Stock discretized structural timber elements

A structural evaluation of a computational optimized timber load-bearing system discretized by available stockpiles

TU Delft
BK Bouwkunde

MSc Thesis by D. Weerdesteijn
18-06-2024

Delft University of Technology
Faculty of Architecture and the Built Environment

Stock discretized structural timber elements

A structural evaluation on a computational optimized timber load-bearing system discretized by available stockpiles

Key Words: Discrete, Timber, Optimisation, Wood, Stockpile, Inventory, Matching, Dynamic programming, Structural

Author
Daan Weerdesteijn | 5699061
Master Thesis Research
Delft | 2024

MSc in Architecture, Urbanism and Building Sciences
Building Technology track

Mentors
Dr. Stijn Brancart | Chair of Structural Design & Mechanics
Ir. Hans Hoogenboom | Chair of Design informatics

Board of examiners delegate
Ir. Leontine de Wit | Architect

TU Delft
Delft University of Technology
Faculty of Architecture and the Built Environment

Abstract

The current sustainability crisis requires a shift in the way material availability is currently considered. The current trend is to dispose of material easily after it has served its first purpose, while often the material still has potential to be reused. The construction sector is a large contributor of the extraction of virgin resources, leading in the case of timber to deforestation. In The Netherlands alone around 1.740 kiloton waste wood is collected annually. 23% of this wood consists of solid non-glued or treated reusable timber, translating roughly to a waste stream of 400kton of reusable wood that is discarded. The current design approach requires a shift in order to accommodate the reuse of components. Instead of design for manufacturing designers should focus on designing with what has already been manufactured. The design concept of discrete timber is well suited for this design approach as it involves building a structure or element out of smaller parts. This method allows for direct reuse and repurposing of a varying stock of discharged timber pieces, effectively enhancing the life-cycle of timber from a currently down-cycling scenario into a circular loop. This will aid in achieving a fully circular economy by 2050 and the reduction of virgin material usage in the construction sector. Using programming and optimisation techniques, this thesis focusses on creating an efficient and adaptable structural system from a varying stock of reclaimed timber pieces while maximizing the future reuse potential of the used parts.

Literature is reviewed and encompasses three domains: Waste wood, Discrete timber, and Optimisation. Firstly, in the "reuse" domain a comprehensive summary of the waste wood market in the Netherlands is given. Questions like: Which waste streams can be identified for potential direct reuse, and what kinds of waste streams can these streams consist of in terms of sizes, lengths, and structural properties? will be investigated. Secondly, in the domain of "discrete timber.", existing discrete structures will be compared and assessed. A variety of factors, including dimensional constraints, mechanical qualities, fire resistance, and fabrication, will be assessed. On top of that, the discrete design ideology will also be emphasised. Lastly, within the domain of "optimisation," an extensive examination of various optimisation techniques and readily available algorithms will determine the best procedure and method of resolving the matching problem involving how to bring all parts together in a viable structural assembly.

The literature review concluded numerous joint options. A massively layered assembly was chosen over a reciprocal structure with the devised design criteria: *Reversible joints, ductile system, minimize re-fabrication, tectonic flexibility and use of unique parts.* The acquired insight in the waste wood market in The Netherlands is put in a database that simulates the available waste wood. The ground structure method is selected as most suitable structural optimisation method and dynamic programming is used to solve the combinatorial problems concerning matching the pieces found in the database.

A modular structural system was designed adhering to the design criteria. The parts are connected by dowels and external plywood links, generating a frictional resistance and enacting, to an extend, composite behaviour between the parts. The dowel holes are drilled in a modular pattern, enhancing the future reuse potential and making the possibility to disassemble the whole system.

An algorithm is created in the visual programming environment Grasshopper using the build-in Python function. The algorithm discretizes a given design space into the pieces found in the database by sequentially solving three combinatorial problems, reflecting the x, y and z direction. The algorithm optimises the placement of the pieces so that higher strength grade pieces are placed in area with higher stress levels. The resulting 3D compositions are translated to a finite element model, for which Karamba3D is used. The assembly is optimised by removing all non vital structural parts resulting in a final efficient structure. The parts are stored in a database to accommodate future reuse.

The algorithm's performance is tested on stability of results, optimisation method, size influence of parts, filling rate of the design space, strength grade influence and buckling. Overall, larger and longer parts provide more stiffness to the structure while smaller parts allow for a better optimal final composition, concluding that a highly diverse composition will result in a global optimum. This work serves as a prove of concept for designing with a highly versatile stock of reclaimed components.

