in-simulation examples are demonstrated in this section, as an evaluation under changes to load cases, target surfaces, and artistic

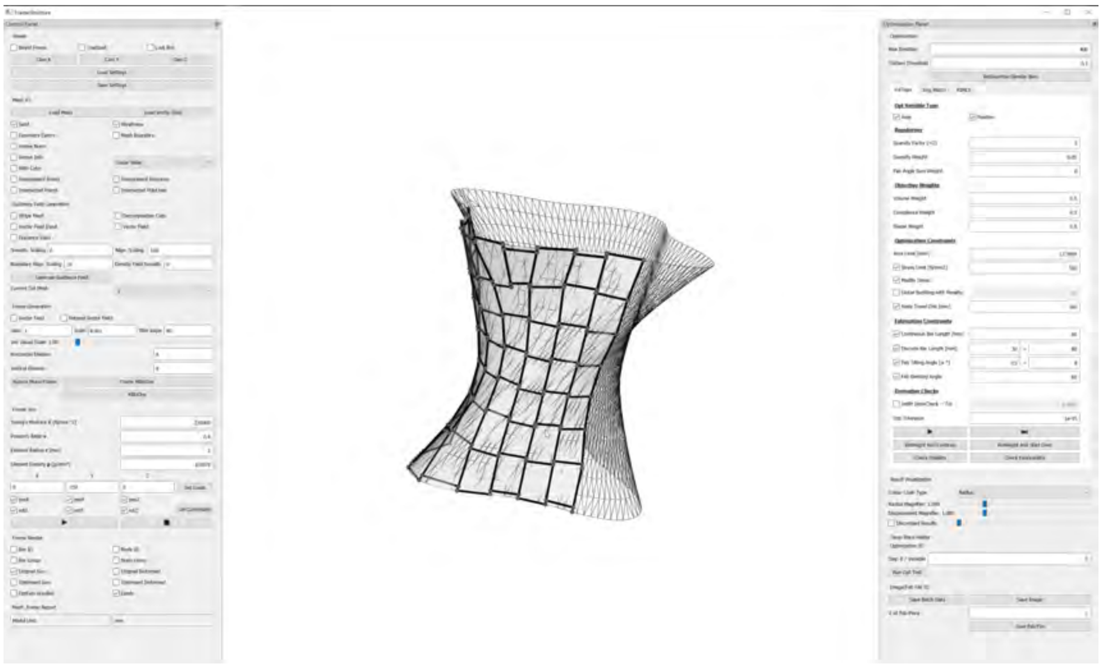

FIGURE 4.20: **The FrameForm GUI.**

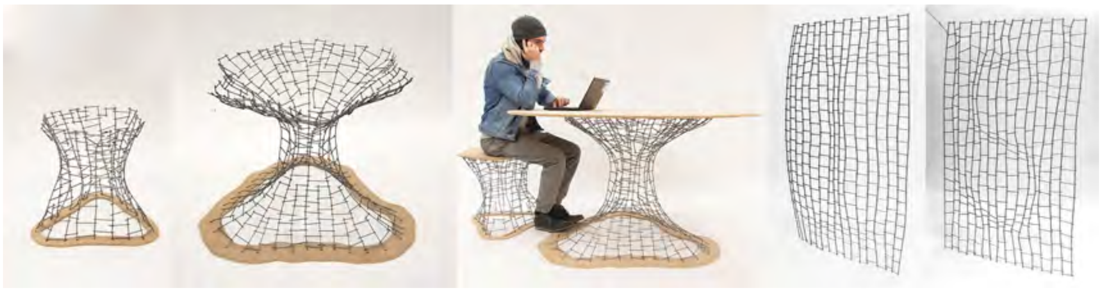

FIGURE 4.21: **A collection of demonstrations optimized and fabricated using the system developed for this project.**

targets, besides an analysis of our decomposition strategy and the direction-dependence of the optimized results. The optimization parameters are reported in Section 4.6.1.

4.6.1 Fabrication Demonstrations

Stool and Table Furniture, such as stools and tables, must withstand extensive live loads from a person sitting on them, or loads of objects placed onto them. For both models, the ground nodes are kept fixed, and a uniformly distributed load of 1500 N is applied at the top, pointing in the direction of gravity (Figure 4.22, Figure 4.24).

As both models are significantly narrower in the mid-region and widen towards the two ends and formed in a circular shape, direct fabrication is thus not possible due to the robot-artefact collision—the robot arm will collide with the fabricated piece before the model closes. A decomposition into at least three units for both models is necessary (Figure 4.23, Figure 4.25). To bridge the gap between the longer *continuous bars* in regions of varying curvature, the initialization inserts additional shorter *continuous bars* as the branches. For the table which owns a higher variation in curvature, up to three *branching* bars are added between the *continuous bars* that run along the full height of the model. During shape and topology optimizations of the individual units, shorter bar segments that are not necessary for structural integrity are removed.

After fabrication and assembly, wooden seats or top surfaces are added to make these furniture functional, as well as the bottom supporting plate (Figure 4.21). As reported in Section 4.6.1, the structural optimization improves the collapse load limits of the structures with minimum or unnoticeable modification to the surface.

Considering principal curvature, the *stool* and *table* examples consist of points of elliptical (both principal curvatures have the same sign), hyperbolic (opposite sign), parabolic (one set to zero), and planar type (both set to zero). In summary, if the input model is decomposed into units that consist of points at which the two principal curvatures are 1) bound from above

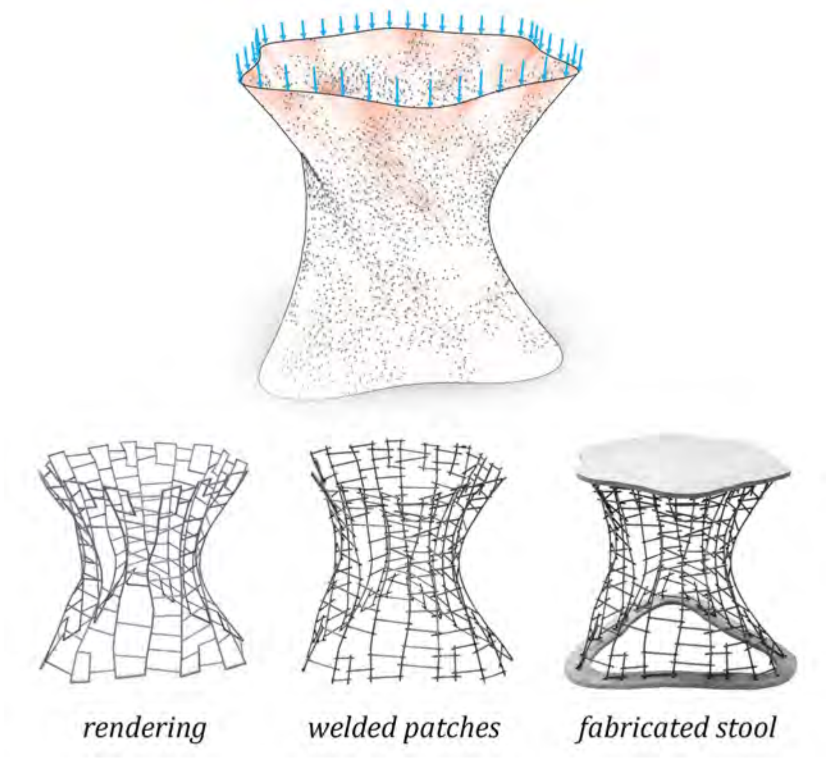

Figure 4.22: **Stool.** A top load of 1500 N is applied to the *stool* example, resulting in regions of high stress at the top of the model (top). The computational technique generates a structurally-sound frame model (bottom left) that is fabricable with the robotic fabrication approach (bottom middle, right).

and below with a constant of moderately high value, and 2) vary sufficiently slowly along the shortest path between any pair of points, the employed global-to-local strategy is expected to produce the desired fabricable output.

Letter Wall Elements Beyond designing fabricable and functional output, the system also supports the embedding of artistic input by applying a user-specified *attraction field* $s(\mathbf{x})$ (Section 4.5.2) to the structure. The optimization will then solve for a frame structure that balances structural properties and fabrication constraints, and complies to the artistic target at the same time.

This approach is demonstrated on curved wall elements with embedded letters (Figure 4.26).

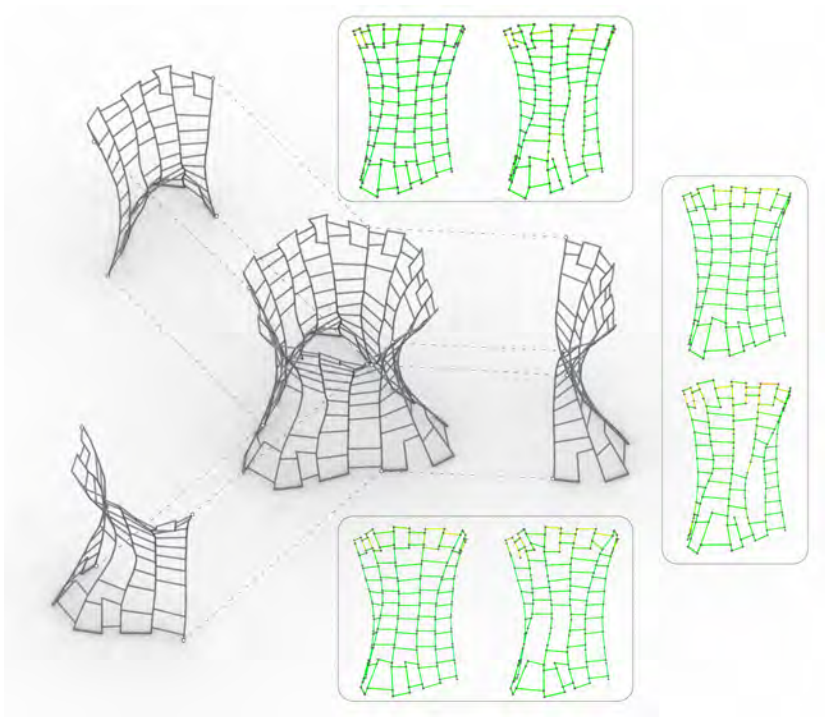

FIGURE 4.23: **Stress visualization of the *stool* example.** During the design process, the *stool* example is split into three units (center). For each unit, the stress distribution is visualized before (left) and after (right) optimization.

These 1.5 m tall wall elements are intended to withstand a moderate vertical load of 300 N. The initialization creates a regular structure from the input surface to ideally absorb these forces. Through an attraction field defined by a texture image, the optimization then computes for the letters *S* and *I* to appear. As removing bars will distract the visual appearance and perception of the letters, only node positions of the structure are optimized, leaving the topology unchanged. These letter examples demonstrate for models of increasing curvature, how weld connections can move significantly along, and in close proximity to the target surface, without getting trapped in local minima.

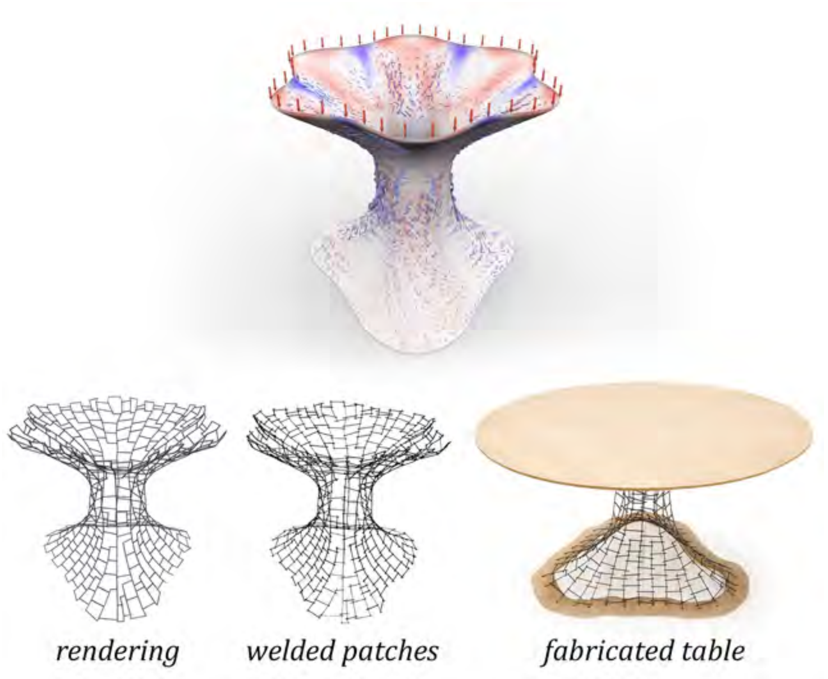

Figure 4.24: **Table.** This example is subject to a 1500 N top load (top). During the initialization, up to three *branching* shorter *continuous bars* are added to bridge the gaps due to the extreme changes in circumference along the height of the model.

Model		**Stool**			**Table**			**Letters**	
	Patch	1	2	3	1	2	3	*S*	*I*
Grid	contin.	5	5	5	4	4	4	15	15
	discrete	10	10	10	17	17	17	19	21
Weights	$w_{\text{s-to-w}}$	2	2	2.2	2	2	2	2	2
	γ	1.5	1	0.9	1.5	1.5	1.5	1.5	1.5
	w_{sparse}	2.5	3.4	3	4	4	4.4	-	-
	w_{target}	5	5	5	5	5	5	10	10
	$w_{\text{reg,length}}$	0.01	0.07	0.01	0.04	0.03	0.03	0.03	0.03
	$w_{\text{reg,sine}}$	-	-	-	10	10	10	-	-
	$w_{\text{reg,cosine}}$	110	150	160	260	200	200	400	400
	w_{art}	-	-	-	-	-	-	1300	1200
Opt Runtime		2m	4m	2m	11m	6m	19m	67m	119m
Collapse	initial	4510 N			5105 N			75 N	365 N
Load	opt.	4695 N			5150 N			305 N	410 N

Table 4.1: *Statistics Summary*: This table reports the size of initialization grids (without additional *continuous bars*), the objective weights, optimization timings for the fabricated examples, as well as the collapse loads before and after optimization. The timings were collected on a 4.0 GHz Intel Core i7-6700K quad-core processor with 32 GB of RAM.

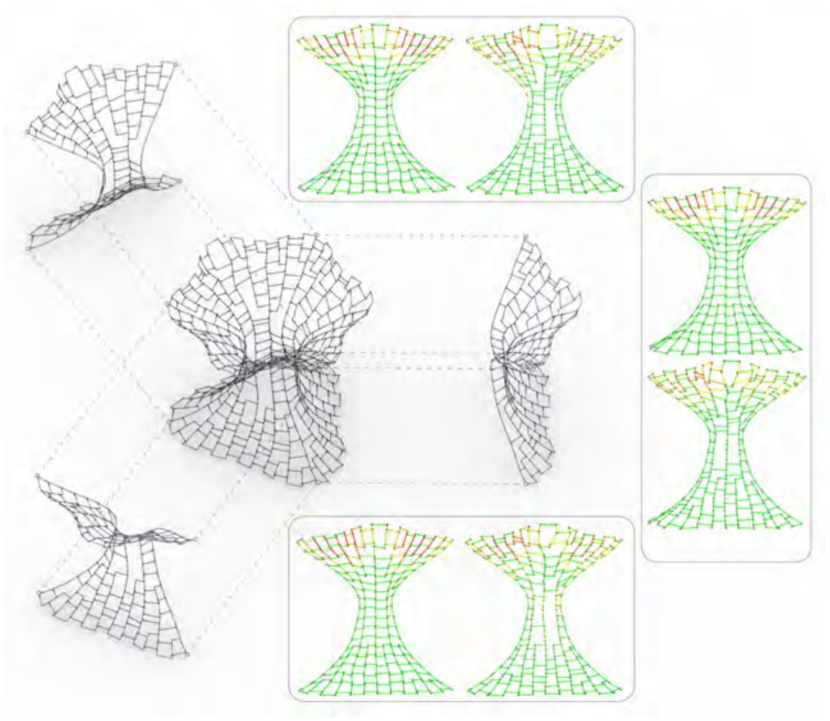

FIGURE 4.25: **Stress visualization of the *table* example.** The visualization of the three patches shows an improvement after structural optimization (compare left to right), resulting in a higher collapse load limit (Section 4.6.1).

4.6.2 Results Comparison

Parameter Modifications Changes to loading scenarios can have a significant effect on the resulting structure. Figure 4.27 shows an optimized wall element under three different loading scenarios: top load, shear load, and cantilever load. Only for the top load case, the *continuous bars* align well with stress lines, leaving shorter bar segments room to align with an artistic target. For the other two load cases, the optimization favors the structural integrity of the result—defined through constraints—at the expense of the artistic intention.

Decomposition Analysis As demonstrated with a curved, uniformly loaded wall element with a bulge in its center in Section 4.6.2, the decomposition strategy results in a very substantial decrease in optimization runtime (10 min vs. 1 h31 min) at the cost of higher average

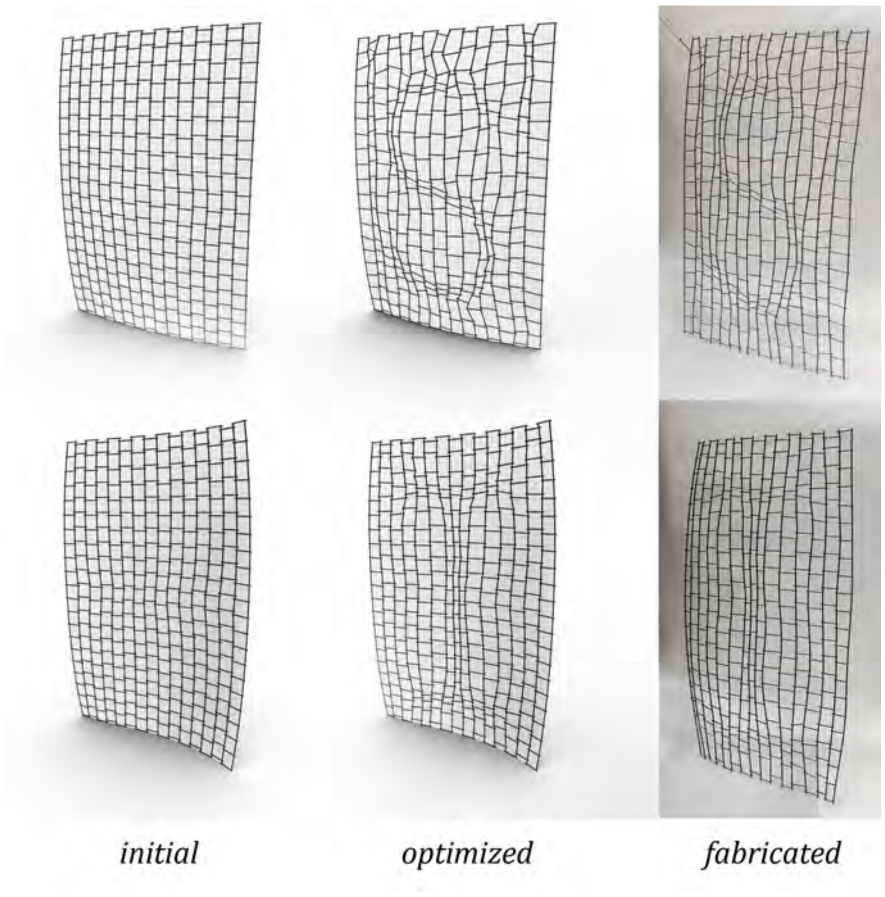

FIGURE 4.26: **Artistic targets embedding into *Wall Elements*, using only shape optimization.** Starting from a regular frame (left), the optimization arrives at a solution that "embeds" the letters *S* and *I* through variations in densities and orientations of bars (center). The fabricated models maintain this visual appeal (right).

displacements and stresses. Note that both optimized models are fabricable.

