TECHNISCHE
UNIVERSITÄT
WIEN

Diplomarbeit

Multi-scalar modelling for robotic prefabrication

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Diplom-Ingenieurs unter der Leitung:

Dipl.-Ing. Dr. Zeynep Aksöz Balzar

E259-02, Interdisziplinäre Tragwerksplanung und Ingenieurholzbau

Eingereicht an der Technischen Universität Wien

Fakultät für Architektur und Raumplanung

von

Georg Lobe

01227063

Wien am

ABSTRACT

Planning practice in the building sector is characterized by repeating similar modelling tasks over and over again. In particular, it is expressed in current modelling strategies by modelling and remodelling digital representations of a building at each stage of the project development. This destructive way of handling data leads to long development processes which loses data and accuracy on its way and increases labour costs. Approaches that combine precision, speed and collaboration are required to keep pace with increasing complexity and volume of construction projects.

This thesis investigates the current practice and development in the field of open collaborative multi-scalar modelling and planning strategies for the AEC industry in the field of robotic fabrication. A prototypical workflow was developed generating a free form grid-shell with doubly curved glulam beams.

Utilizing tools for planning the structure from conception to fabrication in a process thought to be as open as possible for collaboration. Introducing connected multi-scalar modelling in a parametric environment from the design stage to robotic fabrication using a six-axis robot.

By using web-based services, accessibility of the necessary data for each stakeholder can be achieved, building a coherent model open for collaboration and dataflow across different platforms and software packages. Consisting of multiple models of different level of detail (LOD) each reacting on changing parameter and running on cloud services. Making them independently accessible from the location and on hand computational power for qualified personal. Invoking re-computation on one model can inform and, if needed, update models downstream the dataflow.

Focus was on testing usability of current tools and identifying passages where feedback loops are expedient. Needed information as well as possibilities for robotic fabrication and control was evaluated.

ZUSAMMENFASSUNG

Die Planungspraxis im Bauwesen ist geprägt von immer wiederkehrenden ähnlichen Aufgabenstellungen. Insbesondere kommt es in aktuellen Modellierungsstrategien zum Ausdruck, indem digitale Modelle eines Gebäudes in jeder Phase der Projektentwicklung umgestaltet und neu modelliert werden. Dieser destruktive Umgang mit Daten führt zu langen Entwicklungsprozessen, die durch wiederholte überarbeitung information und Genauigkeit verlieren bzw. neu erstellen und somit die Arbeitskosten erhöhen. Um mit der zunehmenden Komplexität und dem Volumen von Bauprojekten Schritt halten zu können, sind Ansätze erforderlich, die Präzision, Schnelligkeit und Zusammenarbeit vereinen.

Diese Arbeit untersucht die aktuelle Praxis und Entwicklungen im Bereich der offenen kollaborativen multiskalaren Modellierungs- und Planungsstrategien für die Bauindustrie. Im Hinblick auf robotergestützte Fertigung wurde ein prototypischer Arbeitsablauf entwickelt, der eine Freiform-Gitterschale mit doppelt gekrümmten Brettschichtholzträgern generiert.

Durch die Verwendung webbasierter Dienste konnte die Zugänglichkeit der erforderlichen Daten für jeden Beteiligten erreicht werden. Dadurch konnte ein kohärentes Modell, welches für die Zusammenarbeit und den Datenfluss über verschiedene Plattformen und Softwarepakete hinweg offen ist, aufgebaut werden.

Bestehend aus mehreren Modellen mit unterschiedlichem Detaillierungsgraden, die jeweils auf sich ändernde Parameter reagieren können, wird so ein Workflow beschrieben der Standortunabhängig und nicht auf lokale Rechenleistung angewiesen ist. Das Aufrufen einer Neuberechnung einzelner Modelle kann nachfolgende

Modelle informieren und so bei bedarf das aktualisieren von Informationen veranlassen.

Der Fokus lag darauf, aktuelle Möglichkeiten zu testen und Schwachstellen zu identifizieren, an denen es Verbesserungspotenzial gibt. Sowie Abschnitte im Designprozess zu ermitteln an denen Feedbackschleifen sinnvoll erscheinen.

INDEX

1 INTRODUCTION

The present thesis examines benefits in multi-scalar model for robotic prefabrication. Utilizing multiple representations of a structural system in the search for novel strategies in collaborative planning for digital fabrication.

1.1 Context

The Architecture, Engineering and Construction (AEC) sector is growing, not only in volume but also in complexity. But continuous digital planning strategies from design to assembly, delivering a final product to the construction site is a practice yet to come. Usually models, geometry and information, are remodelled by different shareholders before and during the construction phase only fulfilling their current purpose and reacting on given conditions during the process. For that, new more detailed information is added to the digital representation of a building and the previous model is declared outdated. This practice increases the amount of work and labour cost but also the error rate in coordination and planning. [25]

In the context of industrialisation, implementing prefabrication and "lean production principles" for the AEC industry the planning effort needs to shift forward where decisions are made early in the process. [24] Therefore gathering and linking information early in the process becomes more important. In most of the industries fabrication starts after the planning is finished. This frontloading can reduce production time and enables automated, time efficient production of the final product.

In Industries like the automotive industry, use of multi axis robots is considered standard since decades working mostly assembly lanes. Industrial robots offer flexibility in design language and precision as well as inexhaustible workforce. The development of digital tools for robotic manufacturing in architecture is gaining momentum with novel prospects in a wide range of applications. Increasing accessibility expands possibilities and prompts questions about how these technologies can help improve material efficiency and advance building industry to novel materials, fabrication techniques and an overall more sustainable practice.

1.2 Hypothesis

This master thesis theorizes that linked multi-scalar modelling can improve the workflow for industrialized architectural planning. By linking different stages of the design process more directly together - thus informing each other from early on – can increase productivity and decrease error rates independent from human capital, and IT capabilities. Developing semi-self-updating models open for human interaction makes it possible to resolve flaws in implementation or to react on unforeseen changes along the planning of building components. Enabling feedback between multiple level of detail (LOD) will further increase optimization possibilities and enable streamlined planning for automated robotic fabrication. It is assumed that building a consecutive interlinked model leads to a more robust planning process. Consequently, making information and knowledge accessible early on is mandatory for the development of robust interlinked adaptive modelling strategies.

The focus lies on developing a modelling workflow assumed to be open for collaboration using web-based services. Supposing that by using web APIs, versioning of different model iterations during the planning process as well as distribution of information between shareholders is enhanced. The main target is to provide an information rich and flexible approach for robotic fabrication.

The thesis aims to answer the following questions:

- What are the limitations of current web-based services in the scope of model complexity and data exchange?
- Overhead and possibilities of feedback using a multi-scalar modelling approach for prefabrication?
- Emerging constraints concerning automation and limitations using a six-axis manipulator?

Former research questions are tested along a series of coherent experiments, exploring a streamlined workflow tested on the case of fabricating doubly curved glulam beams.

1.3 Thesis Outline

In Chapter 2 existing tools are presented that were used in the process and built examples are examined, planned and constructed using similar approaches. By analysing existing projects, a basic outline for the experiments is derived. Chapter 3 introduces the strategies utilized in Chapter 4.

Chapter 4 presents a prototypical approach to fabricate doubly curved wooden beams. This chapter is divided into three parts - digital planning; basic robotic fabrication, and for the last, both previous parts were evaluated and combined.

2 FUNDAMENTALS

2.1 Computational Tools and Methods

2.1.1 Building Information Modelling

The aim of Building Information Modelling (BIM) is to encourage collaboration between different stakeholder during the lifecycle of a building, from conception to maintaining the building. At the heart of BIM is a digital representation of a facility with its physical and functional characteristics – the building information model. Built virtually prior to physical construction figuring out problems and clashes, simulating and analysing possible impacts. Every stakeholder interacts with the model. Adding, extracting, or modifying information to solve emerging conflicts between building systems towards a final virtual model. [28] The common conception of BIM is that it is just a set of tools or a software package like Autodesk's Revit, Nemetschek's Allplan or Graphisoft's ArchiCAD, etc. ..., instead of a methodology that enfolds policies, processes and open standards to promote interoperability and collaboration.[5] Therefore BIM should not be limited to one vendor but instead needs to offer robust possibilities for seamless data exchange in an open environment.

By becoming mandatory for public projects in different countries, at the forefront the UK, Scandinavian countries, USA, Singapore and France [34], the tools developed for AEC have to communicate and funnel in BIM enabled workflows. Adoption of BIM increased steadily but has accelerated in recent years. (Fig.: 1)

BIM supports numerous methodologies and practices – Lean construction,

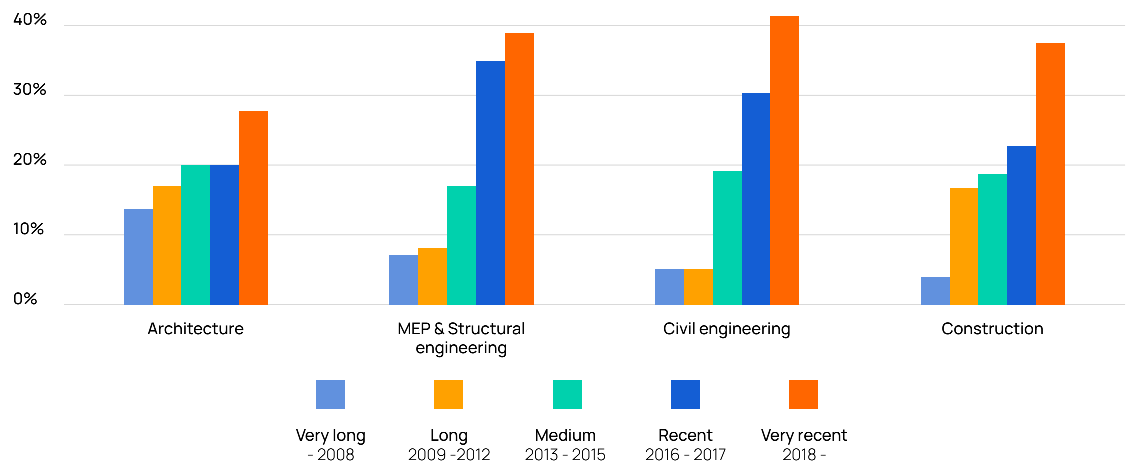


Figure: 1 - BIM adoption curve [5]

Integrated Design Process (IDP) and Design for Manufacture and Assembly (DfMA).

With DfMA being the newest buzzword to add to a long list of methodologies in an architectural context. Developed by Ford and Chrysler it was first used in manufacturing industries and since the last decade is increasingly being discussed and used in AEC industry. [17] Compared to traditional sequential planning processes, DfMA offers a methodology for thinking about fabrication and assembly processes early in the design phase to be able to make informed design decisions. By considering downstream processes it is possible to activate significant potential in increased productivity and reducing production costs. [3]

"Only when manufacture and assembly techniques are incorporated early in the design stage will productivity be significantly affected." [3]

Since the focus of DfMA is on maximizing the overall value by taking production into account, the implementation of the methodology is not tied to a certain scale. It can be applied to individual building component as well as to a housing project. The increased use of off-site prefabrication and on-site construction in particular pushes

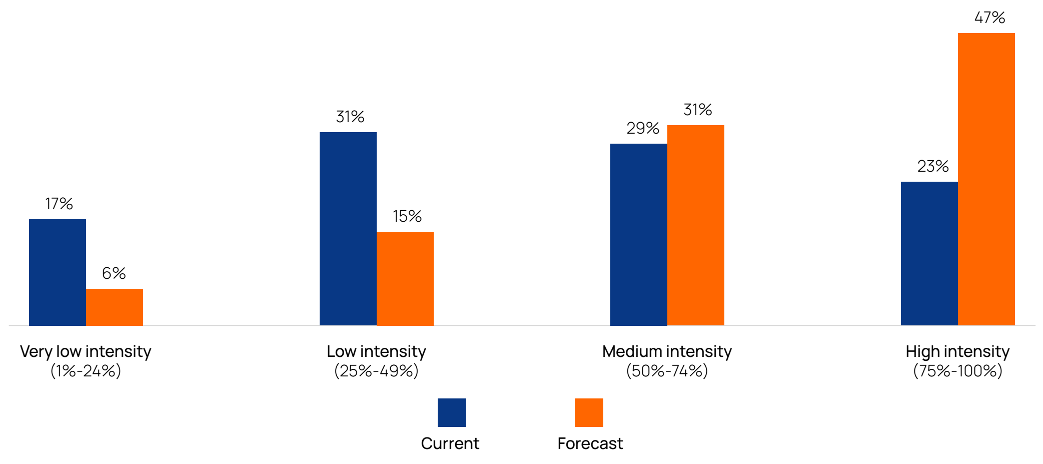


Figure: 2 - BIM usage intensity compared with forecast [5]

DfMA, as the components need to be precisely described early in the design phase. It can assist Lean Construction principles, where the main focus is on eliminating waste and high-quality management of projects, by providing the designer with insight where manufacturing and assembly is inefficient.

DfMA is an extension of current BIM practices which, through parametric design, closes the loop with a detailed description of each individual component. BIM offers DfMA data rich objects with the help of which analyses and simulations can be carried out to identify optimization potential for the fabrication and manufacturing processes. In addition, BIM is an interdisciplinary environment in which different trades can access a model, which makes low-threshold communication possible. [17]

2.1.2 Multi-scalar Modelling

Multi-scalar modelling allows models at different scales to be viewed simultaneously. This should combine the efficiency of the macro level and the accuracy of the micro level, so that complex tasks can be modelled precisely and efficiently.

"Maintaining full detail resolution at all levels of the system mostly becomes inefficient and computationally intensive" [20]

Multi-scalar modelling is widely used across disciplines, including physics, chemistry, meteorology and engineering in general, and it is not a new approach in architecture either. [36] The basically iterative architectural process is excellently suited for the implementation of multi-scalar modelling strategies due to the number of parameters and trades. Due to the complexity that results from several specifications, suitable interfaces must be created or defined in order to be able to do justice to loss-free scaling between the models.

In order to create a consistent multi-scalar model, interfaces, data structures and basic structure must be defined in advance. For this one has to make fundamental considerations about what should be modelled, how it should be modelled and how individual models are connected.

First, the problem and its approaches to solving it should be addressed, this allows guidelines for further steps to be set and a framework for modelling to be created. When modelling, attention must be paid to the interaction of the individual models. The computational designer must plan ahead so that the various objects, data sets and geometries result in a complete picture when implemented.

The way information is connected affects each model. The implemented modelling pipeline affects the flexibility of the holistic model. This is where it is determined how easy it is to make changes, as well as the degree of interactivity and usability. [21] Thus, the planning of the models and defining their relationships to each other in advance is of great importance and requires the necessary experience.

2.1.3 Application Programming Interface (API)

An Application Programming Interface (API) is a description of a library, consisting of functions, methods, enumerations, classes, data types and constants.

"Strictly speaking, an API is simply a description of how to interact with a component." [22]

For example, Microsoft Windows uses files of datatype .lib or dynamic-link library (.dll). APIs define reusable components that allow modular design of applications. Bringing functionality / services to other pieces of software and make it possible to link them together, building highly complex programs using multiple libraries. This component approach makes interoperability easier to achieve using standards when designing APIs.

The programming interface is usually not accessible to the end-user but to the programmer who calls or implements different components of the API, where internal details are hidden, and only useful parts are exposed to the programmer. This is especially useful if internals later change but the calls one must make stay the same.

Modern software engineering strongly depends on robust APIs otherwise common goals would be harder to achieve like code reuse, distributed computing, service-oriented architecture, componentization and modularity.

"As an example of a well-known API, Microsoft's Windows API (often referred to as the Win32 API) is a collection of C functions, data types, and constants that enable programmers to write applications that run on the Windows platform.

This includes functions for file handling, process and thread management, creating graphical user interfaces, talking to networks, and so on." [22]

An API can be a single function to a collection of classes, from low-level operating system calls to highly complex Graphical User Interface (GUI) toolkits.[22]

2.1.3.1 Web API

Web APIs are server-based interfaces which can be accessed using the Hypertext Transfer Protocol (HTTP). Using technologies such as ASP.NET or java it is possible to build RESTful web services. What RESTful services are will be discussed on a later point in this thesis.

Basically, Web APIs are websites or services on the web which can be accessed using a client application which will send HTTP requests to receive data from a server database or trigger calculations on a decentralized cloud environment. A server-side Web API consists of publicly exposed endpoints defining a request-response message system. Popular examples would be the Google Maps API, YouTube APIs and the Twitter APIs.

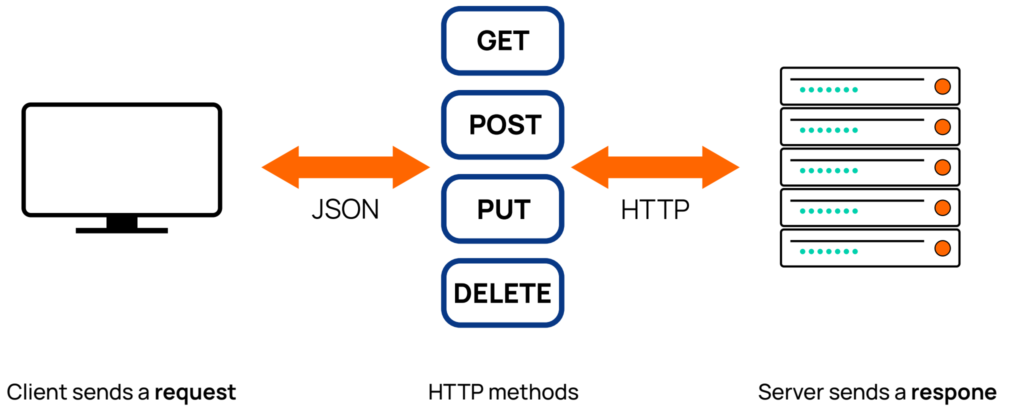


Figure: 3 - Basic web communication using HTTP [19]

A Web API can come in handy if services or applications need to be shipped to multiple devices such as laptops, mobile phones and PCs, or if it is necessary to have an application running on a distributed environment as for cloud computation.[14] Representational State Transfer (REST) API is a software architecture style developed to help standardise web application design, described by Roy Thomas Fielding in his doctoral thesis.

RESTful services are built to be performative, scalable, modifiable - working best in a server-based environment.[12]

The architectural style of REST is composed of a set of constraints defining the fundamental behaviour of systems designed using this approach.

The first constraint would be the separation of concerns using the **client-server** architectural style (Fig.: 4). Separating the user interface from the data storage allows components to advance during development independently, thus being independent from each other and satisfying internet-scale requirements of multiple organizational domains. Further by simplifying server components scalability of the system is improved as well as portability on the client-side.

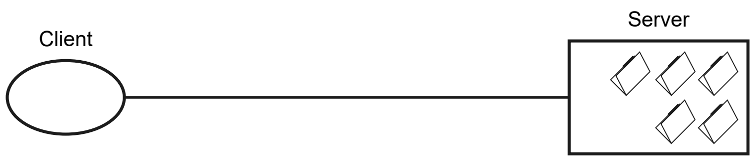


Figure: 4 – Client - Server [13]

Communication between client and server must be **stateless**, meaning that each request from the client must contain all the information necessary to understand and process the request on the endpoint. (Fig.: 5)

The server cannot take advantage of any previous stored information. This brings some advantages – visibility, the full nature of the request is already encoded in the request, there's no need for a monitoring system looking beyond a single request datum; reliability, because it makes it easier to recovery from partial failures; scalability, the server can free up resources between each request and does not store data from previous requests.

But there are also some disadvantages – network traffic might be increased due to the

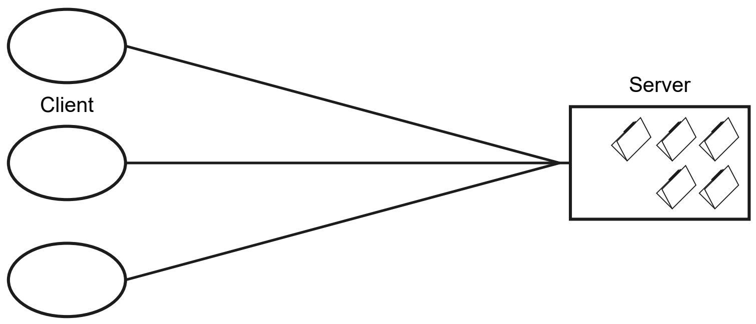


Figure: 5 – Client - Stateless - Server [13]

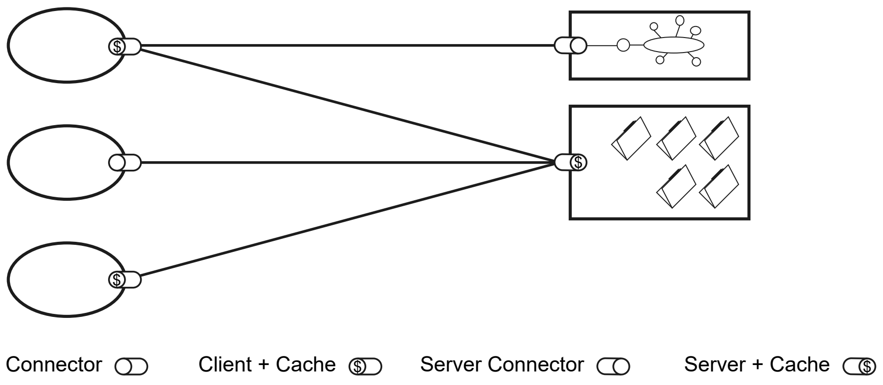


Figure: 6 – Uniform – Client – Cache – Stateless - Server [13] need for every request to send all information in each call. Also, the server's control over consistent application behaviour is reduced since the state the application is in is placed on the client-side, therefore dependent on the correct implementation across multiple client versions.

Adding a **cache** constraint increases efficiency and performance. Data within a response to a request can be labelled cacheable or non-cacheable, partially or completely eliminating the need to interact with the server directly. Reliability is decreased if cached data differs from the data that would have been received if the request had been processed by the server directly. (Fig.: 7)

REST promotes a **uniform** interface between components by applying the principle

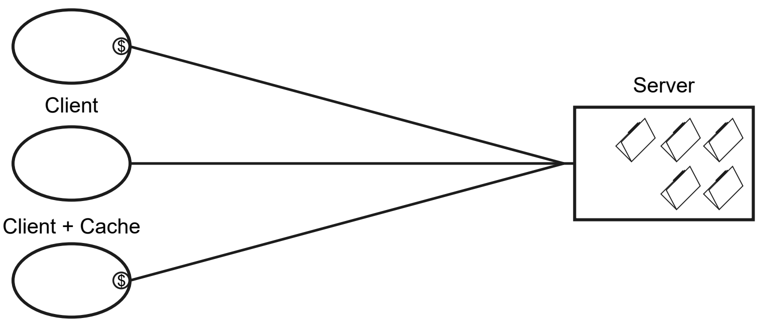


Figure: 7 – Client - Cache - Stateless - Server [13]

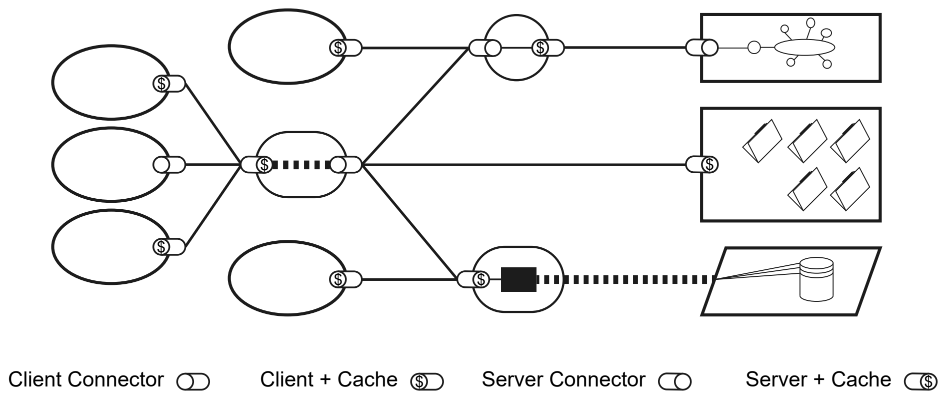


Figure: 8 – REST [13]

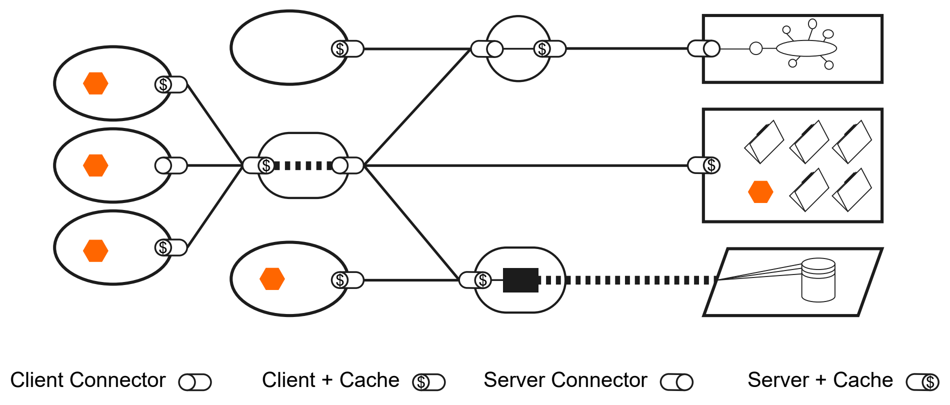


Figure: 9 – Uniform – Layered – Client – Cache – Stateless - Server [13]

of generality. Implementations are decoupled from the services they provide; system architecture is simplified, and the visibility of the requested interactions are improved. (Fig.: 8)

The system is further restricted by hierarchical **layers** where components on one layer are only aware of the immediate layer on which they are operating. This limits the overall complexity of the system and simplifies components by placing infrequently used functions on shared intermediary layers, improving scalability and load balancing across networks and processors. (Fig.: 9)

The last constraint is an optional - code on demand. By allowing clients to execute code or download from scripts, reduces the number of features required to be pre-implemented and improves system extensibility.[13]

2.1.3.2 Rhino.Compute

Based on the Rhino.Inside* technology Rhino.Compute lets you access Rhino's geometry library (RhinoCommon) using a stateless REST API running on Windows servers. [11] Essentially it is a Web API providing the RhinoCommon API/.NET SDK in a non-user interface (headless) environment for performing geometric calculations on remote servers.

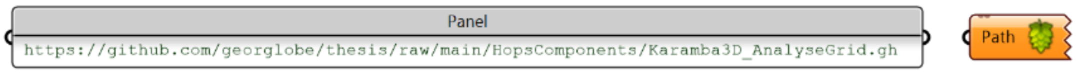
Rhino.Compute utilizes standard HTTP calls to communicate with the server making it independent from the programming language used to interact with said server. [11] Compute is an open source project which lets you manipulate Rhino.3dm files, among other filetypes, and calculate Grasshopper definitions in serial or parallel also sync or async with or without caching. [15]

2.1.3.3 Hops

Hops is an extension for Rhino Grasshopper developed by McNeal; it is designed to use a Grasshopper definition as a function like in programming languages like C#. In the context of Hops the Grasshopper component is a client sending given data to a headless Rhino server on the web or locally, solving the definition using Rhino.Compute.

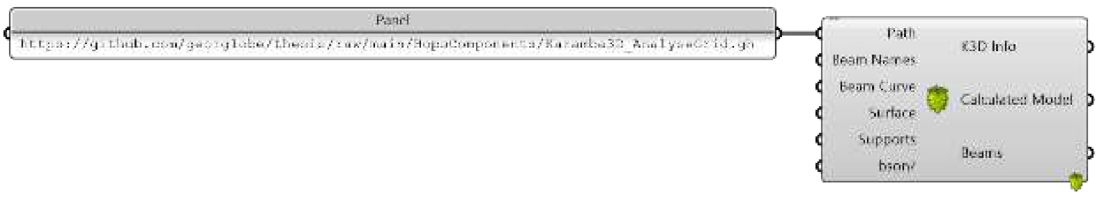
In order to function, Hops needs to locate the Grasshopper definition (Fig.: 10) it is going to solve. Local paths on the hard drive as well as URLs, for example pointing to a GitHub repository, can be used. This property makes it easy to share definitions, reuse them and make collaborative workflows easier. Once given a valid path, the Hops-component will update – changing its appearance according to the loaded Grasshopper definition (Fig.: 11). [18]

Connecting the path to the definition initiates the first interaction with the server. Hops sends the definition to the server addressing the endpoint; the server evaluates the Grasshopper definition and determines which input and output nodes are needed



```
https://github.com/georglobe/thesis/raw/main/HopsComponents/Karamba3D_AnalyseGrid.gh
```

Figure: 10 – Set Path to location of the Grasshopper file



```
https://github.com/georglobe/thesis/raw/main/HopsComponents/Karamba3D_AnalyseGrid.gh
```

Figure: 11 – Loaded .gh definition from github.

on the GUI of Grasshopper. When all the inputs have been connected/populated with valid data – Hops will send another HTTP request to the server addressing the /solve endpoint. The server responds by sending back data which would be also returned from running the Grasshopper definition using the Grasshopper Player or opening the file in Grasshopper itself. [18]

2.1.3.4 Speckle

Speckle is an Open-Source Project developed to enhance interoperability, automation and cooperation in the AEC industry. Providing a generic base datatype which can store different types of data and data-structures. It offers methods to convert native geometry from one program using the Speckle.Base object to another native geometry type from another program: Rhino.Mesh – Speckle.base – Blender.Mesh

It is independent from cloud providers, you can setup your own server and be in control of how and where the data is stored. Modification and extensions to Speckle are easy to implement because of the open-source nature. [33]

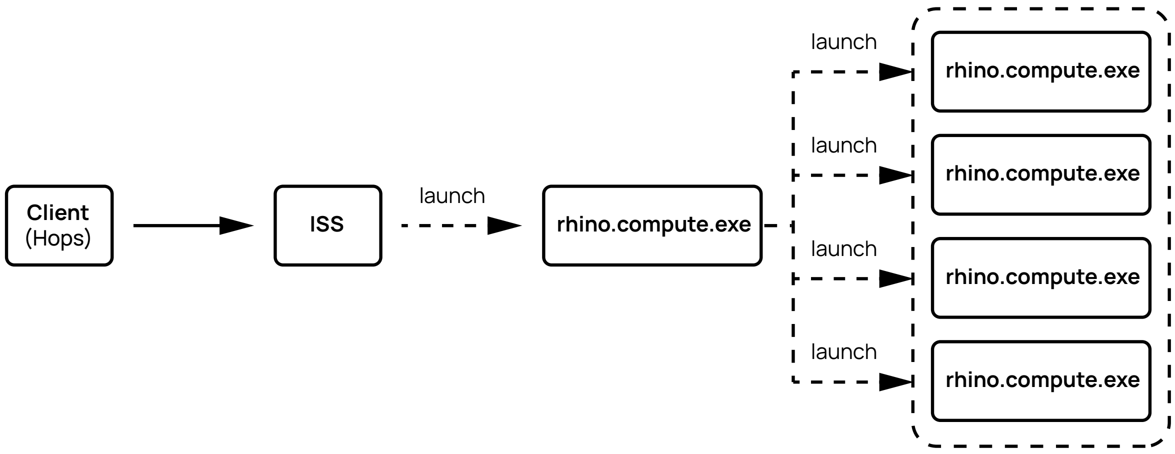


Figure: 12 – Hops API Request [18]

According to the Developer, Speckle is a platform and a product – "Speckle has two distinct parts: the developer platform, and the applications and products built on top."-Speckle. Speckle also provides a repository for geometric data comparable to GitHub. But also includes a set of software solutions, the so called "Connectors", which improve interoperability between different major software products used in the AEC industry (Rhino, Grasshopper, Revit, Tekla, etc...). [32]

Figure: 13 - Swatch Watch Headquarters Interior [25]

2.2 Reference Projects

2.2.1 Swatch Watch Headquarters - Shigeru Ban

Swatch Watch headquarters, completed in 2019 after five years of planning, is located in Biel, Switzerland, and was designed by Shigeru Ban. The almost 200 meter long wooden gridshell with its 27m height spans a four-story office building and a public square. [7]

The load bearing shell was divided into 2,800 approximately 2x2m big quadrilateral pads. Consisting of different types of panels – ETFE domes, glass and solid, the façade takes over this division. Furthermore, the façade is the first fully featured building façade with integrated installations in a free-form wooden gridshell. [25] The supporting structure is composed of doubly curved glulam-beams with an overall span of 240m in length, 35m in width and an enveloping surface area of 11000m2. The 4,500 beams were layered three times on top of each other. To cover the demand of wood, around 6,500 Swiss spruce trees were processed. [1]

In the case of the Swatch Headquarters, a parametric reference model was worked on. By involving all stakeholders, especially specialist planner, early in the project it was possible to recognise and respond to subject-specific requirements much earlier and work them out together. Furthermore, the agile parametric model made it possible to feed in changes relatively quick and check for clashes or errors.