Table of contents

Table of contents

1. Introduction

1.1 Background

In the past century, technology has advanced quickly across all industries. The computer, a significant new innovation that began a technological revolution, was created in the 1970s. The computer allowed for many new possibilities and innovation in every industry. In the field of engineering and manufacturing a considerable deal of flexibility in designs became possible. A new age of computer-aided design and engineering were getting a prevalent place in the design process.

In the fields of design and engineering this allowed for the creation of intricate geometries by 3D software and fast execution of computationally demanding calculations. On the other hand significant advancements in the production sector led to the creation of powerful computer-aided machines, allowing for a new market, distinguished by affordable and previously unattainable designs. Computer-aided machines, like CNC- and robotic drilling-machines, are able to translate a 3D model and create the exact geometry in a piece of material. This makes it possible to create complex shapes quickly and with great accuracy. This allows designers the flexibility to parametrically build structures made up of numerous distinct components. This is where the origin of the theme discrete timber (re)emerges.

Discrete timber is a design concept that involves building a structure or element out of smaller parts that come together to form a whole. (Chen et al., 2021). A system like this is not new; numerous structures are composed of several smaller parts that are joined together. Nevertheless these structures are usually made simple, with a low amount of parts and uniform so that assembly and production can be completed without excessive planning and sorting. The technological innovations in robotic production processes and parametric design have made it possible to effectively design, sort, use and produce more complex structures on large scale. This has revitalised the idea of discrete systems and paves the road for a new way of designing. Discrete timber can be seen on each scale. On a large scale it can be implemented by creating building modules and linking them together. But also on a much smaller scale by creating an element such as a beam or a wall out of smaller pieces. A discrete system's ability to be very flexible and adaptable is one of its main advantages. As the structure composes of separate parts it makes it easy to fit within different sets of requirements and, its topology, is therefore more straightforward to optimise to the forces acting on it. Due to the discrete nature of the system there would be no need to cut solid timber to achieve a optimised topology. In other words, it is convenient to create a composition of parts that are only structurally required. This reduces material consumption and weight.

Furthermore, discrete timber could contribute to a more circular construction industry by using reclaimed timber pieces as parts.

As for now the building industry still produces large amounts of timber waste annually. In many cases, wood has a shorter functional lifetime than its potential technical lifetime and is discarded. This contributes to high amounts of waste while there is a lot of potential for reuse. A study by Tauw estimated that in 2017 in the Netherlands alone 1740 kiloton waste wood was collected of which 23% consists of solid timber (Bruggen & Zwaag, 2017). This translates to a potential reusable material stream of 385 kiloton timber annually. To this day this timber is often not reused as it consist of small unusable pieces and ends up being thrown away and shredded as fuel for the bio-energy industry or as material for the engineered timber board industry. Shredding timber and producing engineered boards is not considered to be circular as the timber can never be reused again due to the addition of glue, see figure 1.1. This while reusing a solid timber beam is considered to be circular.

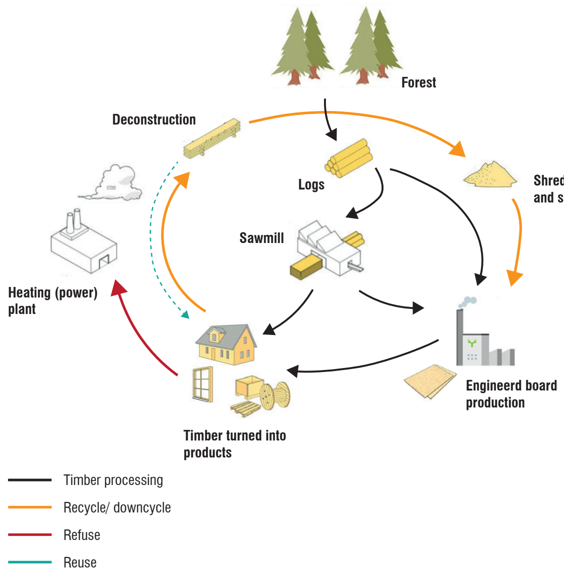


Figure 1.1: common seen end-of-life cycle of timber (own ill.)