Direction-Dependence To study the direction-dependence of the load carrying capacity of structures (Section 4.6.2), a comparison analysis is conducted by uniformly loading a wall element with a bulge in its center from the top (vertical, with nodes at the base fixed), and the side (horizontal, with boundary nodes on the opposite edge fixed). As expected, the performance of the structure is best if long *continuous bars* align with the direction of the load case (vertical). At the cost of additional weight, the optimization can compensate for the

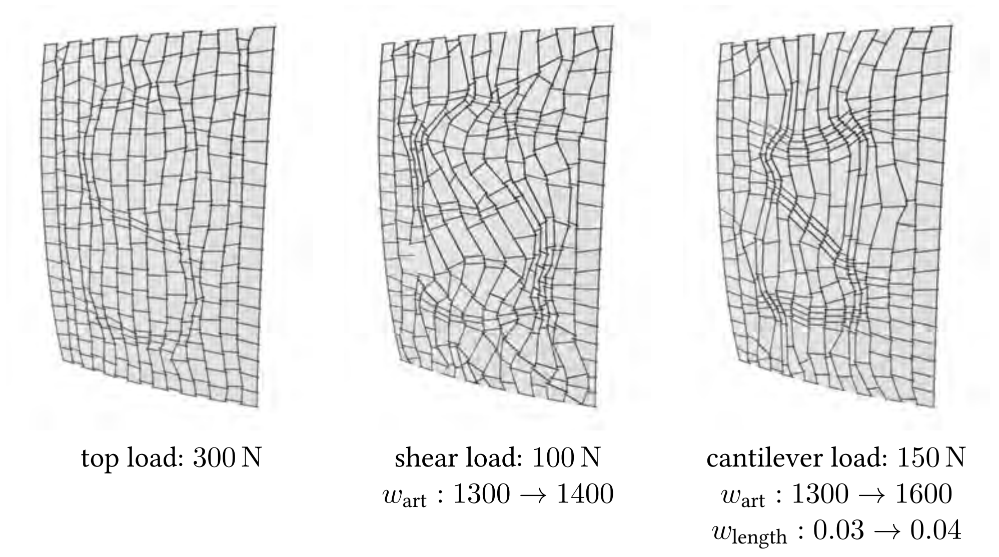
top load: 300 N
shear load: 100 N
$w_{\text{art}}: 1300 \rightarrow 1400$
cantilever load: 150 N
$w_{\text{art}}: 1300 \rightarrow 1600$
$w_{\text{length}}: 0.03 \rightarrow 0.04$

FIGURE 4.27: **Sensitivity to Load Cases.** Changes to the loading scenario of our top-loaded wall element *S* (left) result in significantly different optimization results. Under a shear (middle) or cantilever load case (right), the load carrying capacity of the structure is significantly lower, and the artistic intent can no longer be fully expressed.

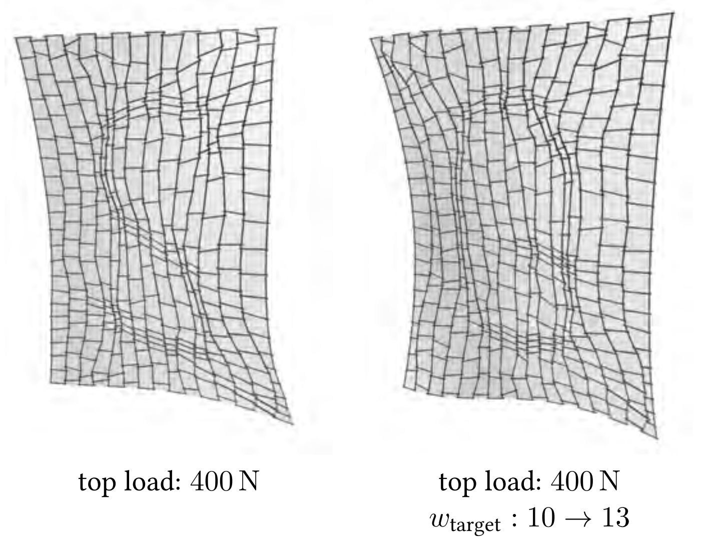

FIGURE 4.28: **Higher-Curvature Target Surface and Artistic Input Modifications.** Replacing the target surface for the *S* with a more curved surface changes the resulting frame structures (left), the artistic intent can still be emphasized by the optimization. Similarly, the artistic target can as well be exchanged, for instance, by using a *G* texture as the input (right).

	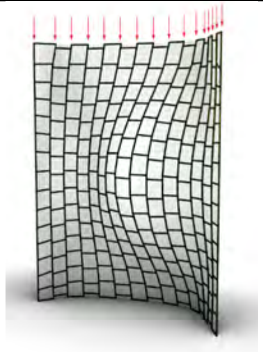	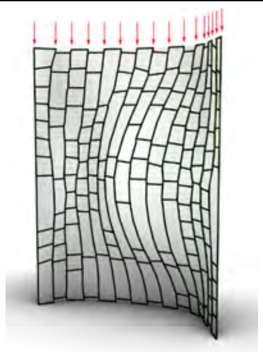	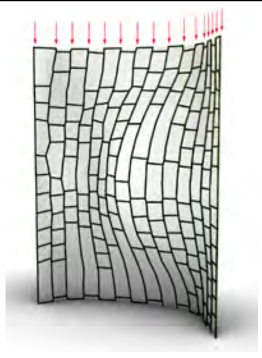
	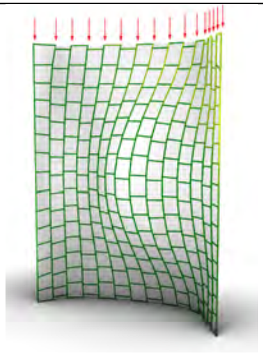	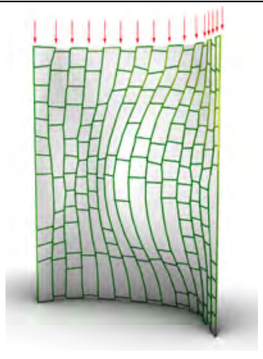	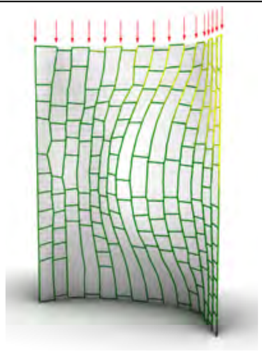
Model	initial	no decomp.	2×2 decomp.
Weight	8.4 kg	7.2 kg	7.2 kg
Avg. Stress	7.1 MPa	6.4 MPa	9.1 MPa
Avg. Displ.	1.61 mm	1.50 mm	2.21 mm
Fabricable	no	yes	yes
Opt. Runtime	-	1 h31 min 1h 31m	10m

TABLE 4.2: **Decomposition Analysis** This table reports the weight, average stress in bars, average displacement of nodes, as well as the runtime of a global optimization, and a subdivision into 2×2 units.

performance gap in structural strength (horizontal).

4.7 Discussion

The *Frameform* project presented in this chapter developed a full system to integrally design and optimize metal fabricable frame structures from an input mesh geometry for an upgraded end effector inherited from the *Mesh Mould* project. The system employs a "global-to-local" strategy for processing the geometry and solves a constrained optimization problem to resolve

	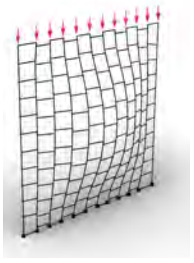	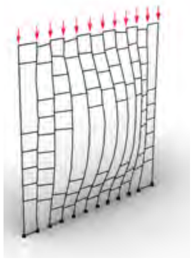	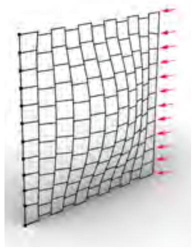	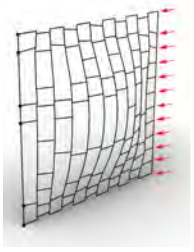
	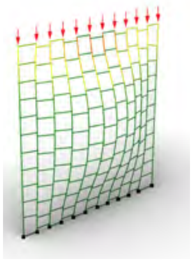	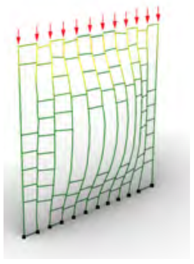	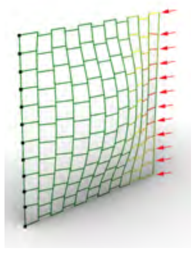	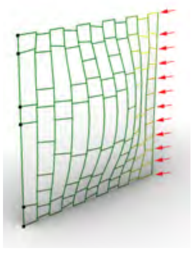
Load	vertical		horizontal	
Model	initial	optimized	initial	optimized
Weight	6.5 kg	5.3 kg	6.5 kg	5.8 kg
Max. Stress	199 MPa	167 MPa	224 MPa	161 MPa

TABLE 4.3: **Direction-Dependence.** This table reports the analysis of the weight and maximum stress for a wall element optimized under either vertical or horizontal loading. The topology objective successfully removes the bars around the curved area, as forces are transferred directly in the flat area. The optimization also maintains some of the bars near the curved area to resist buckling.

all the requirements posed by fabricability or non-fabricability constraints.

Apart from the very hardware-specific contributions concerning geometric topology and fabricability constraints formulation, this project, in a wider perspective, showcases a general strategy for robotic fabrication processes targeting structural assemblies where a discrete set of entities is assembled into a large-scale structure.

Though the "global-to-local" approach scales well with the number of bars, it introduces inconsistencies in forces and moments along unit boundaries. The optimization of the local units also does not guarantee a necessary global optimal result after combination. Since it is not difficult to check the structural stability of the final optimization result, or to identify a scenario where this assumption is responsible for non-fabricable output, the exploration of optimization strategies that alternate between a global analysis and local design optimizations is an exciting future direction to resolve the inconsistencies brought up by the decomposition-to-initialization process.

In terms of fabrication, the complexity of metal frames is restricted by the fabrication constraints imposed by the specific robotic fabrication process. With the fabricated examples, the optimization process also concretely identifies a series of hardware limitations that prevent the fabrication of structurally more efficient forms, indicating by prominent time cost in fulfilling the corresponding constraints, especially the bounds on the insertion and bending plane angles turn out to be limiting factors. Since these constraints are explicitly modelled, the tightly integrated design and fabrication system can identify the ones that are most restrictive, and in turn advise the design of the next generation hardware by taking not only fabricability but also structural performance into consideration for the geometry topology.

With the fabrication constraints, only local robot-structure collisions are avoided. The input model should be assumed self-intersection free, and of moderate curvature and complexity. If pairs of surface regions come too close in a model, it is best to decompose it into units where such pairs are part of different units. Discussions of the future work can be found in Chapter 6.

4.8 Contributions

Due to the interdisciplinary nature of this dissertation Section 1.3.2, this project was conducted in the collaboration between Gramazio Kohler Research (GKR) and Disney Research (DR) under the supervision of Dr. Moritz Bächer, Prof. Matthias Kohler and Prof. Fabio Gramazio.

Dr. Christian Schumacher (DR) provided coding guidance for the optimization framework the author developed, and provided a tremendous amount of valuable feedback throughout the development of the project. Alex Walzer conducted all the fabrication tasks of the project and led the hardware updates to the hardware legacy from *Mesh Mould* in close communication with the author.

Dr. Romana Rust (GKR) was involved throughout the project and provided valuable feedback during the development.

Chapter 5

Project II: RobotSculptor

Style-Driven Robotic Clay Sculpting

We are all in the gutter, but some of us are looking at the stars.

— *Oscar Wilde*

5.1 Introduction[1]

Sculpture is one of the oldest forms of three dimensional visual art. Amongst all the materials used for sculpture, clay is probably the most widespread and frequently used. Its malleability allows it to be formed into any shape imaginable and makes it suitable for both additive and subtractive processes. During a sculpting process, artists utilize a variety of techniques and employ different tools to reform a piece of clay until it satisfies the intent of their artistic expression.

As for the AEC industries, similar materials are mainly utilized for architectural decorations and ornaments, including plaques on ceilings, relievos on historical walls, etc. (Figure 5.1) These artefacts are usually sculpted directly out of concrete, or casted from a sculpted mould with plaster or polymer. Modern techniques such as CNC milling or 3D printing have also been used in modern times as an automated replacement for humans, or for a more accurate procedure to transfer 3D objects designed through CAD software to the physical world.

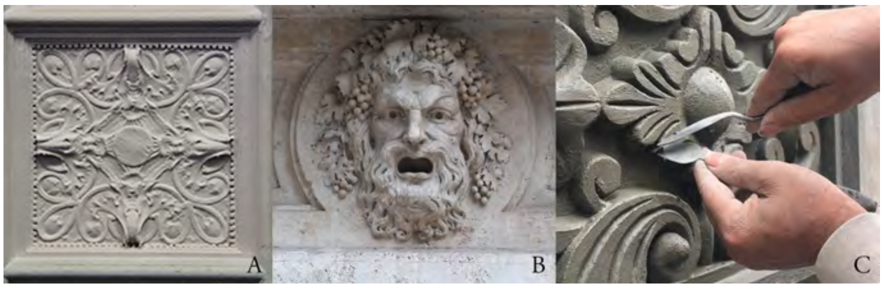

FIGURE 5.1: **Architectural sculptures made of malleable materials**: A. Plaster ornament in the 10th floor library of Roosevelt University designed by Adler and Sullivan; (Image by Terence Faircloth; license: CC BY-NC-ND 2.0.) B. Ungarische Staatsoper in Budapest, Maskaron; (Image by GFreihalter; licence: CC BY-SA 3.0) C. An artist is sculpting an ornament in process.

However, though in today's industrialized context, CNC milling is widely used for the manufacturing of three-dimensional sculptural artefacts and often substitutes traditional processes including stone carving or foam cutting, conventional CNC milling techniques are

[1]Some of the contents of this chapter have been published in Ma, Duenser, et al. (2020)

limited when applied to soft materials like water-based clay. Highly malleable materials are notoriously difficult to cut mechanically, and the strong adhesive tendency of clay greatly hinders the removal of small shavings. A special technique called ”cryogenic machining” uses low-temperature coolant to freeze soft or elastic materials (for instance, rubber) temporarily during the milling process (Dhokia et al. 2011), but no precedence can be found for the clay material.

On the other hand, human sculpting still holds a special place due to its close association with arts & crafts. Due to the non-linear nature of the design process, artists usually rely on an interactive process to think and create through minds and hands simultaneously. Compared to machining, manual clay sculpting satisfies precisely this need, with its modifiable and superimposable layers of sculpting strokes. Additionally, the often imperfect surface finish records the working process of the artist and thus becomes a feature of artistic expression. Such patterns and textures are difficult to achieve and often not considered in CNC machining - and, if required, they are typically only achieved by subsequent special surface treatments.

With the long term goal of endowing robots with human-level skill, this chapter presents the *RobotSculptor*—a user-guided design and path planning framework for robotic clay sculpting. By isolating those parameters related to the aesthetics of the sculpture from the fabrication process, the project in this chapter enables the user to define the sculpting “style” of the result by generating feasible motion trajectories that can be executed by robot arms with the specially designed tool.

In order to make the computational problem tractable, the project described in this chapter focuses on a sculpting process that only involves subtraction of material, using custom-shaped wire loop tools (Figure 5.4). The use of such a customized tool poses two challenges: since the tool is directional, no off-the-shelf algorithm is available, as conventional path planning algorithms for CNC machining often treat the milling bit as central-symmetric. New planning algorithms are needed to suit the chosen tool (*i.e.* manage all 6 DOF). The other challenge is how to support a wide range of possible sculpting strokes that unleash the expression of the user's creativity and comply with the above algorithms simultaneously.

The investigation begins with a set of experiments aimed at identifying the primary parameters that affect the expression of the user's design intentions. Building on the insights gained through these experiments, this chapter then addresses the challenges listed above by separating the entire pipeline into two independent units:

- **User-guided Initialization** exposes a set of parameters for the user to control interactively, and transfers the input style information into a series of initial tool positions, i.e. a toolpath, that matches his/her design intention. This part aims to provide the user freedom to design the sculpting strokes;
- **Path Planning** takes the initial toolpath as input to an optimization and computes the complete robot trajectories. This part aims to find a high fidelity approximation of the input that balances accuracy and design expression. It further resolves any collision and respects all other workspace constraints of the fabrication robot.

This chapter demonstrates the versatility of a computational approach on a set of examples of increasing complexity, and finally fabricates these objects with a Universal Robot with 5kg payload (UR5), to assess the degree to which simulated results translate to the real world.

Recalling the fabrication constraints described in Chapter 1, this project falls into the second category—when the desired robot motion is limited (the fabrication constraints described in Section 4.5.3) and prevents the execution of the desired movements, the target needs to be modified (in the minimum amount for preserving geometry proximity) for compensating the successful fabrication of an alike version of the same target.

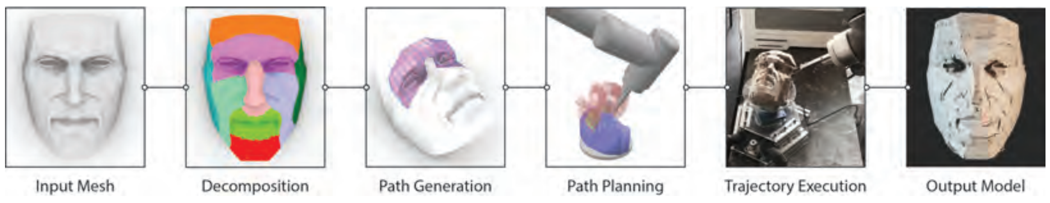

FIGURE 5.2: **System overview.** It takes four steps to design & sculpt a given model with specific styles: 1) the system takes a general triangle mesh as input and decomposes it based on the drawn strokes. 2) The user specifics sculpting styles based on the patch-level parameters and generate a set of initial toolpaths using the system. 3) Using the initialized toolpaths as input, the optimization will compute robot trajectories while maintaining style information and resolving collisions simultaneously. 4) The trajectories are executed on a UR5 to sculpt a physical clay model that matches the optimized results.

5.2 Overview

The general structure of this chapter follows the design-fabrication process that takes a geometry model as the *input mesh* (Figure 5.2) and processes and optimizes the abstracted objectives while considering constraints during the fabrication. After this section, a section specialized to this project is presented , following a "global-to-local" strategy, the model is decomposed into sub-elements for defining the artistic expressions individually which are embedded as a source of input information into a set of toolpaths initialized for each sub-element. These toolpaths are then fed into an optimization component to final optimal toolpaths that can sculpt a physical piece of clay as close as possible to the digital model, preserving the styles defined by the user during as much as possible.

Figure 5.2 illustrates a more detailed pipeline:

1. Given a mesh representation of the target model as input, the user sketches free strokes on the model to define preferred independent areas for further processing.
2. For each disconnected stroke group, the system will compute a decomposed patch.
3. With a minimal set of user input on directional guidance, the developed system generates the initial sculpting paths for each patch.

4. After the toolpaths are initialized, the optimization will adjust the toolpaths to eliminate collisions and smoothen sharp corners, while still maintaining the artistic expression.
5. Finally, once the toolpath is successfully optimized, one can preview the simulated result, or directly execute the trajectory information computed from the optimization to obtain a physical artefact of the "stylized" target geometry in clay.

Since this project aims to explore how optimization-based methods can help robotic fabrication by utilizing the 6 DOF provided by the robot arm and finding optimal trajectories for a specially designed tool, the design iteration of the hardware will be presented first along with necessary experiments. A dedicated section for extracting design parameters follows, introducing how different parameters weighs and how they are selected and incorporated into the design of the system in later sections. Afterwards, the "global-to-local" strategy used in this chapter, namely the mesh decomposition and the toolpath initialization components, will be expanded in detail as a general framework for defining sculpting style per sub-area. Then, the optimization section follows, with a thorough description of how the objectives are formulated, and how the constraints are extracted and incorporated. Unlike the *Project I* where the constrained optimization is utilized, all the constraints in this project are incorporated as regularizers to allow the implementation of an unconstrained optimization problem. The demonstrations and results will be presented in the end, demonstrating the successfully fabricated examples of different shapes with style variations.

5.3 Fabrication Setup

5.3.1 Customized Loop Tool

Unlike *Project I* where the end effector is ready to use, *Project II* requires a design of the end effector from scratch for conducting the sculpting tasks. Inspirations of the tool design are borrowed from professional artists and sculptors, who use various types of tools for different stages of a complete sculpting process (Figure 5.3).