In order to make the communication between the stakeholders easier, a 2D flattened view of the envelope was created. (Fig.: 15) Every change of the façade builder, MEP consultant, structural engineer and so on, could thus be defined in 2D and then

Figure: 14 – Top View, Building Envelope [25]

transferred to the 3D model. The parametric model had to contain 4,500 glulam beams and all the associated details like cut-outs for installations, the support points, cross-section sizes, connection joints and fasteners. [25] By using a parameterized building model, the 16,000 steel parts and 140,000 fasteners could be reduced to a few families. [1]

The starting point of the model was a continuous NURBS-surface, created from eight degree 7 NURBS-curves with only ten control points each. Between these eight curves, the control points were interpolated into a field of 126x10 points. (Fig.: 16) By means of an iterative optimization process, the control points were moved in the plane of intersection until requirements for floor plan, room height and structural stability were satisfied.

The previously mentioned quadrilateral fields of the grid were adapted to a uniform edge length of 2.1m and 2.97m diagonally by using a spring mass relaxation algorithm.

In order to avoid oversizing the individual beams and to allow cross-section sizes

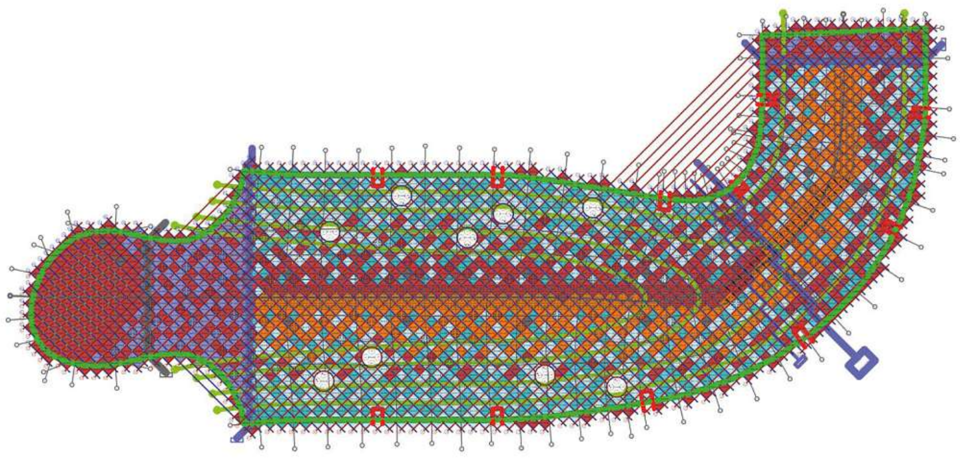


Figure: 15 – Unrolled Building Envelope [25]

to change their size continuously, the reference surface was divided into regions of different heights. Between these regions the cross-section heights, varying within 760mm to 925mm, were interpolated. The result were variably offset NURBS-surfaces serving as a reference for the supporting structure on the one hand but also for the façade. [25]

Production and assembly required absolute precision and meticulous planning on the 3D model. [1] In order to ensure that possible errors in the model are detected timely and thus to prevent delays and an increase in costs, a three-stage quality assurance was introduced. In the first step, the model was checked automatically using algorithms but also manually.

Next, parts lists were created which in turn could be compared with calculations and other data sets, regardless of the 3D model. As a final step, "six-eyes-approval", parts of the model were exported together with their connecting parts and parametric planner, timber builder and structural engineer created issue lists. Only after all problems had been solved, each component was released for production. [25]

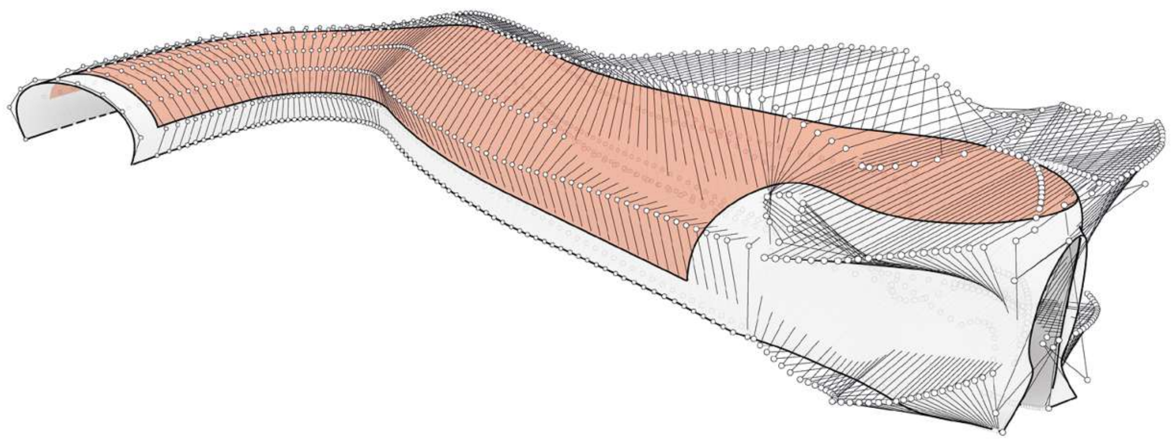


Figure: 16 – Optimization of the reference surface [25]

According to the manufacturer, Blumer Lehman – responsible for planning, production and execution of the wooden structure, one of the greater difficulties was to coordinate the construction process and having the required parts at the right time in the designated place. [1]

Figure: 17 – Assembly [29]

Figure: 18 – La Seine Musical Exterior [38]

2.2.2 La Seine Musical - Shigeru Ban

La Seine Musicale is located on the Ile de Seguin in the west of Paris. In addition to several rehearsal and concert halls, the building includes an auditorium with 1,100 seats. On the wooden construction of doubly curved wooden beams, which is almost 30m high, sits a glass façade. [6]

The wooden construction is ovate – approximately 70m in the longitudinal axis and 45m on the transverse axis. A total volume of 900m3 of glulam was used for the auditorium, assembled in about 10 months. Glass elements of the façade were attached directly to it by the façade builder using a wooden substructure. Hess Timber produced a total of 1,700 glulam components shaped using 3D CNC processes. Due to the required visual quality of the surface of the spruce beams, the blanks were manufactured in such way that the wood fibres follow the geometry exactly and thus visibly milled adhesive splices could be avoided.

The double-curved tension belts are connected to each other using wood-wood connections. For the first time, high strength cam strips made of beech plywood, a development by Hermann Blumer and SJB, were used at the rod ends, which mechanically took over the enormous tensile forces in the face. These “tooth profiles” are CNC manufactured from beech plywood, which has the required shear strength. The connection from beech to spruce is carried out by bonding. Shear-resistant flaps were used for the x-shaped diagonals. [16]

Design-To-Production was commissioned for the digital modelling of the parametric 3D model. The model is detailed down to the last screw and was also used to create a concept for the complex assembly sequence. Furthermore, automatic production data

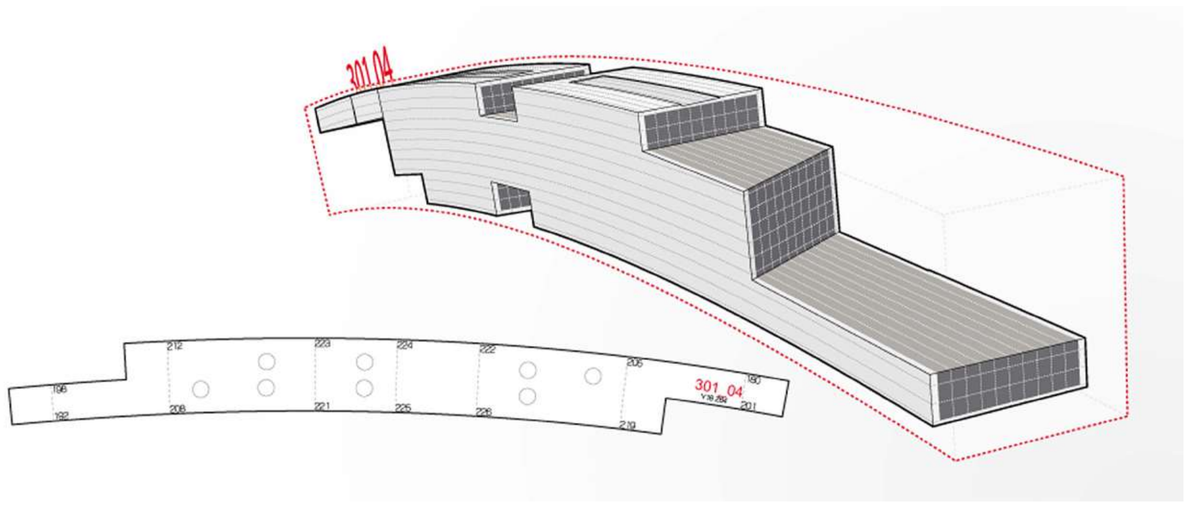


Figure: 19 – Final beam inside of blank geometry. [8]

for gluing and joining of the almost 1,300 beam segments as well as volume models of all 3,300 façade frames were derived to generate a complete set of workshop and assembly plans. Blanks for the beams were produced in a multi-stage process, from rod slats with a cross-section of 32x40mm. Three different gluing methods were chosen depending on the curvature and length.

Each blank required an individual set of detailed drawings, tables and machine-readable production data. Detailing of the 2,800 unique intersection points in the design was done parametrically and clustered into 8 families with a total of 120 subcategories depending on structural and constructive requirements. To ensure the synchronicity of the geometric and the structural model, a table-based interface has been defined. This ensured that the correct type of details was used at all points. [6]

Installation period was ten months from September 2015 till June 2016. [16] In order to require as little support structure as possible and to enable self-supporting of the structure at all times, the diagonal beams were assembled first into x-shaped elements and placed in rows, only selectively supported. (Fig.: 20)

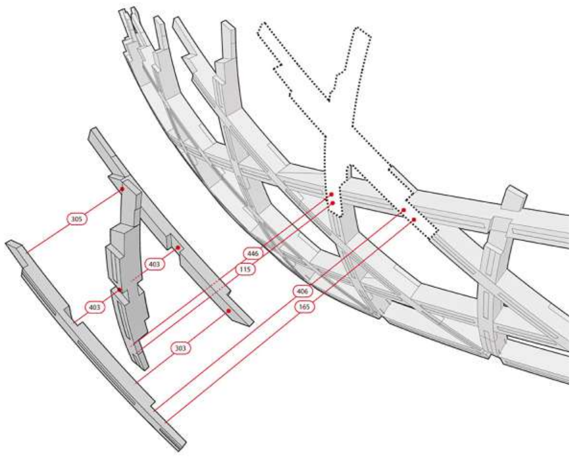


Figure: 20 – Assembly of diagonal beams. [10]

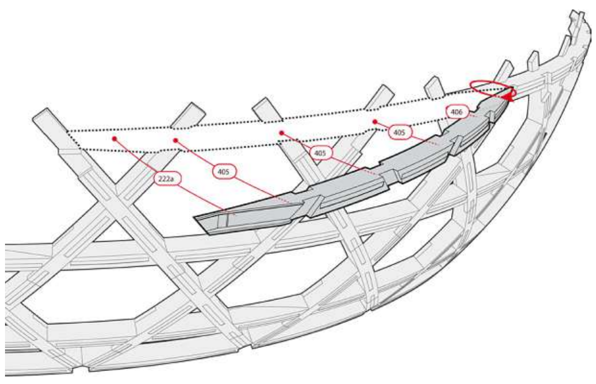


Figure: 21 – Assembly of one ring segment. [9]

Subsequently, the up to 24m long ring segments were inserted. In order to enable the assembly of the ring segments, the flanks of the cut-outs at the intersection points had to be individually bevelled. For this purpose, the exact assembly process of each segment had to be determined.

Movement was defined as turning around one end of each segment in order to be able to retract the intersection points one after the other and to keep the angles of the bevels as small as possible. [6]

3 METHODOLOGY

This master thesis follows a research by design approach because of the complex nature of planning a building and its workflow. Designing, planning and modelling building components requires the ongoing reflection on the synergy of all parts together as well as for itself [39]. So, the process will be a process of multiple feedback and needs to be treated as a continuous development until every problem has been addressed. To clarify the naive approach and allow for reflection on every step the next Chapter 4, describing the experiments, was structured in three parts.

The first is the implementation of a multi-scalar modelling workflow using web APIs in the design process; the second is a proof of concept (POC) of the robotic setup and generated toolpaths; the third will test the accuracy of previous experiments evaluating the quality, by combining both.

In the first experiment a multi-scalar modelling workflow is developed. Consisting of a reference-model in control of smaller sub-models and functions. The sub-models are orchestrated by the designer from within the CAD environment. The reference model represents a graph or data stream - collecting data, coordinates, references, checks for validity, connects, and generates or pulls the data on demand from the sub-models. It also supplies the sub-models with as little data as possible to generate the desired outcome, saving resources.

Each model is built as a standalone Grasshopper Definition but the process itself is not limited to this environment. These definitions get called from the reference model using Hops-Components, a way to communicate with the web services. Each model or component were stored in Github repositories – facilitating access to them. As of means for collaborative workflow and storing important information, Speckle is used as a repository where it is possible to use different streams as contextual input while working on the models separately. This ensures that data won’t be recomputed every time it is requested. Data is stored in, for the sake of simplicity, so called Objects – predefined and structure according to significant information. Therefore, a Beam-Node- and BeamFabrication-Object was implemented. Collaborators work on the same version of up-to-date geometry or data using this dataformat.

Because of the provided versioning through Speckle and Github the possibility to go back to an earlier state is simplified. After each step, finalized information will be validated and pushed to the corresponding Speckle stream, either manually or automated. While still maintaining control over the data from within each sub-model, the reference model can request re-computation and overwrite existing information in the repository.

For the second experiment, fabrication constraints using a six-axis robot for milling are investigated. Given joint geometry was examined for robotic fabrication feasibility.

The last experiment implements findings of the second into the data pipeline of the first experiment, completing the workflow from design to production. By extending the reference model with new models and functions, handling material properties, constraints in fabrication and machine data generation - a consistent workflow from an idea to the physical realization was accomplished.

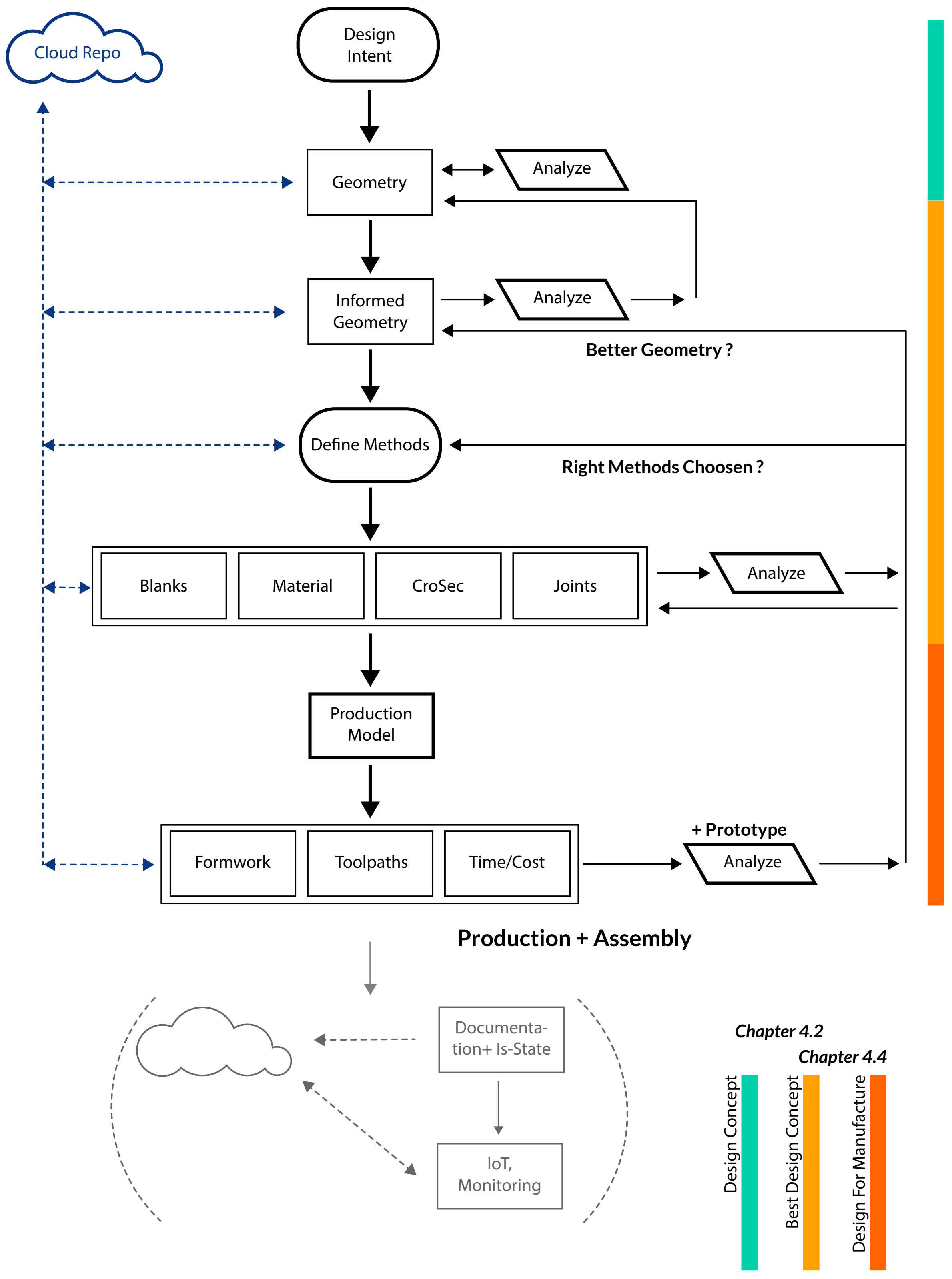


Figure: 22 – Dataflow

4 EXPERIMENTS

4.1 Introduction

Because of the multi-scalar nature of designing and building in an architectural context, it is crucial to develop BIM enabled workflows for all stages in the process. The numerous parties and information involved makes it often hard to handle complexity of planning a building. Involving a multitude of stakeholders and information can become a headache quite easily connecting, communicating and coordinating data, geometry and schedules.

The following study explores possible workflows using decentralized methods, namely Rhino.Compute with Hops as client application and Speckle as repository for detailed informed geometry. Benefits but also shortcomings of the two methods were evaluated and compared to current practice.

In the first experiment (4.2) the basic functionality of each and the combination of both products were examined. Trying to build up knowledge about the products in question, pointing out benefits and finding workarounds for eventual weaknesses. After that, a multi scalar process was developed and applied to a free-form surface generating a timber grid shell. Analysing and optimizing the structure, implementing the possibility for human interacting and automation as far as reasonable.

Content of the second experiment (4.3) is robotically milling a traditional Japanese wood joint.

The last study combines 3.2 and 3.3 applying previous findings to realize two double curved wooden beams. By using the, mostly automated, data generated in 3.2 and already developed algorithms and methods from 3.3 a streamlined workflow for multi-scalar modelling for robotic prefabrication is presented.

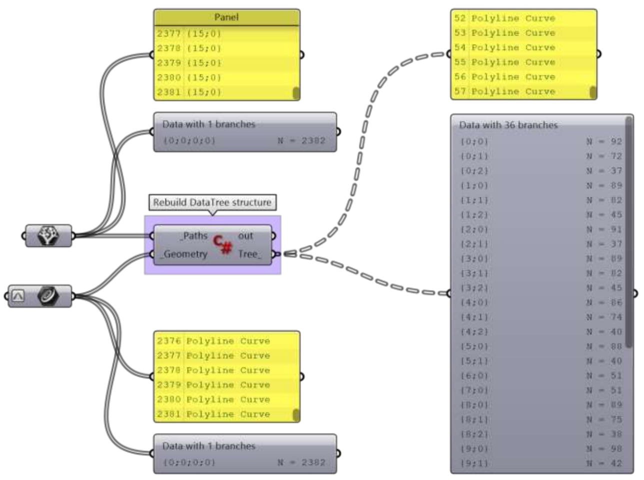
4.2 Experiment 1: Digital Model and Dataflow

4.2.1 Introduction

In the first few experiments with Hops, it became rapidly evident that it is a work in progress system with bugs and the necessity of workarounds at the time writing. One of the first things that came up was the lack of support for data-trees as an access-type accustomed to from C# or Python scripting components in Grasshopper.

The lack of data-tree support or lists of lists as inputs adds some significant overhead to the process. Rhino.Compute is indeed capable of processing data-tree structure internally, simply if input as such it computes for each list as a standalone list where it is not possible to access members of another list in the same tree-structure.

Right now, at the time of writing, the user was limited to a workaround flattening the tree-structure and creating a pointer-list for each item, storing the paths in a new list. (Fig.: 23) The original data-structure can be rebuilt after calling Hops, before the



```
Panel
2377 {15;0}
2378 {15;0}
2379 {15;0}
2380 {15;0}
2381 {15;0}
```

```
Data with 1 branches
{0;0;0;0}          N = 2382
```

```
2376 Polyline Curve
2377 Polyline Curve
2378 Polyline Curve
2379 Polyline Curve
2380 Polyline Curve
2381 Polyline Curve
```

```
Data with 1 branches
{0;0;0;0}          N = 2382
```

```
52 Polyline Curve
53 Polyline Curve
54 Polyline Curve
55 Polyline Curve
56 Polyline Curve
57 Polyline Curve
```

```
Data with 36 branches
{0;0}        N = 92
{0;1}        N = 72
{0;2}        N = 37
{1;0}        N = 89
{1;1}        N = 82
{1;2}        N = 45
{2;0}        N = 91
{2;1}        N = 37
{3;0}        N = 89
{3;1}        N = 82
{3;2}        N = 45
{4;0}        N = 86
{4;1}        N = 74
{4;2}        N = 40
{5;0}        N = 88
{5;1}        N = 40
{6;0}        N = 51
{7;0}        N = 51
{8;0}        N = 89
{8;1}        N = 75
{8;2}        N = 38
{9;0}        N = 98
{9;1}        N = 42
```

Figure: 23 – Rebuilding the data structure inside the Hops-definition actual computation takes place, using a C#-script inside of the Hops-Definition.

Custom types are not fully supported yet. For now, one is basically limited to simple types like Points, Curves, Meshes, Text, Numbers and Boolean operators. More complex data must be serialized on input and de-serialized on output for example transferring the Karamba3D model, which costs some processing power. This probably won’t change because sending textual data to a server-based service for computation saves bandwidth. This practice increases transfer rate and, for sophisticated calculations, decreases calculation time by parallel computing on multiple machines.

Because of the nested list limitations of Hops, Speckle was mostly used in this scenario to send input to models with higher resolution. The models for fabrication data and precise analysis are typically models depending on a high level of detail.

Speckle simplifies collaboration with other people – and acts as a repository like GitHub, where data, once pushed, is easily accessible and trackable in terms of error detection at every stage of the planning process. In addition, Speckle provides interfaces, so-called connectors for different software packages. This facilitates the

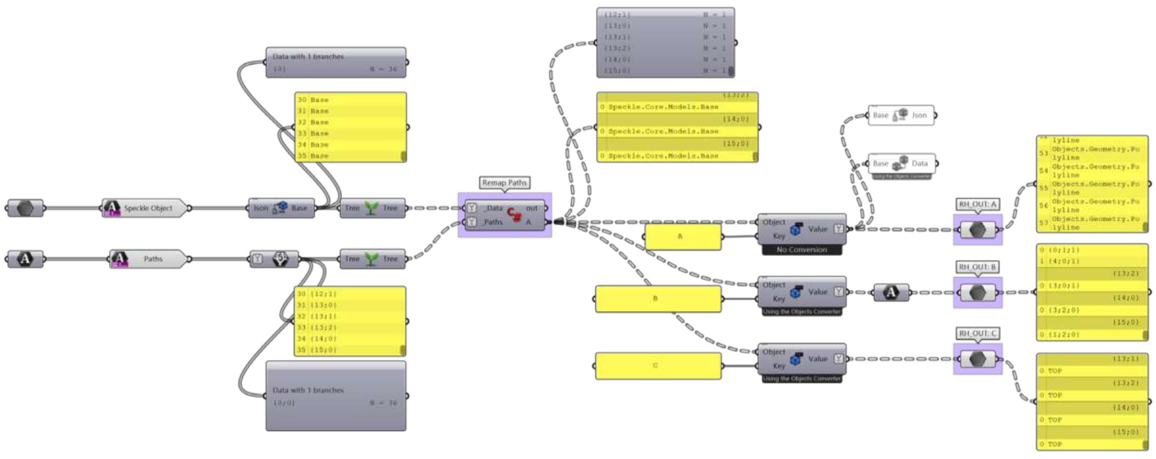


Figure: 24 – Hops, deconstructing Speckle-Object exchange of data and improves collaboration on a project between the companies involved, often requiring specialized software.

4.2.2 Modelling

For the purpose of implementing a prototypical multi-scalar workflow a generic NURBS-Surface was chosen. An regular diagonal grid pattern was defined on the surface using its local coordinate system to formulate the basic design intent. (Fig.: 25 & 26)

These roughly defined geometries represent the pitch, triggering a branched chain of downstream processes. Refining the coarse idea to a highly detailed parametric model providing infrastructure needed to allow the generation of information rich data objects.

The designed workflow consist altogether of twelve different interlinked models each fulfilling its highly specialised purpose. As a controlling body, the reference model acts as a supervising instance orchestrating computation and data flow from a single

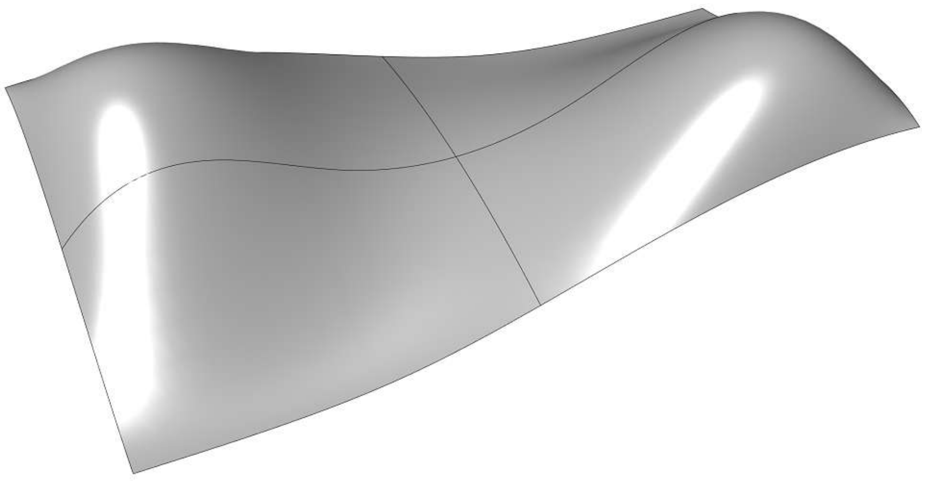


Figure: 25 – Reference Surface

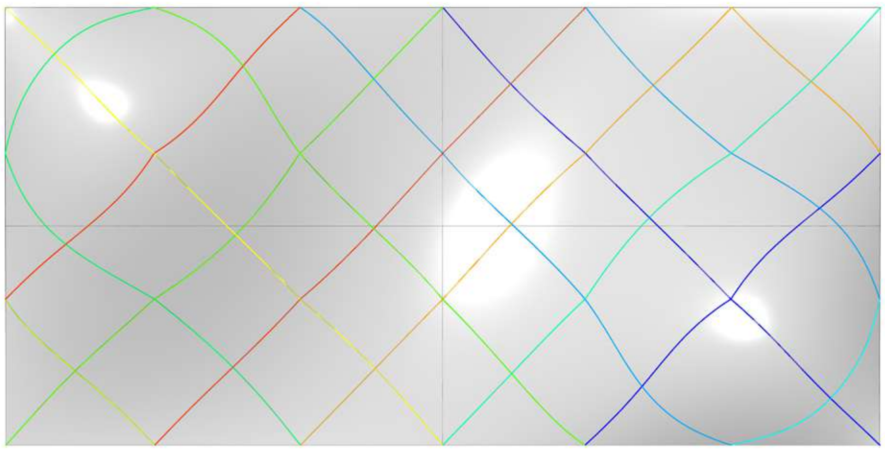


Figure: 26 – Generated diagonal grid

graphical user interface providing overview and control on a macro scale. The user can redirect dataflow and trigger recomputation of certain parts allowing for up-to-date information but also for reverting changes to an earlier state of the project.

It also includes and distributes the design intent and each iteration and refinement step making it way easier to comprehend the decisions made in the course of the project. Because of that, it offers a swift method for diagnostics identifying flaws in design and model architecture. Being able to detect where and why a procedure fails and seamlessly exchange faulty methods facilitates the process of handling complex modelling tasks in an holistically robust manner. Within each model the user keeps control over certain parameter which are of no concern from a macro point of view.

The first step after beeing confronted with the design intent would be optimizing the curves on the surface and defining beam axis as well as basic connectivity information. This is handled by the first model rationalizing, optimizing and analysing the grid extracting key data at a low level.