As discrete timber seems to be a promising system, there is still much unknown and more to discover. The Southern University of Denmark (SDU) recently conducted research on discrete load-bearing systems on a small scale structural element (Kunic, Naboni, et al., 2021), (Hansen et al., 2021). This study represents some of the latest developments in the field of structural discrete timber. Paula (2023) from TU Delft did a similar study on the large scale architectural potential of discrete timber as a building system. Both studies are focussed especially on the production and assembly process of the timber elements. The SDU also did studies on the design and structural understanding of connection type between the discretized elements. However, still a lot is unknown, like a detailed understanding of the structural behaviour of the system as whole, the interaction between different linked discrete elements, the circular aspect and potential, and how efficient compositions can be created. Furthermore, the used parts in the study are often all of the same size and are not linked to an available stock. Eijk (2021) from TU Delft did research to the waste wood market in The Netherlands and created a computational model to create wall elements from an available stock. A recent study of Heijne (2023) shows that with a Best-Fit algorithm a variable stock can be assigned to a structural system. A computational model linked to a stock in combination with a discrete system could have great potential to enhance the circularity of timber. Evaluating the structural behaviour of a discrete system more closely and using varying reclaimed timber parts to create diverse optimised composition could yield valuable new insights. This study will investigate the potential of reclaimed discrete structural timber systems.

1.2 Problem statement

Considering the world population is still on the rise, simultaneously the need for housing within counties all over the world is bigger than ever (United Nations, 2023). The building industry is responsible for a large parts of the extraction of non-renewable resources. (Kunic & Naboni, 2023a) and (Bertin et al., 2022). This sustainability crisis requires especially this industry to reduce virgin material consumption.

As for now the construction sector tries to take its responsibility in lowering its environmental impact and is moving towards the construction of zero energy buildings (Pronk et al., 2022). Many projects and initiatives have started to insulate buildings better, to install PV-panels, use sustainable materials like wood and find smart design solutions to minimize the operational carbon of the building, As this aspect has taken off quite well another, still often underrated, aspect of the sustainable construction process is becoming more pressing: the embodied carbon (Pronk et al., 2022). This is the total amount of carbon emitted in all phases during and after construction and includes also aspects like transport, maintenance and end-of-life processing (LETI, 2020). Ignoring this aspect will prevent the construction industry from being able to sufficiently reduce its impact on the environment.

Bertin et al. (2022) states that 50% of a buildings embodied carbon is the result of its load-