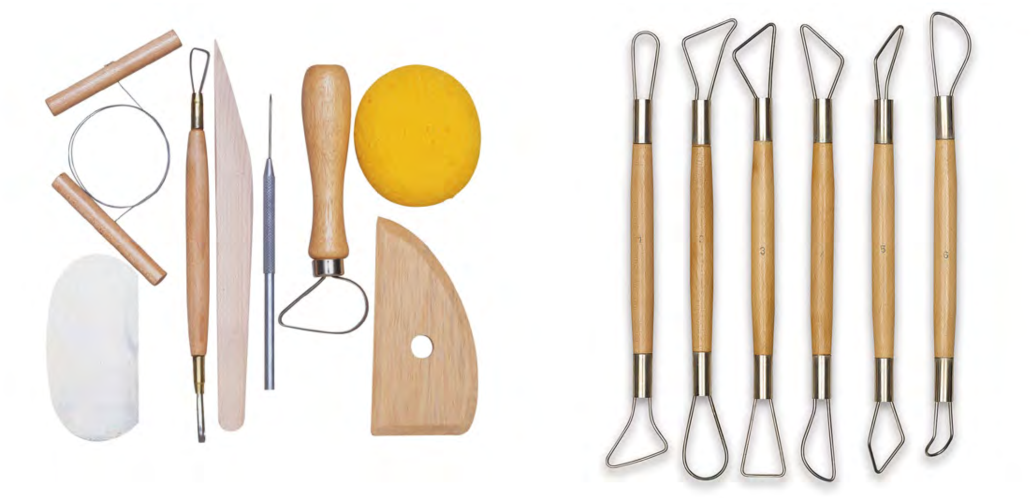

FIGURE 5.3: **Tools used for clay sculpting.** Left: A complete ceramic tool set, including cutting wire and plate tools for rough shaping, loop tools for modelling, detailing for fine-tuning, and sponge for smoothing; Right: Loop tools of various profiles.

While the plate tools are used for shaping and modelling in the beginning stage, and detailing tools with different designs are used for creating detailed surface effects and textures, the loop tools are comparatively used in a more general context. With various sizes and shapes, the usage scenarios of the loop tools range from removing large strips of material to carving general details which do not require special tool treatment.

A conventional *loop tool* (Figure 5.3) for sculpting clay consists of a handle and a planar “loop” that is bent from a piece of steel wire or from a piece of thin, narrow metal strip into a rectangular, triangular, or circular profile to fulfill different cutting needs (size, angle, texture effect, level of detail, etc.). While the sculptor uses his/her hands or the plate tools for the modelling (additive) and formative process, such loop tools are usually used for the subtractive process—cutting a strip of clay off by moving the tool along a desired direction.

The designed tool is composed of two main functional parts: a metal handle with a 3D-printed plastic base that help to connect to the UR5, and a replaceable steel wire “loop” that can be attached to the end of the metal handle. Thus, a sculpting process can be conducted by controlling the robot arm to manoeuvre the loop follow desired toolpaths. After several iterations of prototyping (AppendixA.1), the metal handle is finalized as a customized aluminium piece with a square cross-sectional shape of 10 mm side length. By changing the wire loops, fabrication experiments of various scales can be conducted with the same tool set. The plastic base that connects the metal handle and the robot arm was also updated accordingly, adding positions for an additional screw so as to get better stability. The tool is shown in Figure 5.4.

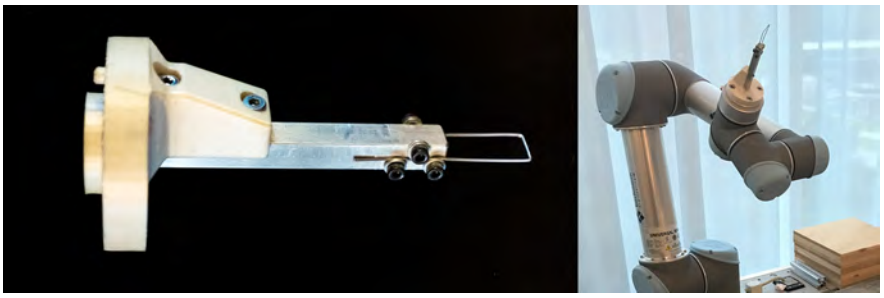

FIGURE 5.4: **The customized loop tool and the UR5 with the tool attached.**

From a fabrication point of view, one major difference between the customized loop tool and a conventional milling bit is in symmetry. While the latter is usually axisymmetric and rotates around the central axis, the former is not. This characteristic of the tool benefits the sculpting process by allowing the loop tool to cut off clay strips of different widths and sizes by simply rotating the tool around its axis during the cutting process. However, while this additional flexibility parameter is trivial for human users to control, it adds significant complexity to the planning algorithm—the additional degree of freedom needs to be managed and exploited.

5.3.2 Turntable

To allow more sculpting poses for 3D models that usually requires the robot to reach from relative extreme angles, a customized turntable was built to expand the accessible workspace of the robotic system. Several iterations (AppendixA.1.1) were made to finalize the design of the turntable through prototyping and validation.

The final version of the turntable controlled by an *Arduino Uno* was built as shown in Figure 5.5. With the help of the gear system, the motor can rotate the table in both directions with 1.8° resolution, and acts as the 7th axis of the robot during the optimization and the fabrication phase.

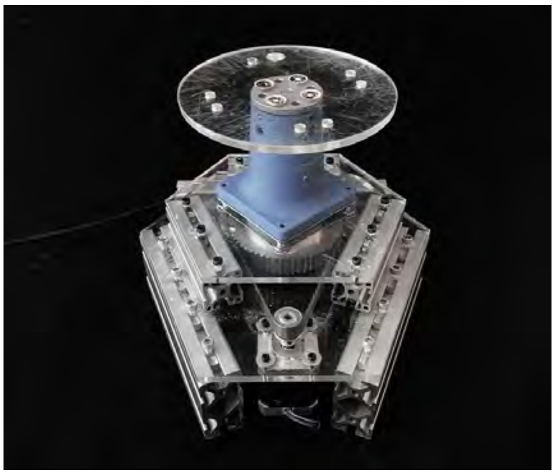

FIGURE 5.5: **The final design of the Arduino-controlled turntable.**

It has been investigated that using the turntable as an additional axis of the robotic system and syncing it with the robot movements require task threading and modification to the low-level UR driver (Andersen 2015). For this project, an asynchronous approach is employed to reduce the engineering complexity—the turntable will stay fixed during each sculpting toolpath and rotate only between two consecutive toolpaths, if needed.

Theoretically, this may limit the accessible workspace of the robotic system when the robot reach cannot fulfil all the positions along one single sculpting toolpath. But no real scenario is constrained by this issue along the whole development of the project. It can be assumed that as long as no single toolpath is extremely geometrically complex, the “rotate-sculpt-rotate” mechanism will fulfil most of the sculpting tasks within the project range.

5.4 Design Factor Extraction

Instead of developing a fully automated system similar to existing path planning software for CAM, this project intends to provide the user with control over certain aspects of the fabrication process that are relevant to the design and appearance of an object. Thus, it is essential to first understand what will affect the fabrication result of a sculpting process so as to define the parameters that can be built into the system.

In order to reveal the most important design parameters that affect the results, a series of experiments involving the interaction between the tool and the clay material is conducted. These experiments are designed to reveal the following three aspects:

- Understand the relationship between the material deformation and sculpting velocity;
- Guide the selection of suitable shapes and sizes of the customized tools;
- Decide on a minimum amount of parameters exposed to the user to exert control on the toolpath generation.

5.4.1 Parameter-extraction Experiments

This chapter categorizes the experiments into two classes: patch-level parameters and path-level parameters. The patch-level parameters affect the selected areas ("patches", see Figure 5.2-2) of the mesh on which the preferred sculpting styles are applied; the path-level parameters affect the toolpaths generated on each patch. The selected patch can be either a portion of the whole mesh or the mesh itself.

Patch geometry (patch-level). This parameter is directly related to how an input model is decomposed. It defines the area and shape to which a particular style can be applied, and can be created with various methods. A sketch-based method is implemented for the interactive design process in the developed system (Section 5.5). And how this parameter can affect the final fabricated results is shown in Figure 5.20. For simple cases, manual decomposition in any mesh operation software may suffice.

Patch overlap (patch-level). Undesired material aggregations are found near the seams between individual patches, causing a clear visual separation. This is caused by the high ductility of the material—when the tool enters or exits the clay at the material interface, it will carry forward some material by pushing or pulling rather than causing a clean separation. This effect is most visible when the enter and exit locations accumulate together. Experiments show that this effect can be reduced to a satisfactory magnitude by introducing an overlap between adjacent patches, as illustrated in Figure 5.6, so that the said material accumulation is eliminated.

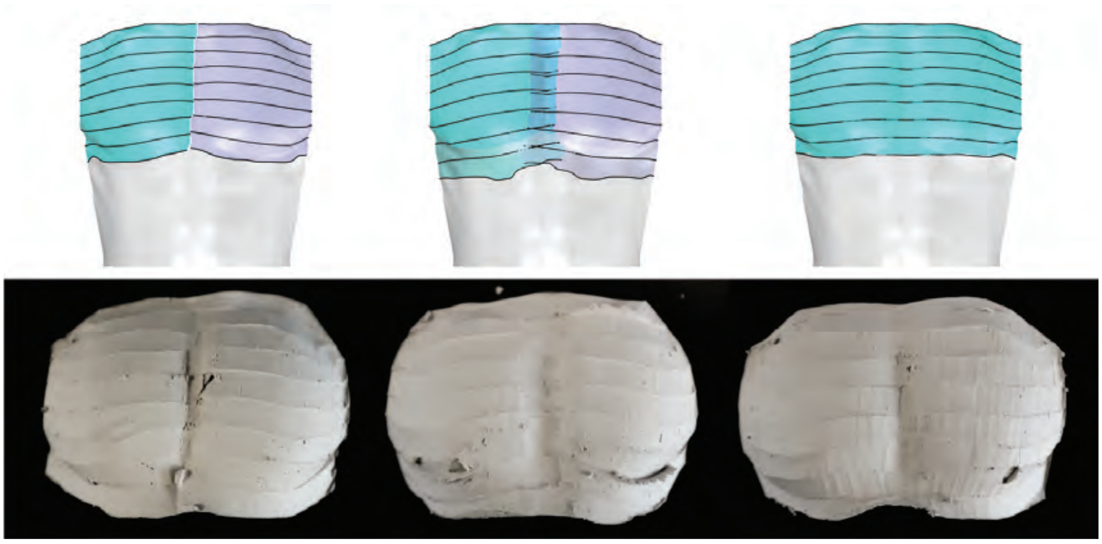
FIGURE 5.6: **Seam comparison models.** Left: sculpting toolpaths intersected at seam area without overlapping; Middle: sculpting toolpaths intersected at seams with overlapping; Right: continuous sculpting toolpaths across the whole surface.

Toolpath length (path-level). The above material property has a similar impact on the toolpaths generated for a specific patch. Regardless of the generation methods, a toolpath will start/end in three circumstances: 1) at the start/end point of another toolpath (toolpaths are connected), 2) at the middle of another toolpath (toolpaths are overlapped), 3) at an empty area (toolpath are disconnected). However, experiments show that for a specific patch, 1) and 2) will always create leftover material at the intersection, and 3) will result in an area not sculpted at the target geometry. As the number of intersection locations is largely decided by the number of toolpaths, long toolpaths are preferred to reduce unnecessary leftovers. Two extreme cases are shown in Figure 5.7, where one contains randomly generated short toolpaths in various directions and the other contains only aligned toolpaths across the entire surface.

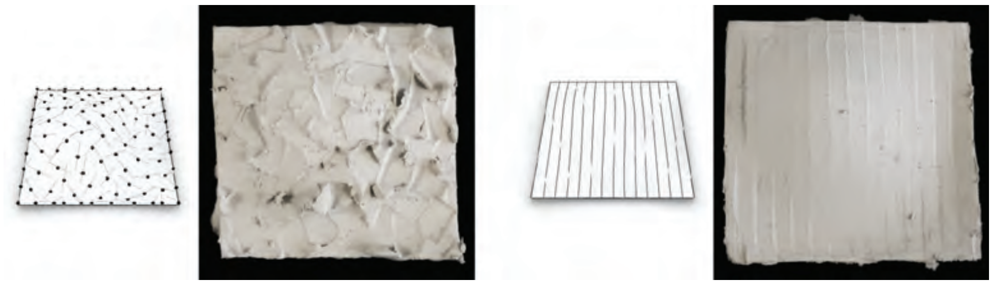

FIGURE 5.7: **Point Sampling Sculpting Test.** Left: a surface sculpted with 100 toolpaths of 18 mm length in random direction, generated from 100 randomly sampled points; Right: the same surface sculpted with 15 parallel toolpaths across the whole width of the patch.

Toolpath direction (path-level). This parameter affects the toolpath generation process as it is the most important one to define the artistic style the user would like to achieve. It affects the visual effects of the sculpted stripe patterns on the final surface as well as the cutting depth into the clay. A Laplacian-based algorithm is used to generate evenly-distributed parallel-aligned toolpaths on top of each patch, and the details are explained in Section 5.5.3.

Tool direction (path-level). As shown in Figure 5.12, the three rotation parameters define the local pose of the tool. Experiments confirm that the *aligning direction* will affect the precision of the target surface, and the *facing direction* will affect the amount of material cut by each toolpath. These parameters together will also affect the final surface quality (Section 5.5.4).

Secondary parameters. Besides the parameters listed above, experiments were also conducted with several other parameters, though these appeared to be less effective to influence the design and fabrication results. For completeness, they are listed below:

- *Density of toolpaths*: This parameter needs to be set to guarantee the sculpted area to cover the whole patch. Beyond that, increasing the density increases optimization time with limited gain for a selected tool. However, this parameter is still exposed to the user to compensate for any change in the tool shape. Figure 5.8 shows the visual effects caused by a loop tool with a semicircle profile.

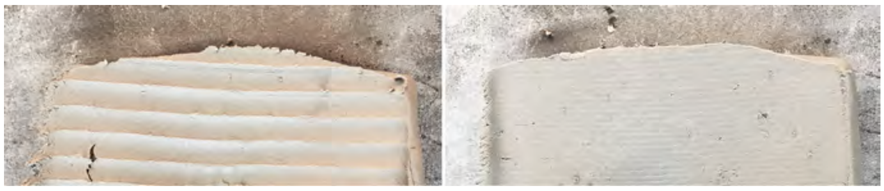

FIGURE 5.8: **Surface quality differences due to a non-flat profile loop.** The local stripe effect caused by the leftovers between toolpaths is homogeneous with the tools. For a loop tool with selected shape, the user can adjust this parameter to obtain a desired final appearance.

- *Inclining direction*: This parameter does not affect the result as much as the other two listed in the *Tool direction* categories, as long as it does not cause any collisions.
- *Tool shape*: As shown in Figure A.2, various tool shapes were experimented with. Since changing the tool during a sculpting task is not allowed, the final appearance sculpted by a selected tool is thus determined by the other parameters. The system thus relies on the optimization to resolve collisions and match the surface, and thus excludes this parameter from the design stage.

5.4.2 Material Properties

As briefly mentioned in Section 5.4.1, it is discovered that the viscosity and plasticity of the clay will affect, on a local scale, how the clay behaves when the tool enters, sculpts, and exits the materialt, which will then directly affect the final appearance of the sculpted model. There are two main effects: 1) the clay is soft enough to be pushed by the tool during the sculpting process. Though a $1\,\mathrm{mm}$ steel wire can be used to reduce the effect as much as possible, leftover clay is still noticeable along the moving paths of the tool; 2) due to viscosity, the forces introduced by the tool will cause a "pulling" effect when the tool leaves the clay, or even cause failure to detach when the remaining material is incapable of absorbing these forces. This effect mainly happens between the subtracted clay (on the tool) and the clay model, resulting in small leftovers on the target surface.

A complete modelling of the clay is extremely challenging as its material properties change over time when the contained water evaporates gradually. However, for a thin ($1\,\mathrm{mm}$) tool made of steel wire, it is discovered that limiting the cutting speed to within $3\,\mathrm{cm/s}$ to $8\,\mathrm{cm/s}$ can reduce these visible defects to an acceptable level. The author thus decided to conduct the fabrication under these settings and formulate the optimization in a purely geometric approach, resolving robot motion and collision issues without involving any simulation of the material behaviour.

5.4.3 Style as an aesthetic feature

One of the core contributions of this piece of work is to embed the user's design expression into an automated robotic fabrication process as the sculpting styles. The author successfully develops a "style function" of the design parameters abstracted from a series of design experiments and integrated it into the developed system so that it can transfer these styles from the digital environment to physical artefacts with ease.

While conventional CNC milling prefers precision, the designed system favours the possibility to create various visual styles with the minimum amount of effort. As evaluating the magnitude of aesthetically pleasing is subjective, the exploration of the design space is left to the user by providing the maximum amount of freedom to realize his creativity and intention.

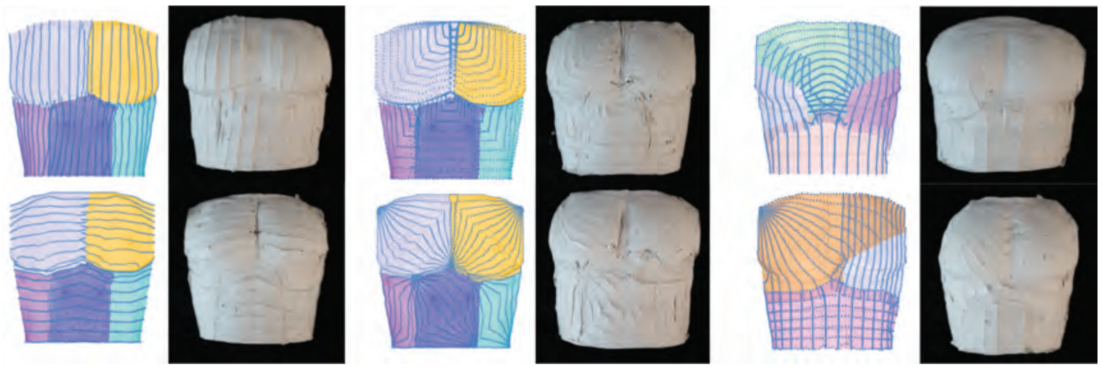

FIGURE 5.9: **Sculpting styles created by *RobotSculptor* with different decomposition schemes and toolpaths.**

Figure 5.9 shows the sculpting style variations of a torso model created by different decomposition schemes or toolpaths. The visual styles formed by the sculpting toolpaths define a unique feature of the sculpting process. The author believes these style variations provide new opportunities to explore new ways of robot control and open up discussions in human-robot interaction. More details will be discussed in Section 5.5 and Section 5.6.

5.5 User-Driven Toolpath Generation

5.5.1 Design Parameters

One important goal of this research is to embed human design choices and expressions as ”styles” into the automated robotic fabrication. It requires the system to maintain a certain magnitude of precision and at the same time generate results that deviate from the homogeneous appearances that are typical for the results of CNC milling. This project relies on the key parameters selected based on the design experiments described in Section 5.4.1 to allow users to generate toolpaths creatively. These parameters will help to transfer essential features into the fabrication process. Following the pipeline described in Section 5.2, the input mesh is assumed as a conceptual quad-mesh (4 corners + 4 edges), and the system will interpret the selected 5 parameters into variables that the user can access and modify in the GUI:

1) Locations of free strokes drawn for model decomposition;
2) Offset distance at the overlapping area between patches;
3) Locations on patch boundaries as conceptual “corners”;
4) Distribution of start and end points of the toolpaths;
5) Number of toolpaths generated on each patch.

1), 2) are *patch-level parameters* and relate to the *Decomposition* process; 3), 4), 5) are *path-level parameters* and relate to the *ToolPath Initialization* process. Both of them will be explained in detail below.

For both CNC milling and the system developed here, one necessary step of the toolpath generation is to develop toolpaths that can cover the whole surface of the input model. While common milling tasks use widely applied strategies including *parallel*, *scallop*, *radial* and *flow-line*, a different procedure is preferred here to generate toolpaths for two reasons: first, existing methods either treat the input model as one global geometry and generate toolpaths with one overall milling strategy for the whole model surface, or rely on the surface/solid component information contained in the CAD file to treat different pieces of the model with different strategies[2]. As the system uses a general mesh representation as input data (surface/solid component information is not provided) and should provide more choices for style variation, a systematic approach is needed to develop different subtracting strategies to suit this case. Second, the robot end effector (*i. e.* the customized loop tool) in this case is not axisymmetric, in contrast to a normal milling bit, and the system needs to additionally align the normal of the cutting plane towards the cutting direction at every sampled point for any effective cut (though it does not need to be aligned fully).