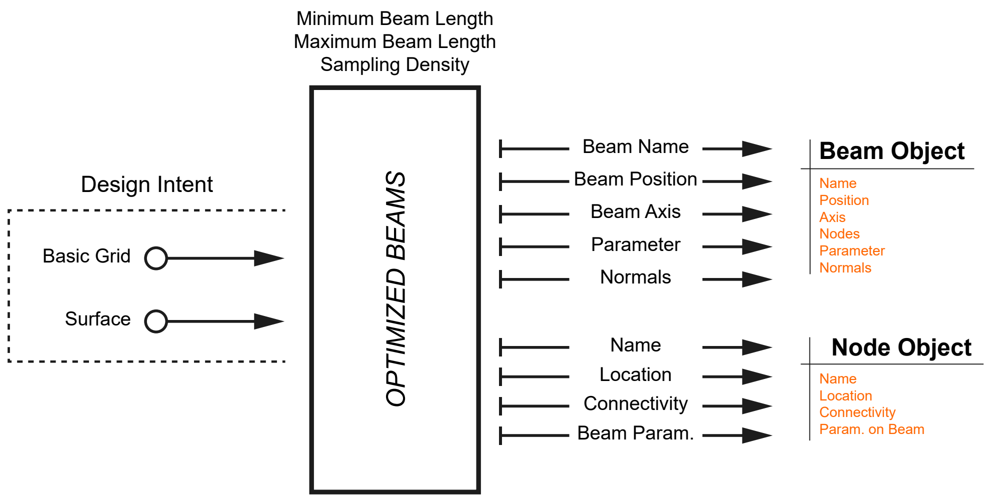


Figure: 27 – Dataflow Defining Beams

4.2.2.1 Definition of Beams

The first model generates the basic beam data on the surface. By using the coarse grid geometry as input. It defines important features of the to be modelled final product. such as minimum and maximum beam length, appearance of the grid, assembly sequence, etc. ... and generating basic information describing each beam.

Figure 27 illustrates the in- and outputs of this model, feeding a Beam and a Node Object which collects necessary data for the description of the end-product and processes down stream. Base datatype for the Beam and Node object is the generic “Speckle Base” object. This generic datatype allows for storing all types of data in a custom user-defined hierachy alleviating how this data is referenced.

At this stage the design intent, consisting of the reference surface and a rough diagonal line grid, is the only input available for processing. Inside of this model the data stream gets split up in concurrent running tasks each computing parts of information required in upcoming models and extending the Beam and Node object.

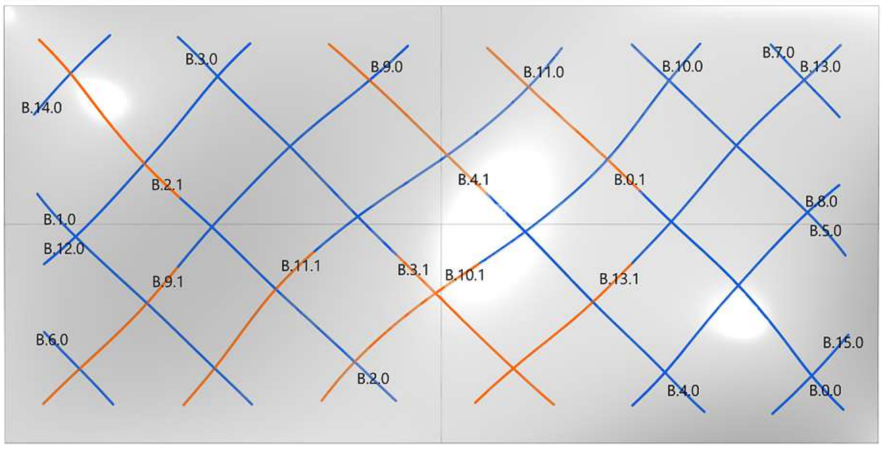


Figure: 28 – Basic beam data. Equal length optimized.

Rationalizing the grid is the first step. Using particle spring relaxation, the lengths of the segments are iteratively adjusted until an equilibrium of same length curves is reached (Fix.: 28). Fulfilling different constraints such as regions where changes shall be kept to a minimum, anchor points and tolerances.

Next, the curves get split multiple times before they are again joined to form the beam axis, constraint to a minimum and a maximum beam length. After that the axes get sampled and the normal orientation corresponding to the surface at each sample point is evaluated.

Parallel, the assembly sequence was defined using an assumed assembly pattern and each beam named accordingly. Subsequent nodes and their connectivity are computed to determine the type of joint. Where no connections are allocated, an invalid placeholder item is added. (Fig.: 29) At the end, generated information is pushed to the repository and issued to the reference model.

Nodes and Beams are separated Speckle objects, referencing each other and can be used to query data from each other, provided there is a corresponding node for that

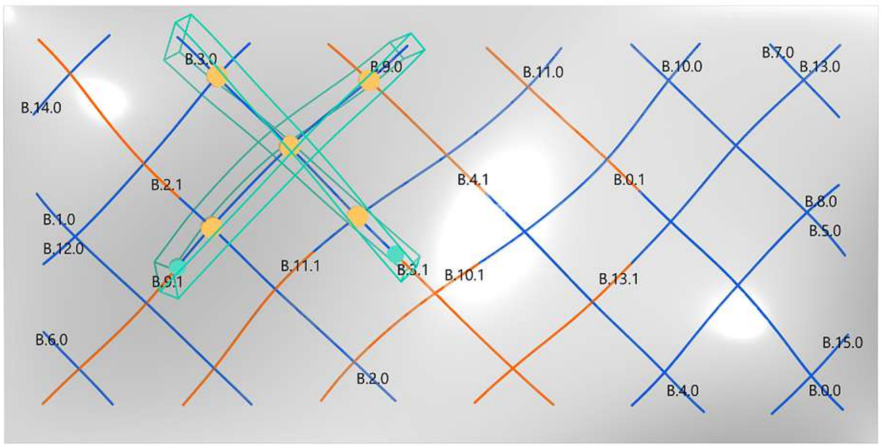


Figure: 29 – Nodes and connectivity of beams. (B.3.0, B.9.0)

beam or vice versa.

With this basic geometric data set, the next steps compute semi-asynchronous with some models waiting for data of a previous models before computing. Next chapter describes a simple structural analysis done using Karamba3D within Grasshopper.

4.2.2.2 Structural Analysis

After defining basic geometry as well as its connectivity the structural analysis using Karamba3D is called.

The goal is to evaluate the structural performance and optimize cross section height of each beam. By trying to maximize structural integrity and minimizing material demand, an efficient timber gridshell is developed.

The model retrieves multiple parameter from the Beam and Node Objects: beam name and corresponding nodes are used to reference to the Node Object defining what type of joint is at given position on a beam for the structural analysis. Normal

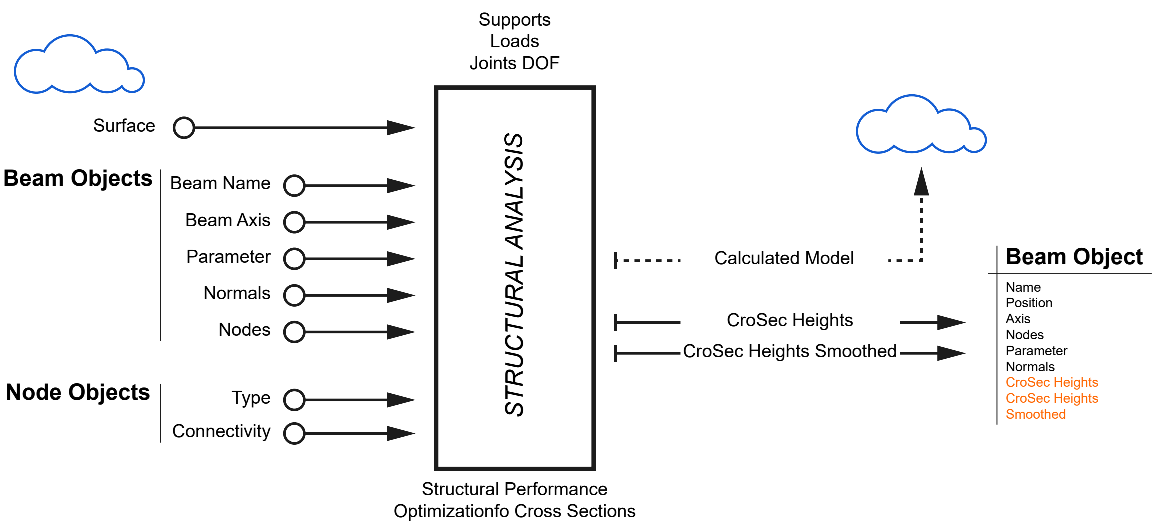


Figure: 30 – Dataflow Structural Analysis

vectors define the orientation of the beam at precomputed parameters along the beams axis.

Return value is the calculated structural model and cross section heights per parameter, smoothly interpolated across the surface as well as rough stepped values for each beam segment. The resulting model gets serialized and store in the Github repository, making it accessible without the need for recomputation. Cross section values were appended to the beam object for immediate access if needed.

For computation, the beam axes get split again into parsable segments and grouped. Grouping the segments enables cross section optimization for the whole beam instead of just a single segment. This way the largest axial forces of all beam segments are considered.

An arbitrary point load is applied near the surfaces center. Using the results from the analysis, regions of different cross section heights were introduced. These regions allowed a smooth design for the heights of the blank beams. Cross section values at the sample points of each beam were collected and stored in designated tables.

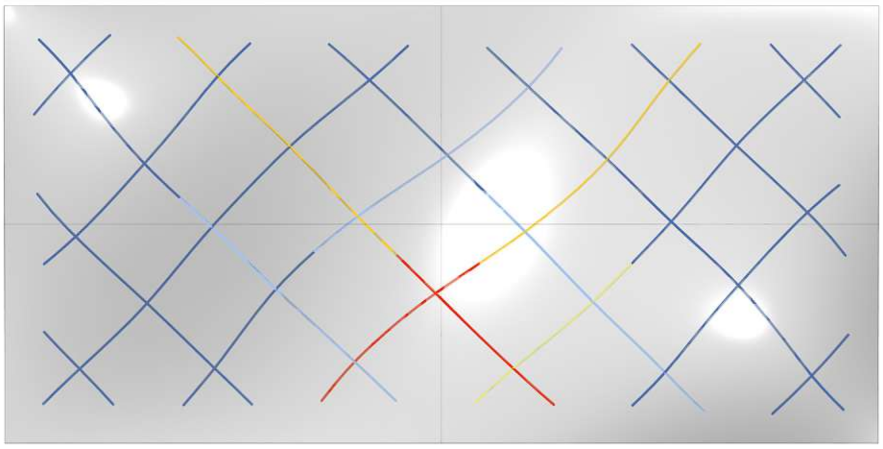


Figure: 31 – Sampling of beams according to cross section heights.

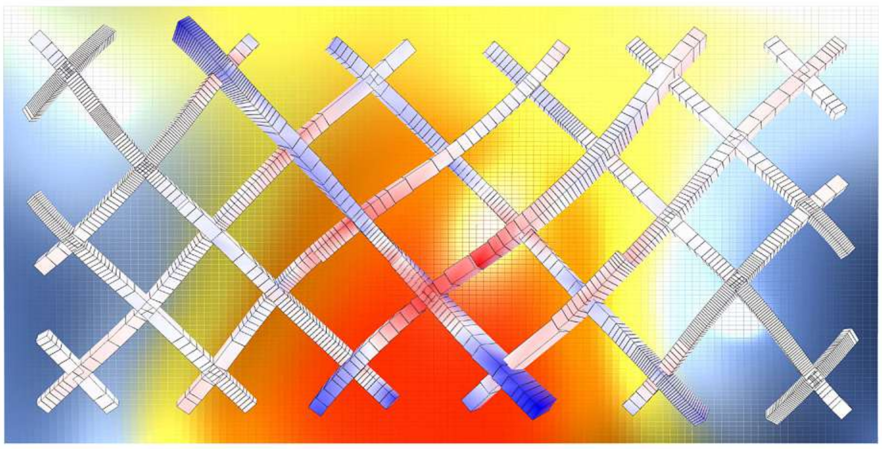


Figure: 32 – Interpolating values over base surface.

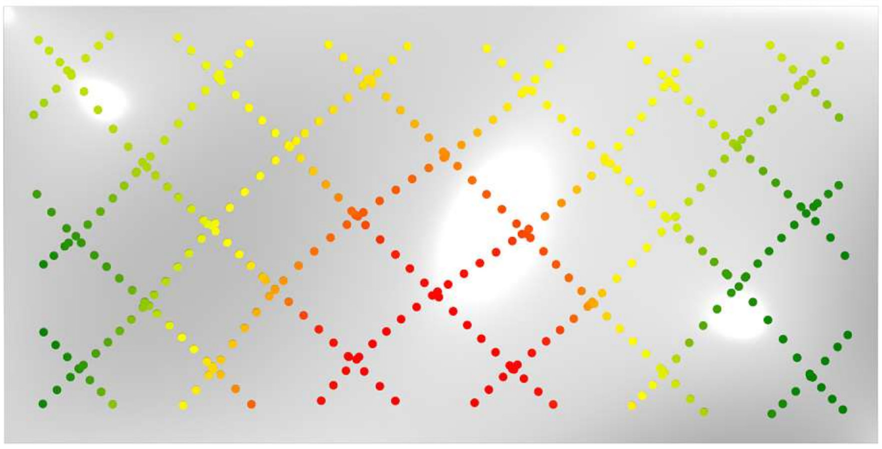


Figure: 33 – Final distribution of cross section heights.

The use of different cross section heights has the advantage of developing a more resource efficient structural system by only using material where it is actually needed. In comparison to applying the same cross section over the whole structure, a reduction of 26.8% in material requirement is achieved simply by adjusting the height of the beams. Further, limited to this arbitrary load case, consisting only of a singular point load and gravity, the weight reduction amounts to 60,4%.

4.2.2.3 Generating Blanks

This model generates on the one hand the beam geometry with all its necessary information, and on the other the blank geometry describing the raw volume of wood from which the beam geometry will be milled from.

"Once the beam segmentation is set, blanks are created. The term 'blank' refers to the raw glue-laminated timber piece, which is milled down to the final shape of the segment. In free form projects, single curved or doubly curved blanks approximate the final piece's curvature. They are produced by bending and gluing raw lamellas on a set of pre-shaped supports."[35]

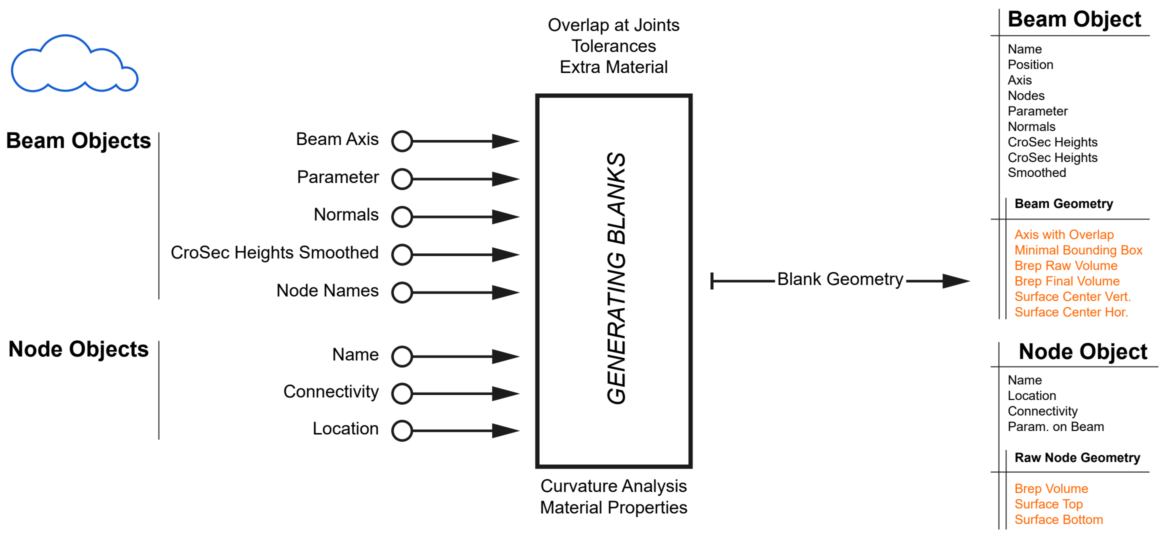


Figure: 34 – Dataflow Generating Blanks

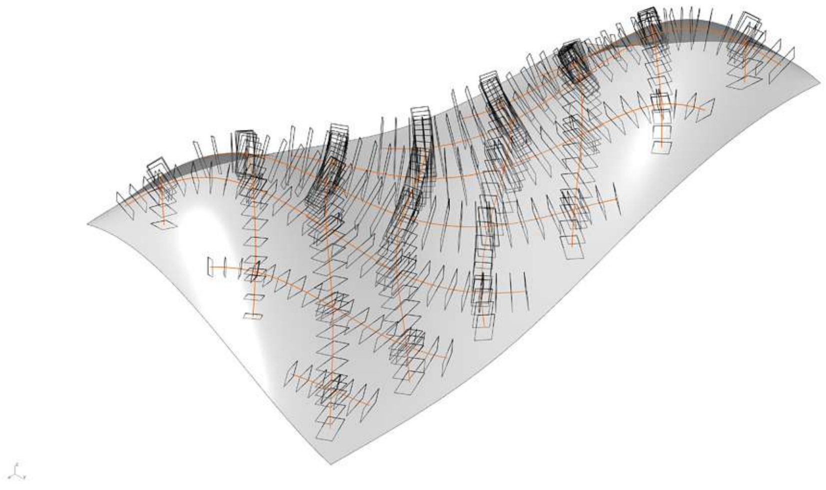


Figure: 35 – Cross Section profile at sample parameters

As part of this process the axis of each beam got overwritten. This is done to extend the curve and create overlapping regions for end joints. Each individual overlapping distance can be input and hereby generate stronger and materially optimized end connections.

Needed cross section heights, previously calculated using Karamba3D, are then used to generate raw beam geometry by lofting rectangular profile curves along the newly extended axis.

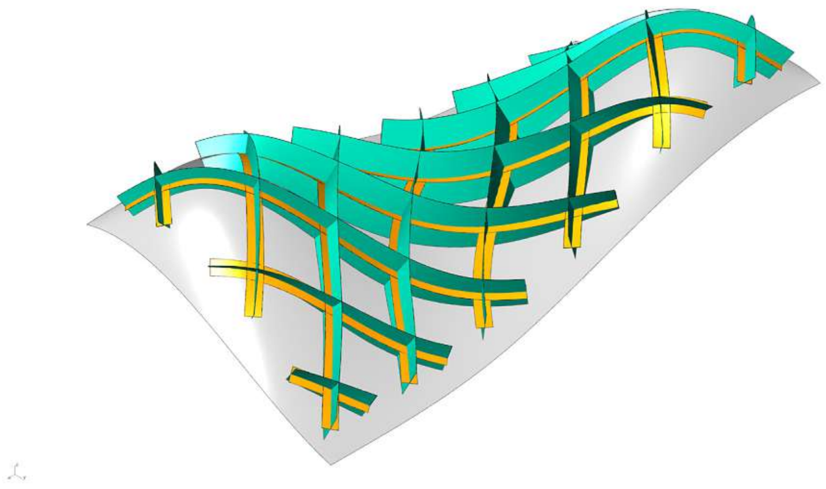


Figure: 36 – Beam Center Surfaces

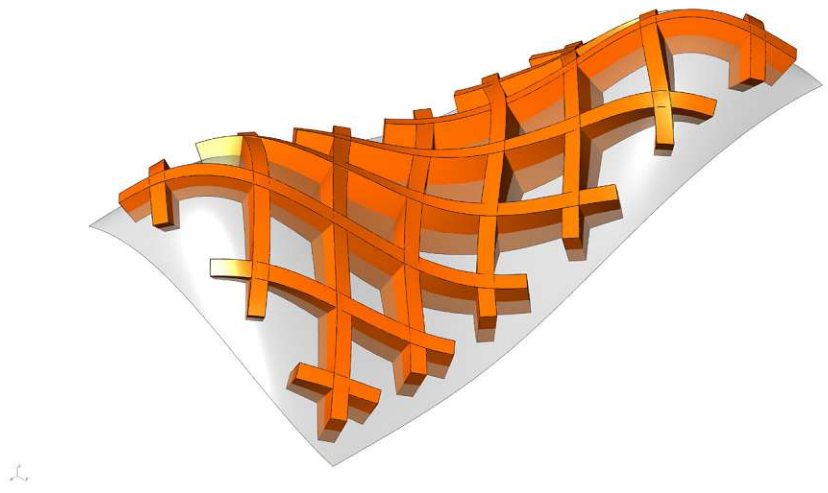


Figure: 37 – Beam Blanks on reference Surface.

Simultaneously, surfaces needed for the next steps – like trimmed top and bottom surfaces for generating the end joints and horizontal as well as vertical surfaces alignd with the center of each beam – are created.

A simple curvature analysis is executed to evaluate overall geometric performance and buildability of given grid. Extra effort in minimizing waste, processing time at the fabrication stage and increasing beam strength by optimizing blank distribution and size with respect to its torsion and curvature was not done because it would not contribute to the current goal of this study and would have unnecessarily increase complexity.

By having set the raw description of the basic beam geometries it is now possible to proceed with adding details to the structure.

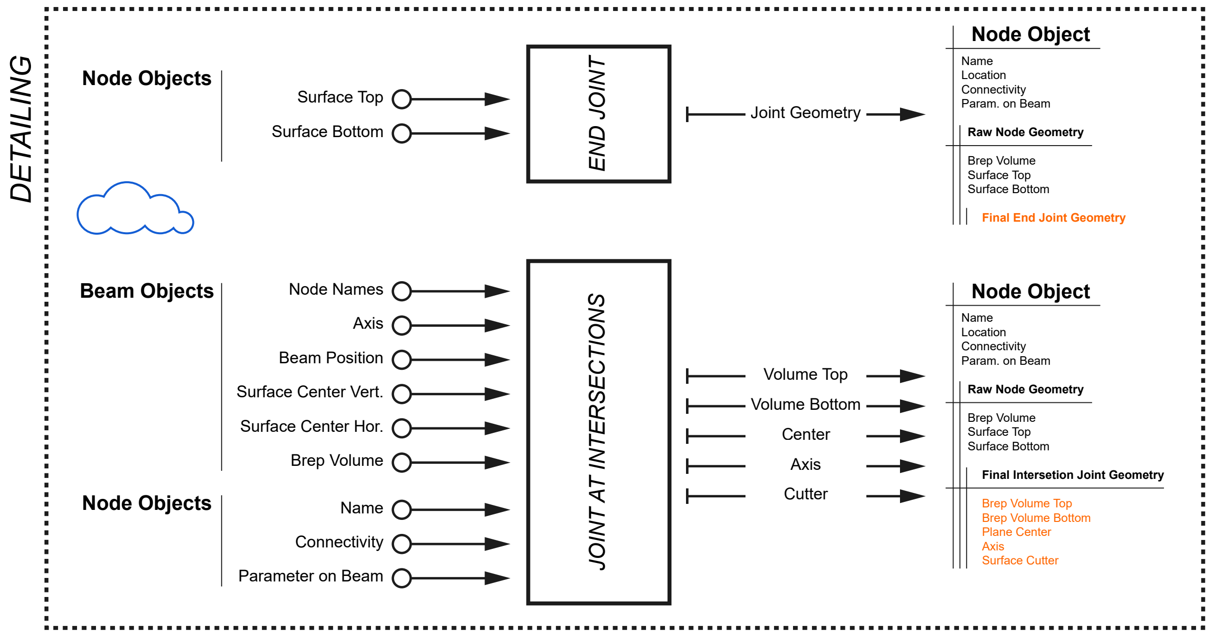


Figure: 38 – Dataflow Detailing

4.2.2.4 DETAILING

In this section the geometric information for the joint connections is added.

Simultaneously end-joints and joints at intersections are generated. These definitions are triggered when changes concerning the blanks are present and directly extract needed information from the Speckle repository as well as updating existing Beam Objects.

Joint at Ends

For modelling joints at each beam´s ends, top and bottom surface of the raw blank is needed. It was possible to define how much overlap and corresponding touching surface area is generated. This was done by defining desired distance on the beam axis and trimming corresponding surfaces.

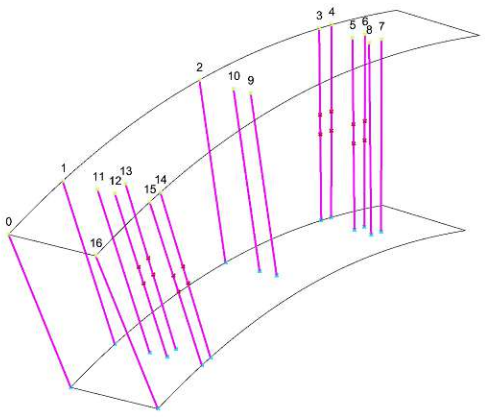


Figure: 39 – UV-Points on joints reference surfaces.

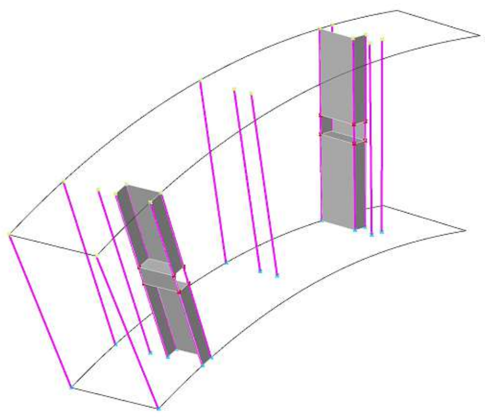


Figure: 40 – Construction of joint geometry.

A traditional Japanese joint, the Kanawa-Tsugi, was chosen. It is a mortised rabbeted oblique scarf joint only tightened by inserting a pin in the opening at the joint center. [23]

Even though this joint would be hard to realise for automated production or assembly because of the needed undercuts and the free form nature of the surface limiting the range of motion during assembly, it was chosen to add complexity to the model and overserve performance of the logic.

Using the input surfaces UV-coordinates, points were generated on the surfaces drawing the outline. Making use of a parametrically defined logic added the possibility to change proportions of the joints on demand. Points are then used to draw lines between the surfaces and construct the three dimensional geometry the joint is composed of. Each joint is built individually with oriented loft surfaces and joined forming a watertight closed BRep*. At each step scripts perfrom checks to validate the generated geometry and ensure no errors appear. Otherwise an exception will be thrown and the user is alert and can intervene in the procedure.

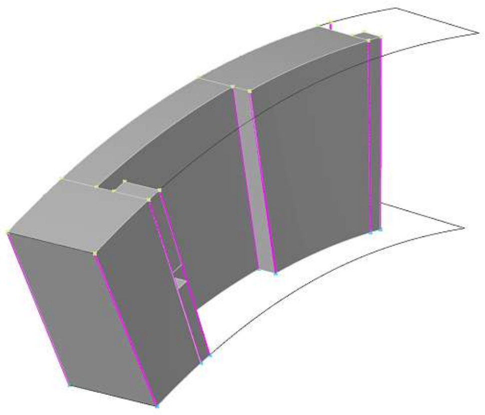


Figure: 41 – Assembled joint.

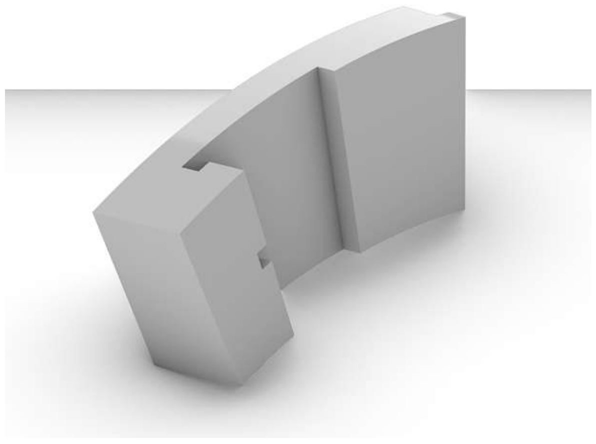


Figure: 42 – Final end joint.

Using connectivity information of the nodes it was possible to link generated geometry to associated beams. Orientation of the joint is predefined by the orientation of the input surfaces so it is impossible that an end part is computed at the beginning of the beam or vice versa. Geometric information about the final beam is then again transferred to the cloud based repository where everyone on the team has access and versioning is enabled.

Joint at Intersections

The definition handling joint information at intersecting beams uses the horizontal and vertical center surfaces of the beam. Also, the connectivity information of the nodes and calculates the cut-away geometry for the defined node geometry. Here the logic for a simple tapered half-lap joint was parametrically modelled and applied.

At first the line intersection of crossing vertical center surfaces was calculated where the normal orientation of the joint and its centroid had been evaluated. From there the full volume enclosing the joint is generated.

Using this twisted box shaped volume, the vertical faces are used to draw the profile curve for the tapered half-lap joint in local coordinate space. Making the profile adapt to free-form geometry. The negative geometry of the joint was generated by cutting the raw joint volume with the extruded profile curve.

Finally, the joint gets "carved" into the beam using a boolean operations, if one wants to visualize the outcome. Otherwise, toolpaths can be generated instead and time computing, potentially a vast number of joints, could be saved.

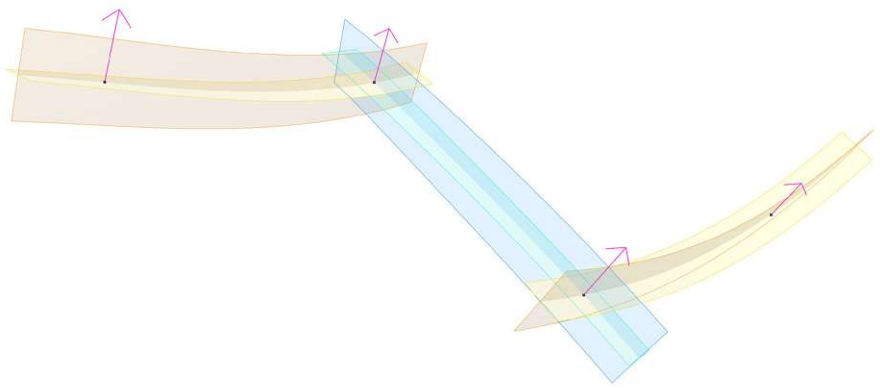


Figure: 43 – Joint orientation.

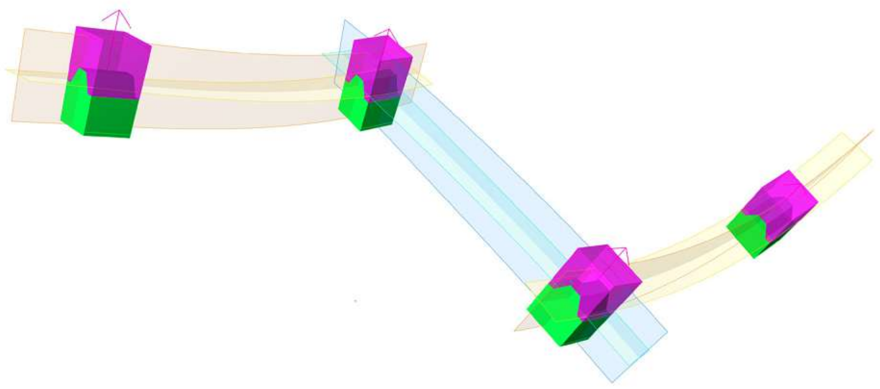


Figure: 44 – Joint volumes.