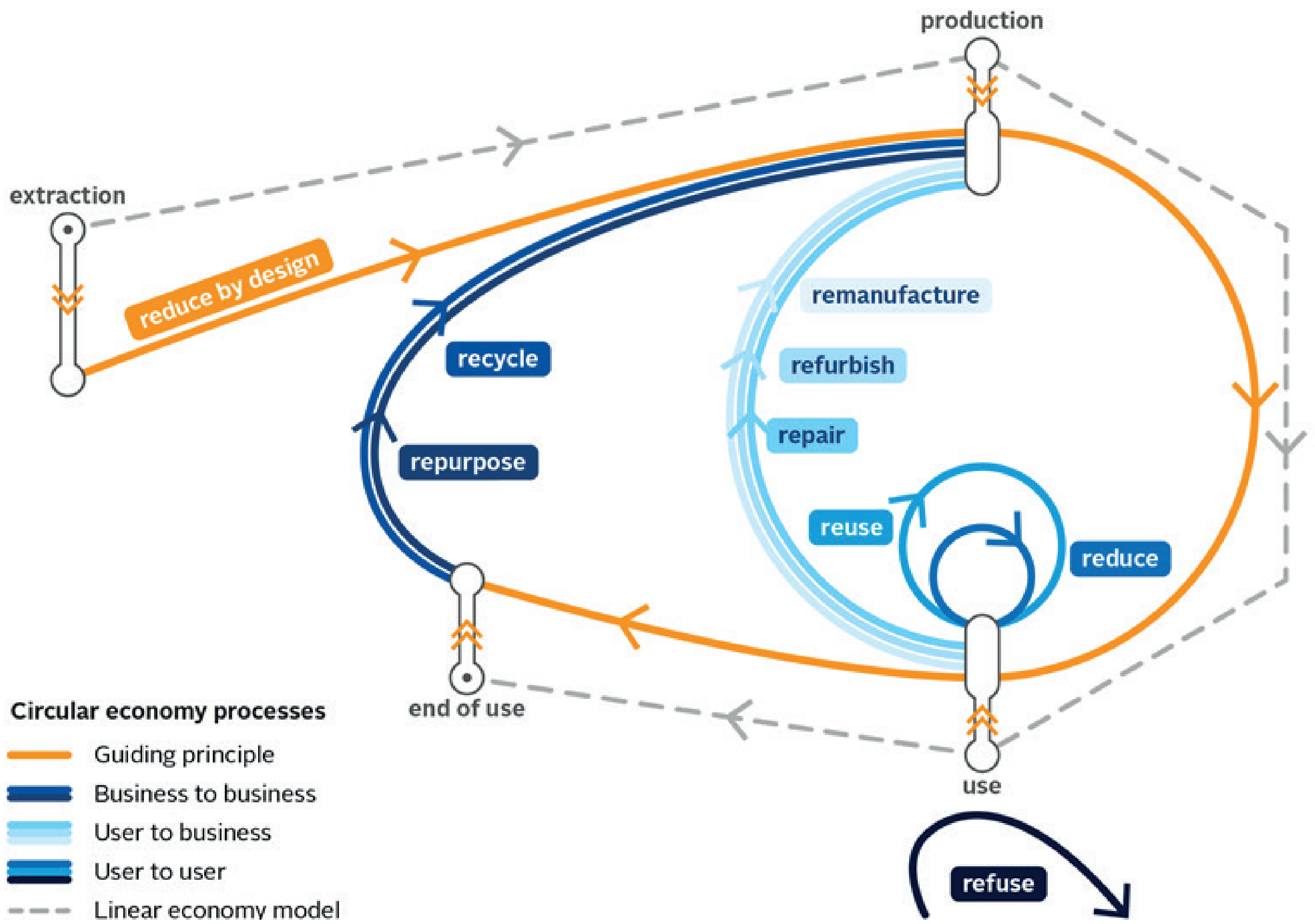


Figure 1.2: An approach to circularity (UN Environment Programme, 2019)

bearing structure. This therefore identifies as a influential area in which substantial impact can be made, and on which this research will focus. Material selection is a highly influencing factor which is able to enhance the embodied carbon of a building significantly (Fang et al., 2023). For this reason timber has become a popular material again after a long history of industrialization (Kunic & Naboni, 2023a). Building with timber helps in cutting carbon due to its positive ecological footprint (Kunic, Naboni, et al., 2021). This means that the material absorbs more CO_2 than is emitted during its processing. As timber is becoming a more predominant building material and the embodied carbon becomes a more important factor, the need for a better organized circular loop gets more pressing and cannot be considered separately from the real impact of the material (McArthur et al., 2015).

Sadly, despite its potential for substantial contribution to a more sustainable industry the life-cycle of timber is still mainly focussed on recycling/ down-cycling instead of reusing see figure 1.2. Currently a lot of waste wood is shredded and used as fuel for bio-energy or for the production of engineered timber boards while these solid pieces have great potential for direct reuse. Timber is not a finite material but forests are no infinite source either. It takes a long time to grow and harvest a tree so it is not considered environmental responsible to simply throw out used timber (Kunic & Naboni, 2023a). In the Netherlands alone around 1.740 kiloton waste wood is collected annually, of which 23% consists of solid non-glued or treated reusable timber (Bruggen & Zwaag, 2017). Translating roughly to a waste stream of 400kton of reusable wood that is discarded.

This shows that timber can give the impression of being a durable material while its impact on the environment is still substantial when the embodied carbon is considered. According to Pronk et al. (2022) the impact of timber is related to the large quantity of extraction of virgin resources and the down-cycling process at its end of life. Furthermore, the demand for timber is nowadays larger than the available supply which speeds up deforestation. While timber is a material that has great potential to make the construction process fully carbon neutral, a better end-of-life scenario is essential. Figure 2 shows the possible end of life scenarios: a shift from business to business which translates to down-cycling, i.e. shredding, to a user to business or even user to user is needed to reduce the embodied carbon.

Direct reuse of timber is often not considered or found to be difficult for numerous reasons. Left-over pieces are bound to have different lengths and cross-sections, which makes it challenging to directly reuse the pieces in a new design (Giordano et al., 2023). Furthermore, the structural capacity of these scrap pieces is often uncertain and sorting and organizing a stockpile of timber is costly and without demand. Therefore, a shift in the current design process is required to allow for direct reuse of components. It is necessary to design using what has already been manufactured rather than designing for manufacturing. (Gorgolewski, 2008).