Therefore, a global-to-local strategy is developed to decompose the input model into small patches that can be incorporated with different sculpting intentions of the user. Treating each

[2]CAD models usually contain either information that help to identify the different surface components of a 3D model or construction history that help to extract the construction sequence. Milling software can select these separate pieces to apply different milling strategies. The number of the strategies or the area where milling strategies can be applied depends on the sub-components that the CAD model contains.

patch individually, the strategy generates toolpaths based on the isolines of a scalar field, which in turn is defined through user-provided boundary conditions for each patch. If no decomposition is given, the strategy will treat the whole mesh as a single patch and conduct the toolpath generation over the whole area.

5.5.2 Decomposition

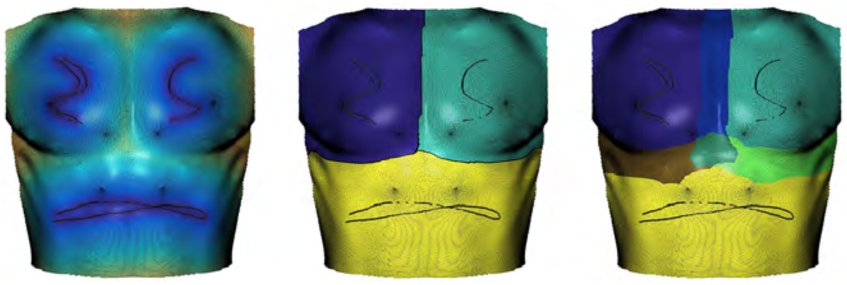

FIGURE 5.10: **Decomposition comparison w/o overlapping boundaries.** Left: distance field calculated from the drawn strokes; Middle: decomposed patches without overlapping boundaries; Right: decomposed patches with overlapping boundaries of 15 additional triangle loops.

The *Decomposition* aims to allow the user to select different areas that can be treated separately for the toolpath generation. A GUI is developed to facilitate this task. The user can draw strokes on the model using a mouse, and the system will compute a distance field for each disconnected stroke. This field later helps to compute separate surface patches using a priority queue. Once the result is visualized, the user can accordingly decide to either draw additional isolated strokes to create more patches, or to intersect existing strokes with further stroke(s) to modify the shape of the corresponding patches (Figure 5.10).

Once the *patch-geometry* is defined, the user can modify the overlapping areas around the borders where patches intersect. As the dimension of the input model may vary, this is achieved by adjusting the number of facets in the overlap areas (Figure 5.10). This adjustment aims to prevent the aggregation of entry/exit locations of the loop tool, which will produce inferior surface quality due to the material behaviour discussed in Section 5.4.2.

5.5.3 Initialization

The *Initialization* process aims to provide intuitive toolpath generation for each decomposed patch. Each patch is treated as a ”quad-like” patch and asks the user to provide four ”cutting” points near its boundary. These points are used to segment the closed boundary curve into four segments, *i. e.* two facing pairs. The vertices of the two segments in one of the pairs are assigned with the value 0 and 1 respectively, and those in the other pair are assigned with values interpolated from 0 to 1.

To generate the isolines, a technique similar to those described in Ma, Walzer, et al. (2020) and Pereira, Rusinkiewicz, and Matusik (2014) is used. For each surface patch a scalar field is computed by solving the common Laplacian equation with boundary constraints:

$$\begin{cases} \mathbf{L}\mathbf{z}(x) = \mathbf{0}, & x \in \Omega \\ \mathbf{z}(x) = \mathbf{z}_0(x), & x \in \partial\Omega \end{cases} \tag{5.1}$$

where $\mathbf{L}$ is the $n \times n$ discrete Laplace–Beltrami operator and $\mathbf{x}$ is the coordinates of mesh vertices. $\mathbf{z}(\mathbf{x})$ and $\mathbf{z_0}(\mathbf{x})$ are vectors of per-vertex values of all the vertices and boundary vertices, respectively. Those elements of $\mathbf{z}$ that correspond to the interior vertices are unknown, while the elements corresponding to the boundary vertices are given as constraints. The user is allowed to modify the toolpaths directions and orientations by adjusting the position of the cutting points, the distribution of the assigned values, and the number of toolpaths (Figure 5.11).

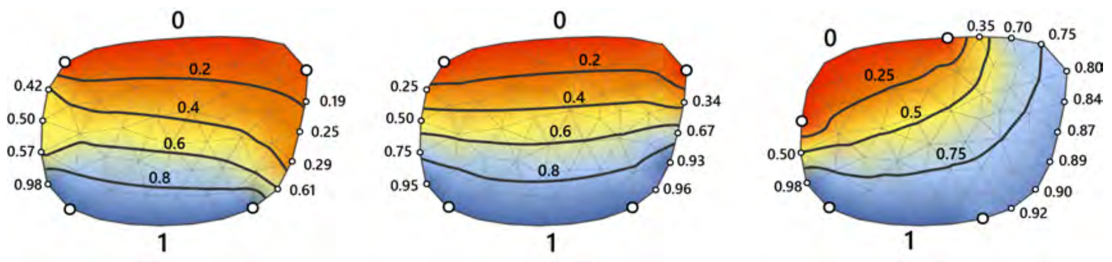

FIGURE 5.11: **Toolpath initialization.** Left & Middle: same cutting point locations, different distributions of assigned boundary values; Right: different cutting point locations, distribution of assigned boundary value and density of toolpaths.

A series of isolines is then interpolated from the scalar field. The user can set the parameters interactively to find a toolpath initialization that matches his vision. Experiments show that an overlap of more than 30 % of the tool width between adjacent paths is needed to allow for the optimization to modify the toolpaths sufficiently in order to avoid collisions or to match the target geometry more closely.

5.5.4 Tool direction modification

Though the *Decomposition* and *Initialization* processes successfully help us transfer the artistic intention to a set of initial toolpaths, the initialization can be further improved through local adjustments of the tool directions. Though the user can generally rely on the optimization to compute the results, experiments show that a better initialization will often lead to better surface quality and faster optimization time, especially in high curvature areas where local minima occur.

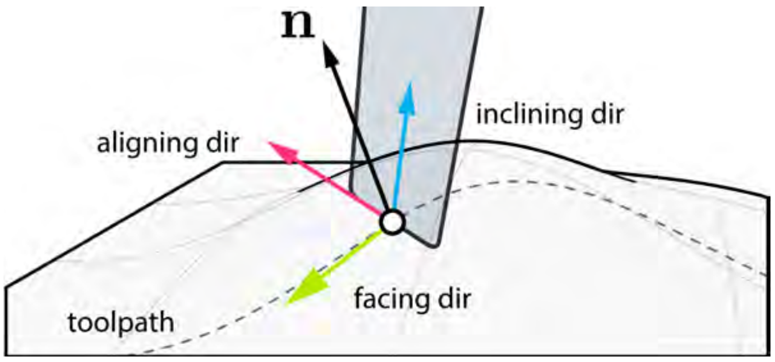

FIGURE 5.12: **Tool vectors.** During a sculpting movement, the tool pose is defined by three vectors: the facing direction, the aligning direction, and the blade direction.

As the loop tool has 6 DOF, the 3 directions that are not constrained by a given toolpath (in fact, a series of tool positions) are defined as *facing direction, aligning direction* and *inclining direction* (Figure 5.12). A milling bit has no facing direction as it always cuts at the width of the tool's diameter. For the loop tool, the cutting profile depends on the projection of the tool profile to the material along the toolpath direction and can be adjusted by its relative angle to the tangent direction of the toolpath.

For a sampled tool location along a toolpath, the *inclining direction* is initialized using the normal direction of the patch, and the tangent direction of the toolpath is projected to the tangent plane of the patch at the referenced point to initialize the *facing direction*.

Additionally, some of the tool's *facing directions* are re-aligned perpendicular to the averaged principal curvature (Meyer et al. 2003) directions near high curvature areas:

$$\mathbf{n}_f = \frac{1}{N} \sum_{r<r_{near}} \mathbf{n}_p \tag{5.2}$$

where N is the number of samples of the principal curvature $\mathbf{n}_p$ within a pre-defined sphere of radius r_{near} around the tool location. The benefits of this post-processing step are illustrated in Figure 5.13.

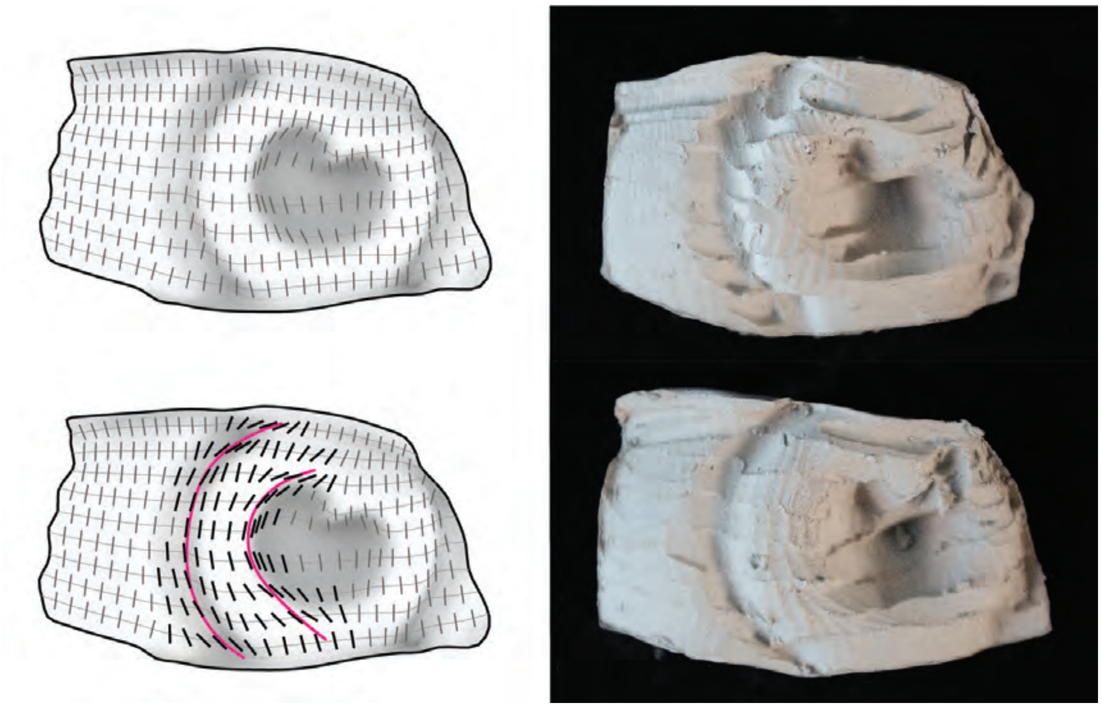

FIGURE 5.13: **Local adjustment of the *facing direction* using curvature information.** The fabrication results illustrate noticeable improvements of the surface quality.

With the above procedures, a general initialization of both toolpaths and tool directions are obtained. However, there is no guarantee that these results can be executed with a specific robot without any collision[3] or reachability problems. It would thus require the user to manually modify the toolpaths iteratively for a specific robot in use to resolve all relevant collision issues, or to use a simplified version / allow minor robot-target collision to exist to some extent (Figure 5.19 middle, Figure 5.20 upper right.)—one of the main reasons that lead to the development of the optimization process described in Section 5.6.

5.5.5 User Interface

Along the development of this project, three independent GUIs and plugins are developed for different corresponding stages. As this project intends to allow users to design and generate sculpting styles with ease, two different approaches are employed as proof of concepts for future development.

For the decomposition component, an independent GUI is developed to allow users interactively conduct the decomposition task (Figure 5.14), as described in Section 5.5.2. The interface is written in C++, using open-sourced 3rd-party libraries listed in AppendixB.3.

The initialization component employs a "plugin" approach that a set of customized scripts are written inside the visual programming platform *Grasshopper* which is built on the CAD software *Rhinoceros*. As shown in Figure 5.15, these scripts take the decomposed patches as the input and initialize toolpaths as described in Section 5.5.3. This interface may not be as easy to use as the independent GUI for the decomposition process, but allows more control for advanced users, especially those who are experienced in CAD software. Additionally, it also helps to foster the development speed as the developer only needs to take care of the non-GUI context.

[3]Collisions referred to in this chapter include both the self-collision of the robot between its different parts and the collisions between the robot and target object, which will create imperfection on the target surface. Please also refer to the figures in AppendixA for more details.

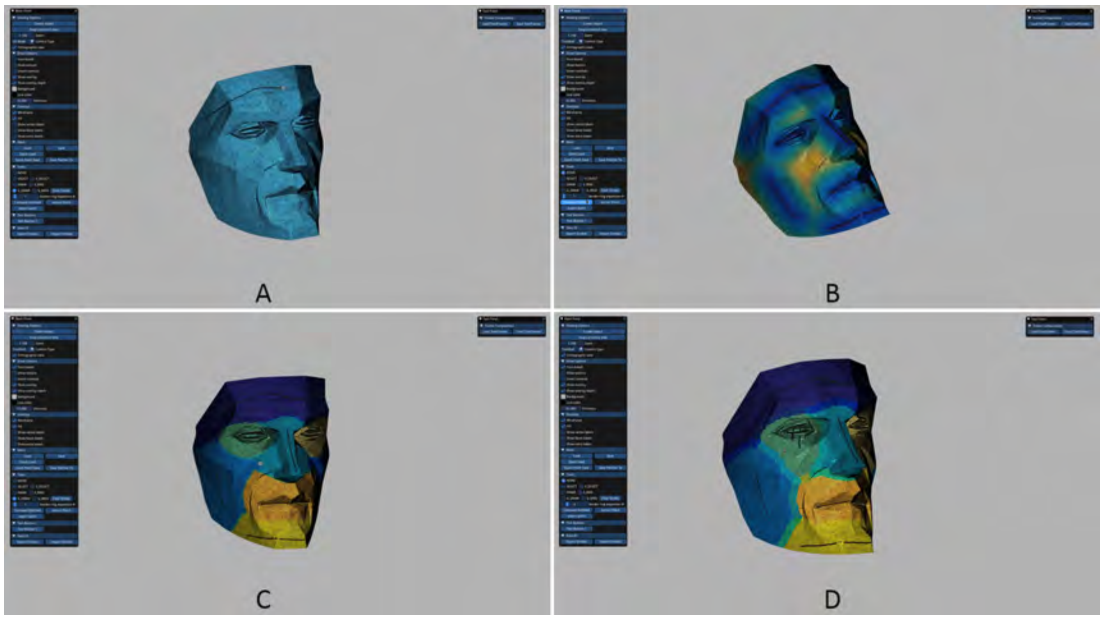

FIGURE 5.14: **The customized GUI for the decomposition task:** A. Drawing strokes; B. Computing distance fields; C. Generating patches; D. Adjusting overlapping borders.

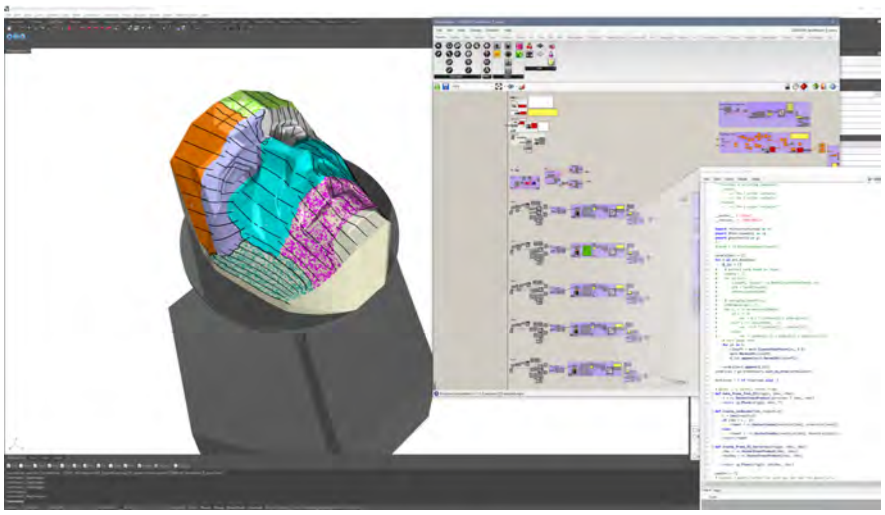

FIGURE 5.15: **The initialization interface in Rhinoceros platform.**

The GUI for the optimization component is built on the one originally developed in Duenser et al. (2020). While the objectives are changed for our project, the GUI has not been modified much, except for adding parameters relevant to the turntable. A screenshot is provided here for reference:

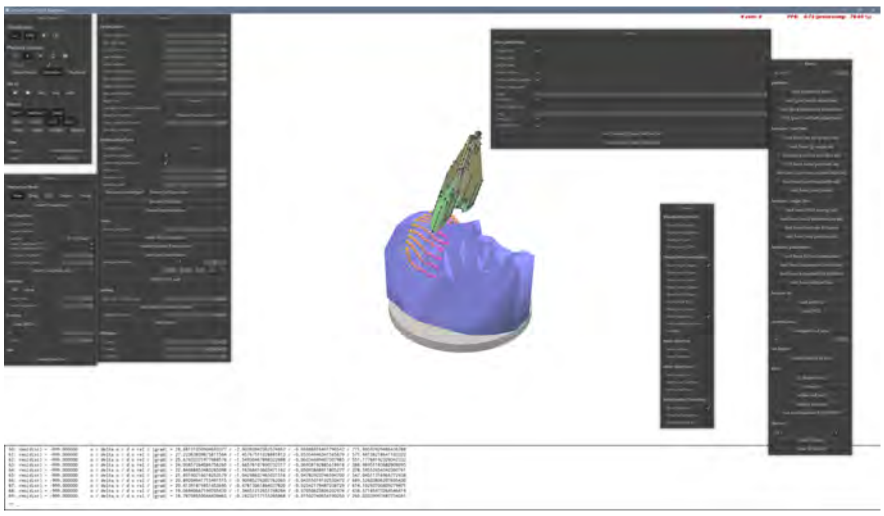

FIGURE 5.16: **The optimization interface.**

5.6 Optimal Path Planning

The toolpath generation in the previous sections defines a path that sweeps the target surface closely and expresses the aesthetic preferences of the user. It is, however, not guaranteed to be feasible, in the sense that it can be executed by a given robot without causing collisions (both self-collision of the robot and robot-target collision) or exceeding the robot's reach. Therefore, given a patch of the target surface and the associated toolpaths (collectively referred to as *input toolpath* in this chapter), a robot trajectory needs to be found that 1) is feasible, 2) produces a cut surface that best approximates the target surface and 3) maintains the overall aesthetics of the cut surface implied by the input toolpath.

The development follows an approach similar to the one proposed by Duenser et al. (2020) for computing cut trajectories for an elastically deformable tool, manipulated by a two-armed robot. At the core of this approach lies the formulation of an optimization problem which matches the surface swept by the tool during movement (*toolsurface*) with the surface of the input model (*target shape*). In particular, the author uses similar formulations for the physical model of the system, the final *primary objective*, the *constraint objectives* and the last two of the *secondary objectives*, as introduced below.