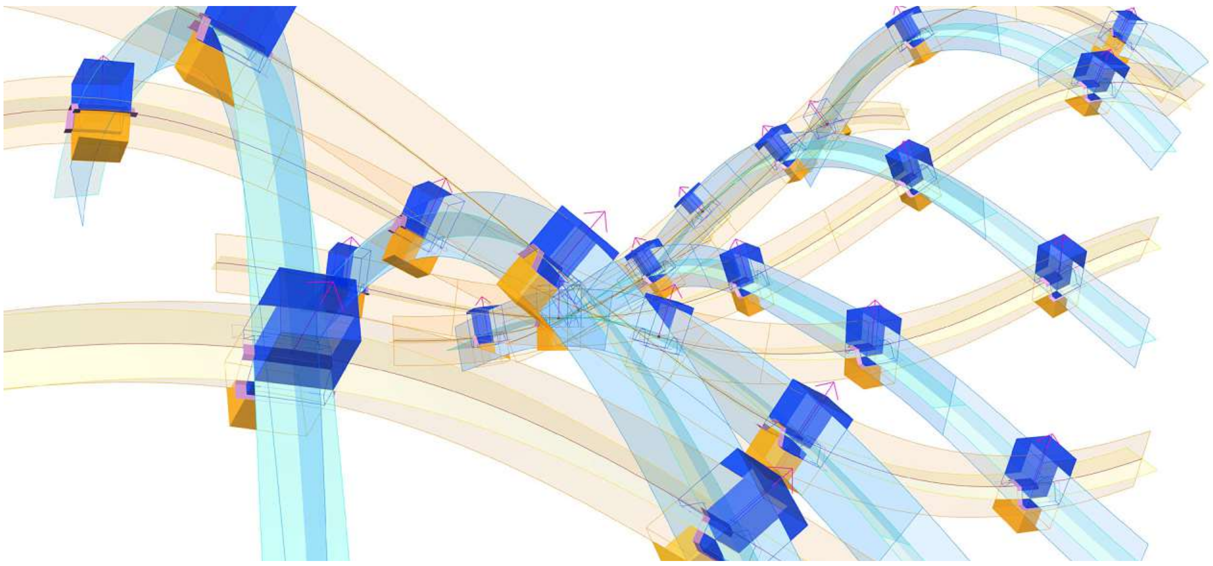


Figure: 47 – Joint geometry.

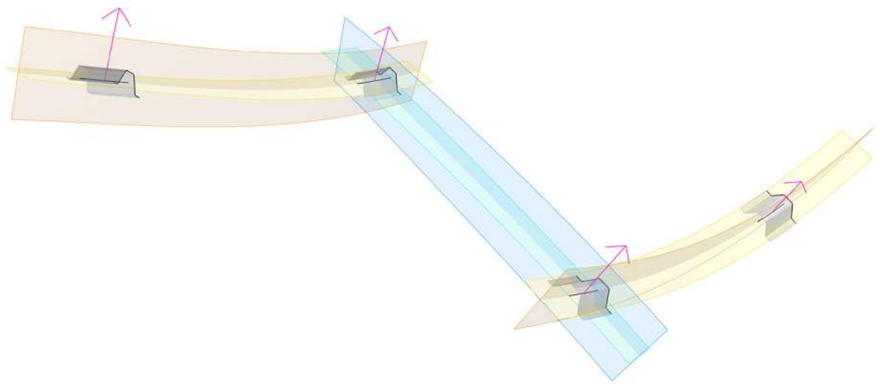


Figure: 45 – Joint profile.

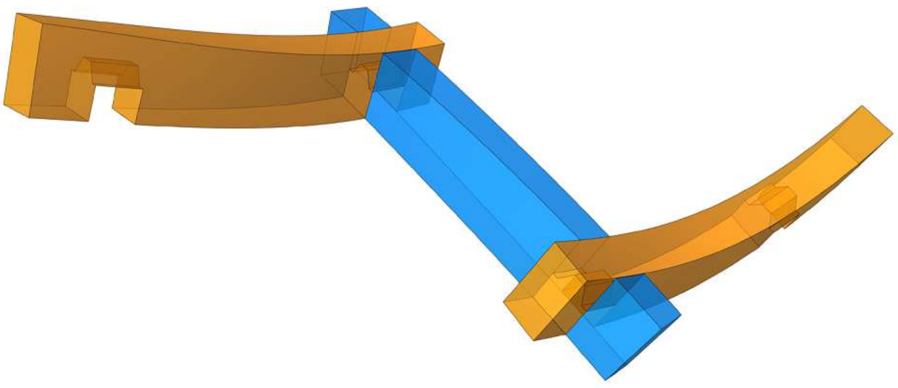


Figure: 46 – Joints booled from beam geometry.

4.2.2.5 Generating Data for Fabrication

With every information in place, it was now possible to compute information needed for fabrication or directly compute toolpaths, layouts and massing of demanded material. For this purpose, another Speckle Object - BeamFabrication Object - was introduced, storing fabrication data which can be referenced with the Beam and Node Object.

This chapter describes the approach to massing, formwork and a prototypical approach of developing toolpath planing strategies for the fabrication on the robotic manipulator. Retrieving geometric Beam information - blanks, final beam, final surface, etc. - and joint information from the Node Object - describing volume, enclosing surfaces, etc. - data rich information down to 0.1mm was generated and overloaded into to the BeamFabrication Object and pushed to the cloud service. From there the user can retrieve needed information to monitor, modify or fabricate certain components. For each step of the fabrication, information can be queried and used more or less directly without the need for further modification.

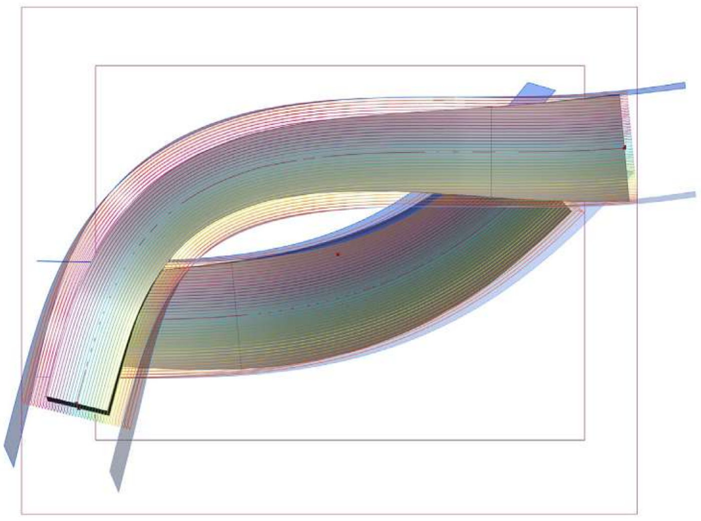


Figure: 48 – Boundary volume, layer and boundary surfaces of beam blanks.

Formwork

At this stage, because it was not quite clear what kind of machinery would be available - two different ideas for the formwork were developed.

The first being, using XPS-blocks as mold on bottom and top of the blank to be able to press down and glue together individual layers. Those negatives could haven been cut using a CNC styrocutter following the ruled top and bottom surfaces of the raw beam blanks. Major concern using XPS or similar materials was, if the material would have been sturdy enough to withstand the potentially high and inhomogeneous distributed pressure needed to shape multiple layers. On the other hand the upside of quick fabrication, by only needing to make one cut, seemed advantageous.

Second idea was to describe the negative of the blanks using multiple layers of individually CNC milled MDF-blocks following the bottom surface. Thereby, it would have been possible to use a simple 3-axis machine to approximate the curvature. Layerheight was maximised regarding the on-hand machines workspace, leaving 5mm tolerance to the hard z-axis end-stop. The layers were optimized to be the same

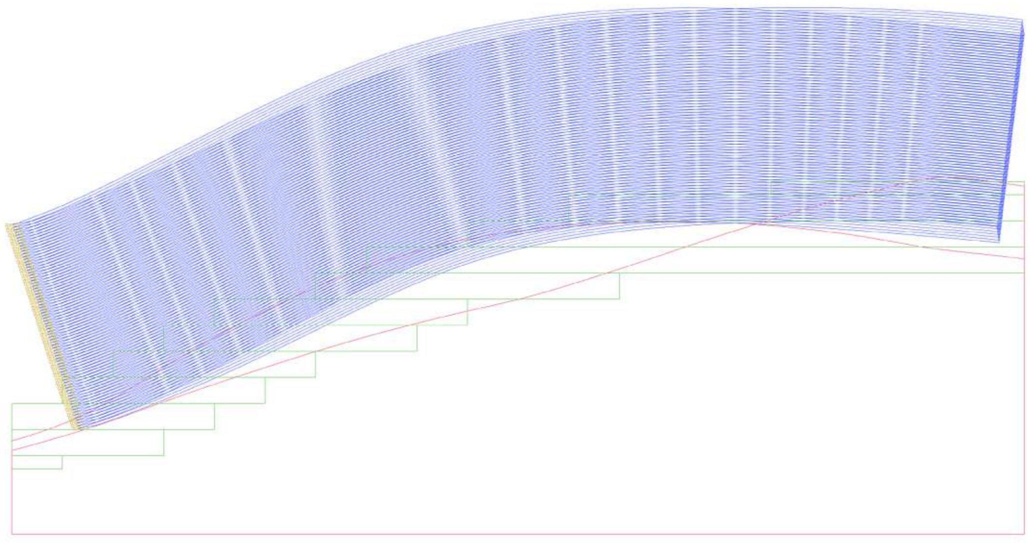


Figure: 49 – Layers for one beam (blue); MDF blocks for Formwork (green).

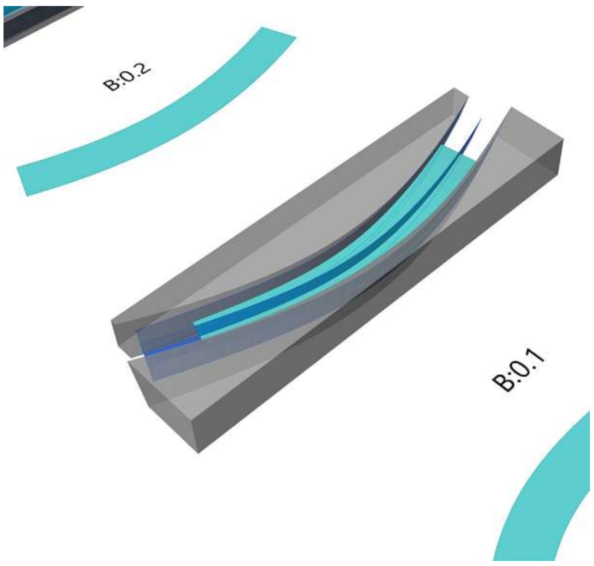


Figure: 50 – XPS mold.

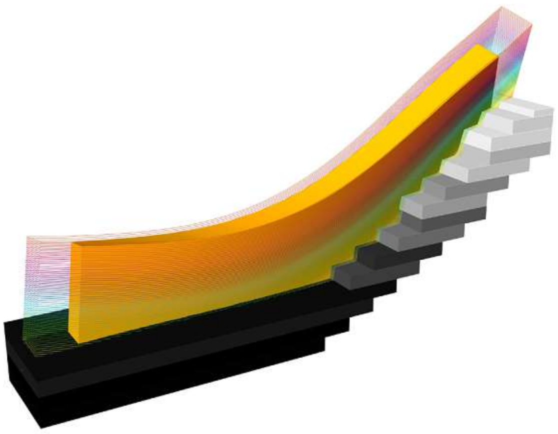


Figure: 51 – MDF mold.

width and only cover areas necessary to describe the beam geometry. Each MDF-layer represents one milling task. By assembling all corresponding elements one would have got a single mold. However, this would have significantly increased material demand and lengthened overall time spent on fabrication slowly milling needed formwork.

Both ideas were included in the BeamFabrication Object albeit later omitted as described in Chapter 4.4.1.

Massing

Each beam was divided in 4mm height layers each represented as a developable ruled surface. These surfaces were unrolled resulting in plane boundary curves of said surfaces. By doing so, it is possible to cut each layer from a flat sheet of wood using a conventional 3-axis CNC milling machine. Further, having the flat layer-surface already following one direction of the beams, reduces strength needed when gluing and ‘assembling‘ beam blanks.

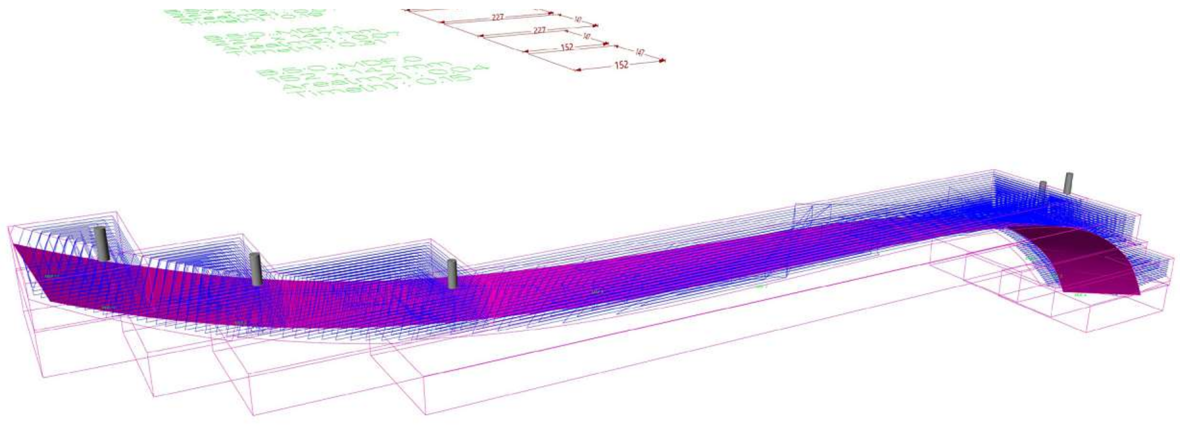


Figure: 52 – MDF mold: toolpaths.

After surface flattening, each flat layer was named according to its sequence position during gluing and optionally sorted along the x-axis for validation. By nesting the beam layers onto available plywood sheets, production ready toolpaths were generated as well as needed number of plywood sheets was retrieved.

Toolpaths

First strategies for planning toolpaths were developed. By using primitive geometric parts of the Beam Object different strategies were implemented in Grasshopper. Namely, four operations for different stages of refinement of the blanks were chosen and further tested in Chapter 4.3. By extracting surfaces and generally BReps together with describing curves strategies for drilling holes, roughing and finishing were implemented.

For drilling holes, helical ramping was chosen. By cutting along a helical arc the cutter is slowly driven into the material until a certain depth is reached. After the last helical cut a circular interpolation was used to clean up residual material. In contrast to just diving straight in, the cutting tool engages all three axes, distributing the cutting forces across these. Also the cutter can be flat because using small helical motions the cutting is done in by its sides. [27]

Fine cut or surface finish was implemented using a zig-zag pattern generated using the desired face surface of a beam. Because each finished beam surface is curved the cutter needs to be aligned with the surface curvature at each step. With this and the limitation of only having a flathead cutter the curvature needs to be compensated with a offset of the cutter calculated using simple trigonometry. In respect to diameter of the tool and the angle it is oriented to the surface a good approximation of desired surface is achieved as well as a nice finish

For roughing the basic shape of the beam two strategies were realized. One being the simplest toolpath strategy - constant stepover. The volume to be removed is divided in plane surfaces orthogonally to the z-axis separated by a constant z-step or plunge depth value. Using a constant value for stepover, in this case 1/4 of the cutter diameter, contours of the surfaces are offset parallel to each other to clear one layer of

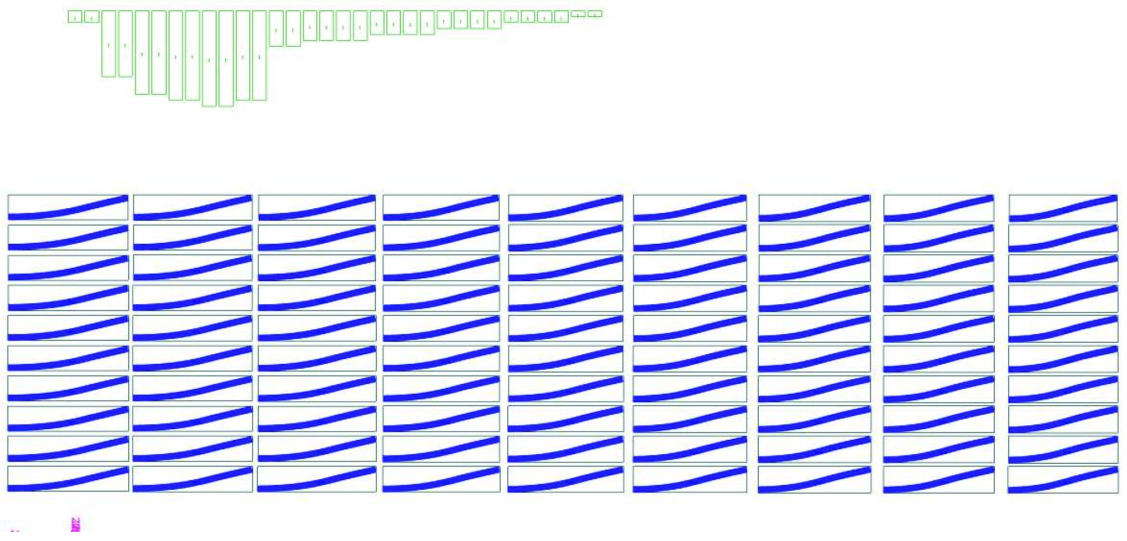


Figure: 53 – Evaluation of material demand.

material from the raw blanks. This 2.5D milling technic is a conventional approach to clear large amounts of material.

The second approach was a combination of the finishing technic and previous roughing strategy. Instead of using surfaces orthogonal to the z-axis, offset surfaces were generated from the desired finished surface. Offset with constant z-steps the blank volume was filled. By using the raster toolpaths from the fine cutting strategy, each layer was cleared in a 3-dimensional way, orienting the cutter with the surface normal at each step. This approach was thought to be a two in one toolpath finishing the surface directly at the end of roughing.

Further, feedrates were ‘dynamically’ adressed, allowing slow down of the tool at corner points. Limiting forces on the cutter at these points trying to keep an accurate result as well as not damaging the cutter, spindle or robot. [4][27]

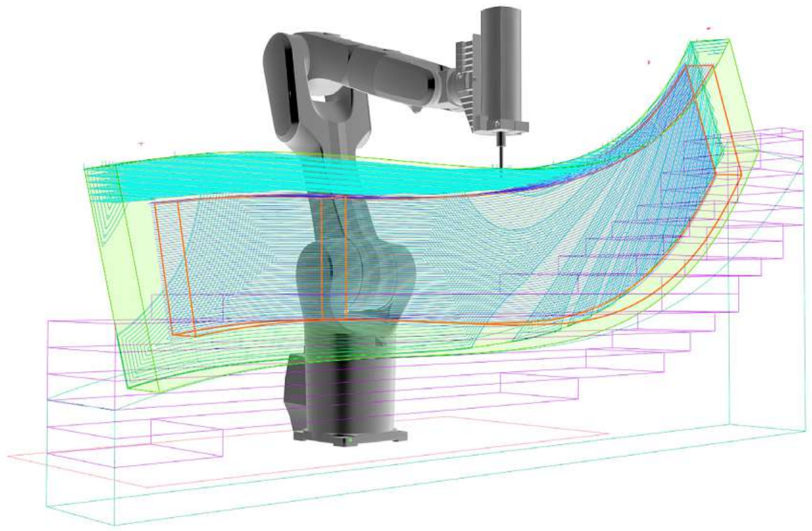


Figure: 54 – Prototypical toolpaths using Grasshopper.

4.2.3 Conclusion

This experiment demonstrates the implementation of multi-scalar modelling using multi-layered data objects shared by remote services. The biggest benefit of using this method is the exchangeability of every model. By the time a model changes it can be seamlessly replaced with the updated version making the process more consistent while keeping information up to date. As the model gets changed and published, a notice including the description of the modifications will also be accessible. Therefore, each stakeholder can work with the latest information without the need for consultation of previous changes. The multi-scalar modelling process for robotic fabrication is further extended in Chapter 3.4 Robotic Fabrication of Doubly Curved Beams.

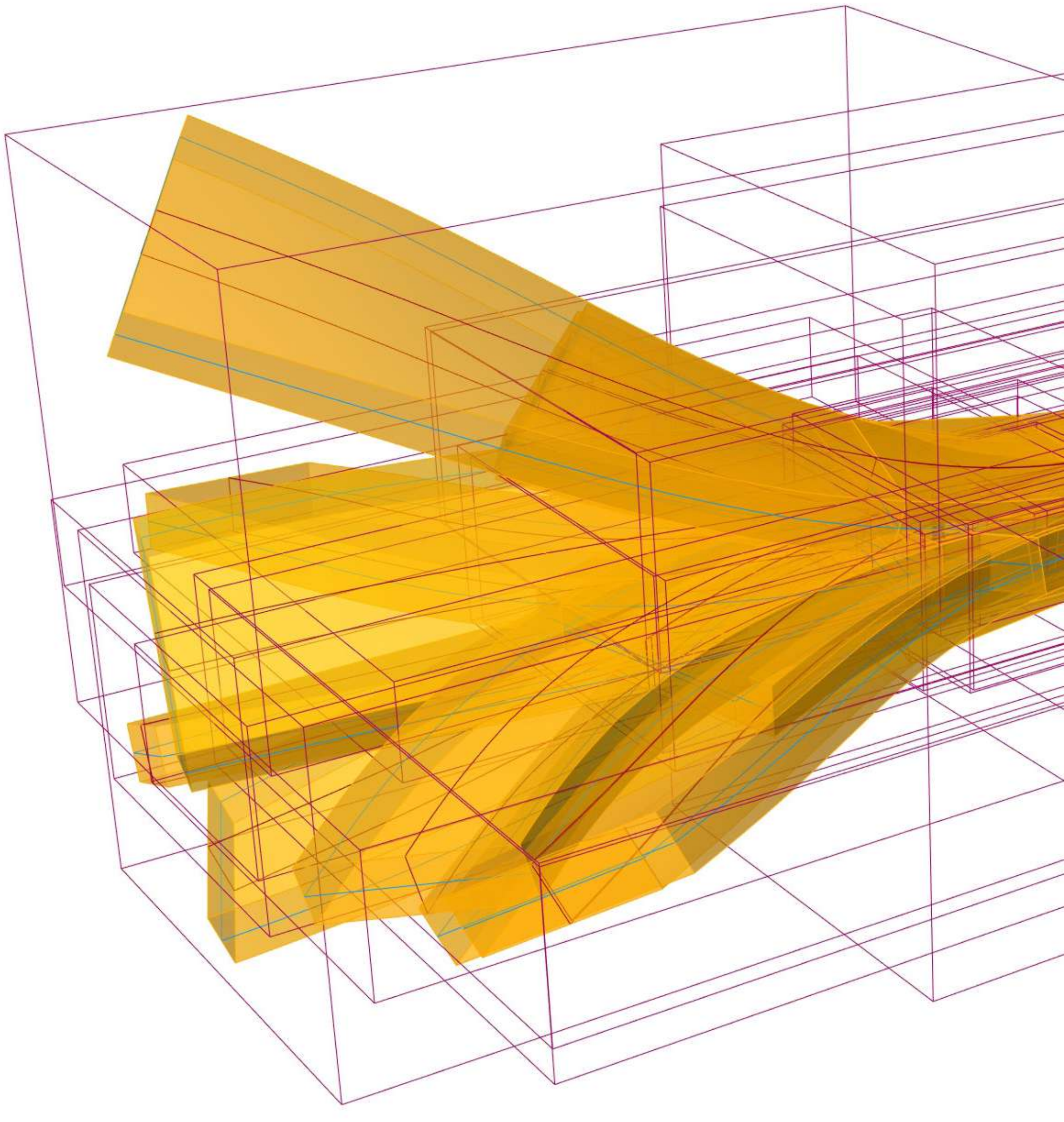
One of the shortcomings in this experiment were the limitations of the tools, at time writing. Because of their 'Work In Progress' status the user was confronted with some bug, performance issues and missing basic features. Alternative solutions and workarounds had to be developed in order to be able to make use of the full potential. Also, the basic Speckle Object seemed to have issues with multiple layers of information while excessive reading and writing of large data, which caused crashes of the software from time to time.

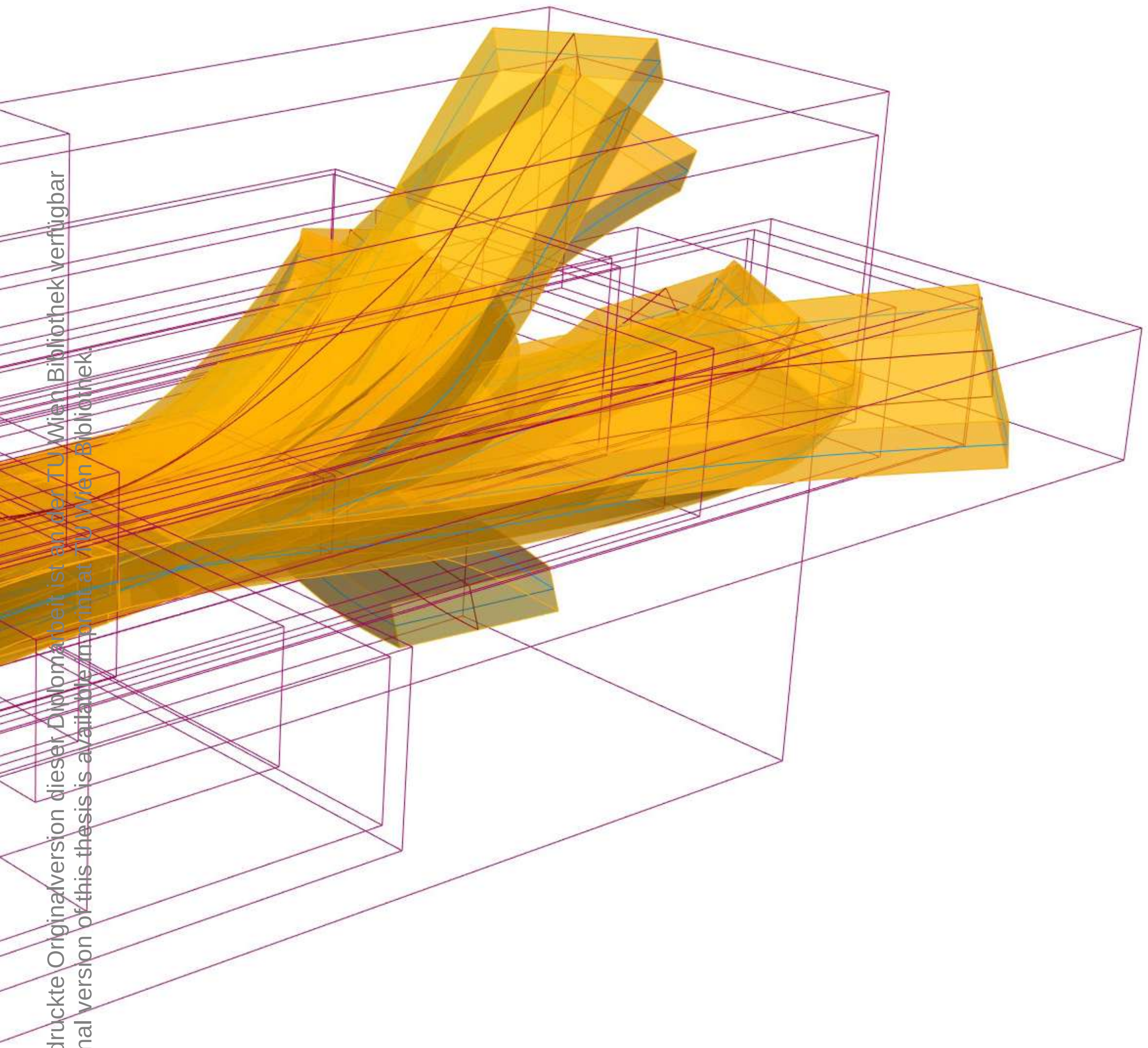
Managing data over the course of a project is a hard endeavour which requires already established expertise of processes downstream. During development of a project the need for changes during modelling or between stages happen on a regular basis. Switching back and forth implementing changes to previous models restructuring the data objects and extending generated information to the needs of the upcoming models and processes is effortless using this approach. But in attempt to minimize refactoring of data structures it is necessary to know early in a project what kind of information is needed at what state of the project. For this purpose, a coordinator similar to a software architect or BIM manager is needed, responsible for design choices related to the overall system/model structure.

"The software architecture of a system represents the design decisions related to overall system structure and behaviour. Architecture helps stakeholders understand and analyse how the system will achieve essential qualities such as modifiability, availability, and security." [26]

Partly generalization of common methods, models and datatypes would be beneficial. For this, the creation of open-source guidelines and standards are mandatory to ensure fast and interdisciplinary development for the industry.

Another important finding was that automation is limited to certain models or parts of a model. Further, it is time consuming trying to define logic robust enough to handle most uses cases, given the complexity of architectural projects. Knowing this, it gets more important to provide good solutions for handling exceptions. Here small C#-scripts detected when operations failed and pushed notifications about the problem. This way it was relatively easy tracking faulty parts of models. However, one must notice that this was a small project and still manageable by one person, compared to real building projects. As complexity increases more sophisticated solutions are essential to a robust process.





4.3 Fabrication of the Kawai Tsugite

4.3.1 Introduction

During the first tests in robotic milling for the doubly curved beams, Peter Bauer (Werkraum Ingenieure ZT-GmbH) asked to fabricate a POC of a generalised Japanese wood joint they were working on. In its original form, the joint is a so-called Kawai Tsugite (Fig.: 55). This joint uses rotational symmetry of a cube allowing to be assembled every 120°. With the original design, it is possible to connect joint to joint in two directions – aligned straight or right angled.

The given Grasshopper script can utilize the principles of the Kawai Tsugite in a way that a higher variety of angles can be achieved. (Fig.: 56) An odd number of faces is required for the joint to work. By using the parametric model a variety of small joint prototypes with different settings were 3D-printed to validate their functionality and the geometry generated.

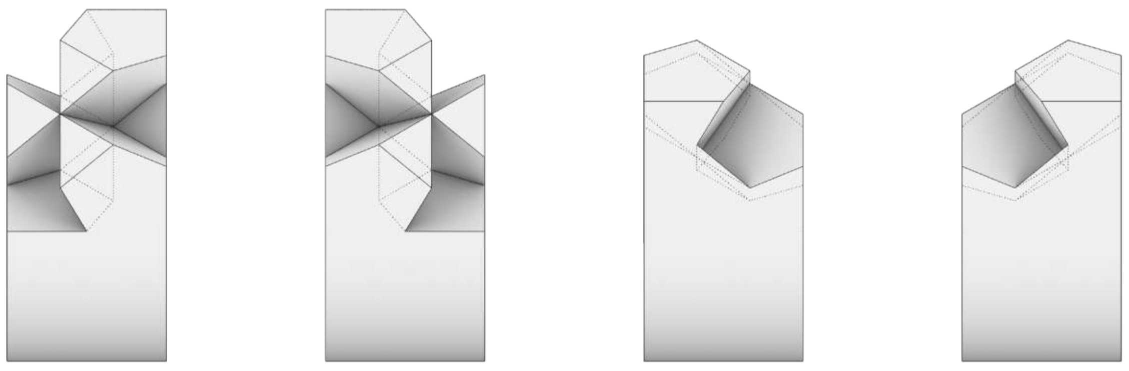


Figure: 55 – Original Kawai Tsugite

4.3.2 Setup

Aim of this experiment was, besides fabricating a proof of concept (POC), to learn about limitations of the tools at hand and how to approach robotic milling using custom pathing strategies. Because of the experimental nature of the robotic setup some constraints were given at this instant of time. First and foremost, the spindle holder was 3D-printed using PET-G filament which works well but was not sturdy enough to compensate for vibrations and forces applied when moving the tool fast through wood. Also, there was a good chance to bend or break the plastic at the weak spots. Therefore, moderate velocity was chosen to start with, then slowly increased which was expected to have an impact on the accuracy.