Discrete timber systems can start having an more important role in the construction sector. A discrete system composes of smaller elements and with advances in computational design and planning, it is possible to create structures with parts that can differ in size and length. By introducing a stock-constrained design a catalogued stockpile can be directly projected on the design reducing cutting and material costs. Direct reuse allows for immediate reduction of waste production and virgin material use, lowering the embodied carbon instantly and closing a now down-cycling material scenario (Brütting et al., 2020). Discrete timber systems can assist in repurposing waste wood, enabling the realisation of a net zero carbon emissions and a fully circular economy by 2050 (European parliament, 2021).

Additionally, a discrete system can also aid in achieving a material efficient yet flexible structural system. Topology optimization is employed to reduce self weight and material usage. This is a technique which optimizes an element by removing parts where forces are negligible or small (Holmberg et al., 2013). This results in light weight constructions and respectively lower amounts of required concrete for foundations. A sustainable way of designing, however by optimizing on one scenario the element usually has no excess capacity left to accommodate future changes in functions or loads. In other words, a design becomes fully constricted by its load-bearing structure and would have to be demolished at any future adaptation. Discrete systems can be designed for disassembly so that damaged parts can be removed or new parts can be added if requirements change. The structural element can therefore profit from both future flexibility while being optimised to the current scenario.

As described in “1.1 background” some research has already been done on a structural discrete load-bearing system and stock-constrained design. This study will built further on already existing research and aims to identity the potential structural usage and limitations of a discrete element. This study will add value to the already existing knowledge by exploring the circular and structural potential of the system by creating a program that allows for the design of a stock-constrained structural system.

1.3 Research questions

Based on the above stated problem concerning the embodied carbon of timber and still unknown limitations of discrete structural elements a research question is formulated as:

How can programming be utilised to create a discrete structural system using reclaimed timber parts that maximizes efficiency and adaptability but minimizes the need for virgin materials in construction?

To help answer this question a few sub-questions have been formulated. These questions can be divided into a literature review and in research through design. The first three questions serve to help guide the literature review and are formulated as:

- Which timber waste streams can be identified to be useful for the concept “direct reuse” and what are typical components, lengths, sizes and quality of this available timber?
- What connection types can be identified to create a adaptable discrete system considering flexibility, structural performance and manufacturability?
- What method for structural optimisation constrained by the fixed dimensions of a stock is best suited for a discrete timber system?

The following question are sub-questions that will help in answering the main research question and are formulated as:

1. How can the discrete system accommodate future adaptations and requirements?
2. What are the structural limitations and recommended configurations for such a system?
3. How can a program be created that accommodates a stock constrained discrete design method effectively in practice?
4. What are the gains and limitations of this discrete system in comparison to traditional systems?

1.4 Research aim

This research aims to get a better understanding in the structural behaviour of discrete timber systems. A computational tool will be developed which can generate a structural system from a versatile stock of waste wood. This stock will have to be stored in a database and linked to this tool to simulate real time conditions. This thesis will be a proof of concept for the construction sector, ultimately sparking other innovative ideas for implementing this new design strategy. Hereby this study paves the way in making the life cycle of timber more circular, reducing virgin material consumption. This contributes to achieving the goals set in the Paris agreement from 2015 to become net zero by 2050 and obtain a fully circular economy. It tries to accommodate a shift in the design process from a linear, down-cycling economy to a circular economy in which the concept, design with what already is manufactured, is the guiding principle.

1.5 Research methodology

The study is divided into three phases. Firstly, a literature review will be conducted in which general information will be gathered on three topics: Reuse, discrete timber and optimization. The results of this review will outline the boundaries and constraints for the next phase, the development phase. In this phase the acquired knowledge will be combined to design a structural system and create a parametric optimisation model that is able to create a discrete composition of varying stock and can indicate its structural performance. The performance of this model will be tested in the next phase, the test phase. In this phase different scenario’s can be tested, evaluated and the impact calculated. With these results a recommendation can be made on the feasibility of a reclaimed structural discrete timber system in practice. A global overview with research topics can be seen in figure 1.3.