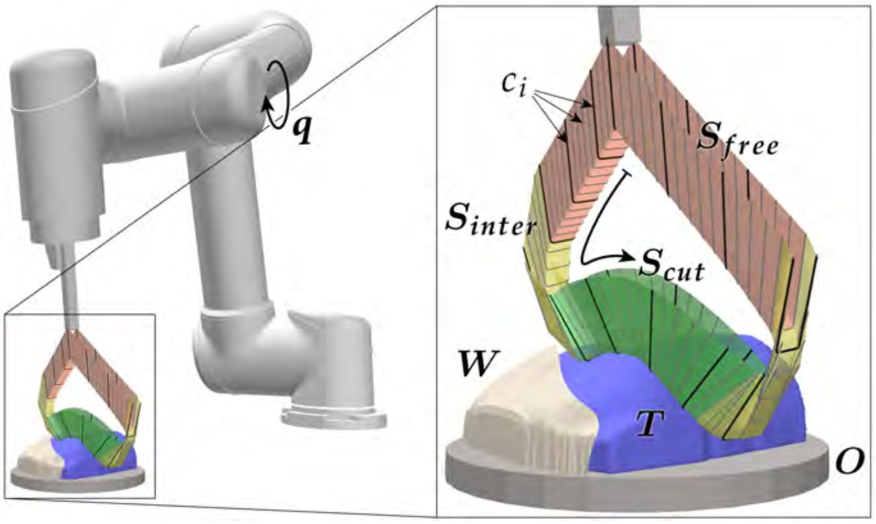

FIGURE 5.17: **An overview of the main components of the optimization model.** The robot is shown in its rest pose, from where it traverses towards the workpiece (toolpath $\mathcal{S}_{free}$ and $\mathcal{S}_{inter}$) and performs the cut ($\mathcal{S}_{cut}$). The robot then moves back to its rest pose—though typically it would loop around and perform a number of successive cuts, optimized simultaneously, to carve out the entirety of a given surface patch.

Model description The robot trajectory is represented through a sequence of robot poses, each defined by the set of joint angles $\boldsymbol{q}_i$, collectively forming the full trajectory $\boldsymbol{q} = (\boldsymbol{q}_i)$. In case a turntable is used, the turntable is simply viewed as an additional robot joint, and its orientation is included in $\boldsymbol{q}$. The tool is rigidly attached to the robot end effector and modeled by its center line $\boldsymbol{c}_i$, such that the path swept by the tool forms the *toolsurface* $\mathcal{S}$. Between the discrete steps of the trajectory this surface is approximated as piecewise linear. Using a kinematic model for the robot, the toolsurface is then fully defined through the joint angles as $\mathcal{S} = \mathcal{S}(\boldsymbol{q})$. For a full description of the setup, the author further considers the target shape $\mathcal{T}$ and its currently processed subsection $\mathcal{T}^*$, the current shape of the workpiece $\mathcal{W}$, as well as any other obstacles $\mathcal{O}$ in the scene, such as the turntable. If the target mesh is split into several patches, the shape of the workpiece is updated after applying each of the corresponding cuts. See Figure 5.17 for an overview of the simulated setup.

Optimization problem Similar to Duenser et al. (2020), an unconstrained optimization problem[4] of the form

$$\min_{\boldsymbol{q}} \quad E(\boldsymbol{q}) = E_{prime} + E_{constr} + E_{sec}, \tag{5.3}$$

is formulated, where all physical constraints are enforced through penalty terms, collectively denoted as E_{constr}. The principal design objective E_{prime} defines a cost for the distance between the toolpath and its target, while E_{sec} collects several secondary objectives, as laid out in more detail below. This minimization problem is solved using Newton's method with line search and a Levenberg-Marquardt type regularization.

The trajectory and the corresponding toolsurface to be optimized consists of several distinct, predefined subsections: One or more cut portions $\mathcal{S}_{cut}$, in accordance with individual cuts of the input toolpath, which are designated to carve out the target shape. Transitional portions $\mathcal{S}_{free}$, which describe the free movement in-between individual cuts, as well as from and to a fixed robot rest pose. And finally, intermediate portions $\mathcal{S}_{inter}$, which are short connecting sections at the interface between $\mathcal{S}_{cut}$ and $\mathcal{S}_{free}$. While the toolsurface of these sections may take part in cutting through the material, it is not optimized to match the target shape.

[4]For using unconstrained formulation to solve problems with constraints, please refer to Section 3.1.2 for more details.

5.6.1 Primary Objective and Constraints

Surface Matching The primary objective E_{prime} measures the closeness between the toolpath and the given target. This is viewed as a non-rigid surface registration problem and match the target surface $\mathcal{T}^*$ with the toolsurface $\mathcal{S}_{cut}$. Starting from a dense set of sample points on the target surface $\mathcal{T}^*$, the optimization penalize the absolute distances to their respective closest points on the toolsurface $\mathcal{S}_{cut}$. In principle, a simple quadratic penalty could be used for this. Though in a case where portions of $\mathcal{T}^*$ can not feasibly be cut, this choice can lead to an undesirable overemphasis on these regions. So instead, a smooth step function of the form below is chosen:

$$H_\tau(d) = \begin{cases} 3\left(\frac{d}{\tau}\right)^2 - 2\left(\frac{d}{\tau}\right)^3 & 0 \leq d < \tau \\ 1 & d \geq \tau. \end{cases} \tag{5.4}$$

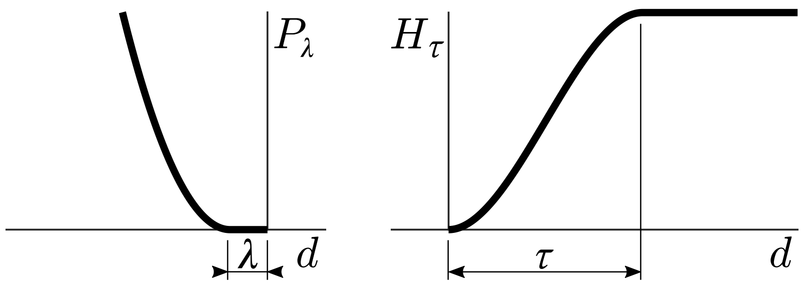

FIGURE 5.18: **Penalty functions on distance used for collision avoidance (left) and surface matching (right).**

This function acts similar to a quadratic penalty for a distance d close to zero, but smoothly transitions to a constant penalty over a transitional region of size τ (Figure 5.18). Thereby, regions that are decidedly uncuttable, i.e. with a distance larger than τ, are simply ignored.

Initialization Procedure Due to the relatively fine-scaled geometry of the toolsurface and its very low rigidity, the outlined surface matching is prone to a large number of undesirable local minima. It therefore relies on a fairly good initialization, for which the input toolpath is used. To this end, the optimization process is split into two distinct stages. During the first, the surface matching objective is not applied as the primary objective. Rather, the cut portion of the toolpath is matched to the input toolpath, with regards to the position and orientation of the tool, using a quadratic penalty. Once a toolpath is found which resembles the input path as close as possible but has a feasible trajectory, this initial objective is gradually dropped and surface matching is gradually applied instead. Using the input toolpath as initialization also establishes the desired global path layout, and experiments showed that this layout is generally well preserved during the surface matching stage, even once the initial objective is removed entirely. At the same time, matching only the toolsurface provides a larger degree of freedom for the robot trajectory, allowing the robot to gracefully avoid collisions even in challenging situations.

Physical Limits The considered constraints are the robot's limitations on joint angles, as well as collisions of the robot and the tool. These collisions are namely: 1) self-collisions of the robot, 2) collisions between the robot and the workpiece $\mathcal{W}$ and obstacles $\mathcal{O}$ 3) collisions between the toolsurface $\mathcal{S}$ and the obstacles $\mathcal{O}$ 4) collisions between $\mathcal{S}_{free}$ and the workpiece $\mathcal{W}$ 5) penetration of $\mathcal{S}_{cut}$ and $\mathcal{S}_{inter}$ into the target shape $\mathcal{T}$. For the implementation of robot collisions the robot model is equipped with a number of spherical collision primitives, typically eight per link. From each collision primitive the signed distance is computed to any of the other collision spheres, as well as to the closest point on each of the objects in the scene. The latter are accurately represented through triangle meshes. A negative sign of the distance thereby signifies penetration. Similarly, proximity of the toolsurface $\mathcal{S}$ is evaluated on a dense set of sample points on the surface, for each of which the smallest distance to the relevant objects is computed. These distances are then penalized with the one-sided quadratic function

$$P_\lambda(d) = \begin{cases} (d-\lambda)^2 & d < \lambda \\ 0 & d \geq \lambda, \end{cases} \tag{5.5}$$

where λ is a safety margin. The same type of penalty is applied directly to the joint angles of the robot. The weighted sum of all penalties constitutes the full constraint objective E_{constr}, whereby the weights are chosen large compared to any of the remaining objectives, such that the constraints are enforced rigidly.

5.6.2 Secondary Objectives

Several more criteria for the quality and practicability of a toolpath are identified, enforced through additional objectives E_{sec}.

Orthogonal tool orientation For the fabrication process, it is favorable to keep the cutting direction orthogonal to the tool plane. Though a cut can be produced when the tool plane is aligned with the cutting direction, this would produce only a narrow slit, often without fully removing a portion of clay from the workpiece. There is a high risk the clay will reattach subsequently, effectively undoing the cut. By only cutting orthogonal to the tool plane, long, narrow shavings are produced which can be removed immediately. Let $\boldsymbol{c}_{ij}$ be the sample point j of the tool of time step i. For each $\boldsymbol{c}_{ij}$, the deviation of the tool facing direction $\mathbf{u}_i$ from the local cut direction $\mathbf{v}_{ij}$ is penalized for those sample points on the tool engaged in the cutting, as

$$E_{orth}^{ij} = l_{ij}\ \sin^4(\angle(\mathbf{u}_i, \mathbf{v}_{ij}))\ H^*_{a,\tau}(d_{\mathcal{W},ij}). \tag{5.6}$$

The symbol $\angle(\cdot,\cdot)$ is the angle spanned by two vectors. The cut direction is computed as $\mathbf{v}_{ij} = 1/2\ (\hat{\mathbf{v}}_- + \hat{\mathbf{v}}_+)$, where $\mathbf{v}_- = \boldsymbol{c}_{i,j} - \boldsymbol{c}_{i-1,j}$, $\mathbf{v}_+ = \boldsymbol{c}_{i+1,j} - \boldsymbol{c}_{i,j}$, and $\hat{\cdot}$ represents a normalized vector. The associated step size $l_{ij} = 1/2\ (\|\mathbf{v}_-\| + \|\mathbf{v}_+\|)$ is used to weight the objective. And finally, the last term of the equation represents a weight in the range $[0, 1]$ indicating whether the sample point is inside or close to the workpiece $\mathcal{W}$ and therefore is relevant for the cut. Herein $H^*_{a,\tau}(d) = 1 - H_\tau(d - a)$ is an inverted smooth step function shifted by a tolerance a, and $d_{\mathcal{W}}$ is the signed distance between the sample point and $\mathcal{W}$.

Smooth discrete toolpath To ensure smoothness of the discretized toolpath, the angle spanned by the piecewise linear path of a tool sample point at each time step through is penalized:

$$E^{ij}_{smooth} = l_{ij}\ \alpha^2_{ij}\ H^*_{a,\tau}(d_{\mathcal{T},ij}), \tag{5.7}$$

where $\alpha_{ij} = \angle(\mathbf{v}_-, \mathbf{v}_+)$. This angle can essentially be viewed as the ratio between the local, approximated curvature of the toolpath (i.e. α_{ij}/l_{ij}) and the sampling density (given by $1/l_{ij}$). Thereby, the objective does allow for an arbitrarily large curvature of the path, provided that the temporal resolution is adequate locally. As above the objective is weighted with the path length l_{ij}; and also according to the closeness $d_{\mathcal{T}}$ to the target shape, such that only portions of the cut are affected which may be visible in the final object.

Limited joint angle step size While for the optimization, the toolpath is assumed to be given by linear interpolation of the tool geometry at discrete time steps, and the robot trajectory is interpolated linearly in joint angle space during fabrication. For the k^{th} joint of the robot, a step of $\beta_{i,k} = \boldsymbol{q}_{i,k} - \boldsymbol{q}_{i-1,k}$ in joint angle space induces a maximum interpolation error of

$$\epsilon_{i,j,k} = r_{i,j,k}\ \big(1 - \cos\big(\frac{\beta_{i,k}}{2}\big)\big), \tag{5.8}$$

where $r_{i,j,k}$ is the distance between a tool sample point $\boldsymbol{c}_{i,j}$ and the k^{th} robot axes. For simplicity, a rough, fixed estimate $\tilde{r}_k$ is assumed for this distance for each joint angle, and penalize the corresponding approximation error through

$$E^{i,k}_{joint} = \big(\tilde{r}_k\ \big(1 - \cos\big(\frac{\beta_{i,k}}{2}\big)\big)\big)^2. \tag{5.9}$$

Limited tool step size Collision avoidance of the toolsurface is based on a fixed number of sample points. In order to maintain an adequate sampling density it is necessary to limit the step size of the tool. Again, we simply apply a one-sided quadratic penalty

$$E^{i,j}_{step} = P_{-\delta}\big(- \|\boldsymbol{c}_{i,j} - \boldsymbol{c}_{i-1,j}\|\,\big) \tag{5.10}$$

to roughly ensure an upper bound of δ.

Quadratic regularization Finally, a weak quadratic regularization is applied to the tool step size, such that all portions of the toolpath which are not governed by any of the above objectives remain short and smooth:

$$E_{reg}^{i,j} = \|\boldsymbol{c}_{i,j} - \boldsymbol{c}_{i-1,j}\|^2 \tag{5.11}$$

5.7 Demonstrations and Results

To demonstrate the versatility of the system, four physical prototypes featuring different geometric characteristics are designed and fabricated. The decomposition of the input model by drawing strokes in the GUI and generating toolpaths for each patch takes around $0.5\,\mathrm{h}$ on average, depending on the number of patches of the decomposition and the number of attempts made to match the user's intention. The optimization takes $1\,\mathrm{h}$ to $4\,\mathrm{h}$ on average for the models presented here (torso, eye, face, 3D Möbius ring). The fabrication takes around $1\,\mathrm{h}$ on average with a joint speed of $1\,\mathrm{rad/s}$ for the leading axis (the robot is controlled through the *movej* command (Robots 2015). Each process is dependent on the number of patches of the decomposition, and the toolpaths density. After fabrication, the clay model needs around one day to dry until its surface solidifies and more than two days to be fully dried.

The optimization framework is implemented in C++, making use of the `Eigen` library (Guennebaud, Jacob, et al. 2010) for matrix algebra. Searches for closest points on surfaces, as required for collision avoidance, are performed through an axis aligned bounding box tree, using the `libigl` library (Jacobson, Panozzo, et al. 2018). This operation accounts for the largest part of the computational costs in the procedure, with roughly $50\,\%$. Another $15\,\%$ to $20\,\%$ of costs can be attributed to the forward kinematics of the robot, and the respective first- and second order derivatives. For context, it should be noted that collisions between toolsurface and target surface are tested on 140 sample points per trajectory point, and the distance function for surface matching is evaluated with similar density. Computation times for all examples are reported in Table 5.1, obtained on a standard PC with a 3.4GHz Intel Core i7-3770 CPU.

5.7.1 Fabricated Models

Beside the torso model (Figure 5.9), the simplest of the examples is the eye model (Figure 5.19), which contains concave features that are nearly impossible to generate collision-free toolpaths for. Several attempts are made through a CAD-modelling process but failed as collisions cannot be fully resolved. A smoothed version of the model is used in the end. However, the optimization component from the *RobotSculptor* resolves all the collisions and generates toolpath trajectories that achieve fabricated results in decent quality, even with a tool that is oversized for the details around the iris area.

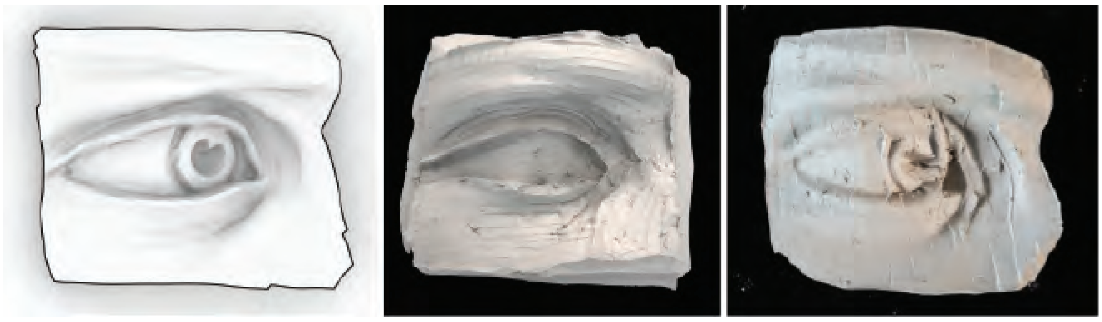

FIGURE 5.19: **The *eye* model.** Left: the input geometry; Middle: a model by executing hand-modelled toolpaths based on a CAD model; Right: a model by executing trajectories generated from *RobotSculptor* using the mesh model of the same geometry as input.

The interactive, user-guided design method is further used to decompose and generate toolpaths for a face model that contains more challenging geometric features around the eye area (concave feature with large curvature) and the nose area (sharp edges). Similarly, CAD-modelled toolpaths fail to resolve collisions around the eye corner, but the *RobotSculptor* system successfully fabricates the different styles that are desired (Figure 5.20).

The *RobotSculptor* system even allows the use of different parts of the tool for the sculpting

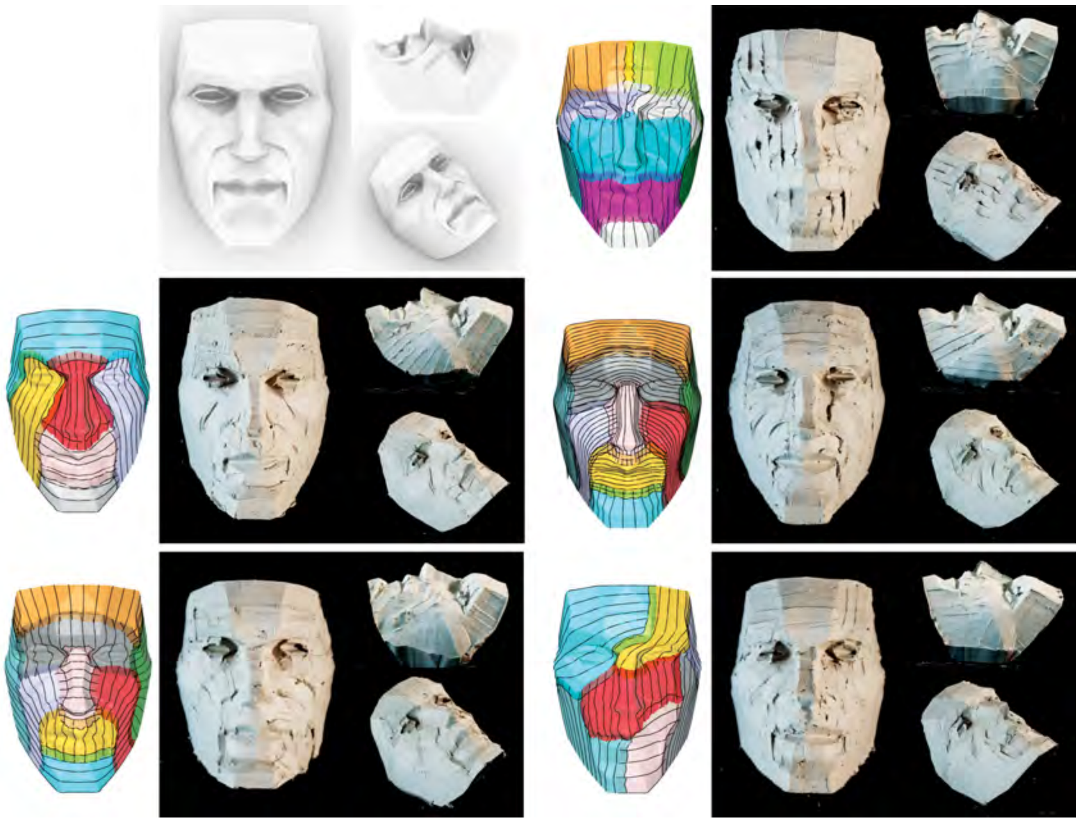

FIGURE 5.20: **Sculpting results of the *face* model.** Top-left: the input geometry; Rest: initialized toolpaths and the results with different styles by executing robot trajectories generated from *RobotSculptor*.