Using an cutter with 8mm in diameter, maximum plunge depth was limited by the length of the cutting edge at 45mm deep. Two types of wood were used for the tests – pine and oak. The raw wood beams were cut into multiple pieces with a length of 250mm. Because of varying cross-section dimensions (Fig.: 57), digitally an idealized cross-section of 120x120mm was assumed for the oak wood. The pine wood was more accurate with dimensions 98x98mm.

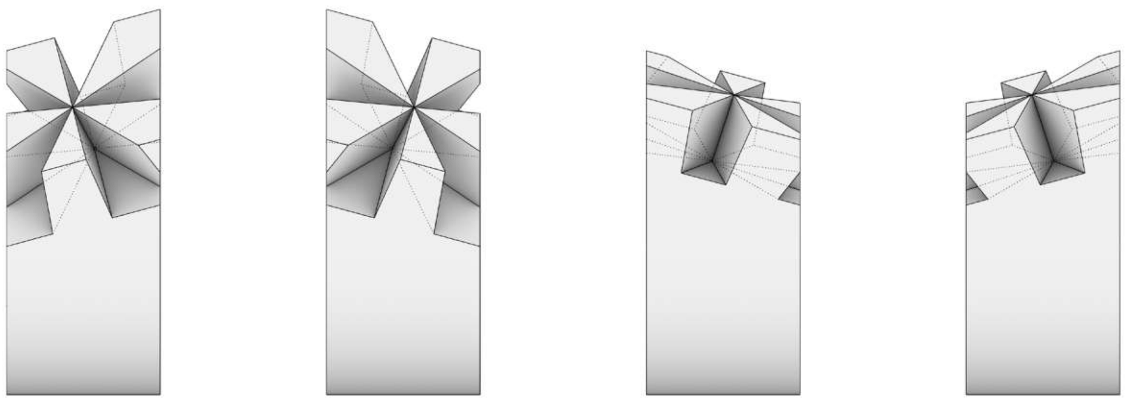


Figure: 56 – Generalized Kawai Tsugite

4.3.3 Fabrication

One of the more finicky problems in digital fabrication is to match reality with the digital model and vice versa. To synchronize the physical workpiece in the digital model, four prominent features were located using the coordinate space of the robot. These points can be anywhere on the blank as long as they don't change position and can be easily approached. Here the corner points of the raw workpiece, facing towards the robots arm, were chosen but some kind of other physical mark, like a nail or a painted dot would have also been suitable. This provided enough information to precisely place the wood blanks and align the two realms.

Toolpath information was then transformed from the global non-referenced coordinate system the joint was modelled in, to the coordinate space the robot is aligned to. Making use of this technique, precise manufacturing can be achieved but is far from being fast or automated because all points must be measured manually.

Toolpaths were generated using Rhino and Grasshopper, therefore the joint geometry was placed at the world origin (0,0,0) in model space. At first the needed steps were

Figure: 57 – Raw oak

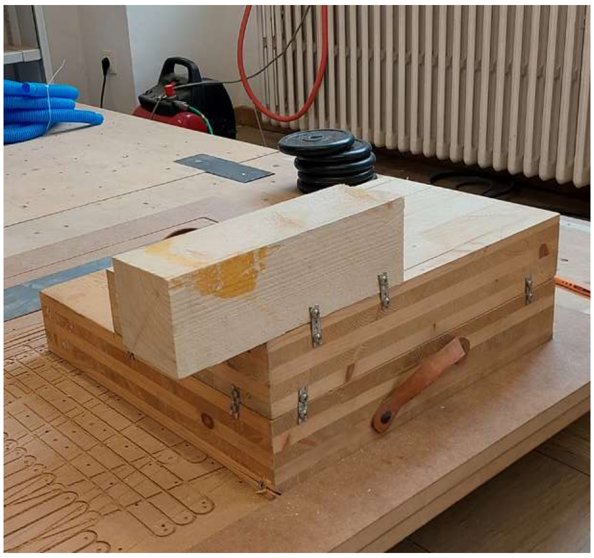


Figure: 58 – Position of workpiece.

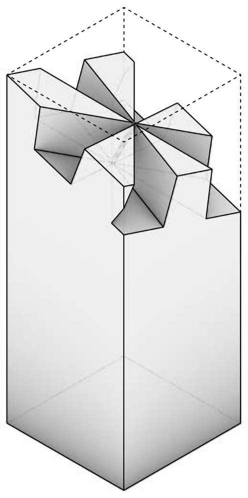


Figure: 59 – Toolpaths - Generalised Kawai Tsugite.

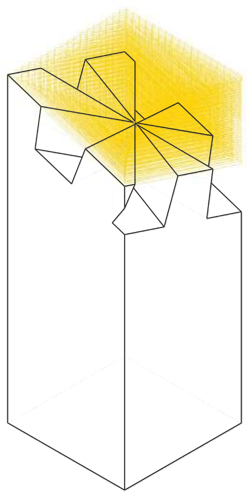


Figure: 60 – Toolpaths - Roughing.

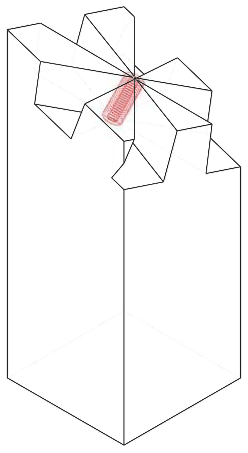


Figure: 63 – Toolpaths - Drilling center.

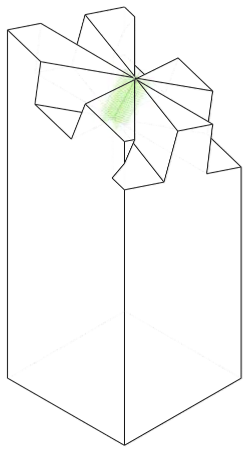


Figure: 64 – Toolpaths - Cleaning center.

evaluated - roughing, cleaning and drilling. (Fig.: 60 - 66). For roughing a relatively simple C#-script was written which generates paths for fast removal of large quantities of material. At this step the plunge depth per layer was set to 2mm and velocity to 50mm/s which was save enough to ensure viability of the tools but made sure the process would be finished in an acceptable amount of time.

Cleaning was implemented by using given finished surfaces. The plunge depth and velocity were reduced to 1mm and 20mm/s. This reduction helped minimizing forces

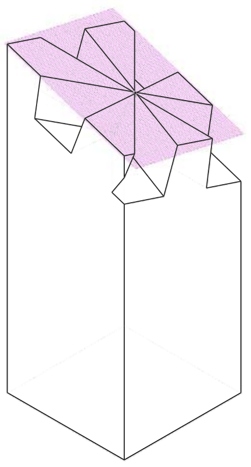


Figure: 61 – Toolpaths - Cleaning plane.

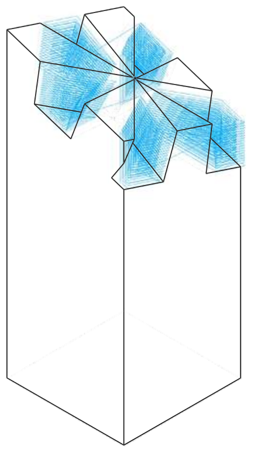


Figure: 62 – Toolpaths - Cutting divider.

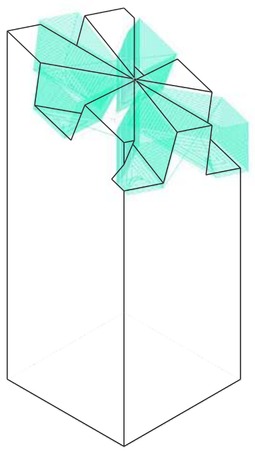


Figure: 65 – Toolpaths - Cleaning divider.

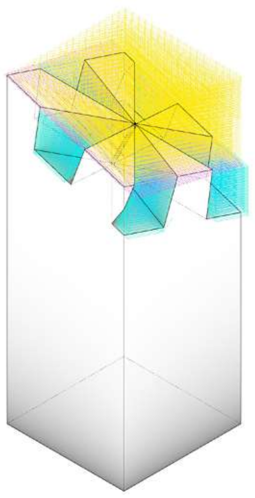


Figure: 66 – Toolpaths - All steps.

acting on the cutter to ensure an accurate and smooth finish of the desired surface. Drilling the hole in the center was accomplished by generating a helical movement into the wood.

Because of the tilted but planar nature of the joint it was relatively easy to generate corresponding toolpaths. Collision detection was not a huge problem because of that - each step was performed after another, so for each travel move the robot retreated to a save reference plane 10mm above the final product.

4.3.4 Conclusion

The test setup shows how much overhead is involved in fabricating complex geometries with basic knowledge from the fields of milling technology and robotics to achieve good results. Here, the focus was on the need for necessary knowledge in the aforementioned areas for the further course of the work.

The eight manufactured connectors vary greatly in their accuracy. The reason for this lies in the different settings in the path generation. Special attention was paid to the penetration depths of the tool, both perpendicular and lateral, as well as the speed at which the cutting edge moves through the material. Furthermore, observations were made in which work sections conventional or climb milling are more suitable.

Only using Rhino and Grasshopper for the toolpath generation proved as a robust alternative to expensive CAM software packages. Developing and utilizing custom tools provided further insight on fabrication processes and required data.

It is worth mentioning that the generalized form of the Kawai Tsugite was easier to machine than the original. Due to the modification of the geometry, two surfaces have been omitted which would otherwise have been difficult to approach with the milling cutter. Towards the end of the project, the products were compared and analysed. This evaluation helped to estimate settings most suitable in the further course.

Overall, the experiment with the necessary prior knowledge was a straightforward process, which is also due to the complex but still easy to produce geometry. Without major problems, it would be possible to produce this wooden joint on a three-axis milling machine, provided you can tilt the spindle perpendicular to the base plane of the knot or place the workpiece accordingly in the working area.

4.3.5 Discussion

For further simplification and acceleration of the process, an automation of the position determination of the workpiece in the working area would be advantageous. Since this step is particularly laborious and error-prone, positioning systems, for example - LiDAR (Light Detection And Ranging, visual) and so on, should be established as already mentioned by Svilans et. al. [31]

In addition, artificial intelligence could be implemented controlling and monitoring the process. Making pathing more automated and possibly enabling feedback during fabrication. By monitoring outside parameters concerning wood properties or its surroundings, AI enabled robotic control could react on changes during fabrication – reducing faulty products or making the process more intuitive and therefore more accessible for a wider range of applications.

Figure: 67 – KT2 - Raw Wood.

Figure: 68 – KT2 - Roughing.

Figure: 71 – KT2 - Cutting Divider #4

Figure: 72 – KT2 - Cutting Divider #5

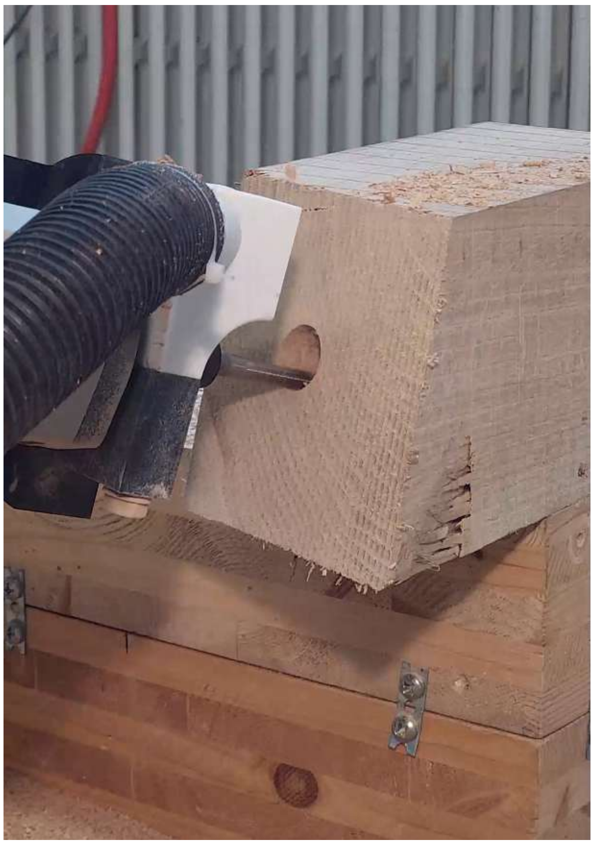


Figure: 69 – KT2 - Drilling Center.

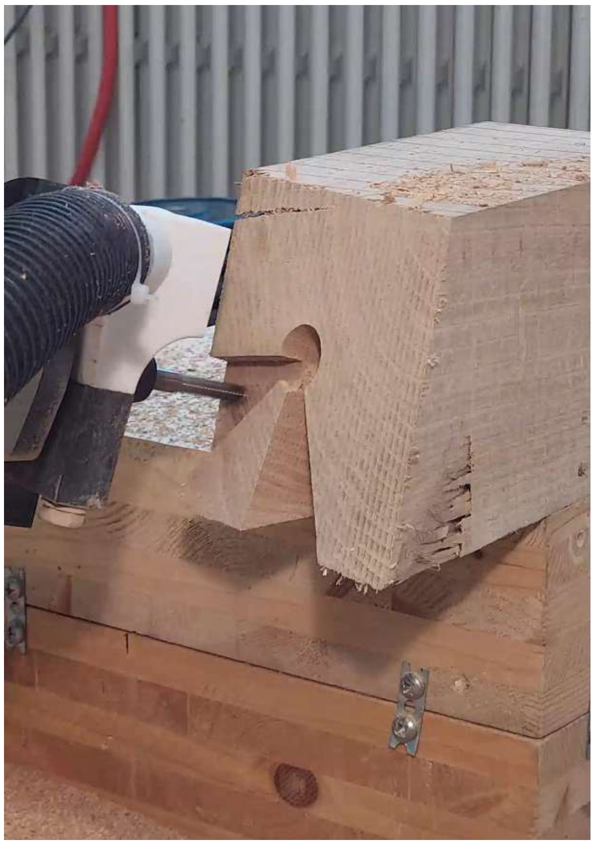


Figure: 70 – KT2 - Cutting Divider #2

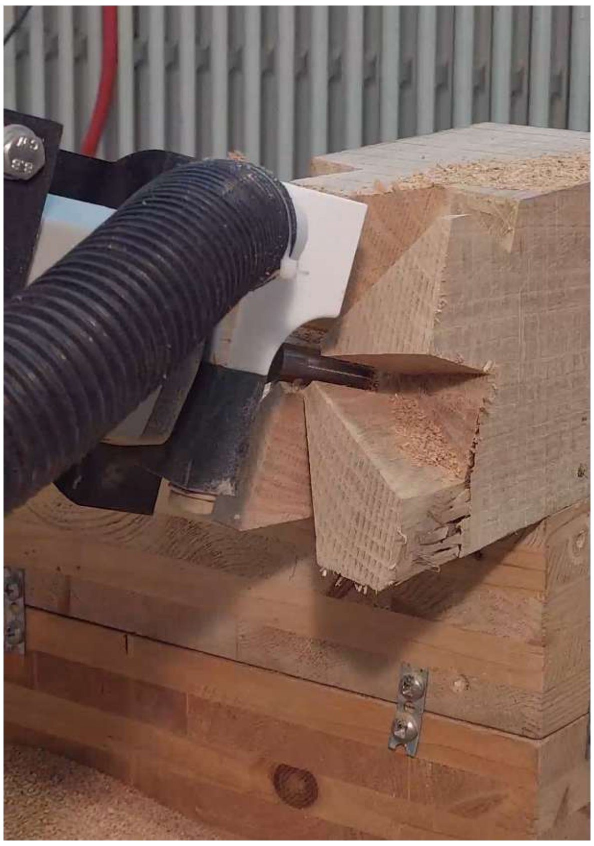


Figure: 73 – KT2 - Cleaning Center.

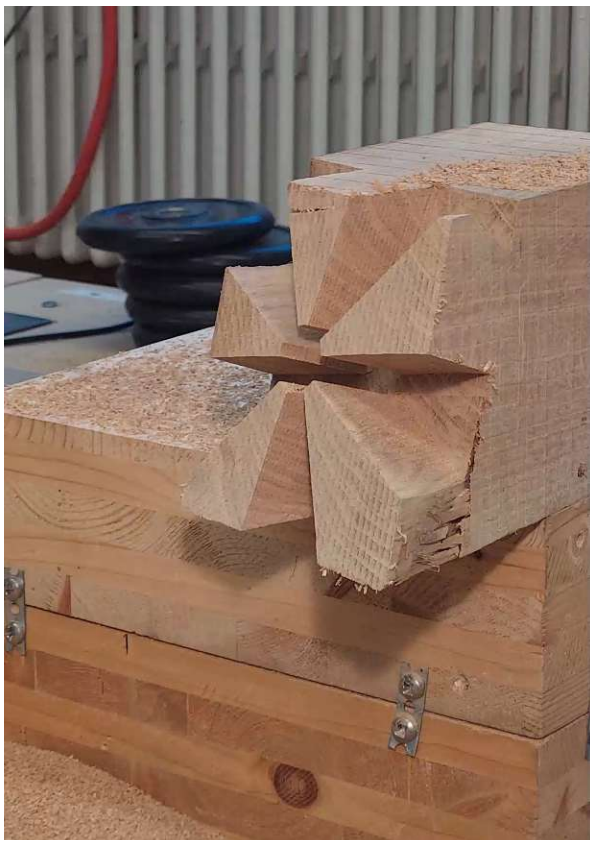


Figure: 74 – KT2 - Finished Joint.

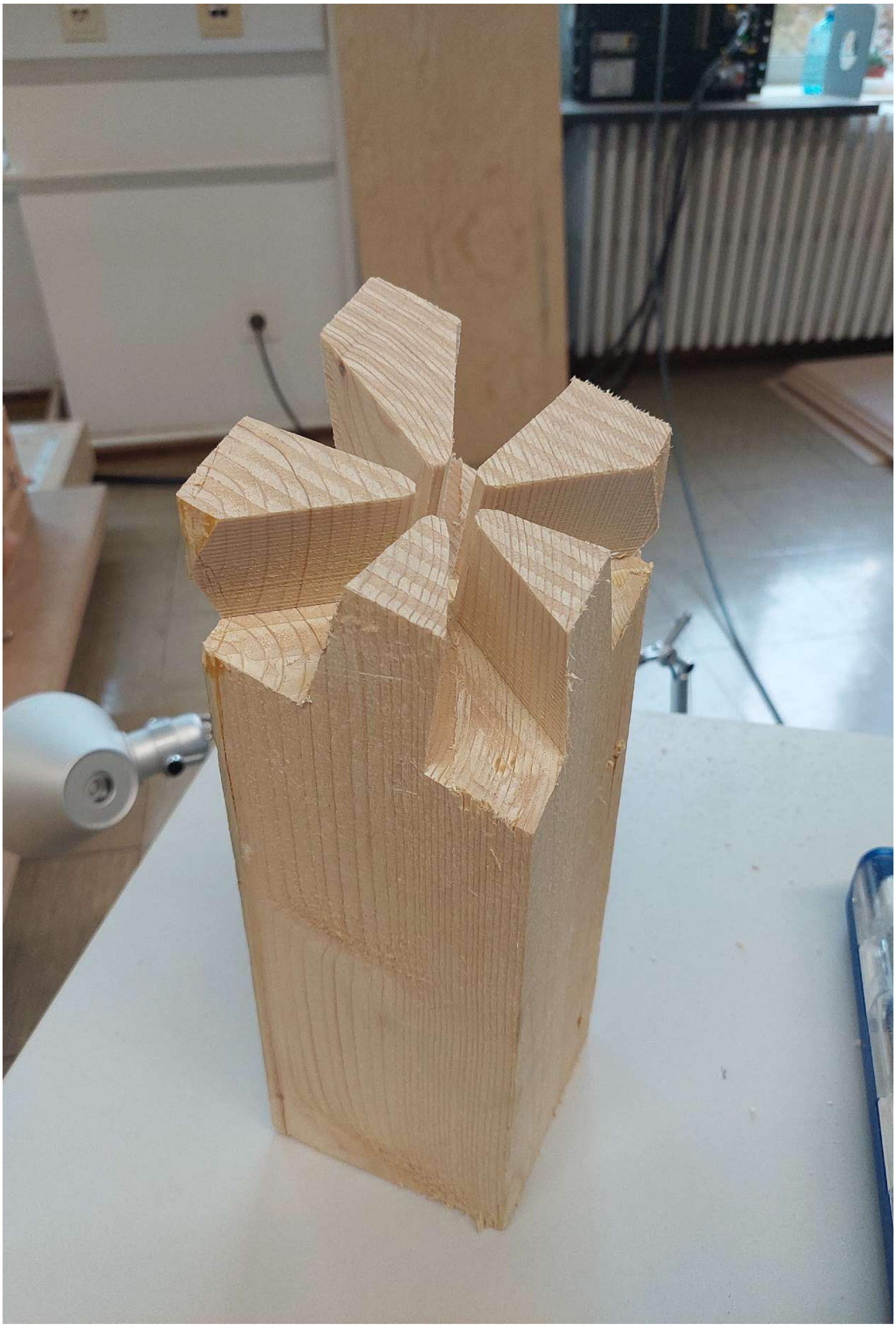


Figure: 75 – Finished KT0 - oak

Figure: 76 – Finished KT0 - configuration 1

Figure: 77 – Finished KT0 - configuration 2

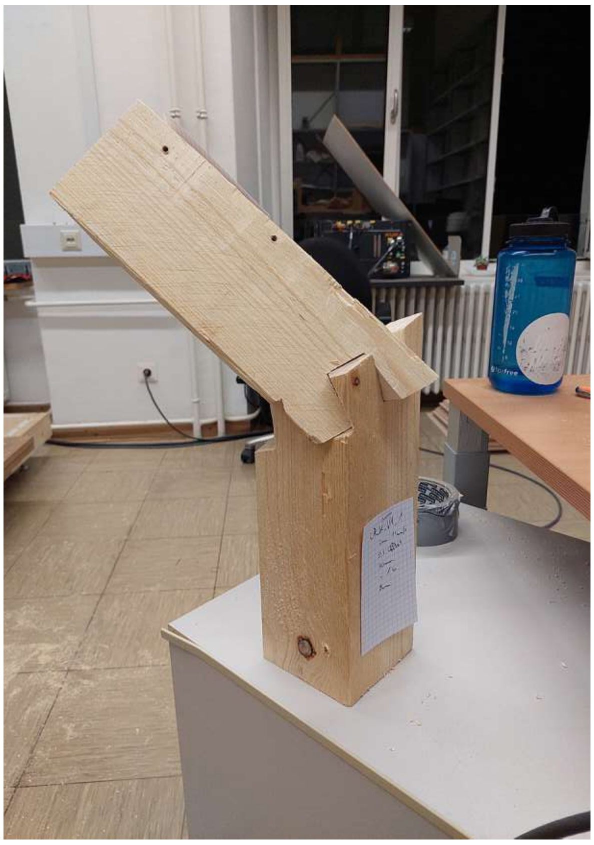


Figure: 78 – Finished KT0 - configuration 3

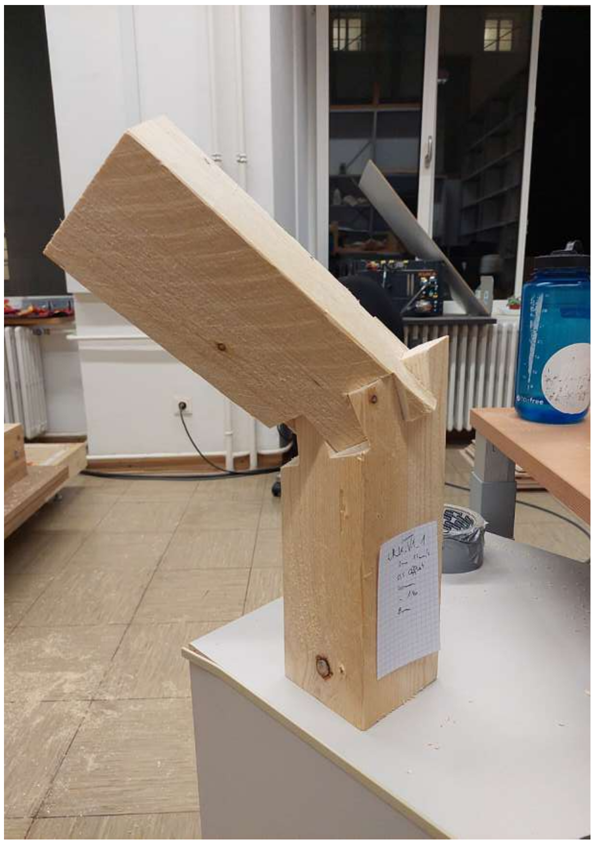


Figure: 79 – Finished KT0 - configuration 4

4.4 Robotic Fabrication of Doubly Curved Beams

4.4.1 Introduction

The following chapter brings the two previous chapters together. The generative multi-scalar workflow from Chapter 4.2 is expanded by the findings on robotic fabrication from Chapter 4.3. For this purpose, two beams of the prototypical test surface were selected to be produced as a proof of concept in order to be able to subsequently evaluate the digital workflow presented. These were crossing beams – B.3.0 and B.9.0 (Fig.: 80) – at a strongly curved area of the surface connected to each other in the middle using a cross lap joint.

In order to be able to manufacture the beams, some information had to be added. Among other things, models for the production of the mold, quantity determination of the materials as well as a detailed description of the work steps were required. Further adjustments in the previous models were necessary because more and more restrictions emerged in the course of the work.

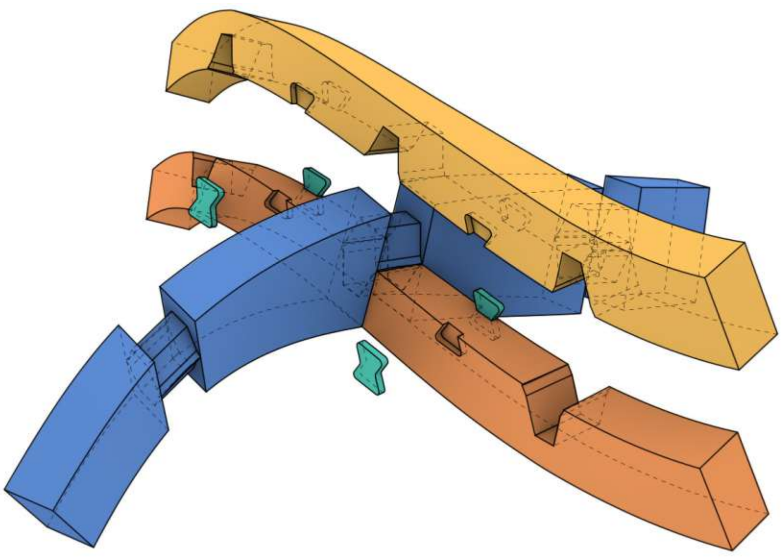


Figure: 80 - Selected Beams: B3.0 (orange), B.9.0 (blue)

In advance, an attempt was made to identify as many sources for errors as possible early on and thus to create a straight-forward workflow. The evaluation of the accuracy of the end product and the robustness of the process itself were finally checked by laser scanning the finished beams.

4.4.2 Setup

Because of the limited operating range of the robot, being a radius of 725,9mm, a scaling of 1:5 was chosen so each point on a beam can be reached. Furthermore, the range of motion of the robot, although it is a 6-axis system, had to be considered as many motion sequences can lead to collisions. Thorough preparation and control of the paths to be generated is therefore another focus.

Due to the knowledge gained during the project, adjustments and extensions had to be carried out in the previously models created in Chapter 4.2. Mostly affected were models needed to provide direct information for manufacturing. Massive changes had to be made, especially for the creation of the blanks and the wood joints. However, minor changes were also made at the beginning of the workflow. For example, the curvature of the reference surface and the individual beams had been slightly reduced.

The following paragraphs detail the changes made.

4.4.2.1 Modification at Intersections and Beam Blanks

Because of the chosen type of wooden connections at points where the beams cross, the models for wood joint production and the generation of the blanks for the

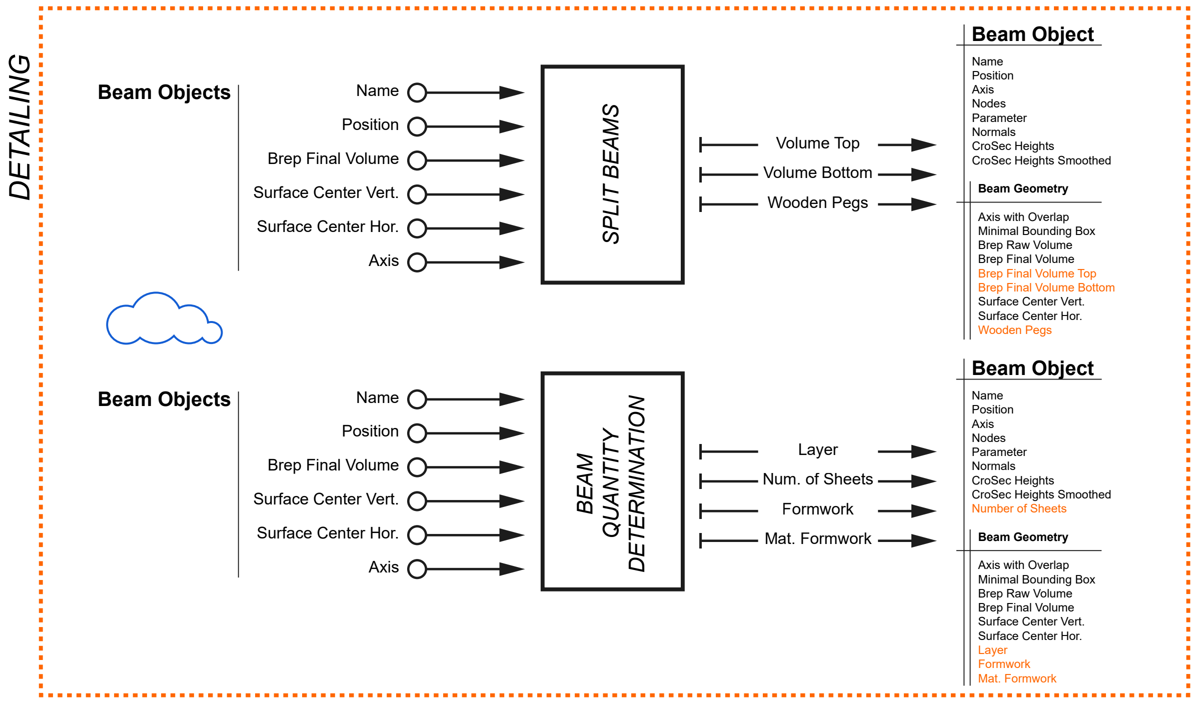


Figure: 81 - Modified detailing.

fabrication had to be modified. Since one girder always encloses the other in one point, the blanks for the enclosing beams had to be divided along the horizontally central surface. Furthermore, it had to be ensured that the beams were held together when installed. For this purpose, wooden dowels were designed and included into the models.