More specifically, three domains are reviewed in the first phase, as illustrated graphically in figure 1.4. Current literature in the "reuse" domain will be reviewed in order to give a comprehensive summary of waste wood in the Netherlands. Questions like: Which waste streams can be identified for potential direct reuse, and what kinds of waste streams can these streams consist of in terms of sizes, lengths, and structural properties will be investigated? This literature study's findings will produce a database of available stocks that the

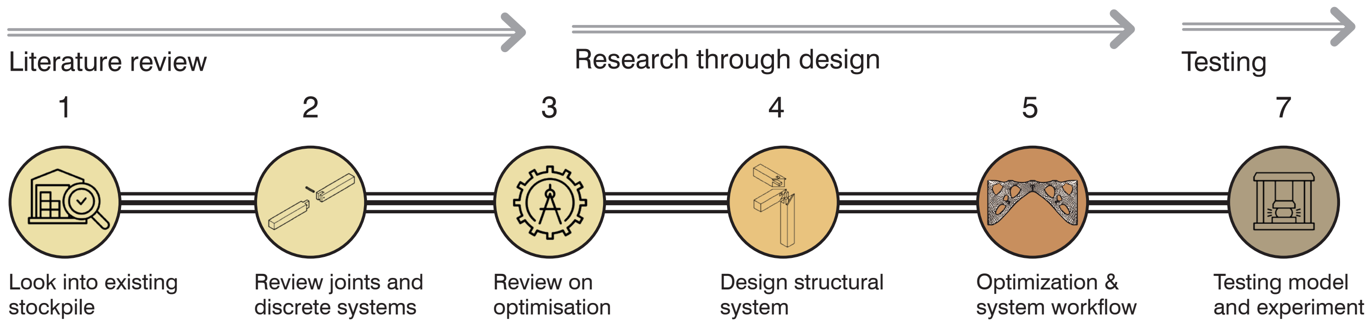


Figure 1.3: Global research plan, the colours represent research themes, light yellow: circularity, yellow: design, brown: structural, orange: computational

computational model can be connected to. Within the domain of "optimisation," an extensive examination of various optimisation techniques and readily available algorithms will determine the best procedure and method of resolving the matching problem. The design of the structural element will be covered in the final domain of the literature review, "discrete timber." Existing discrete structures will be compared and assessed in this section in order to identify knowledge gaps and implement current solutions. A variety of factors, including dimensional constraints, mechanical qualities, fire resistance, and fabrication, will be assessed. On top of that, the discrete design ideology will also be emphasised. The limitations and guidelines for designing a stock-constrained structural discrete timber element will be produced by the findings of this literature review.

The findings of the literature review will also serve as a basis for the design process in the following phase, "design". Initially, the joints and components geometry are designed. After that, an algorithm will have to be created to solve the matching problem. For preliminary structural analysis and behaviour, this matching tool will be expanded to a structural model. The computational tool can be updated and the final detailed model created based on new insights.

Once the model is ready for testing a case study can be created aiding in testing the performance of both the computational model as well as the discrete structure. This case study will help to highlight problems related to connectivity between other elements, foundations or floors.