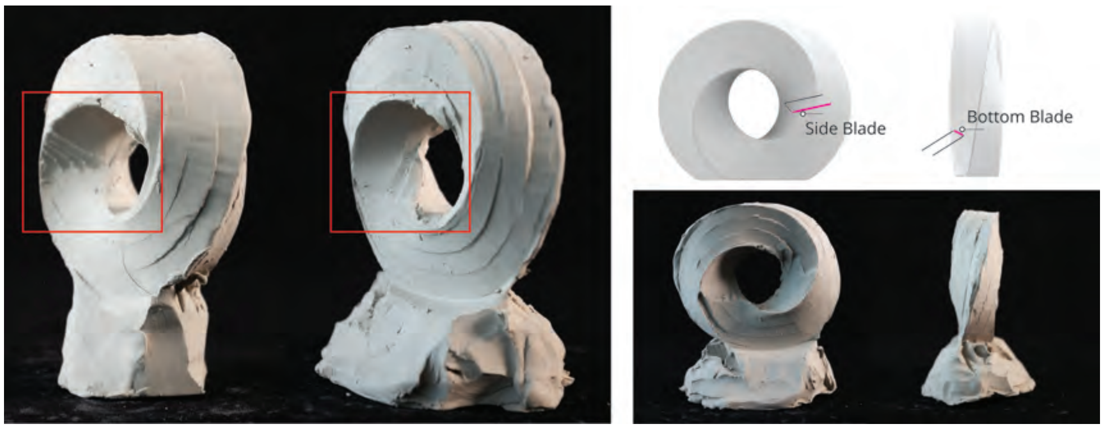

FIGURE 5.21: **The 3D Möbius model.** Left: Reachability limitation from inadequate *side blade* length. Right: Illustration of model areas cut by *side blade* or *bottom blade*.

process. In the 3D Möbius ring example[5] (Figure 5.21-right), the bottom blade is used to sculpt the outer patches, and the side blade is used to sculpt the inner patches which are inaccessible to the bottom blade due to collision issues. However, two limitations are noticed: 1) Models with a thin connection to the base are likely to be deformed during the fabrication, resulting in lower precision results. In this example, it is compensated by manually supporting the model. 2) Sculpting with the *side blade*, the maximum cut depth cannot exceed the length of the tool. This becomes a limiting factor when cutting the innermost portion of the ring and constraints the allowable size of the model.

Preference for using a specific edge of the tool can be set by simply choosing the appropriate tool-local frame used for the initialization phase. The subsequent toolpath optimization on the other hand is agnostic to the notion of distinct blades. That is, it treats the entire tool as one blade. Similarly to the overall path layout, experiments show that any preferences implied by the initialization are typically well preserved.

As shown in Figure 5.21-left, the reachability limitation is verified by fabricating two Möbius models with different thickness.

TABLE 5.1: Statistics of presented examples.

model	# patches	avg traj. pts/patch	opt. time	fab. time
Torso	5	334	1h 57m	7m
Face	6	445	4h 11m	26m
	7	473	4h 30m	31m
	9	412	5h 33m	33m
Eye	3	624	1h 21m	16m
Möbius	8	339	1h 39m	31m

[5]For this specific model, robot reachability and robot-target collisions are the main challenge. Since the model itself contains clear edges for cutting the model into sub-patches, a simple decomposition by hand is employed instead of using the decomposition tool for the decomposition process.

5.8 Discussion

The project in this chapter presents an interactive design and fabrication system that allows the user to design different styles for sculpting clay models with a 6-axis robot. A set of key parameters is identified and extracted from a series of sculpting experiments and then exposed to the users in an interactive GUI developed through the project Figure 5.14. The interface allows the user to decompose a input mesh into desired patches by drawing free sketch strokes and embed his/her design expressions as different sculpting styles individually by generating a set of corresponding initial sculpting toolpaths. Additionally, the developed GUI provides intuitive and instant feedback to the user, allowing a seamless design process for the sculpting styles on an arbitrary mesh model.

After the toolpaths are initialized, the developed system conducts optimal path planning to resolve robot collision and reachability issues while still maintaining a maximum match to the given input surface. To obtain a higher success rate, an Arduino-controlled turntable is also developed and integrated into the optimization pipeline. The capacity of the system is demonstrated through a set of fabricated desktop-size clay models: torso, eye, face, and 3D Möbius ring. Additionally, as evidenced by the wide variety of styles for the same model, the system successfully enlarges the magnitude of expression incorporation during the design stage.

It is the combination of the initialization and optimization components that makes the *RobotSculptor* system not only a robotic extension of the human hand, but a system with certain intelligence that fulfills certain design intentions. Yet there is still a long way to go to merge the system seamlessly in the human endeavours of design and creation, and some limitations are identified.

First, the developed system utilizes a subtractive strategy for the sculpting process that assumes the clay to be rigid. This assumption works well when sculpting thick areas, but may cause imprecise results due to material deformation at thin sculpted areas (*e.g.* the nose area in the face model).

Second, limited by both material properties and the optimization framework, this project only touches the sculpting styles in the manner of stroke density and directions for selected sub-areas, while many other possibilities exist for “artistic expressions”.

Third, the current system can only predict the sculpted geometry after running the toolpath optimization. As the optimization process is computationally demanding, the current pipeline cannot present a predicted representation of the final appearance to the users instantly.

Fourth, compared to human artists who conduct a combination of various modelling techniques during an entire sculpting process (additive, subtractive, formative), the system only utilizes the subtractive process, using one type of tools. While combining both additive and subtractive techniques in the fabrication process is not difficult, predicting the material behaviour under formative processes (modelling, pushing) to fulfill the optimization tasks will require a simulation component, additional to the need mentioned in the first point above.

Fifth, this project developed an interactive GUI for non-expert users to design sculpting styles for the clay material. However, similar robotic processes that involve interaction with geometries have more applications, such as foam wire cutting and wax cutting, where the user group may also extend to experts. Instead of asking these users to incorporate the new tools into their work process, parallel alternatives for doing the same task could be provided to gain better popularity. As the project decoupled the *Decomposition & Initialization* and the *Optimization* components, migrating the former into existing CAD software (for instance, *Rhinoceros*) as plugins and keeping the optimization (currently written in C++) as it is for computation efficiency is feasible. Possible future work is discussed in Section 6.3.4.

5.9 Contributions

Due to the interdisciplinary nature of this dissertation Section 1.3.2, this project was conducted under the collaboration between Gramazio Kohler Research (GKR) and Computational Robotics

Lab (CRL) under the supervision of Prof. Dr. Stelian Coros, Prof. Matthias Kohler and Prof. Fabio Gramazio, with additional support from Disney Research (DR).

Section 5.6 is mainly based on the work of Simon Duenser (CRL), the second author of the published paper Ma, Duenser, et al. (2020) related to this chapter. Dr. Espen Knoop (DR) built the turntable and manufactured the end effector used in this project.

Dr. Christian Schumacher (DR), Dr. Romana Rust (GKR) and Dr. Moritz Bächer (DR) were involved throughout the project and provided valuable feedback during the development.

Chapter 6

Conclusion

To build a future, you have to know the past.

— Otto Frank

6.1 Summary

In this dissertation, it has been argued that optimization-based approaches can facilitate the robotic fabrication processes of complex tasks that are difficult and constrained to conduct in the current architectural research domain. Based on the realization of the two complementary projects embedded in this specific context, it has been demonstrated that with careful abstraction of the physical world into mathematical formulations, as well as the identification of the constraints in either local or global level, computational optimizations are able to resolve complex robotic fabrication problems with various types of constraints simultaneously and in a integrated way.

The challenges emerging from the development of the projects identify several key areas for investigation: geometric processing and modelling, material simulation for robotic fabrication, decomposition and reassembly for fabrication across scales, and design-to-fabrication pipelines. Addressing these aspects, this dissertation proposes solutions for these topics in the two specific projects, as well as the generalization of some solutions for similar tasks.

This chapter summarizes the key contributions and findings arising from both projects and provides cross-topic conclusions and guidelines for future work.

6.2 Contributions

6.2.1 Optimization-Enhanced Integrated Design and Robotic Fabrication Process

For both of *Project I: FrameForm* and *Project II: RobotSculptor*, a complete pipeline consisting of an integrated design to robotic fabrication process was developed, helping to fulfil the successful fabrication with designate fabrication technologies. With optimization as an essential component, the systems in both projects were able to resolve constraints that emerged from a fabrication process in an integrated way.

For *Project I*, the developed system supported the construction of metal frames from input shapes of moderate curvature and complexity. The design component took the fabrication technology embedded in the robot end effector into consideration, and created designs considering the topology constraints of the frame geometries. The optimization component then managed to resolve the topology constraints simultaneously, as well as others extracted from structural properties, material properties, etc., to produce results that can be used by the robot for successful fabrication tasks. The results are usually not easy to obtain without the optimization component.

As for *Project II*, the system supported the robotic sculpting of clay material with user-customized styles. The design component in this project allowed the user to design "sculpting styles" for different areas of a given model, while the optimization component planned the robot trajectories by simultaneously maintaining the design of the user and resolving constraints emerged from the fabrication process.

While the optimization components are tailored for specific purposes in each of the two projects, the conceptual framework can be applied universally. By formulating complex fabrication tasks into optimization problems in a mathematical way, the optimization-based approaches offer great potential to increase the geometric complexity of the fabricated objects and the number or variety of constraint types that a common robotic fabrication system can manage, and can usually resolve these issues in a more systematic and integrated manner.

6.2.2 Divide-and-Conquer Strategy to Increase Design Diversity

For both of the projects developed in this dissertation, the *divide-and-conquer* strategy was employed in the design stages for different purposes.

Since the final frame structure in *Project I* is usually larger than the reachability of the fabrication system, the introduction of the *divide-and-conquer* strategy assists to decompose the overall geometry into smaller fabricable pieces, to design, optimize and fabricate each of the pieces independently with the consideration of the joining area. The fabricated pieces are then combined after the fabrication to form the global structure completely. Additionally, the *divide-and-conquer* strategy also decreases the time cost for the optimization stage, as the relationship between the objectives is nonlinear to the number of variables.

For *Project II*, on the other hand, though the employment of the *design-and-conquer* strategy increased the reachability of the fabrication system to some extent (by allowing the robot to fabricate each sub-piece in an individual turntable position), the main goal for introducing this approach is to increase the variety of *styles* that a user can design, thus to increase the expressiveness of the overall sculpture.

Again, while the *divide-and-conquer* strategy is employed differently in each of the two projects, the conceptual framework applies to general design tasks in a similar manner—it allows the treatment to the sub-elements of an overall object to be conducted separately, and thus provides design diversity at a global scale.

6.2.3 Geometry Processing and Data Transformation

As mentioned in Section 2.1, the two projects developed in this dissertation intensively use geometry processing methods from the CG field that apply on triangle meshes. While designers and researchers in architecture rarely use the triangle-based mesh representation, the work of this dissertation provides interoperability between the two fields.

Project I is mainly developed in the CG context, except for the design of the initial geometry, which is conducted in the architectural CAD software *Rhino*. The geometry transfer was one-directional as it is exported to the main system using triangle mesh as the intermediate data format.

Project II, on the other hand, employs a bi-directional transformation of the geometric data. Besides using triangle mesh data as the intermediate data format, the project also develops a set of *Grasshopper* plugin components that manages to call functions and algorithms written in C++ code in the *C#* environment. These components can be used together with other *Grasshopper* plugins for geometry processing and help to transfer data between the two systems through the use of *WebSockets*. A more detailed description of the implementation can be found in AppendixB.2 and AppendixB.1.

While the main development of geometry processing happened in the CG field, tools and libraries for these tasks are mostly written in C++ for performance reasons and disciplinary conventions. As the language is not user-friendly for people without sufficient skills, many of them are not exposed to researchers outside the CG field, and re-implementing the corresponding algorithms in other languages like *Python* brings additional cost with reduced performance. The tools for geometry processing and data transformation developed in this dissertation, though initialized for the author's own use to enjoy the high performance of geometry processing algorithms written in C++, pose promising direction to allow *Grasshopper* users to access the vast amount of geometry processing algorithms available in the CG field with ease.

6.3 Future Work

Optimization-based methods for robotic fabrication with multi-axis robot arms offer a wealth of possibilities for future work. While this dissertation presented two projects with different characteristics in the hardware setup and of different scale in the final results, only a small portion of the possibilities that optimization-based methods may assist has been touched, and more are likely to explore in the future. Many exciting questions are still left open for future work.

6.3.1 Simulation for the Robotic Fabrication

To formulate an optimization process that aims to resolve a robotic fabrication problem, the simulation of the fabrication process is necessary. This includes the simulation of both the fabrication process and the prediction of the fabricated artefacts during fabrication. Depending on the material that the robot interacts with, true material simulation may sometimes be impossible due to the computation cost, and a reduced macro model has to be used as a compensation between precision and speed during an optimization process.

As *Project I* uses steel bars as the target material, whose material model is well established, a precise simulation in the project's use case is possible and thus implemented, which supports the precise prediction of the material behaviour for the target object during the optimization process.

However, a precise simulation of the clay used *Project II* is not worthwhile due to the computation cost for the complex material property. The project thus employs a geometric-based approach (geometric Boolean operations) to simulate the fabricated artefacts during the optimization process without considering the true material behaviour. Though this decision serves the main purpose of the project, investigations on methods that incorporate true material simulation during the optimization with acceptable computation cost are still meaningful for better prediction of the optimization results. It would contribute to controlling small portions of leftover materials caused by the material deformation, and the visual intensity of the sculpting styles.

Furthermore, While the choice of the simulation method for fabrication process is project-dependent, material simulation for material operations with large deformation (non-linear material) or formative material (*e. g.* clay) is always worth exploring for fabrication simulation. As long as the computation cost is acceptable, these improvements will always contribute to the quality and precision of the optimization results.

6.3.2 Optimization Facilitated Design and Robotic Fabrication

Since this dissertation intensively employs optimization approaches to facilitate the design-to-fabrication process, one of the global goals is to consider the robotic fabrication already in the design stage rather than just before production.

Through the two projects, it is shown that the combination of initialization and optimization establishes a general conceptual method that facilitates the convergence of an initial design to a similar, but fabricable design. To be specific, *Project I* allows the user to generate frame structures with choices of density, decomposition, etc. and *Project II* allows the user to design specific sculpting styles over a target model. None of the initial designs are guaranteed to be fabricable, until the optimization components resolve the constraints.

Future work shall stress both parts, by providing more freedom on the design side and improving the ability to handle more complex constraints on the optimization side. For *Project I*, the research only avoided local robot-structure collisions and assumed the input to be self-intersection free. Further improvements could extend the current technique to handle global collisions. For *Project II*, the research only touches the sculpting styles in the manner of stroke density and directions for selected sub-areas, while many other possibilities exist for surface treatment to demonstrate the artistic expression. Further developments could focus on providing different initialization strategies to enlarge our toolpath generation library to support more styles, and enrich the optimization component accordingly.

In a more general context, the author expects a picture that paints, in the not-too-distant future, an ideal robotic system not only as a robotic extension of human hands, but also a system with intelligence that helps to fulfil design intentions and overcomes design limitations.

6.3.3 Optimization Algorithm for Design-Oriented Tasks

Due to the specific character of the problem defined in the two projects, only continuous-based methods are explored in this thesis. However, a larger body of work using sampling-based methods exists for motion planning (Şucan, Moll, and Kavraki 2012). In many circumstances, especially with large-scale data and uneven data space where many local minima occur, sampling-based methods may provide better results in even shorter time. Research on the employment of optimization algorithms for design-oriented targets is a direction with high potentials for design explorations and uncertainty-based solution findings.

Though not always achievable, instant or near-instant feedback from the optimization is always favourable for research exploration as it will greatly increase the iterative design-optimization cycle. However, it depends largely on the problem scale, optimization type, and other mathematical characteristics and may not be possible by performance increase. Nonetheless, faster optimization should always be remembered when developing optimization-based systems.

6.3.4 Design Interactivity in Optimization-based Design Processes

While optimization-based approaches truly improve the range of tasks that robotic fabrication can conduct, they also break the desired interaction with instant feedback due to the computation cost. For designers, instant feedback of the designed results weighs very high in their design process. Though achieving interactivity and precision simultaneously for any project might not be possible, specific projects may prefer different solutions depending on the weights of the interactivity and precision.

For *Project I*, the interactivity happens mostly in the design stage for decomposition and initialization. The optimization produces results that are close to the original geometry in shape, and topologically different to the initialized frame structures. While the optimization takes long, the initialized frame structure already depicts the final results relatively close. Thus, the interactivity is preserved well for the design process in this project, as the instant feedback is provided through the whole design process and the intermediate results before feeding into the optimization are visually close to the final results coming out of the optimization process except for the structural topology[1].

Unlike *Project I* where structural stability serves as the main objective, *Project II* seeks for sculpting styles, a visual feature that depends on both the material behaviour and the

[1]Structural topology has to be obtained from the optimization

way robot sculpts. Due to the method that the robot-material interaction is modelled, the instant feedback of the sculpted results is not provided. Though the developed GUI provides interactivity through the design stage of the sculpting styles for non-expert users, the missing link between the designed styles and a visualization of the fabricated result is still unfortunate. Future work shall look for solutions that can approximate the sculpting results with instant feedback. Further investigation of replacing the modelling of the robot by direct modelling of Boolean operations between the path geometry and the target geometry is necessary.

Additionally, as mentioned in Section 5.8, plugins for other CAD software as parallel alternatives of the independent GUI can be provided in a general context. This will help newly developed fabrication processes to gain better popularity in existing design fields.

Appendices

Appendix A

Tool Iteration for Project II

A.1 Design Iteration for the Customized Loop Tool

A conventional *loop tool* (Figure 5.3) for sculpting clay consists of a handle and a planar "loop", which is bent from a piece of steel wire or from a piece of thin, narrow metal strip into rectangular, triangular, or circular profiles to fulfill different cutting needs (size, angle, texture effects, level of detail, etc.). While the sculptor uses his/her hands or the plate tools for the modelling (additive) and formative process, such loop tools are usually used for the subtractive process—cutting a strip of clay off by moving the tool along a desired direction.

Thus, the design of the robot end effector initiates from a similar concept, as shown in Figure A.1.

The designed tool is composed of two main functional parts: a metal handle with a 3D-printed plastic base that helps to connect to the UR5, and a replaceable wire "loop", made of steel, that can be attached to the end of the metal handle. Thus, a sculpting process can be conducted by controlling the robot arm to manoeuvre the loop follow desired toolpaths. The metal handle is composed of a standard aluminium 40 mm T-slot profile and two customised aluminium plates, connected with *BN-610 M2* hex socket head cap screws. By changing the wire loops, fabrication experiments of various scales can be conducted with the same tool set.

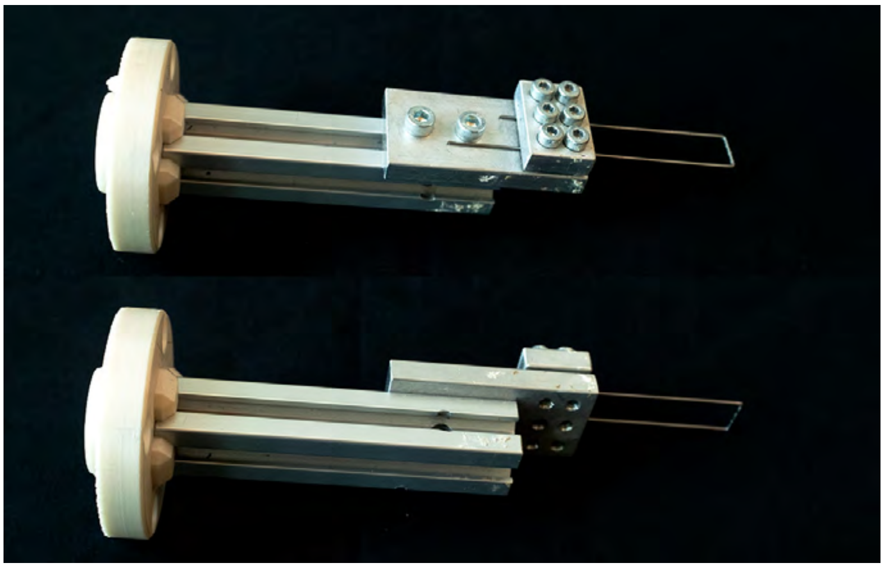

Figure A.1: **The initial design of the loop tool.**

Figure A.2 shows various designs of the replaceable "loops". To remove the inaccuracy caused by manual bending process and material elasticity, this project employs the *D.I.Wire Pro* (Pensa Labs n.d.) desktop bender for fabricating complex-shaped 2D steel wires (Figure A.2). The programmable bender allows customized bending commands in G-code, and can fabricate physical wire pieces that match the digital inputs after calibration due to the material elasticity.