Additionally, certain tolerances are necessary for assembly. It must be possible to

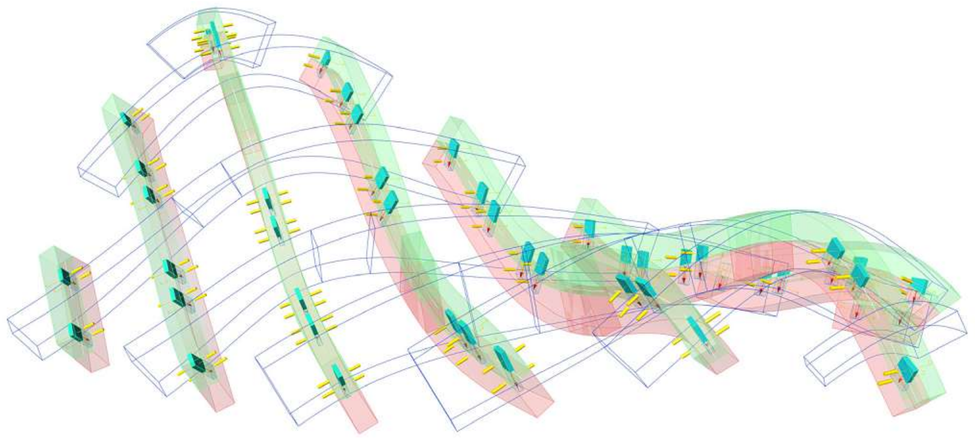


Figure: 82 - Modified beam blank.

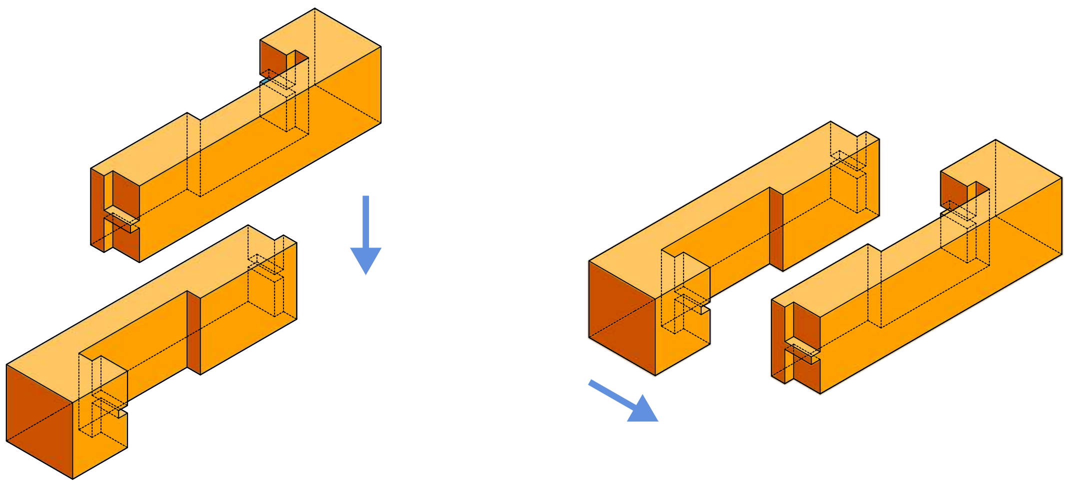
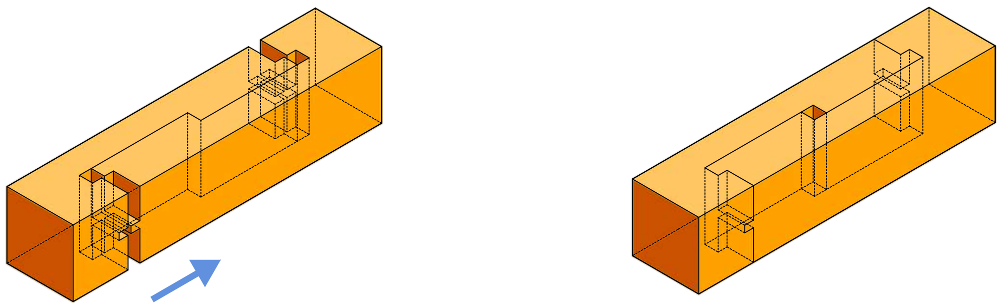



Figure: 83 - Assembly DOFs of original Kanawa Tsugi.

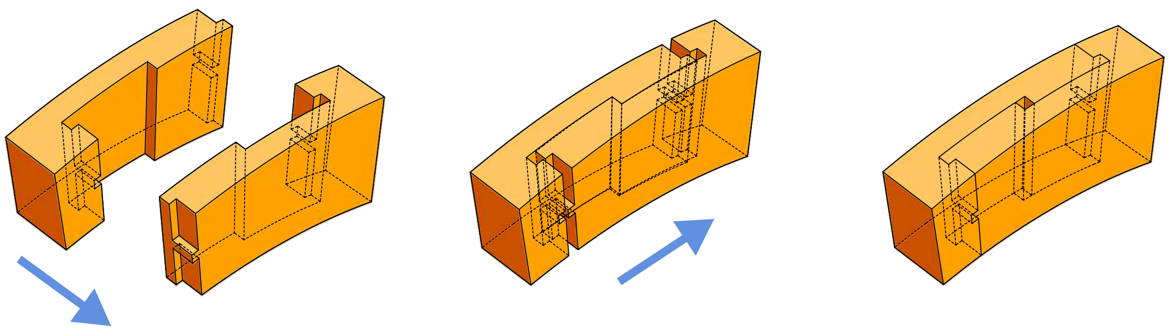


Figure: 84 - Assembly DOFs of doubly curved Kanawa Tsugi.

approach all connection points of the beams exactly from one direction, which affects the geometry of the joints. This direction must be found and determined beforehand.

To do this, each connection point as well as connected beams must be considered in order to determine a vector that works for the entire local area. To then chamfer the edges of the joint with the help of this vector in order to have enough tolerances available during assembly. [25]

This problem was dealt with here purely during the toolpath generation. By moving beyond the actual surface of the wood joint, additional material was removed from desired surfaces. This problem was addressed for the prototype and worked in this limited case.

Modification at the End-Joints

In contemplating how the girders could be made with all their details, it has been found that the end connections chosen - the Kanawa Tsugi - are neither suitable for use in double curved form nor suitable to be produced satisfactorily with the given means.

By bending and twisting the original geometry and logic of the knot, one loses a degree of freedom for assembly in the structure (Fig.: 84). Due to the required tolerances mentioned above, the knot would lead to further difficulties. And further iterations would be needed before the design of the joint would be developed enough to be suitable for use in this framework. Since these were not necessary for the prototype, consisting of two crossing beams, they were omitted for production.

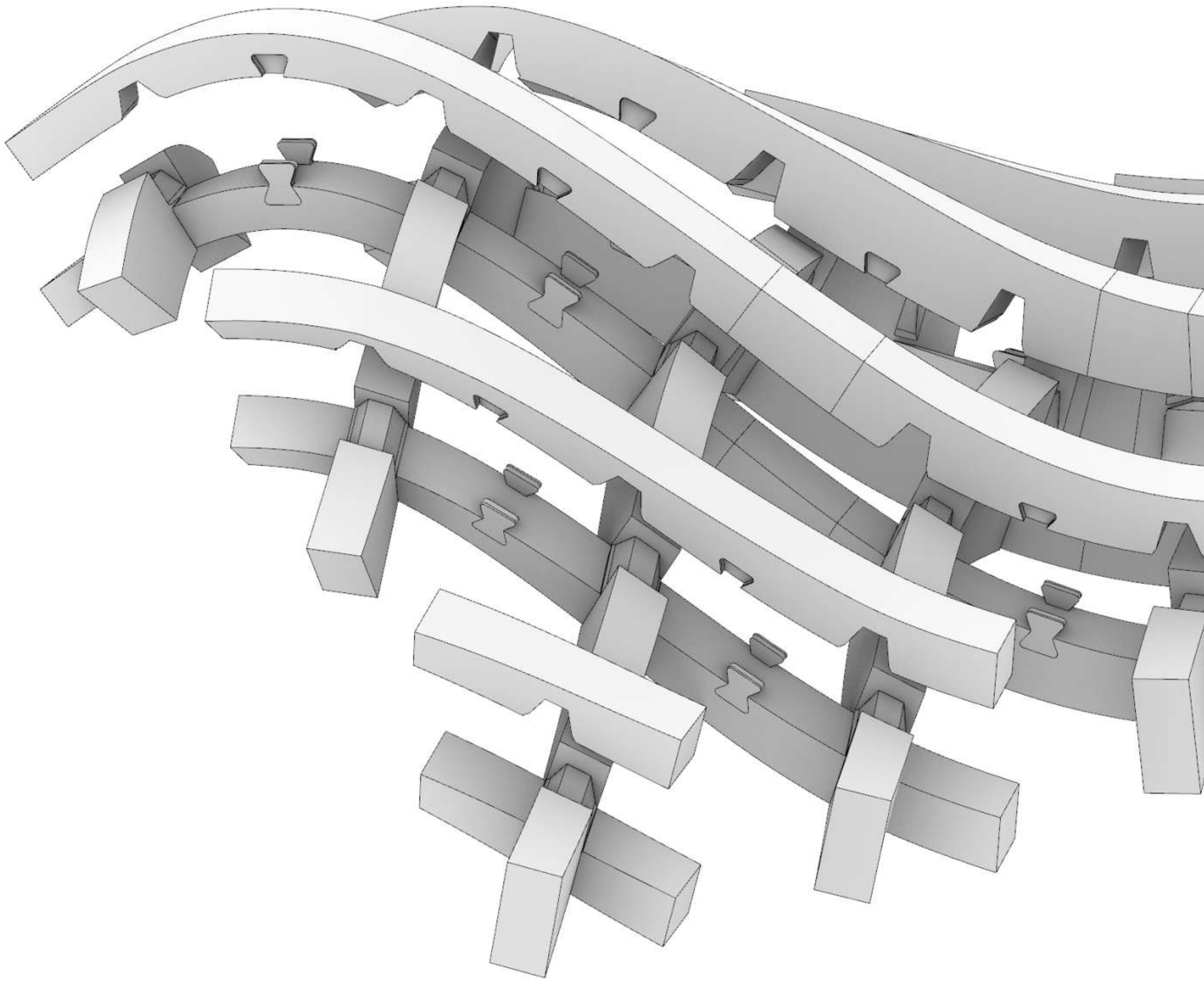
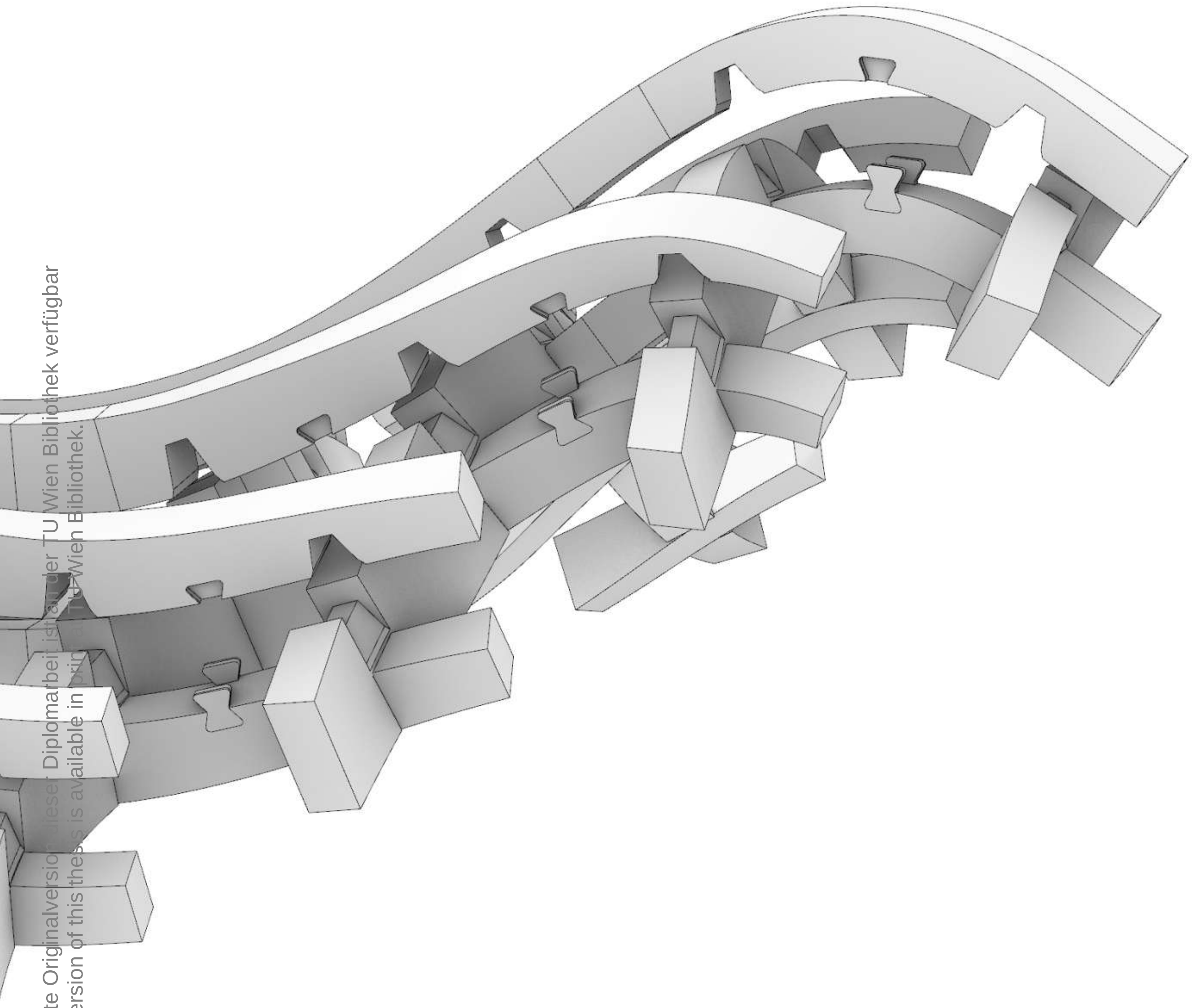


Figure: 85 - Final grid geometry for prototypical fabrication.



4.4.3 Fabrication

4.4.3.1 Formwork

The first approach for a suitable formwork for the blanks, was based on the extended section of the lower beam surface resting on the reference surface. This was slightly enlarged and enclosed by a minimal bounding box. The box was then cut using this surface, resulting in an upper and lower half. The lower half was connected to the surface to form a closed volume and from this a plug-in waffle grid was created which follows the curvature of the beam.

The individual elements were abstracted in order to minimize the number of cuts on the milling machine and later to achieve an exact surface. Then the elements were milled out of 18mm MDF boards. After assembling together, they were placed in front of the robot and the exact surface was milled out of the grid.

The first attempt to glue four 4mm leftover pieces of board using the jig was very

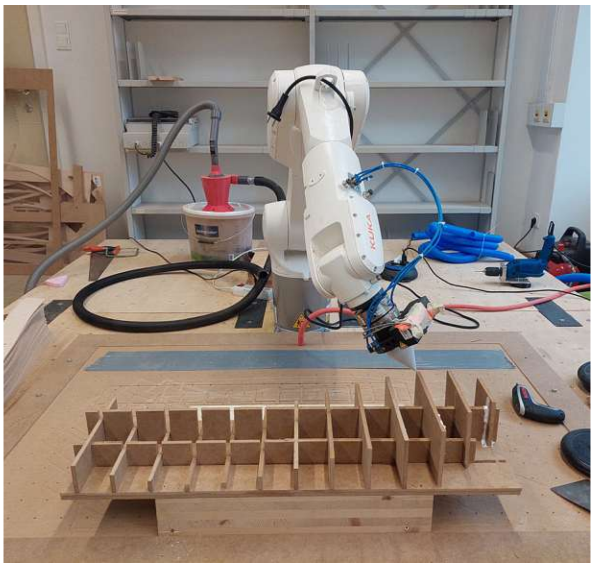


Figure: 86 - First MDF-Form.

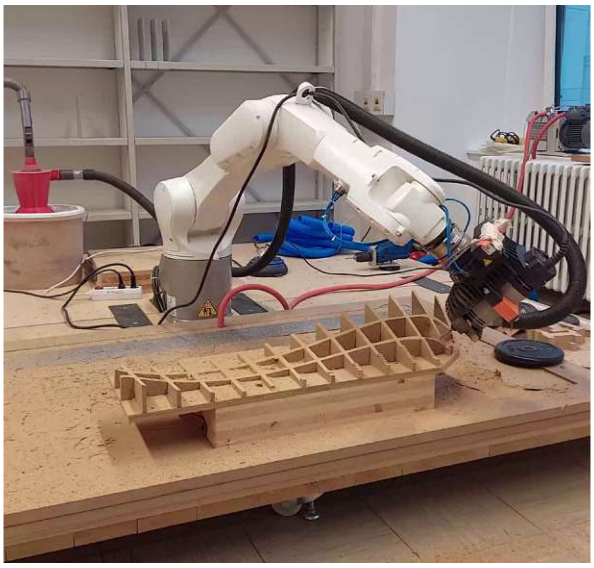


Figure: 87 - Cutting freeform surface from MDF-grid.

successful and showed that the desired shape was maintained very well from the material. However, it also became apparent that it would be impossible to fix the exact position of the wooden layers over the entire cross-section of the beam because there were no endstops along the vertical distance. This was then confirmed when trying to glue the layers of the beam blank.

Due to the lack of endstops following the shape, the individual layers began to slide on the glue film, which made a precise product impossible. In addition, work was done purely with clamps, where it was also difficult to find a position in which they could apply the necessary pressure on the curved surfaces.

For the second approach to building formwork, the waffle-grid was retained, but significant extensions were made. Based on the minimal bounding box, however, this time the entire beam was cut out of the volume, which described the shape of the beam on all side surfaces. This was used to generate stops on five sides of the beam blank, only the upper surface remained open to ensure that the individual layers of wood could be inserted.

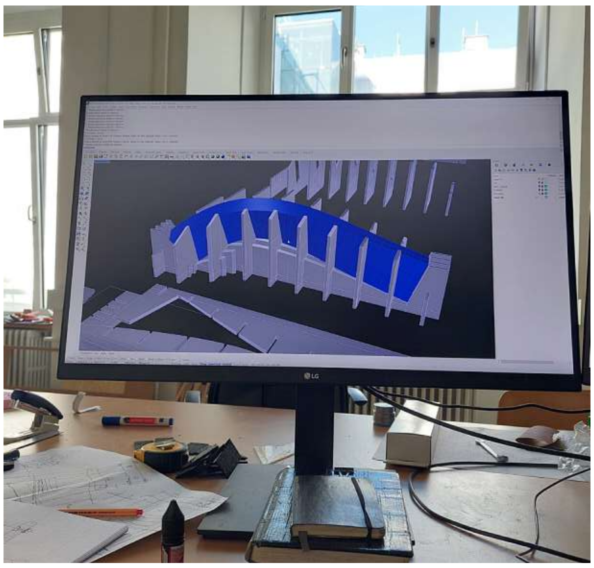


Figure: 88 - Revised formwork.

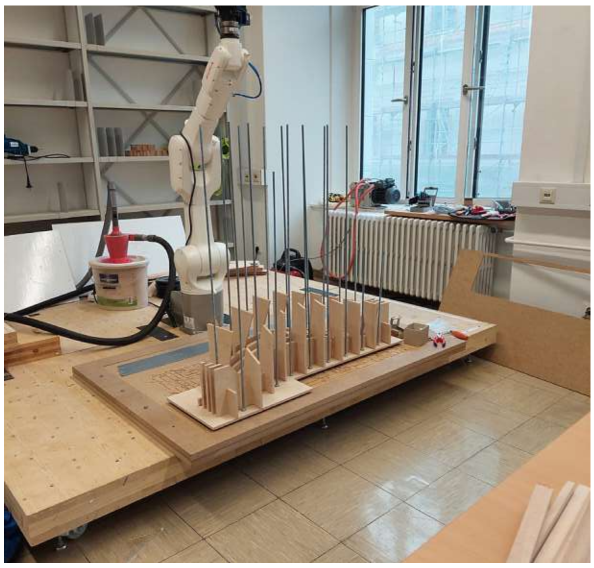


Figure: 89 - Final revised formawork.

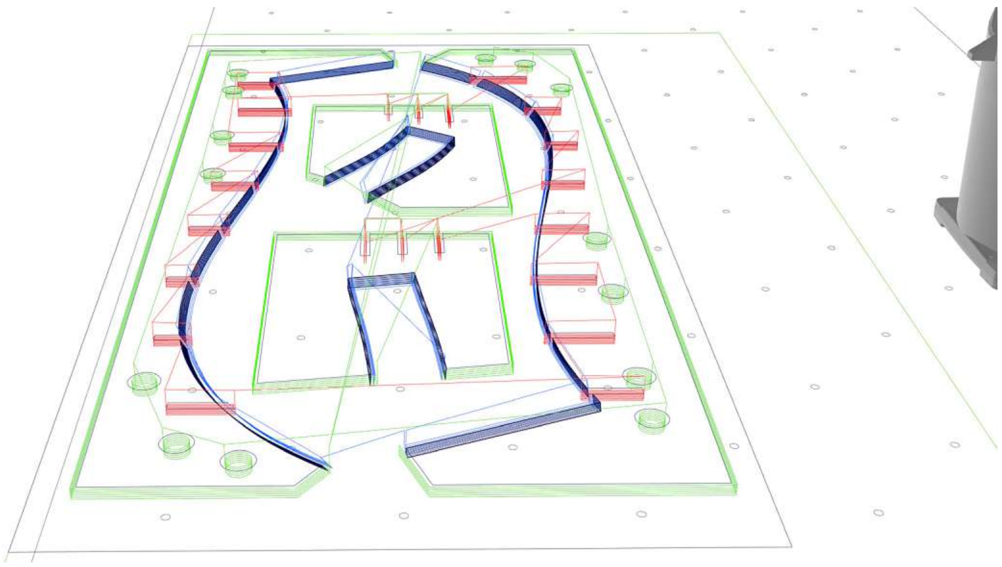


Figure: 90 - Toolpaths for milling of formwork.

In addition, the points at which the support structure should be placed were evaluated in advance, thus reducing the number of individual lattice elements to the essentials.

Moreover, holes were added in the side surfaces of the grid elements, which allowed clamps to be placed regardless of the curvature of the surface. In addition, the original version was extended by a base plate. Threaded rods could be placed in these, with the help of which it was possible to bring in additional force in the gluing

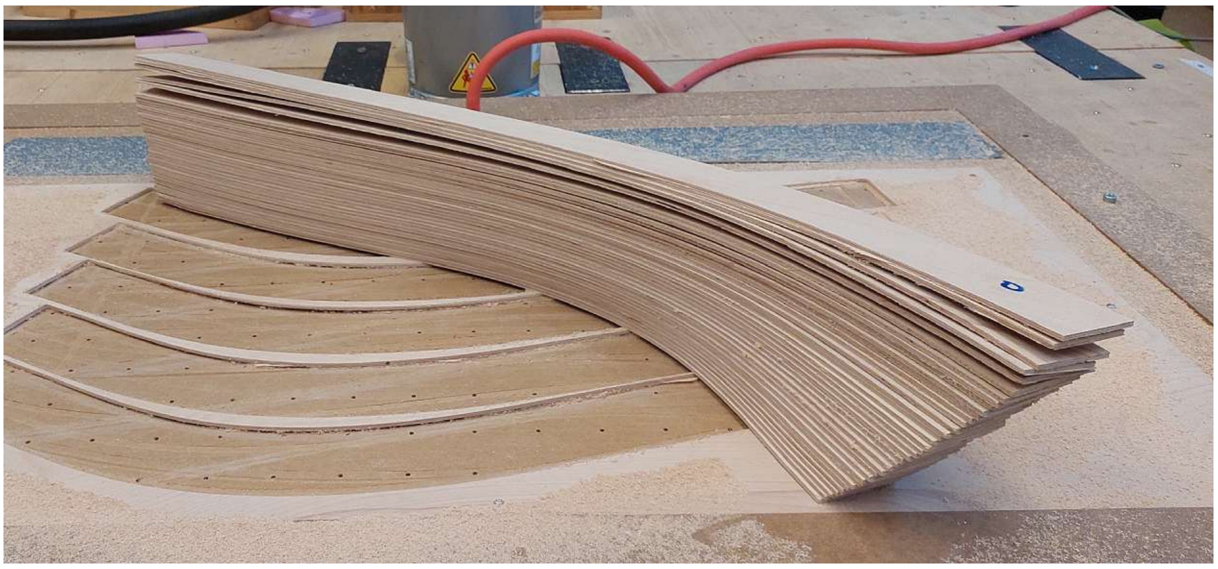


Figure: 91 - Finished blank layers.

process. The generation of the necessary data for mold construction was largely set up manually on the basis of the final geometry, created and manufactured semi-automatically.

4.4.3.2 Glueing

4mm plywood panels made of beech served as the raw material for the beams, as these could just about be brought into the desired shape with the available means. The scaled cross-section height of the selected beams varied between 120 and 200mm. Accordingly, for a raw double-curved wooden beam, for B.3.0 - 45 layers; and for B.9.0 - 34 layers of beech plywood were prepared and glued.

The individual layers were processed in Rhino + Grasshopper and provided with beam names and layer numbers to simplify identification and gluing. They were then conventionally cut out in the already curved shape on a three-axis milling machine.

The blanks were oversized in the previous steps. On the long sides, 50mm of material

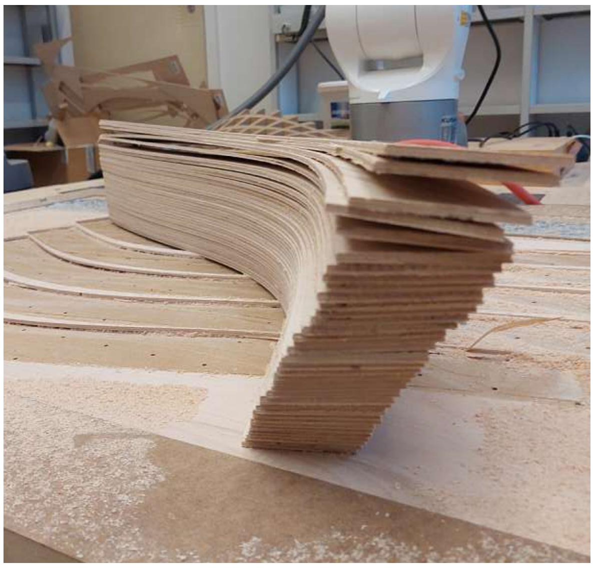


Figure: 92 - Blank layer B.9.0.

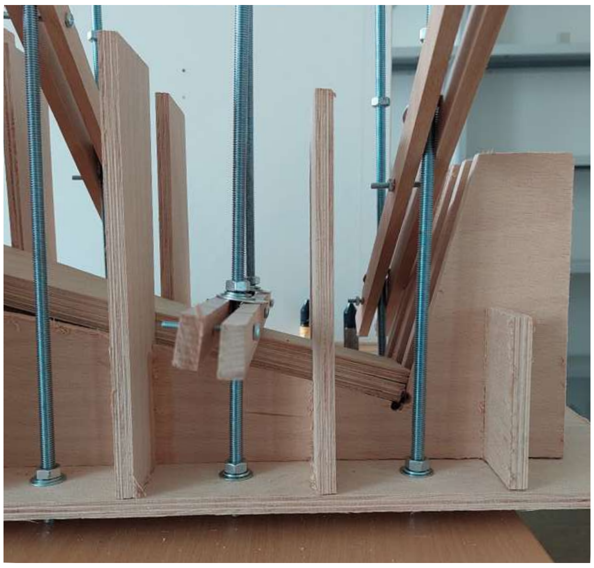


Figure: 93 - Testfitting the mold.

was added at the ends and almost 10mm across the beam. This leads to more tolerances for fabrication.

As already described above, gluing the individual layers proved to be more difficult than expected. The improved variant of the formwork at least prevented individual layers from slipping, but still did not hold them in their position quite accurately. Layers were placed in the mold in bundles of four - this was necessary as that number was just about bendable and manageable. To do this, glue was first applied to the entire underside of each layer with a spatula and then positioned on top of each other. Then this package was placed in the mold to be held in shape with the help of threaded rods and screw clamps until the glue had hardened. Each of these steps took around 30 minutes and the effort of manual tensioning was challenging.

It was expected that individual layers could slip slightly, and that the material would want to spring back into the originally flat starting position after stripping. Fortunately, the spring-back of the finished blanks was minimal and the gluing fixed the shape.

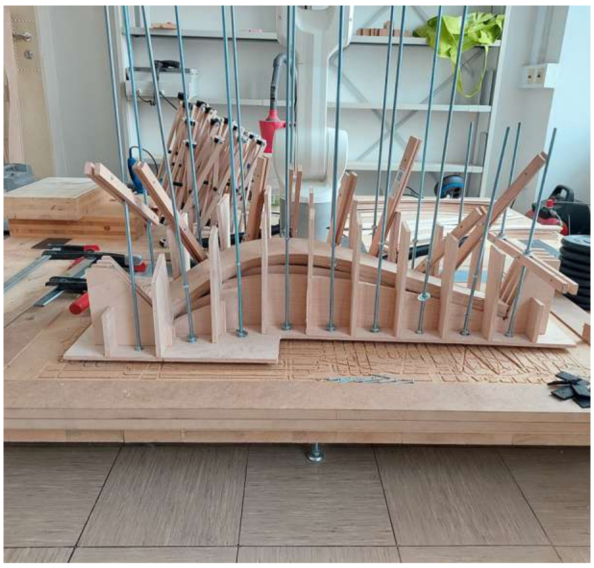


Figure: 94 - Gluing B.9.0 layer: 8.

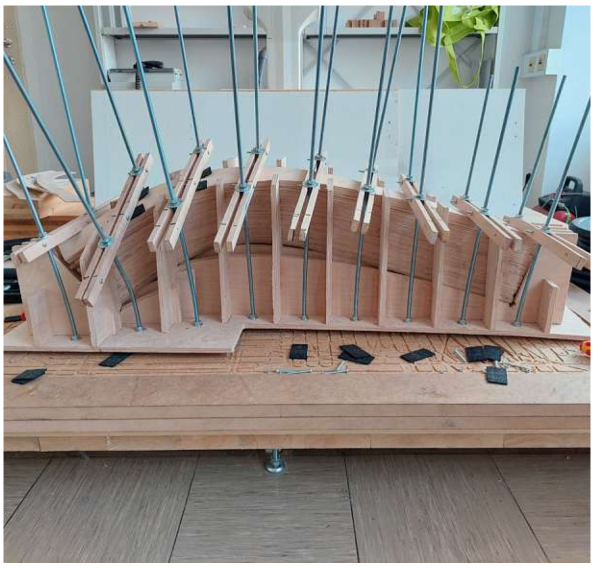


Figure: 95 - Gluing B.9.0 layer: 34.