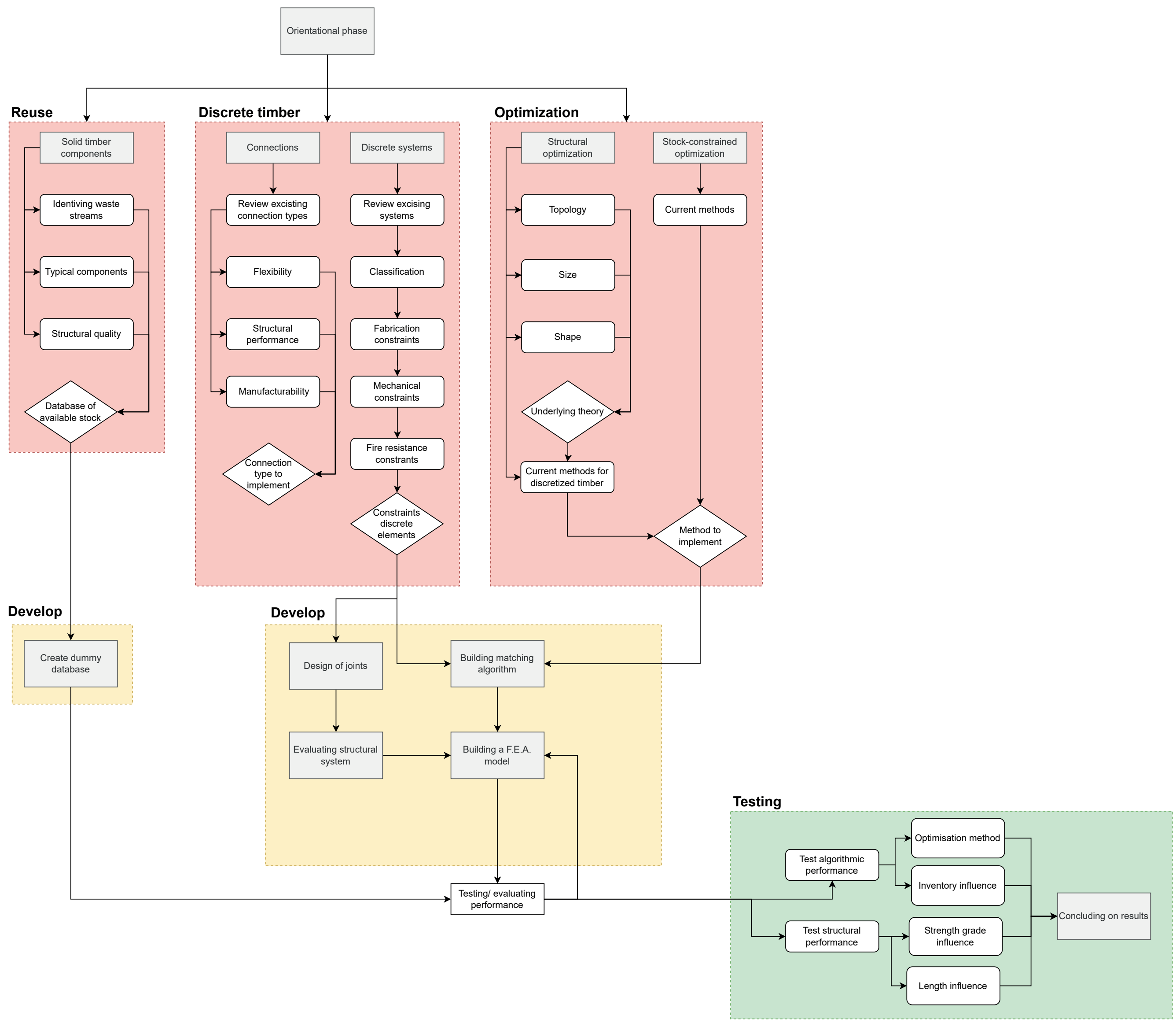


Figure 1.4: Detailed research plan, the colours represent research methods, red: literature review, yellow: design, green: development

I

Research Phase

2. Timber and discrete systems

2.1 Wood as a building material

Timber has been the main building material for the biggest part of history, more than 80% of all buildings were constructed with timber up to the nineteenth century (Cheret et al., 2013). In this time period the versatile and heterogeneity properties of wood were accepted and with slow but precise processing techniques the varying pieces were carefully brought together in a building. Unlike now, where lesser pieces are discarded rather than used, employing structural lesser pieces was common practise and were allocated to specific non load-bearing elements (Menges et al, 2016). When the industrial revolution took off, wood became less and less interesting to use as a building material. New synthetic materials like iron, steel and concrete which were more homogeneous, strong and could be harvested in great quantities became more dominant and caused engineers to question the traditional accepted understanding of wood. Over two centuries this shift in the design paradigm lead to the loss of valuable knowledge of wood construction techniques and material specific design which, furthermore resulted in a very polluting construction industry. The recent digital revolution, and climate crisis have revived the possibilities and the demand for wood construction yet again (Menges et al, 2016).

Mechanical behaviour of timber

Next to the environmental advantages of wood it also performs structurally remarkable. It has similar compressive strengths as lower-end strength classes of concrete and is much lighter than either steel or concrete (Cheret et al., 2013). Moreover, wood is a reasonable insulating material as it consists on a microscopic level out of small fibrous “tubes” filled with air or moisture. These “tubes” are naturally oriented in only one direction, the fibre direction. That makes wood an orthotropic material what means that the properties of wood are different in two or more directions. In the fibre direction (referred to as 0°), wood has different properties than transverse to the fibre direction (referred to as 90°) see figure 2.1. Wood is quite strong along the grain in both tension and compression. Compression along the grain is characterised with strain softening. This is an effect that causes a decrease in the material's strength due to increasing levels of stress and extensive deformation. The larger the deformation the weaker the material becomes.