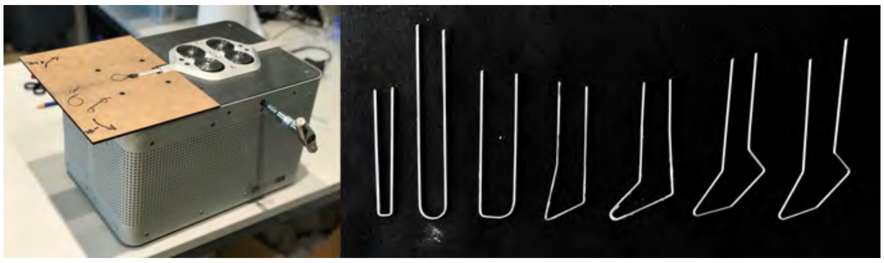

Figure A.2: **The *D.I.Wire Pro* desktop bender and the bent loops with different profiles.**

Handle Update While the designed tool fulfils some of the needs as expected during the conducted experiments described in Section 5.4.1, collisions between the handle and the clay material are often observed. Though the shape of the target geometry is one of the causes, the relative size of the metal handle to the wire is also one important factor. Shown in Figure A.3, the extra part of the handle outside the extension of the wire loop's profile often prevents the tool from cutting in an inclined position.

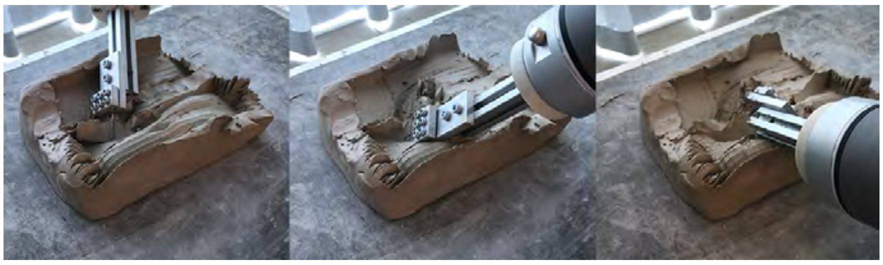
FIGURE A.3: **Tool-target collisions observed in various sculpting paths.**

Hence, an upgrade of the metal handle was conducted by replacing the original T-slot metal handle with a slimmer metal piece. The dimension of the square cross-sectional area was reduced from 40 mm to 10 mm. The wires were adjusted accordingly, guaranteeing the profile of the connecting section is aligned with the side face of the handle. The plastic base that connects the metal handle and the robot arm was also updated accordingly, adding positions for an additional screw so as to get better stability. The new tool is shown in Figure A.4.

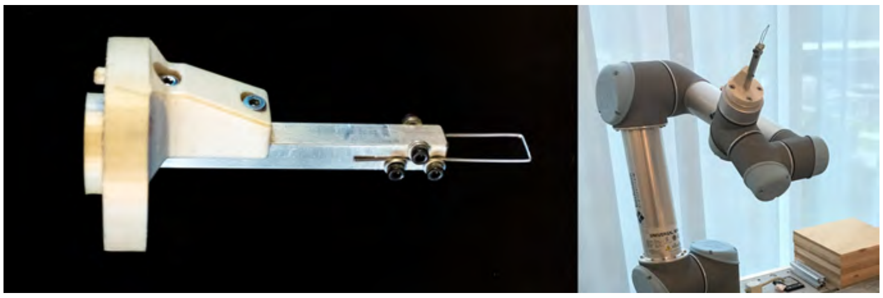
FIGURE A.4: **The upgraded version of the customized loop tool.**

A.1.1 Design Iterations for the Turntable

Prototype Design Two initial prototypes were quickly built and iterated with standard aluminium T-slot profiles and laser cut 2 mm MDF boards. The first prototype (Figure A.5) contains no rotating parts and requires manual alignments to the additional reference (the temporary plywood) to allow four 90° positions.

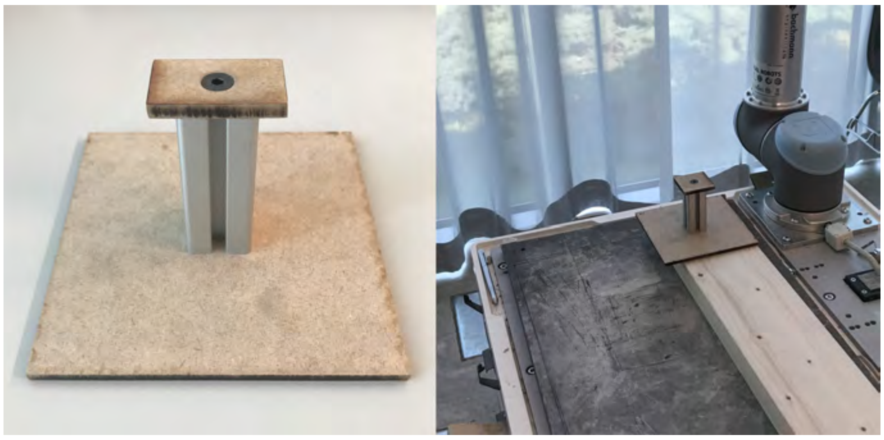

FIGURE A.5: **The first iteration of the turntable prototype.**

A 3D model which was unable to fabricate before turned fabricable, was fabricated as a proof of concept under the this setup (Figure A.6). During this fabrication task, the model was 4 times rotated by 90° for the corresponding robotic toolpaths.

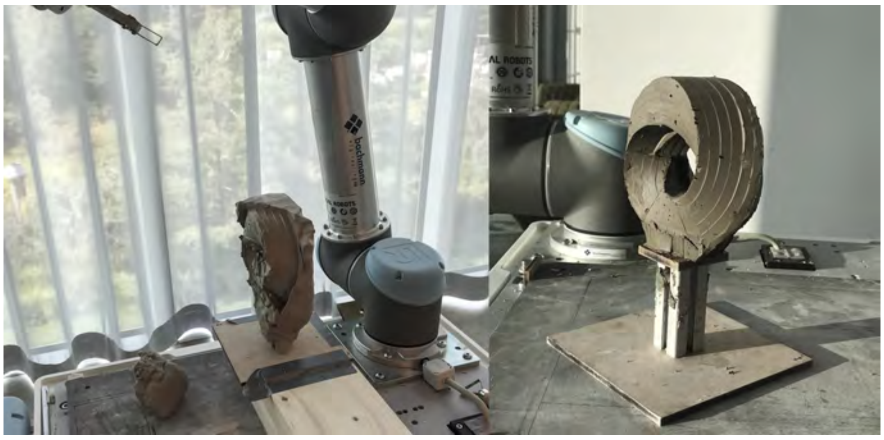

FIGURE A.6: **The proof-of-concept model fabricated with the turntable prototype.**

As the fabricated results showed successful improvements of the accessible workspace of the robotic system, the current prototype of the turntable was far from a "turnable table" and required accurate manual alignment during the fabrication process. This often results in surface imperfection at the connecting seams of two sub-areas on the same surface fabricated from two turntable positions. Thus, an improved version of the turntable was then built with a turnable component (Figure A.7) that eliminates the required tedious alignment procedure. While this prototype still allows only four different turntable positions, it paves the road to the final design of the turntable.

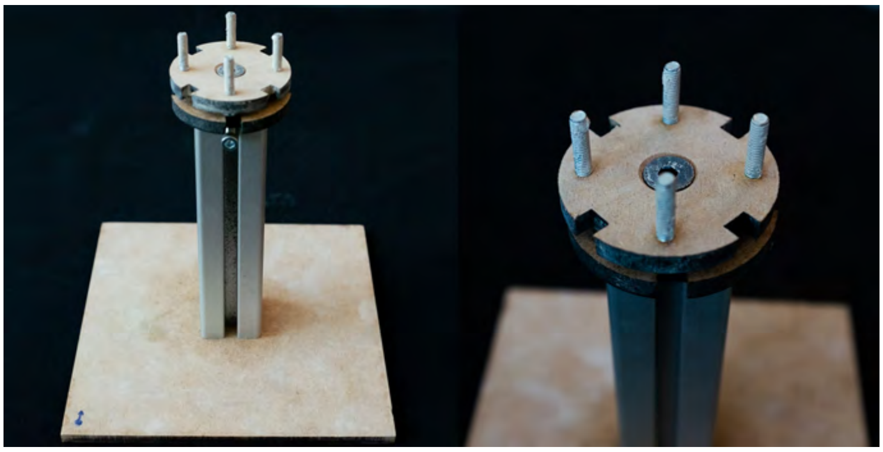

FIGURE A.7: **The second iteration of the turntable prototype.**

Enhanced Design A customized turntable controlled by an *Arduino Uno* was built after the prototyping stage. With the help of the gear system, the motor can now rotate the table in both directions with 1.8° resolution, and acts as the 7th axis of the robot during optimization and fabrication phase (Figure A.8). Notice that the initial version of this Arduino-controlled turntable uses a thin shaft and causes undesired vibration when the sculpting tool interacts with the model. An upgraded version with thicker support is then developed, helping to keep the model stable during sculpting and rotating stages.

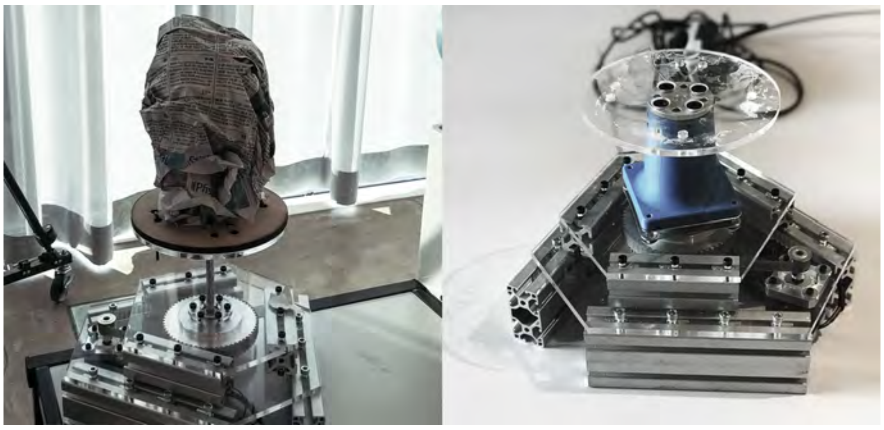

FIGURE A.8: **Two iterations of the Arduino-controlled turntable design.** The initial design (left) uses a thin aluminium shaft to support the table plate and causes undesired vibration during the sculpting process. The final design (right) employs a thicker 3D printed plastic support to stabilize the whole table.

Appendix B

Technical Details for Programming Implementation

B.1 WebSocket Communication between Grasshopper and C++ Environment

Due to the compilation nature of the C++ language, using native C++ libraries for geometry processing breaks the instant feedback that designers usually prefer. Additionally, using these libraries directly with any process of geometry modelling in CAD software like *Rhino* is almost impossible. Thus, the author developed a communication approach that uses *Websocket* to open a two-way communication channel and integrate the algorithms available in the *libigl* (a library written in C++) into the geometry processing pipeline the *Grasshopper* platform inside *Rhino*.

As the connection established by the *Websocket* approach requires a *server* side and a *client* side, this approach will run an online program written in C++ in the background as the *server* side. The program keeps listening to a shared port for incoming data. Once the geometry data comes, the program will execute the function and send the processed data back to the source (*client* side).

On the *Grasshopper* (*client*) side, geometry data can be sent through the shared port using any plugin that has the *Websocket* communication functionality. The processed data is collected from the *server* side for visualization and post-processing. Data transmission is conducted through a common *json* format.

A diagram of the processing pipeline is given below:

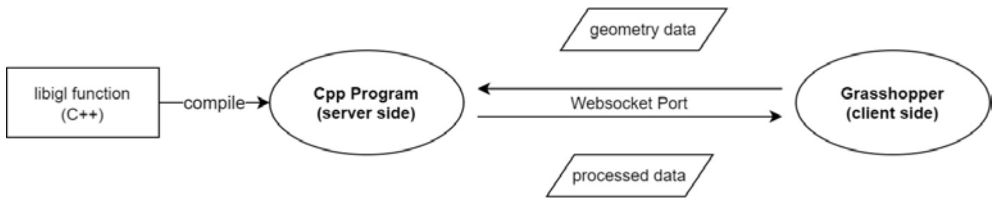

FIGURE B.1: **The pipeline of integrating C++ code *Grasshopper* platform by using *Websocket* communication.**

The *Websocket* approach (Figure B.2) provides a fast and relatively easy way to call C++ function for geometry processing. However, the framework requires the researcher to develop ad hoc codes during the development, and the external program needs to be running constantly in the background. The program sometimes needs to be restarted depending on the size of the data transmitted through the port, and processing time.

From the author's experience, though this approach provides an interactive geometry processing pipeline in the *Grasshopper* platform after the C++ program is running, it suits better for ad hoc use and testing of the C++ functions for research purposes, and is not a stable solution for permanent use.

B.2 Grasshopper Components for Geometry Processing

Unlike the approach described in Section B.1, the approach described in this section provides an independent solution of using native C++ files inside the *Grasshopper* platform.

Grasshopper components are written in *C#* language, with APIs provided by the McNeel company. As code written in *C#* are "managed code", and code written in C++ are "unmanaged

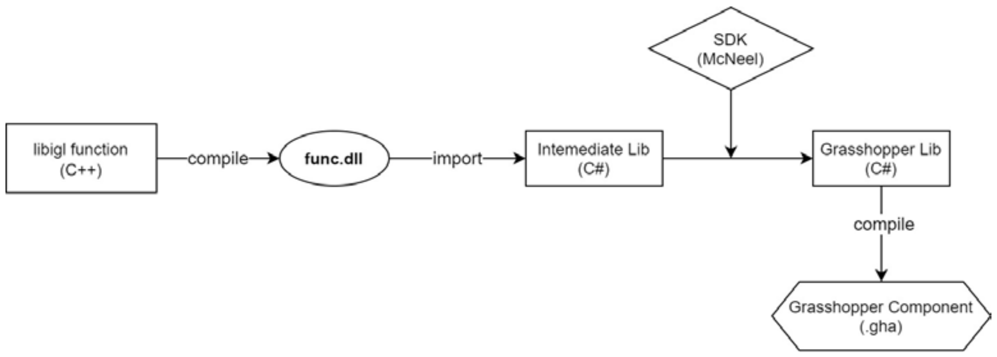

FIGURE B.2: **The pipeline of integrating C++ code *Grasshopper* platform by writing *Grasshopper* component.**

code"[1], additional steps are required to call the function written in C++ in a *C#* environment, so that the compiled *Grasshopper* component can provide the same functionality as the original C++ function.

As shown in Figure B.2, a C++ function is first compiled into a "dll" (Dynamic-link library) file, which can be imported into a *C#*, an exemplary code snippet is shown below:

LISTING B.1: DllImport Example

```
public static class CallingCppFuncInCSharp
{
    private const string DllFilePath = @"func.dll";

    [DllImport(DllFilePath , CallingConvention = CallingConvention.Cdecl)]
    private extern static int cSharpFunc(Type x);

    public static int cSharpFunc(Type x)
    {
        return iglFunc(x);
    }
}
```

[1]For difference between "managed code" and "unmanaged code", please refer to the Microsoft document for details. It is not the purpose for this appendix to provide explanation to these technical concepts in the computer science fields.

Once the code can be called in a *C#* environment, a *Grasshopper* Library Program can be created using the Software Development Kit provided by the McNeel company[2]. Once the conversion pipeline of different data types among different languages is completed, a *Grasshopper* plugin can be compiled, and loaded into the *Grasshopper* platform.

Compared to the *Websocket* approach, this approach requires more effort for the implementation, but provides a native and smooth way of using C++ functions in the *Grasshopper* platform, and requires no other programming running in the background.

From the author's experience, this approach suits better for mature geometry processing pipelines and can be combined with the *Websocket* approach so that a C++ function is first tested through the *Websocket* approach, and then implemented into a *Grasshopper* component solution for permanent use. Once a considerate amount of functions is implemented, the component set can even be published as an independent *Grasshopper* library.

B.3 Key C++ Library Used in the Dissertation

Below is a table for the key C++ libraries used in the development of the two complimentary projects in this dissertation.

[2]The company that made *Rhino* and *Grasshopper*.

TABLE B.1: Main C++ libraries used in the Dissertation

libigl	A simple C++ geometry processing library.
Eigen	A computer programming library for matrix and linear algebra operations.
Qt5	A free, cross-platform and open-source widget toolkit for creating graphical user interfaces.
WebSocket++	A header only C++ library that implements RFC6455 The WebSocket Protocol.
Dear ImGui	A bloat-free graphical user interface library for C++.
spdlog	A very fast, header-only/compiled, C++ logging library.
nlohmann-json	JSON for Modern C++.
suitesparse	A suite for sparse matrix algebra library.
Artelys Knitro	An library solving large scale nonlinear mathematical optimization problems.

Glossary

ADRL Agile and Dexterous Robotics Lab

AEC Architecture, Engineering and Construction

BFGS Broyden–Fletcher–Goldfarb–Shanno

BREP Boundary Representation

C++ the C++ programming language

CAD Computer-Aided Design

CAM Computer-Aided Manufacturing

CG Computer Graphics

CRL Computational Robotics Lab

CSG Constructive Solid Geometry

DOF degrees of freedom

GKR Gramazio Kohler Research

GUI Graphic User Interface

ICD Institute for Computational Design and Construction

NURBS Non-Uniform Rational B-Splines

UR5 Universal Robot with 5kg payload

List of Figures

Bibliography

Achtziger, W. (1999). "Local Stability of Trusses in the Context of Topology Optimization Part I: Exact Modelling". In: *Structural Optimization* **17** 4, pp. 235–246. DOI: 10.1007/BF01206999.

Achtziger, W. (2007). "On Simultaneous Optimization of Truss Geometry and Topology". In: *Structural and Multidisciplinary Optimization* **33** 4-5, pp. 285–304. DOI: 10.1007/s00158-006-0092-0.

Alexa, M. and M. Wardetzky (2011). "Discrete Laplacians on General Polygonal Meshes". In: *ACM SIGGRAPH 2011 Papers on - SIGGRAPH '11*, p. 1. DOI: 10.1145/1964921.1964997.

Andersen, T. (2015). *Optimizing the Universal Robots ROS driver.* English. Technical University of Denmark, Department of Electrical Engineering.

Aste, N., M. Manfren, and G. Marenzi (2017). "Building Automation and Control Systems and Performance Optimization: A Framework for Analysis". en. In: *Renewable and Sustainable Energy Reviews* **75**, pp. 313–330. DOI: 10/ghc83j.

Austern, G., I. G. Capeluto, and Y. J. Grobman (2018). "Rationalization Methods in Computer Aided Fabrication: A Critical Review". en. In: *Automation in Construction* **90**, pp. 281–293. DOI: 10.1016/j.autcon.2017.12.027.

Bechthold, M. (2016). “Ceramic Prototypes – Design, Computation, and Digital Fabrication”. en. In: *Informes de la Construcción* **68** 544, p. 167. DOI: 10.3989/ic.15.170.m15.

Beliaev, M. et al. (2016). “Simulation of pulling the reinforcing bar from concrete block with account of friction and concrete damage”. In: *MATEC Web of Conferences*. Vol. 73. EDP Sciences, p. 04010.

Bermano, A. H., T. Funkhouser, and S. Rusinkiewicz (2017). “State of the Art in Methods and Representations for Fabrication-Aware Design”. In: *Computer Graphics Forum* **36** 2, pp. 509–535. DOI: 10.1111/cgf.13146.

Bouaziz, S. et al. (2014). “Projective Dynamics: Fusing Constraint Projections for Fast Simulation”. In: *ACM Transactions on Graphics* **33** 4, 154:1–154:11. DOI: 10.1145/2601097.2601116.