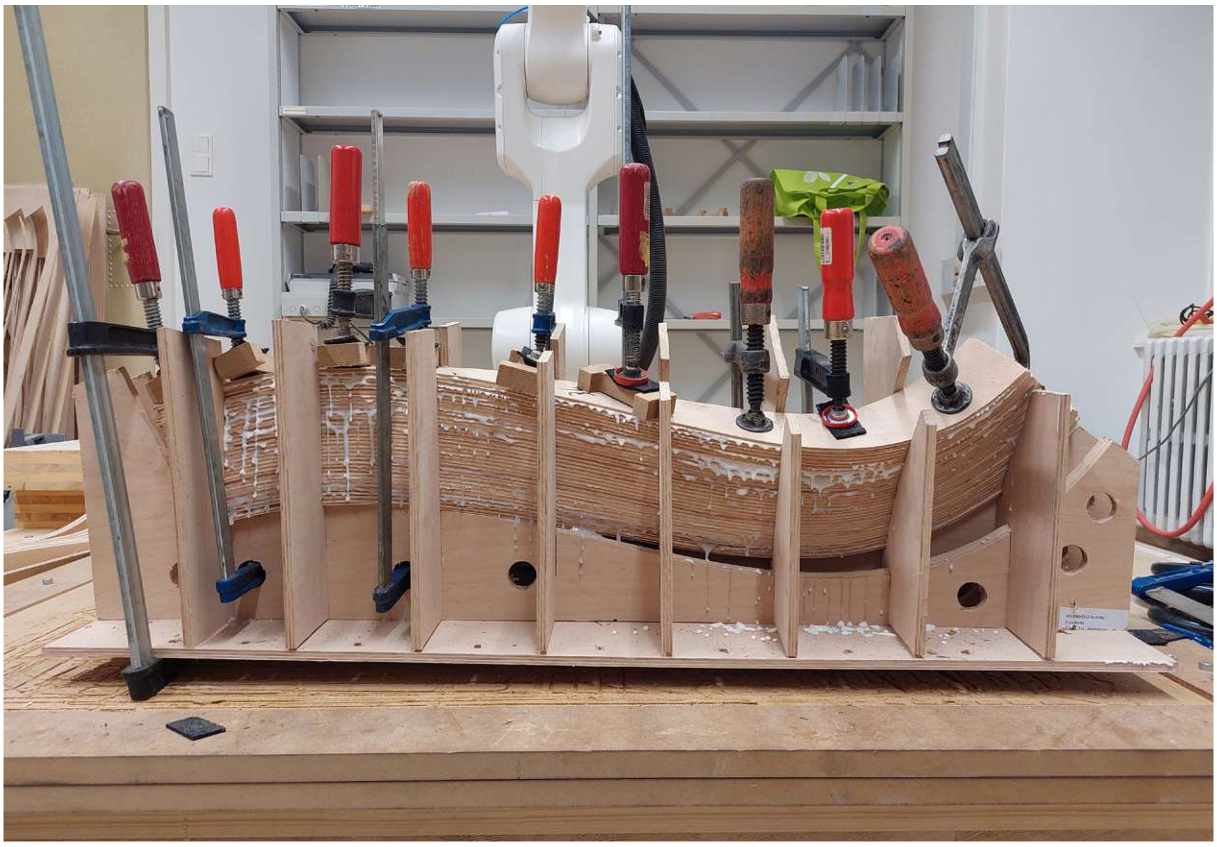


Figure: 96 - Gluing B.3.0 blank.

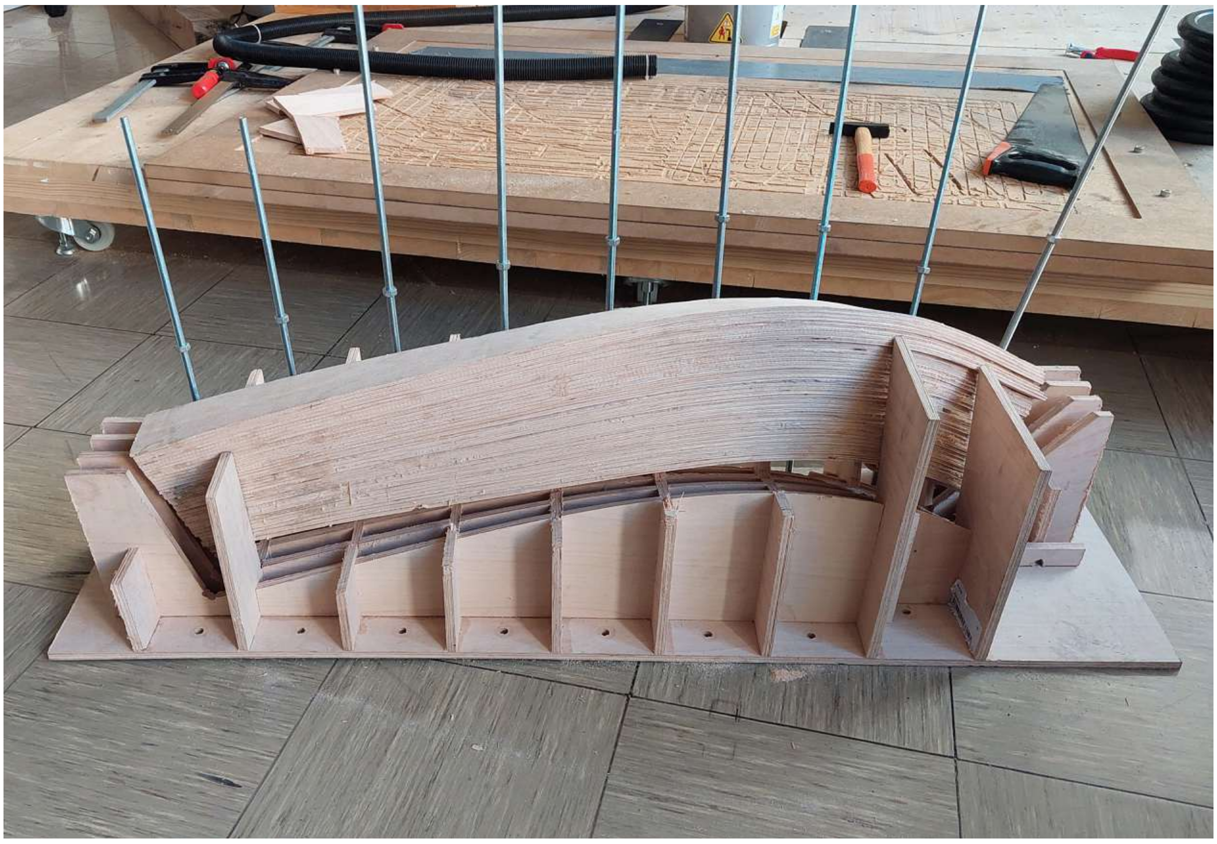


Figure: 97 - Finished B.9.0 blank.

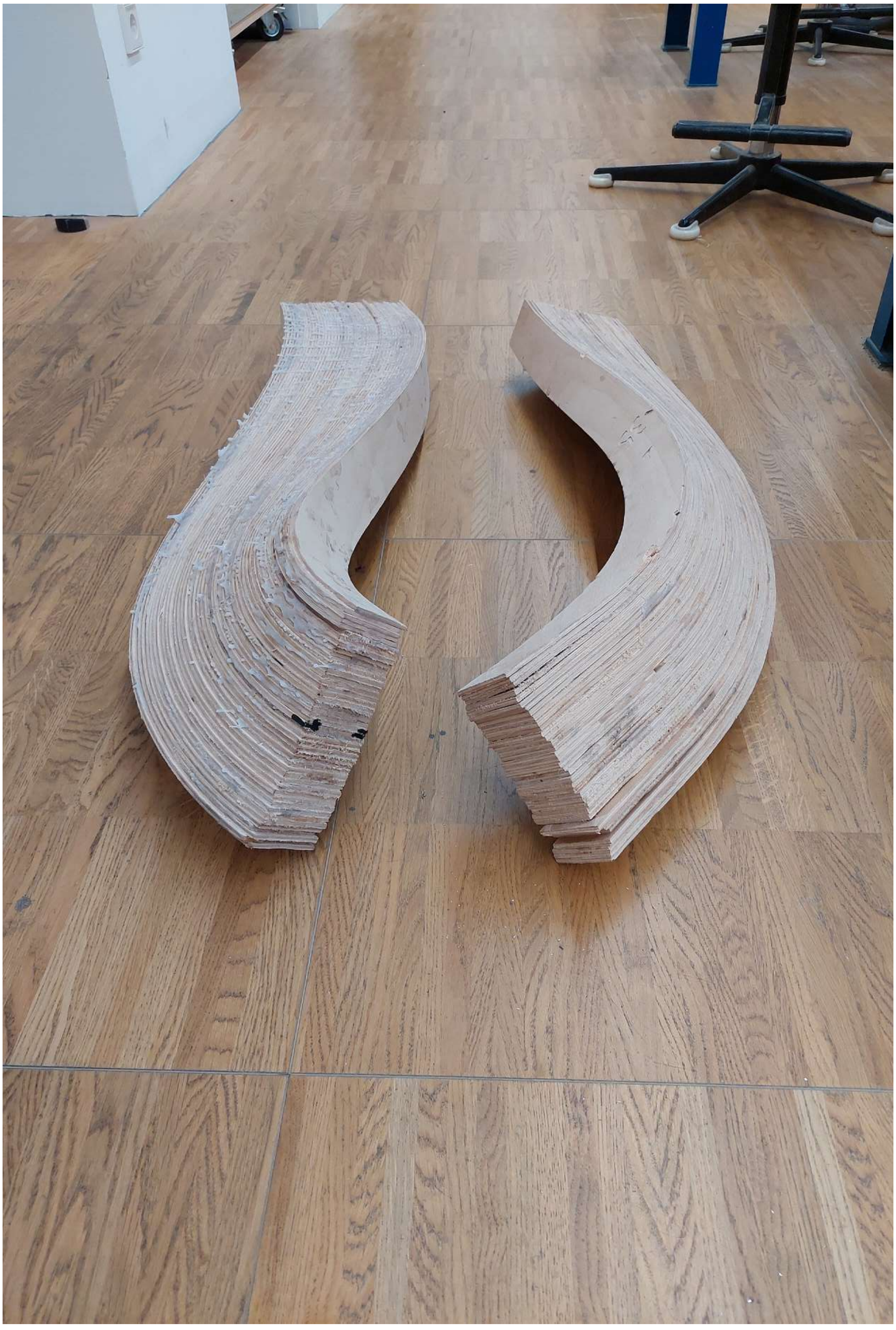
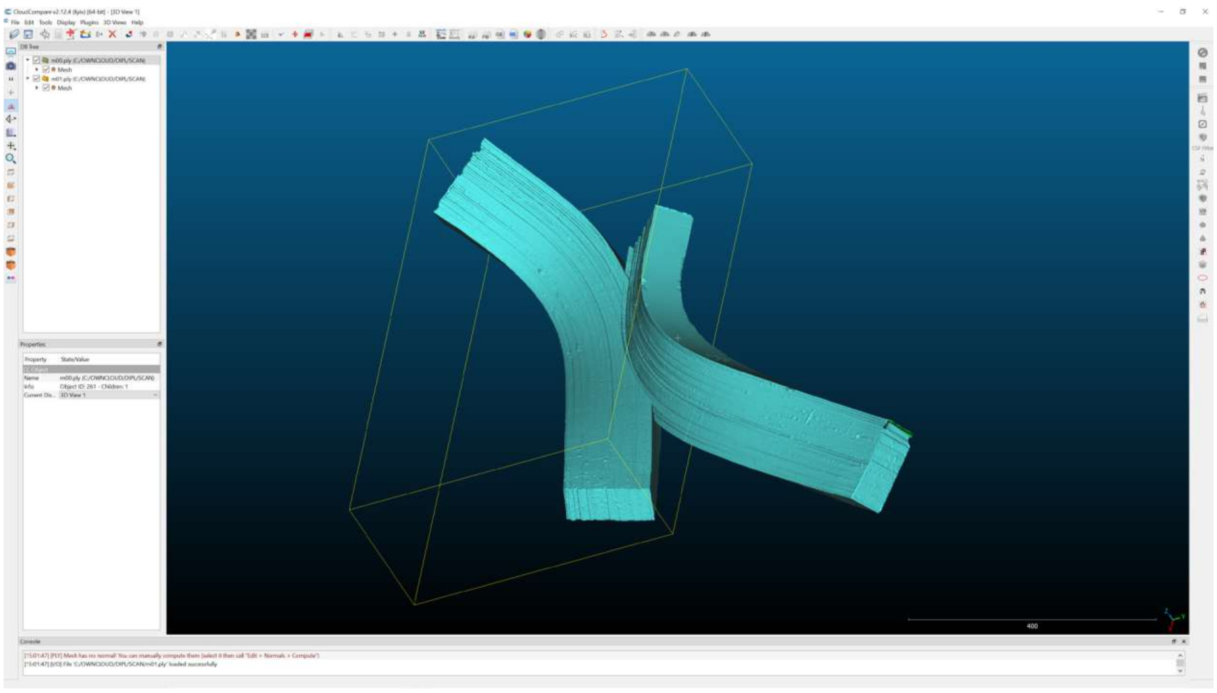


Figure: 98 - Finished Blanks - left: B.3.0; right: B.9.0


CloudCompare v2.12.4 (Kyiv) [64-bit] - [3D View 1]
File Edit Tools Display Plugins 3D Views Help
DB Tree
m00.ply (C:/OWNCLOUD/DIPL/SCAN)
Mesh
m01.ply (C:/OWNCLOUD/DIPL/SCAN)
Mesh
Properties
Property State/Value
Name m00.ply (C:/OWNCLOUD/DIPL/SCAN)
Info Object ID: 261 - Children: 1
Current Dis... 3D View 1
400
Console

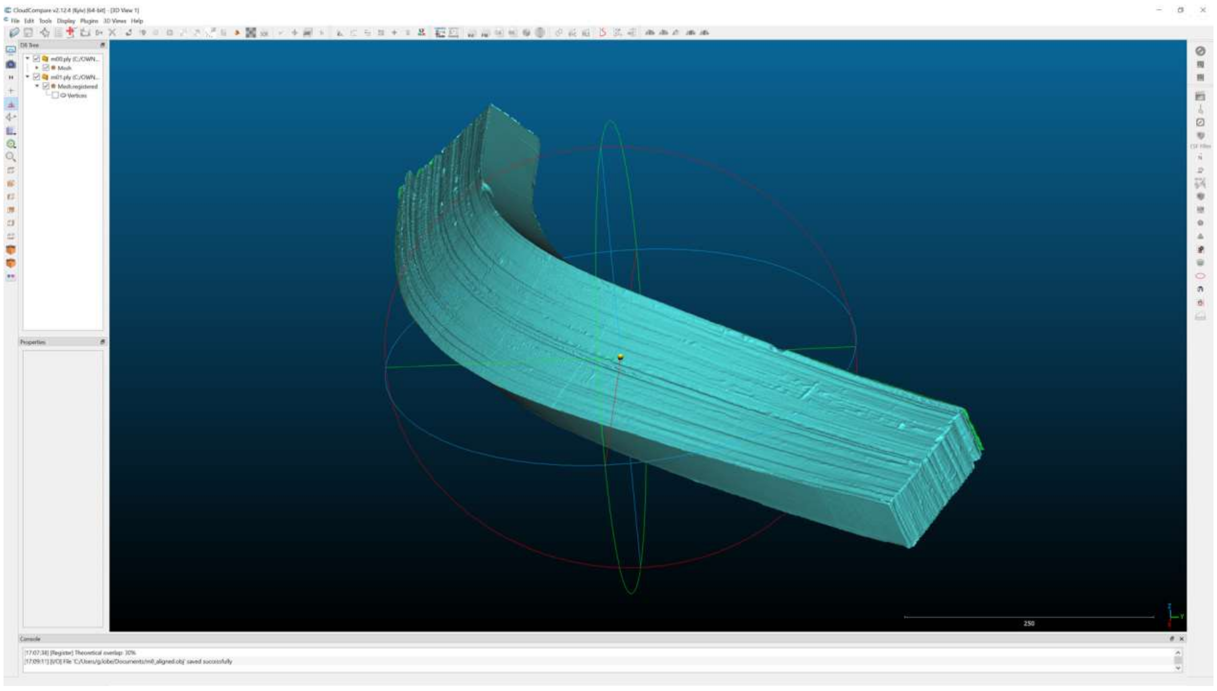
```
[15:01:47] [PLY] Mesh has no normal! You can manually compute them (select it then call "Edit > Normals > Compute")
[15:01:47] [I/O] File 'C:/OWNCLOUD/DIPL/SCAN/m01.ply' loaded successfully
```

Figure: 98 - B.9.0.: Pointclouds of both sides.

4.4.3.3 Laserscanning

The ready-glued blanks were then digitized using a manual laser scanner to be able to guarantee sufficient precision. Two point-clouds were created from each blank. To record the entire beam in as much detail as possible, it had to be turned once. Five of the six faces were recorded in each pass. Then the point clouds had to be registered, combined, and meshes created from them. This was done with the help of CloudCompare (Fig.: 98 + 99).

The open-source software enables the processing of point clouds and meshes. Resulting closed meshes were saved in two resolutions - original and reduced. In Rhino the reduced version was imported so that the digital model could be verified, by overlaying scanned and modelled geometry. Critical areas were identified, and geometry was repositioned enhancing the use of the volume and neutralizing spots too close to each other. To do so, the distance between the mesh vertices and the


CloudCompare v2.12.4 (Kyiv) [64-bit] - [3D View 1]
File Edit Tools Display Plugins 3D Views Help
DB Tree
m00.ply (C:/OWN...
Mesh
m01.ply (C:/OWN...
Mesh.registered
Vertices
Properties
250
Console

```
[17:07:38] [Register] Theoretical overlap: 30%
[17:09:11] [I/O] File 'C:/Users/g.lobe/Documents/m0_aligned.obj' saved successfully
```

Figure: 99 - B.9.0.: Aligned Pointclouds for mesh generation.

surface of the modelled blank was measured digitally. In Figure 100 critical areas in which this distance is less than 2mm are shown in red. The minimum distance from beam B.9.0 is roughly 2mm, which should not be a problem later.

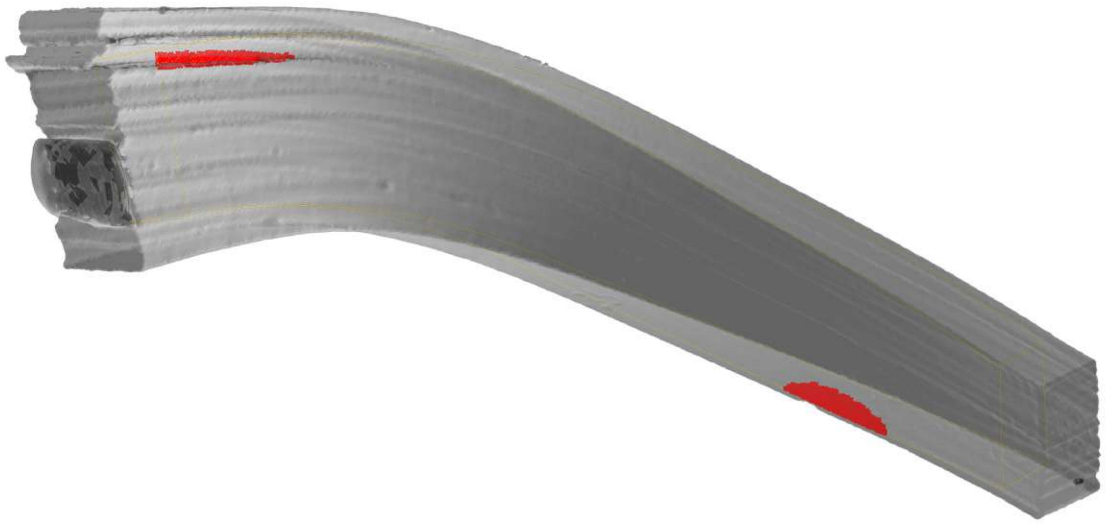


Figure: 100 - Critical areas

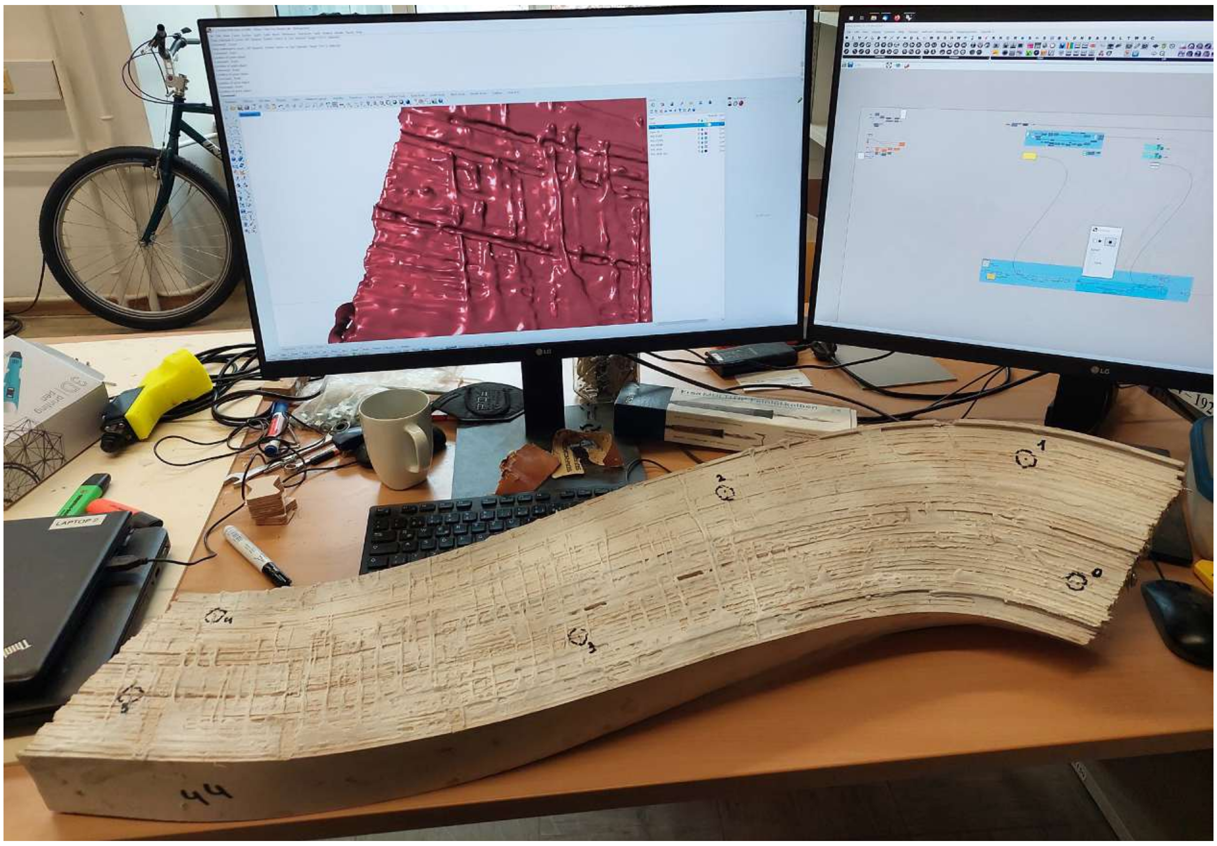


Figure: 101 - Prominent points on B.3.0.

4.4.3.4 Positioning

To be able to locate the blanks in the robot's work area, nails were hammered into the end faces. These served as reference points for measuring and finding the correct orientation of the beam.

Since the nails had not yet been set at the time of the laser scan, they had to be manually located afterwards in the digital model. For this purpose, prominent points on the blanks surface were used with which the positions of the nails on the beam were successfully determined (Fig.: 101 - 105). Figure 105 shows the located nails as orange points.

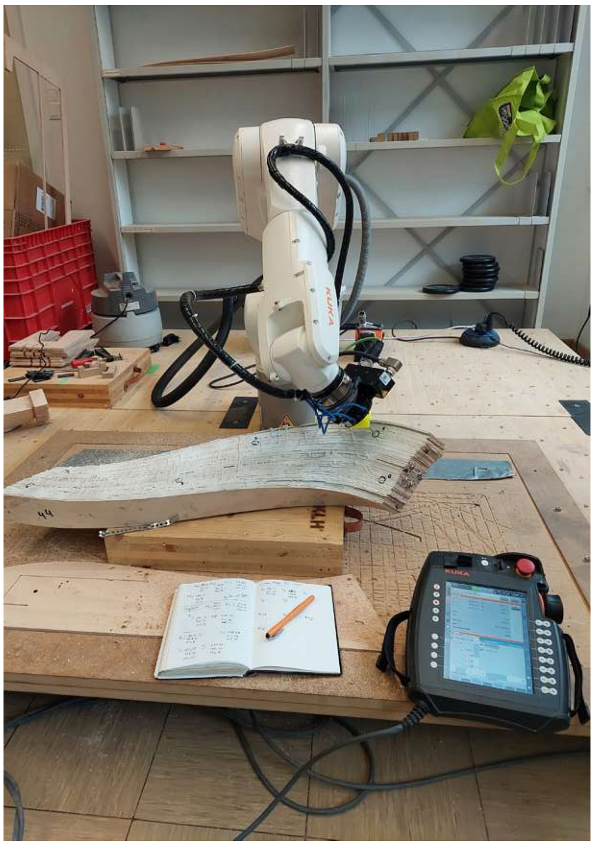


Figure: 102 - Localizing points in robots coordinate space 01.

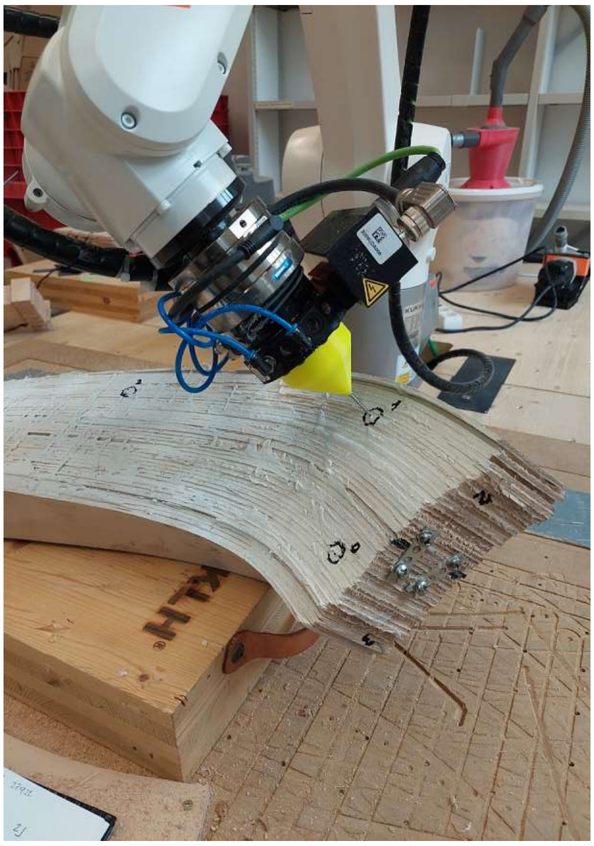


Figure: 103 - Localizing points in robots coordinate space 02.

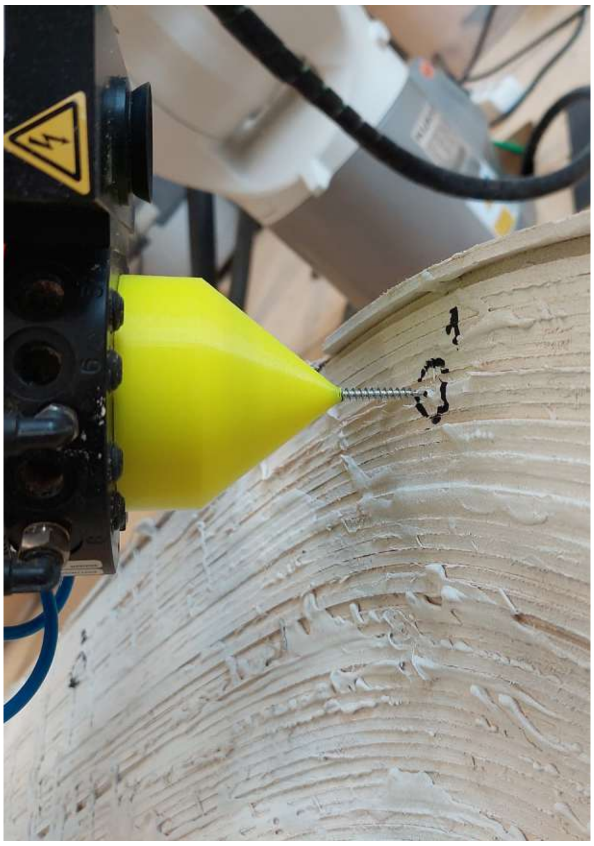


Figure: 104 - Localizing points in robots coordinate space 03.

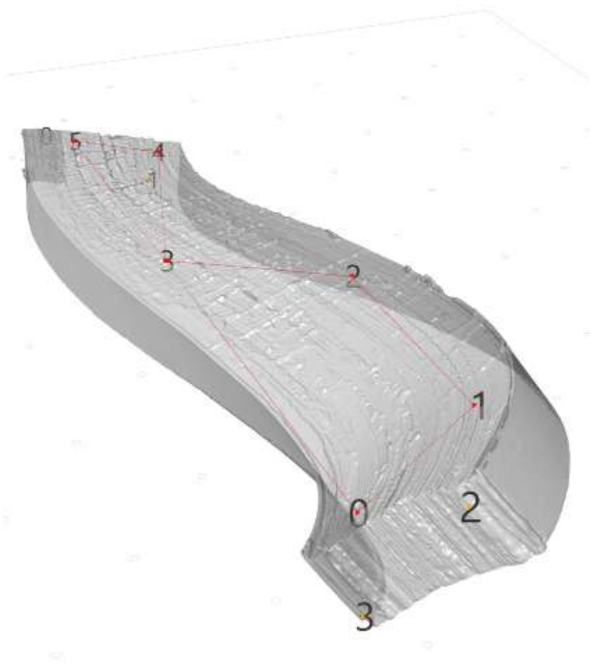


Figure: 105 - Transfered points onto the laser scanned model.

4.4.3.5 Toolpathing

As in the previous chapter, each work step was planned, visualized, and checked in Grasshopper, which turned out to be a robust and flexible way of working. Once created, sequences could be reused with few or no changes. Since the basic methods and scripts had already been developed, relatively little effort was required to plan the milling process for an entire beam in one go.

The main challenge was that only four sides of a beam could be milled in one pass, meaning the workpiece had to be repositioned once in the work area. Initial concerns that the two milling passes would then not fit on top of each other were not confirmed. Due to the planned tolerances in the form of excess material and the digital comparison using laser scans, both beams were manufactured without any major complications.