Brugnaro, G. B. (2016). “Robotic Softness: An Adaptive Robotic Fabrication Process for Woven Structures”. In: *Proceedings of the 36th Annual Conference of the Association for Computer Aided Design in Architecture, Ann Arbor 27-29 October, 2016, Pp. 154-163*. CUMINCAD.

Bunge, A. et al. (2020). “Polygon Laplacian Made Simple”. en. In: *Computer Graphics Forum* **39** 2, pp. 303–313. DOI: 10.1111/cgf.13931.

Byrd, R. H., J. Nocedal, and R. A. Waltz (2006). “KNITRO: An integrated package for nonlinear optimization”. In: *Large Scale Nonlinear Optimization*. Springer Verlag, pp. 35–59.

Chen, X. et al. (2014). “An Asymptotic Numerical Method for Inverse Elastic Shape Design”. In: *ACM Transactions on Graphics* **33** 4, pp. 1–11. DOI: 10.1145/2601097.2601189.

Chiou, C.-J. and Y.-S. Lee (2002). “A Machining Potential Field Approach to Tool Path Generation for Multi-Axis Sculptured Surface Machining”. In: *Computer-Aided Design* **34** 5, pp. 357–371. DOI: 10.1016/S0010-4485(01)00102-6.

Chiriatti, L. et al. (2019). “A study of bond between steel rebar and concrete under a friction-based approach”. In: *Cement and Concrete Research* **120**, pp. 132–141.

Cortsen, J. et al. (2014). “Automated Fabrication of Double Curved Reinforcement Structures for Unique Concrete Buildings”. en. In: *Robotics and Autonomous Systems* **62** 10, pp. 1387–1397. DOI: 10/f6gf8j.

Cosenza, E., G. Manfredi, and R. Realfonzo (1997). “Behavior and modeling of bond of FRP rebars to concrete”. In: *Journal of composites for construction* **1** 2, pp. 40–51.

Crane, K., M. Desbrun, and P. Schröder (2010). “Trivial Connections on Discrete Surfaces”. en. In: *Computer Graphics Forum* **29** 5, pp. 1525–1533. DOI: 10.1111/j.1467-8659.2010.01761.x.

Crane, K., C. Weischedel, and M. Wardetzky (2013). “Geodesics in Heat: A New Approach to Computing Distance Based on Heat Flow”. In: *ACM Transactions on Graphics* **32** 5, 152:1–152:11. DOI: 10.1145/2516971.2516977.

de Goes, F., M. Desbrun, and Y. Tong (2016). “Vector Field Processing on Triangle Meshes”. In: *ACM SIGGRAPH 2016 Courses*. SIGGRAPH ’16. New York, NY, USA: Association for Computing Machinery, pp. 1–49. DOI: 10.1145/2897826.2927303.

De Maeyer, J., B. Moyaers, and E. Demeester (2017). “Cartesian path planning for arc welding robots: Evaluation of the descartes algorithm”. In: *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–8.

Dhokia, V. G. et al. (2011). “A Process Control System for Cryogenic CNC Elastomer Machining”. en. In: *Robotics and Computer-Integrated Manufacturing*. Conference Papers of Flexible Automation and Intelligent Manufacturing **27** 4, pp. 779–784. DOI: 10.1016/j.rcim.2011.02.006.

Dörfler, K. et al. (2016). “Mobile Robotic Brickwork”. en. In: ed. by D. Reinhardt, R. Saunders, and J. Burry, pp. 204–217. DOI: 10.1007/978-3-319-26378-6_15.

Dragomatz, D. and S. Mann (1997). “A Classified Bibliography of Literature on NC Milling Path Generation”. In: *Computer-Aided Design* **29** 3, pp. 239–247. DOI: 10.1016/S0010-4485(96)00060-7.

Duenser, S. et al. (2020). “RoboCut: Hot-Wire Cutting with Robot-Controlled Flexible Rods”. In: *ACM Transactions on Graphics* **39** 4, 98:98:1–98:98:15. DOI: 10.1145/3386569.3392465.

Eckert, C. and M. Stacey (2014). “Constraints and conditions: drivers for design processes”. In: *An anthology of theories and models of design*. Springer, pp. 395–415.

Edwards, S. and C. Lewis (2012). “Ros-industrial: applying the robot operating system (ros) to industrial applications”. In: *IEEE Int. Conference on Robotics and Automation, ECHORD Workshop*.

Elber, G. and E. Cohen (1994). “Toolpath Generation for Freeform Surface Models”. en. In: *Computer-Aided Design* **26** 6, pp. 490–496. DOI: 10.1016/0010-4485(94)90070-1.

Faraut, P. and C. Faraut (2013). *Figure Sculpting: Planes & Construction Techniques in Clay*. v. 1. PCF Studios.

Feng, H.-Y. and H. Li (2002). “Constant Scallop-Height Tool Path Generation for Three-Axis Sculptured Surface Machining”. en. In: *Computer-Aided Design* **34** 9, pp. 647–654. DOI: 10.1016/s0010-4485(01)00136-1.

Friedman, J., H. Kim, and O. Mesa (2014). “Experiments in Additive Clay Depositions”. en. In: *Robotic Fabrication in Architecture, Art and Design 2014*. Ed. by W. McGee and M. Ponce de Leon. Cham: Springer International Publishing, pp. 261–272. DOI: 10.1007/978-3-319-04663-1_18.

Galilei, G. (1914). *Dialogues concerning two new sciences*. Dover.

García de Soto, B. et al. (2018). “Productivity of Digital Fabrication in Construction: Cost and Time Analysis of a Robotically Built Wall”. en. In: *Automation in Construction* **92**, pp. 297–311. DOI: 10.1016/j.autcon.2018.04.004.

Garg, A. et al. (2014). “Wire Mesh Design”. In: *ACM Transactions on Graphics* **33** 4, 66:1–66:12. DOI: 10.1145/2601097.2601106.

Gramazio, F., M. Kohler, and J. Willmann (2014). *The Robotic Touch: How Robots Change Architecture*. en. Park Books.

Graser, K. et al. (2020). “DFAB House: A Comprehensive Demonstrator of Digital Fabrication in Architecture”. In:

Guennebaud, G., B. Jacob, et al. (2010). *Eigen v3*. http://eigen.tuxfamily.org.

Hack, N. and W. V. Lauer (2014). “Mesh-Mould: Robotically Fabricated Spatial Meshes as Reinforced Concrete Formwork”. In: *Architectural Design* **84** 3, pp. 44–53. DOI: 10.1002/ad.1753.

Hack, N., T. Wangler, et al. (2017). “Mesh Mould: An On Site, Robotically Fabricated, Functional Formwork”. In: *Proceedings of the Second Concrete Innovation Conference (2nd CIC)* March.

Hamid, M., O. Tolba, and A. El Antably (2018). “BIM Semantics for Digital Fabrication: A Knowledge-Based Approach”. en. In: *Automation in Construction* **91**, pp. 62–82. DOI: 10.1016/j.autcon.2018.02.031.

Huang, Y. et al. (2016). “FrameFab: Robotic Fabrication of Frame Shapes”. In: *ACM Trans. Graph. Article* **3516** 1112, pp. 978–1. DOI: 10.1145/2980179.2982401.

Jacobson, A., D. Panozzo, et al. (2018). *libigl: A simple C++ geometry processing library*. https://libigl.github.io/.

Jiang, C. et al. (2017). “Design and Volume Optimization of Space Structures”. In: *ACM Transactions on Graphics* **36** 4, 159:1–159:14. DOI: 10.1145/3072959.3073619.

Jun, C.-S., K. Cha, and Y.-S. Lee (2003). “Optimizing Tool Orientations for 5-Axis Machining by Configuration-Space Search Method”. In: *Computer-Aided Design* **35** 6, pp. 549–566. DOI: 10.1016/S0010-4485(02)00077-5.

Kazhdan, M., M. Bolitho, and H. Hoppe (2006). “Poisson Surface Reconstruction”. In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. SGP ’06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, pp. 61–70.

Kazhdan, M. and H. Hoppe (2013). “Screened Poisson Surface Reconstruction”. In: *ACM Trans. Graph.* **32** 3, 29:1–29:13. DOI: 10.1145/2487228.2487237.

Kilian, A. (2006). “Design Innovation through Constraint Modeling”. en. In: *International Journal of Architectural Computing* **4** 1, pp. 87–105. DOI: 10.1260/147807706777008993.

Kilian, M. et al. (2017). “Material-Minimizing Forms and Structures”. In: *ACM Transactions on Graphics* **36** 6, 173:1–173:12. DOI: 10.1145/3130800.3130827.

Knöppel, F. et al. (2013). “Globally Optimal Direction Fields”. In: *ACM Transactions on Graphics* **32** 4, 59:1–59:10. DOI: 10.1145/2461912.2462005.

— (2015). “Stripe Patterns on Surfaces”. In: *ACM Transactions on Graphics* **34** 4, 39:1–39:11. DOI: 10.1145/2767000.

Ko, M. et al. (2019). “InFormed Ceramics: Multi-Axis Clay 3D Printing on Freeform Molds”. In: *Robotic Fabrication in Architecture, Art and Design 2018.* Cham: Springer International Publishing, pp. 297–308. DOI: 10.1007/978-3-319-92294-2_23.

Kumar, N. et al. (2017). “Design, Development and Experimental Assessment of a Robotic End-Effector for Non-Standard Concrete Applications”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1707–1713. DOI: 10.1109/ICRA.2017.7989201.

Lira, W., C.-W. Fu, and H. Zhang (2018). “Fabricable Eulerian Wires for 3D Shape Abstraction”. In: *ACM Transactions on Graphics* **37** 6, 240:1–240:13. DOI: 10/ggsmx3.

Logan, D. L. (2011). *A first course in the finite element method.* Cengage Learning.

Ma, Z., A. Walzer, et al. (2020). “Designing Robotically-Constructed Metal Frame Structures”. en. In: *Computer Graphics Forum* **39** 2, pp. 411–422. DOI: 10.1111/cgf.13940.

Ma, Z., S. Duenser, et al. (2020). “RobotSculptor: Artist-Directed Robotic Sculpting of Clay”. In: *Symposium on Computational Fabrication.* SCF ’20. New York, NY, USA: Association for Computing Machinery, pp. 1–12. DOI: 10.1145/3424630.3425415.

Marry, V., B. Rotenberg, and P. Turq (2008). “Structure and Dynamics of Water at a Clay Surface from Molecular Dynamics Simulation”. en. In: *Physical Chemistry Chemical Physics* **10** 32, pp. 4802–4813. DOI: 10.1039/b807288d.

Meyer, M. et al. (2003). “Discrete Differential-Geometry Operators for Triangulated 2-Manifolds”. en. In: *Visualization and Mathematics III*. Ed. by H.-C. Hege and K. Polthier. Mathematics and Visualization. Berlin, Heidelberg: Springer, pp. 35–57. DOI: 10.1007/978-3-662-05105-4_2.

Michell, A. (1904). “LVIII. The Limits of Economy of Material in Frame-Structures”. In: *Philosophical Magazine Series 6* **8** 47, pp. 589–597. DOI: 10.1080/14786440409463229.

Mitchell, J. S. B., D. M. Mount, and C. H. Papadimitriou (1987). “The Discrete Geodesic Problem”. In: *SIAM Journal on Computing* **16** 4, pp. 647–668. DOI: 10.1137/0216045.

Mueller, S. et al. (2014). “WirePrint: 3D Printed Previews for Fast Prototyping”. In: *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. UIST ’14. Honolulu, Hawaii, USA: Association for Computing Machinery, pp. 273–280. DOI: 10/ggsmwv.

Muntoni, A. et al. (2018). “Axis-Aligned Height-Field Block Decomposition of 3D Shapes”. In: *ACM Trans. Graph.* **37** 5, 169:1–169:15. DOI: 10.1145/3204458.

Nan, I. C., C. Patterson, and R. Pedreschi (2016). “Digital Materialization: Additive and Robotical Manufacturing with Clay and Silicone”. In:

Nealen, A. et al. (2006). “Laplacian Mesh Optimization”. In: *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*. GRAPHITE ’06. New York, NY, USA: Association for Computing Machinery, pp. 381–389. DOI: 10.1145/1174429.1174494.

Nocedal, J. and S. J. Wright (2006). *Numerical Optimization*. Springer.

Öztireli, A. C., G. Guennebaud, and M. Gross (2009). “Feature Preserving Point Set Surfaces Based on Non-Linear Kernel Regression”. In: *Computer Graphics Forum* **28** 2, pp. 493–501. DOI: 10.1111/j.1467-8659.2009.01388.x.

P. Faraut and C. Faraut (2009). *Mastering Portraiture: Advanced Analyses of the Face Sculpted in Clay*. PCF Studios, Incorporated.

Pellis, D. and H. Pottmann (2018). “Aligning principal stress and curvature directions.” In: *Advances in Architectural Geometry*, pp. 34–53.

Pensa Labs (n.d.). *Pensa Labs I CNC Wire Formers*. en-US. https://www.pensalabs.com.

Pereira, T., S. Rusinkiewicz, and W. Matusik (2014). “Computational Light Routing”. In: *ACM Transactions on Graphics* **33** 3, pp. 1–13. DOI: 10.1145/2602140.

Pérez, P., M. Gangnet, and A. Blake (2003). “Poisson Image Editing”. In: *ACM Transactions on Graphics* **22** 3, p. 313. DOI: 10.1145/882262.882269.

Peters, B. (2010). “Acoustic Performance as a Design Driver: Sound Simulation and Parametric Modeling Using SmartGeometry”. In: *International Journal of Architectural Computing* **8** 3, pp. 337–358. DOI: 10.1260/1478-0771.8.3.337.

Pietroni, N. et al. (2015). “Statics Aware Grid Shells”. en. In: *Computer Graphics Forum* **34** 2, pp. 627–641. DOI: 10.1111/cgf.12590.

Potstada, M. et al. (2016). “An Alignment Approach for an Industry in the Making: DIGINOVA and the Case of Digital Fabrication”. en. In: *Technological Forecasting and Social Change* **102**, pp. 182–192. DOI: 10.1016/j.techfore.2015.07.020.

Preisinger, C. and M. Heimrath (2014). “Karamba–A Toolkit for Parametric Structural Design”. In: *Structural Engineering International* **24** 2, pp. 217–221.

Rael, R. and V. S. Fratello (2017). “Clay Bodies: Crafting the Future with 3D Printing”. en. In: *Architectural Design* **87** 6, pp. 92–97. DOI: 10.1002/ad.2243.

Robots, U. (2015). “The URScript programming language”. In: *Universal Robots A/S, version* **3**.

Rosenwasser, D. M. (2017). “Clay Non-Wovens: Robotic Fabrication and Digital Ceramics”. In: *ACADIA 2017: DISCIPLINES & DISRUPTION [Proceedings of the 37th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA) ISBN 978-0-692-96506-1] Cambridge, MA 2-4 November, 2017), Pp. 502- 511.* CUMINCAD.

Schwartz, M. and J. Prasad (2013). “RoboSculpt”. en. In: *Rob — Arch 2012.* Ed. by S. Brell-Çokcan and J. Braumann. Springer Vienna, pp. 230–237.

Schwartzburg, Y. et al. (2014). “High-Contrast Computational Caustic Design”. In: *ACM Trans. Graph.* **33** 4, 74:1–74:11. DOI: 10.1145/2601097.2601200.

Skipper, N. T., F.-R. C. Chang, and G. Sposito (1995). “Monte Carlo Simulation of Interlayer Molecular Structure in Swelling Clay Minerals. 1. Methodology”. en. In: *Clays and Clay Minerals* **43** 3, pp. 285–293. DOI: 10.1346/ccmn.1995.0430303.

Skouras, M. et al. (2013). “Computational Design of Actuated Deformable Characters”. In: *ACM Transactions on Graphics* **32** 4, 82:1–82:10. DOI: 10.1145/2461912.2461979.

Slaughter, W. S. (2012). *The linearized theory of elasticity.* Springer Science & Business Media.

Smith, J. et al. (2002). “Creating Models of Truss Structures with Optimization”. In: *ACM Transactions on Graphics* **21** 3, pp. 295–301. DOI: 10.1145/566654.566580.

Solomon, J., K. Crane, and E. Vouga (2014). “Laplace-Beltrami: The Swiss army knife of geometry processing”. In: *Symposium on Geometry Processing graduate school (Cardiff, UK, 2014).* Vol. 2.

Sorkine, O., D. Cohen-Or, et al. (2004). “Laplacian Surface Editing”. In: *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing - SGP ’04*, p. 175. DOI: 10.1145/1057432.1057456.

Sorkine, O. (2005). “Laplacian Mesh Processing”. In: *Eurographics - State of the Art Reports* Section 4, pp. 53–70. DOI: 10.1128/JVI.00468-10.

Sorkine, O. and M. Alexa (2007). "As-rigid-as-possible Surface Modeling". In: *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*. SGP '07. Barcelona, Spain: Eurographics Association, pp. 109–116.

Stolpe, M. (2016). "Truss Optimization with Discrete Design Variables: A Critical Review". In: *Structural and Multidisciplinary Optimization* **53** 2, pp. 349–374. DOI: 10.1007/s00158-015-1333-x.

Şucan, I. A., M. Moll, and L. E. Kavraki (2012). "The Open Motion Planning Library". In: *IEEE Robotics & Automation Magazine* **19** 4. https://ompl.kavrakilab.org, pp. 72–82. DOI: 10.1109/MRA.2012.2205651.

Sullivan, A. et al. (2012). "High Accuracy NC Milling Simulation Using Composite Adaptively Sampled Distance Fields". In: *Computer-Aided Design* **44** 6, pp. 522–536. DOI: 10.1016/j.cad.2012.02.002.

Tan, R. a. S. D. (2016). "Clay Robotics: Tool Making and Sculpting of Clay with a Six-Axis Robot". In: *Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2016) / Melbourne 30 March–2 April 2016, Pp. 579-588.* CUMINCAD.

Tang, C. et al. (2014). "Form-Finding with Polyhedral Meshes Made Simple". In: *ACM Transactions on Graphics* **33** 4, 70:1–70:9. DOI: 10.1145/2601097.2601213.

Tournier, C. and E. Duc (2005). "Iso-Scallop Tool Path Generation in 5-Axis Milling". en. In: *The International Journal of Advanced Manufacturing Technology* **25** 9, pp. 867–875. DOI: 10.1007/s00170-003-2054-7.

Tryfonos, O. K. a. G. (2018). "Integrating Parametric Design with Robotic Additive Manufacturing for 3D Clay Printing: An Experimental Study". en-US. In: *ISARC Proceedings*, pp. 918–925.

Wang, Z. et al. (2019). “Design and Structural Optimization of Topological Interlocking Assemblies”. In: *ACM Transactions on Graphics* **38** 6, 193:1–193:13. DOI: 10.1145/3355089.3356489.

Weichel, C. et al. (2015). “ReForm”. In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST ’15*, pp. 93–102. DOI: 10.1145/2807442.2807451.

Wu, R. et al. (2016). “Printing Arbitrary Meshes with a 5DOF Wireframe Printer”. In: *ACM Transactions on Graphics* **35** 4, 101:1–101:9. DOI: 10.1145/2897824.2925966.

YiHong, W. and L. YongJiang (2010). “An Isoline Generating Algorithm Based on Delaunay”. In: *2010 2nd International Conference on Computer Engineering and Technology*. Vol. 7, pp. V7-173-V7–176. DOI: 10.1109/iccet.2010.5485290.

Young, D. A. and D. E. Smith (2000). “Simulations of Clay Mineral Swelling and Hydration: Dependence upon Interlayer Ion Size and Charge”. In: *The Journal of Physical Chemistry B* **104** 39, pp. 9163–9170. DOI: 10.1021/jp000146k.

Zehnder, J., S. Coros, and B. Thomaszewski (2016). “Designing Structurally-Sound Ornamental Curve Networks”. In: *ACM Transactions on Graphics* **35** 4, 99:1–99:10. DOI: 10.1145/2897824.2925888.

Zhu, W. and Y.-S. Lee (2004). “Five-Axis Pencil-Cut Planning and Virtual Prototyping with 5-DOF Haptic Interface”. en. In: *Computer-Aided Design* **36** 13, pp. 1295–1307. DOI: 10.1016/j.cad.2004.01.013.