Only with B.3.0 the digital final beam was positioned slightly rotated in the model, a human error, which meant that the position of the two locks on one side was not exactly in the middle axis. However, since this was noticed in the first few minutes of the milling process, it was possible to fix positioning and losing the blank was avoided.

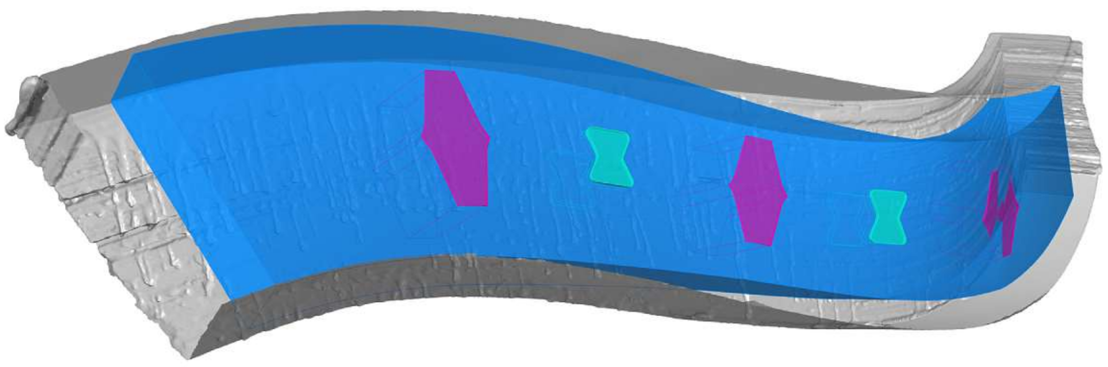


Figure: 106 - Relevant information for generating toolpaths.

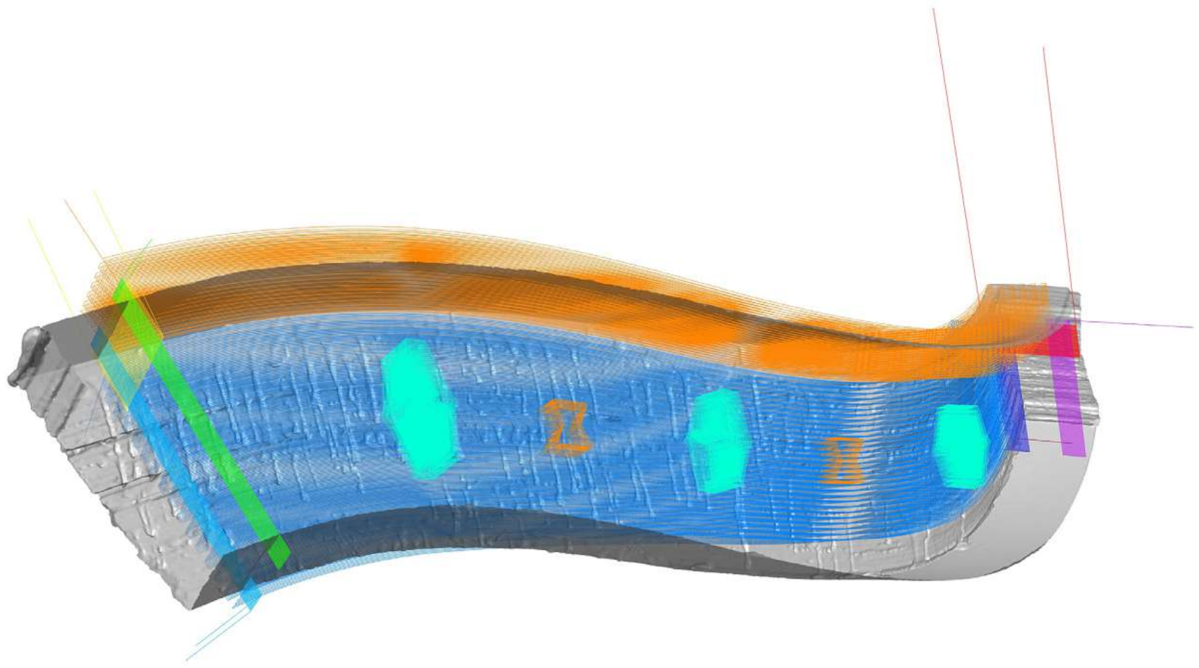


Figure: 107 - Final toolpaths for one side of the beam.

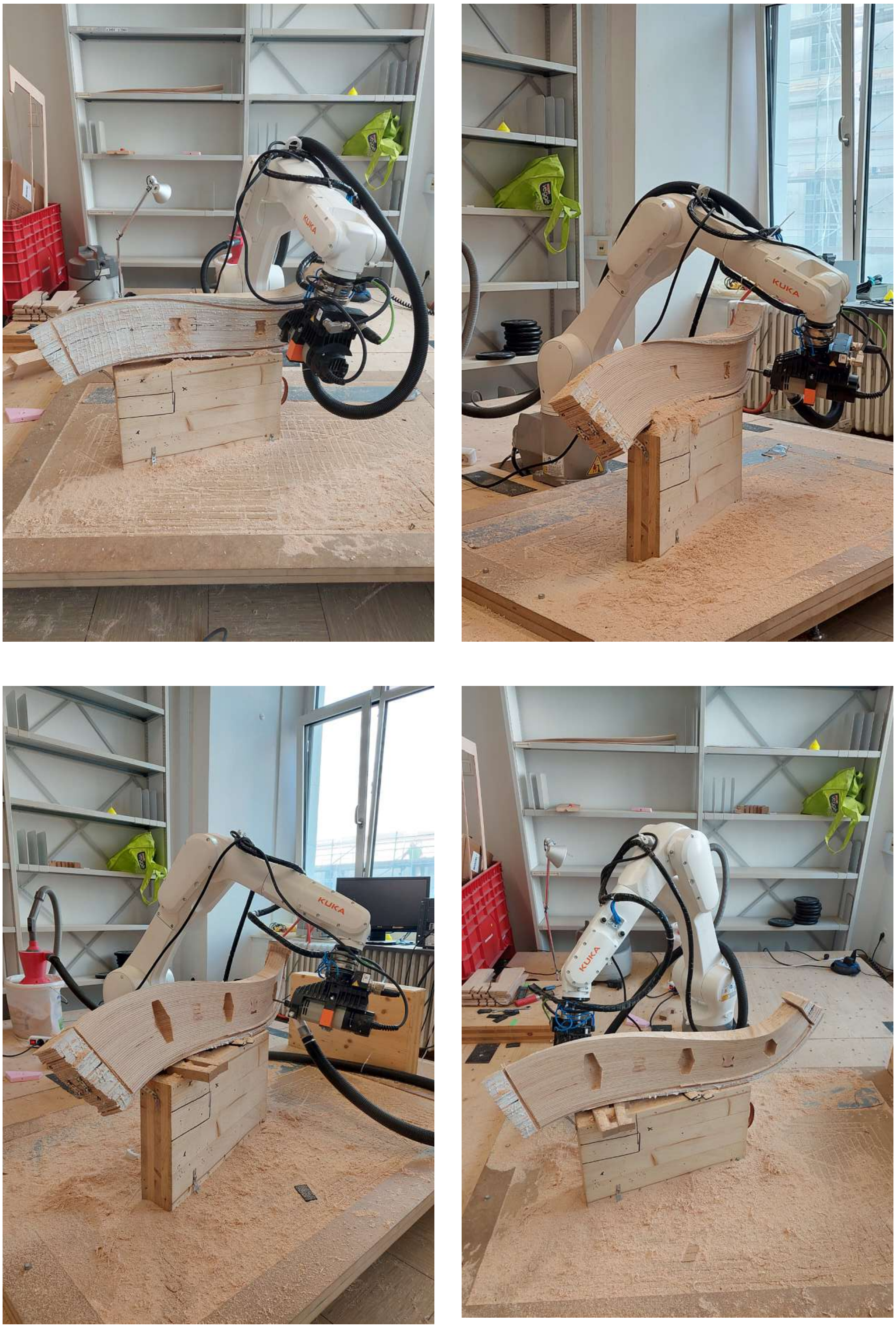
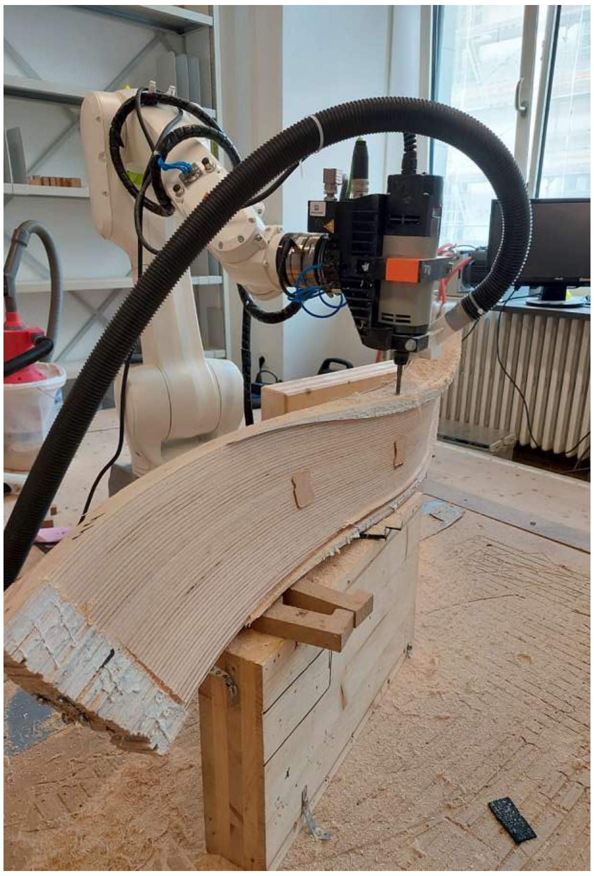
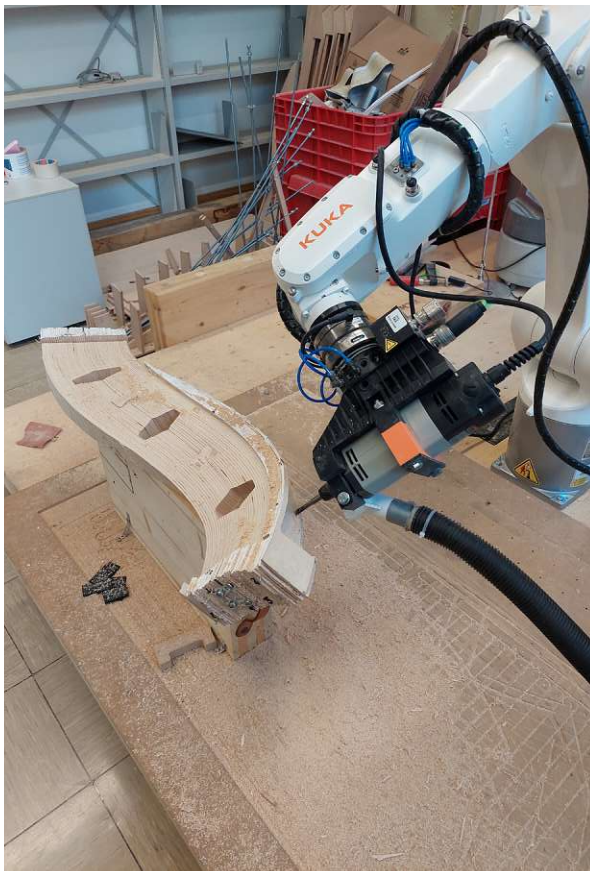


Figure: 108 - Robotic.fabrication of B.3.0.



KUKA


KUKA

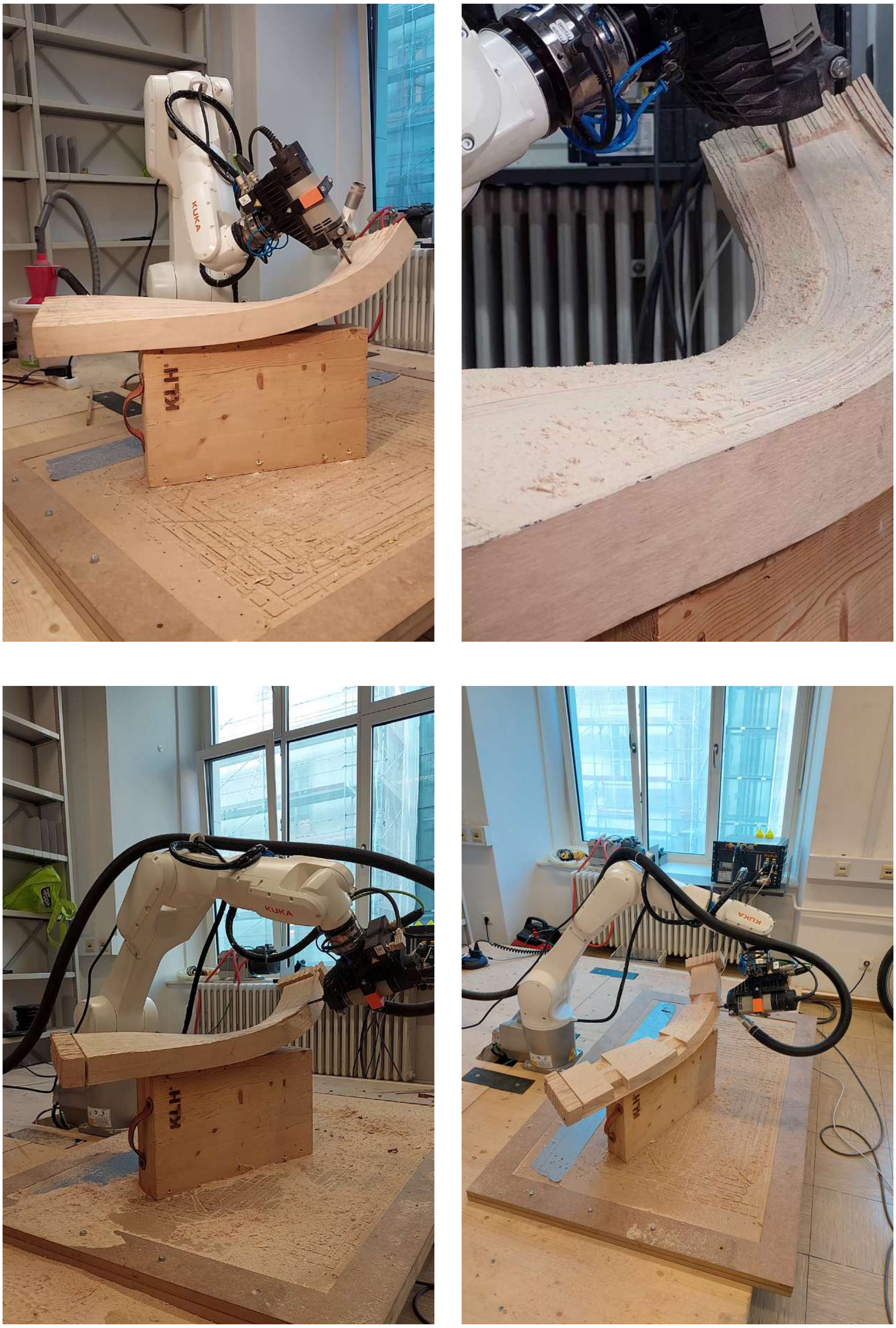
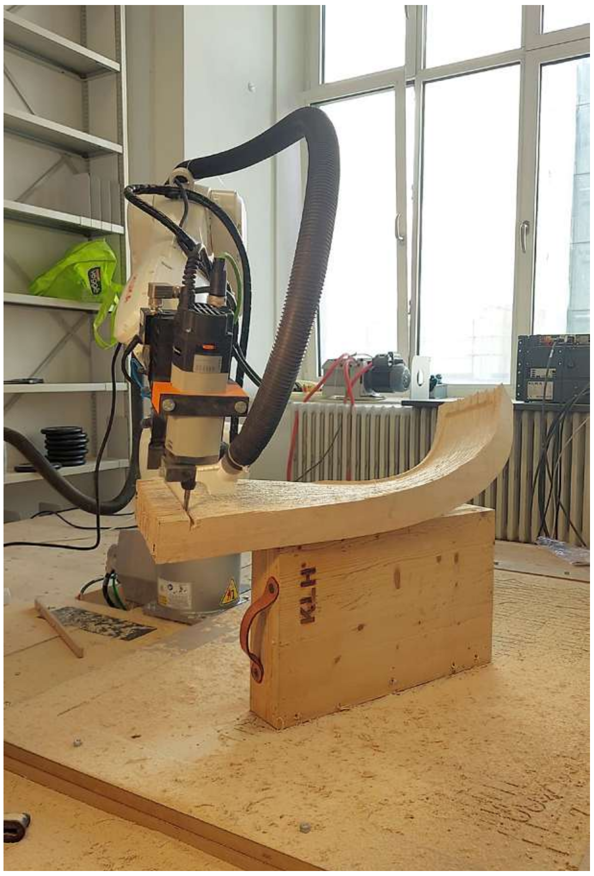
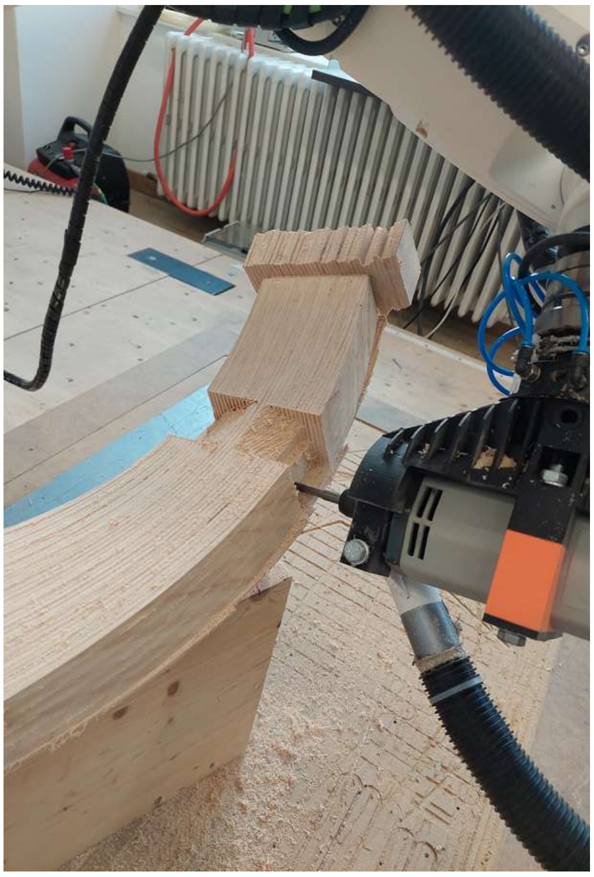
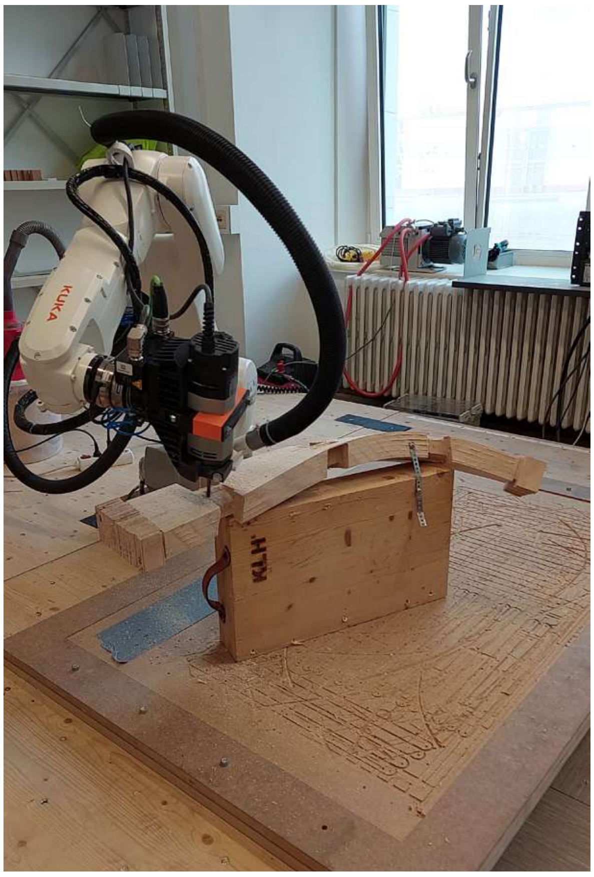


Figure: 109 - Robotic.fabrication of B.9.0.


KLH

KUKA


KUKA
KLH

4.4.4 Conclusion

This part demonstrates the basic procedure of fabricating doubly curved glulam beams, while utilizing the workflow developed in Chapter 4.2. Here the structured planning strategy proved to be very versatile, allowing for quick and easy manipulation of processes upstream affecting the end-product but also allowing for reversion of wrong decisions made along the chain of development. Because of the straightforward replacement of models, alternative modelling strategies could have been implemented with nearly no drawbacks.

Further, because of early implementation - feedback between different stage of the planning process are enabled, allowing for design iterations based on fabrication data once the data flow is established.

Although the complexity of planning for the project or specifically the end-product – two beams – is not that overwhelming, when it comes to geometric and machineability constraints it became quite tricky especially with limited resources. The complexity of controlling and monitoring a six-axis manipulator adds significant overhead to the process. Requiring the tracking of certain parameters like feedrate, possible collisions and how to avoid them in a generalized manner, material properties and so on. Each influencing the outcome or at least how fast and accurate one can fabricate.

Although manual laser-scanning proved to be a robust and accurate method to align the digital component with reality, lack of automation results in a highly laborious task. Firstly, scanning complex geometry to achieve good results needs patience and time, if not done right one ends up repeating this step until a satisfactory point cloud is produced.

Further, meshing can only be as good as the generated point cloud, also aligning multiple scans can get tedious. After this procedure one is faced with the problem of aligning the mesh with the modelled beam representation with each other and yet again both together in the work area of the robot. This step is labour-intensive and error-prone for untrained personal.

5 CONCLUSION

The presented project demonstrates an agile modelling workflow for design to production. Making use of web-based services eases data transfer as well as computational demand. Also, allowing for collaborative strategies apart from conventional file-sharing reducing possible merging conflicts and enabling remote stakeholders to operate on up-to-date information.

Although the project was mostly developed within Rhino and Grasshopper, in view of ongoing development on open-source interfaces and data exchange formats, the same workflow can already include a lot of non-/parametric environments.

In the first experiment, Chapter (4.2), an adaptive modelling workflow was established providing needed infrastructure for the course of this thesis. The principles described allowed the user to switch back and forth between different design iterations to evaluate possible solutions or to find the point where all went downhill.

Through the use of Speckle, it was possible to transfer data rich objects without loss of information from one stakeholder to another (also while switching between workplaces). By doing so, a collaborative workflow with separated concerns was established. More detailed modelling, interested in specialized information, was executed apart and unpaired from the reference model or the general planning process. This allowed consideration of only a limited amount of parameter required to solve part of the general design problem into sub-models/tasks.

By decoupling general design intentions from detailed design descriptions – the models, seamless replacement of later was enabled. Therefore a more robust and interlinked workflow was established, where components of the overall workflow can be exchanged or computed on demand.

Using Speckle also allowed the user to query needed information using text-based commands receiving just requested data, saving time, bandwidth, and processing power. Drawback hereby is that the user needs to know how the data is labelled, presupposing prior knowledge of how the data objects are structured.

However, architectural projects are usually one of a kind, distinct and highly individual projects. But this must not imply that each modelling workflow needs to be rethought or rebuilt from ground up. Guidelines and a good data base for generalized datatypes, representing common and regularly used components should be mandatory. By providing these as opensource, the AEC industry could boost ongoing development of interdisciplinary environment where seamless collaboration is eased.

Through the experiments with robotic milling flaws in the prior process were discovered. Despite that the process of robotic milling itself is straight forward, preparing data as well as materials is not. To meet required prerequisites a lot of parameters must be set early in the design process always keeping in mind how one wants to fabricate. Creating a lot of overhead to the process, only economically manageable with enough expertise involved from the beginning. Avoiding certain pitfalls which would consequentially lead to reworking potentially huge parts of the design.

Unfortunately, still a large portion of robotic fabrication involves manual labour. Right now, automation is still limited to certain tasks or for production with industrial fabrication lanes - producing the same item repeatedly. Developing appropriate techniques for more automated processing of complex singleton tasks, will help to make robotic fabrication for architectural applications feasible. By doing so, more direct feedback loops during the design process could help evaluate techniques and buildability of components.

6 REFERENCES

[1] Blumer Lehman
lehmann-gruppe.ch/holzbau/referenz/swatch-hauptsitz-biel.html
Accessed: 23.03.2022

[2] Boothroyd G. (1996)
“Design for Manufacture and Assembly: The Boothroyd-Dewhurst Experience.”
In: Huang G.Q. (eds) Design for X. Springer, Dordrecht,
DOI: 10.1007/97-94-011-3985-4_2

[3] Boothroyd G. (2005).
“Assembly Automation and Production Design”.
CRC Press. ISBN: 978-1-57444-643-2

[4] cnccookbook.com/online-cnc-training-courses-guides-help/
Accessed: 15.05.2022

[5] DCN - Dodge Construction Network, (2021)
“SmartMarket Report – Accelerating Digital Transformation Through BIM”,
Dodge Data & Analytics, https://proddrupalcontent.construction.com/
s3fs-public/DigitalTransformationBIMSMR_111021.pdf
Accessed: 10.03.2022

[6] Design To Production - Factsheet La Seine Musicale
https://www.designtoproduction.com/fileadmin/projektdownloads/d2p_cite_
musicale_factsheet_de.pdf
Accessed : 27.03.2022

[7] Design To Production - Cite du Temps
https://www.designtoproduction.com/
Accessed: 27.03.2022

[8] designtoproduction.com/fileadmin/images/projektbilder/d2p_cimu_04.jpg
Accessed: 27.03.2022

[9] designtoproduction.com/fileadmin/images/projektbilder/d2p_cimu_05.jpg
Accessed: 27.03.2022

[10] designtoproduction.com/fileadmin/images/projektbilder/d2p_cimu_03.jpg
Accessed: 27.03.2022

[11] discourse.mcneel.com/t/start-here/58566, McNeal
Accessed: 04.04.2022

[12] docs.oracle.com/javaee/6/tutorial/doc/gijqy.html,
Accessed: 17.02.2022

[13] Fielding R. T., (2000)
“Architectural Styles and the Design of Network-based Software Architectures”, https://www.ics.uci.edu/~fielding/pubs/dissertation/top.html,
University of California,
Accessed: 05.05.2022

[14] geeksforgeeks.org/what-is-web-api-and-why-we-use-it/,
Accessed: 18.02.2022

[15] Gillespie B., (2020)
“Compute Features”, https://developer.rhino3d.com/guides/compute/features/
Accessed: 04.04.2022

[16] Hess Timber
hesstimber.com/fileadmin/medien/aktuell/vortraege/IHF/IHF_2017_HESS_TIMBER_Hofmann_La_Seine_Musicale_Das_neue_Musikkulturzentrum_an_der_Seine.pdf
Accessed: 27.03.2022

[17] Lu W., Tan T., Xu J., Wang J., Shang G., Xue F.. (2020).
“Design for Manufacture and Assembly (DfMA) in construction: the old and the new. Architectural Engineering and Design Management.”
DOI: 10.1080/17452007.2020.1768505.

[18] Payne A., (2022)
https://developer.rhino3d.com/guides/compute/what-is-hops/,
https://developer.rhino3d.com/wip/guides/compute/how-hops-works/,
https://developer.rhino3d.com/guides/compute/deploy-to-iis/
Accessed: 04.04.2022

[19] phpenthusiast.com/blog/what-is-rest-api
Accessed - 05.04.2022

[20] Poinet P., Nicholas P., Tamke M., Ramsgaard Thomsen M., (2016)
“Multi-Scalar Modelling for Free-form Timber Structures”, Conference Paper,
https://www.researchgate.net/publication/308695849

[21] Poinet P. (2019)
“Enhancing Collaborative Practices in Architecture, Engineering and Construction through Multi-Scalar Modelling Methodologies “

[22] Reddy M., (2011)
“API Design for C++”,
ISBN: 978-0-12-385003-4

[23] Sato H., Nakahara Y., (1995)
"The Complete Japanese Joinery", Hartley & Marks Publishers Inc.,
ISBN: 978-0-88179-121-1

[24] Scheurer F., Stehling H., (2020) "New Paradigms for Digital Prefabrication in Architecture.", In: Design Transactions: Rethinking Information Modelling for a New Material Age (42 - 49), UCL Press,
ISBN: 978-1-78735-489-0

[25] Scheurer F., Stehling H., (2020)
"Large-Scale Free-Form Timber Gridshell: Digital Planning of the new Swatch Headquarters in Biel, Switzerland",
In: Fabricate 2020: Making Resilient Architecture (210 - 217), UCL Press.
ISBN: 978-1-78735-812-6

[26] sei.cmu.edu/our-work/software-architecture/,
Accessed - 23.11.2022 12:04

[27] Smid P., (2007)
"CNC Programming Handbook", Industrial Press,
ISBN: 0-8311-3347-3

[28] Smith D. (2007)
"An Introduction to Building Information Modelling (BIM)"
In: Journal of Building Information Modelling (12 – 14)

[29] structurae.net/en/media/334384-swatch-headquarters-cite-du-temps
Accessed: 23.03.2022

[30] Svilans T., Poinet P., Tamke M., Thomsen M. (2017).
"A Multi-scalar Approach for the Modelling and Fabrication of Free-Form Glue-Laminated Timber Structures.",

[31] Svilans T., Tamke M., Thomsen M., Runberger J., Strehlke K., Antemann M. (2019)."New Workflows for Digital Timber".
10.1007/978-3-030-03676-8_3.

[32] Speckle.Systems
https://speckle.guide/dev/
Accessed: 18.02.2022

[33] speckle.systems
Accessed: 18.02.2022

[34] united-bim.com/leading-countries-with-bim-adoption/
Accessed: 11.02.2022

[35] Usai S., Stehling H., (2017)
"La Seine Musical: A Case Study on Design for Manufacture and Assembly", In: Humanizing Digital Reality: Design Modelling Symposium Paris 2017, De Rycke K., Gengnagel C., Baverel O., Burry J., Mueller C., Ngyuen M., Rahm P.,Ramsgaard Thomsen M., eds. Springer Singapore

[36] Weinan E., (2011)
"Principles of Multiscale Modelling", Cambridge University Press

[37] Zadeh P.,Staub-French S., Calderon F., Chikhi I., Poirier E, Chudasma D., Huang S. (2018).
"Building Information Modeling (BIM) and Design for Manufacturing and Assembly (DfMA) for Mass Timber Construction."

[38] https://images.adsttc.com/media/images/5952/1c47/b22e/38e9/2900/0781/slideshow/LA_SEINE_MUSICALE_183.jpg?1498553409
Accessed: 27.03.2022

[39] Hauberg, Jørgen & Tamke, Martin & Thomsen, Mette. (2012).
"Research by Design - a Research and Teaching Concept..",
Proceedings of the Conference Theory by Design 29-31 (pp.335-342)

List of Figures