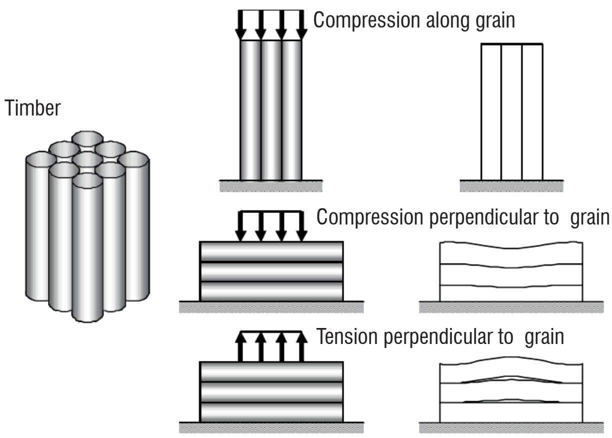


Figure 2.1: Timber structural behaviour on microscopic level (Munck et al., 2011)

Perpendicular to the fibre direction, the wood is much less strong. On compression, the wood can take a reasonable load because the fibres are flattened, what causes strain hardening. This is the opposite effect of strain softening and causes the materials strength to increase due to plastic deformation. The deformation causes dislocations within the material's fibres to become more densely packed, leading to increased strength. When pulled, wood is very weak and cracks can quickly appear because the fibres are pulled apart.

This concludes, that in tension timber will show brittle behaviour and in compression ductile. For shear strength along the grain the four components are almost identical. Except for the shear perpendicular to the grain, also known as the rolling shear, which is significantly lower. These properties are also illustrated in the stress-strain diagram in figure 2.2 and can lead to complicated calculations at junctions where several forces come together in different directions (J. L. Hansen et al., 2023) and (Munck et al., 2011).

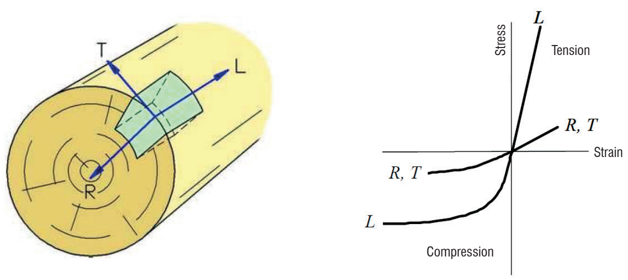


Figure 2.2: stress strain diagram of wood (Munck et al., 2011).

Loading wood at an angle causes a combination of stresses in the parallel and perpendicular directions as well as both on the shear plane. Strength values at an angle to the wood fibres are somewhere between those parallel and perpendicular to the fibre. The strength at an angle is ultimately limited by the tensile strength parallel to the fibre, the shear strength and the tensile strength perpendicular to the fibre. Figure 2.3 shows the behaviour of the strength of wood at different angles. The graph indicates a sharp decrease as the angle increases. The tensile strength of wood at an angle of 20 degrees is only 25% of the original tensile strength parallel to the fibre. The effects for compressive strains are less dramatic but it shows that strength values can differ a lot and that realising this trait is crucial for the design process (Munck et al., 2011).

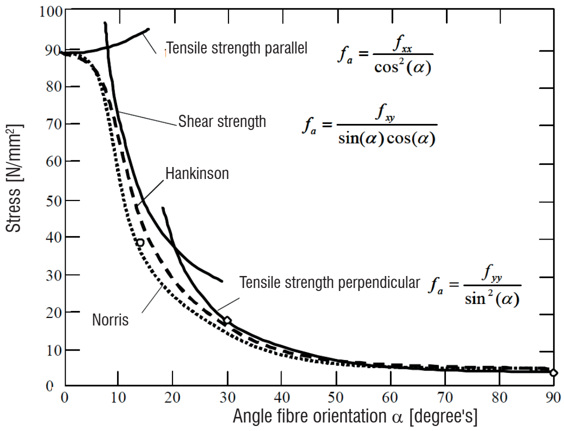


Figure 2.3: strength degrease in various angles, stress values are based on 100% flawless wood and do not represent the design strength, adapted from (Munck et al., 2011).

Structural properties of timber

Wood, like concrete, is available in different types and different strength classes. Mainly, all wood species are classified as softwood or hardwood. Hardwood is much heavier and denser and therefore more resistant to moisture than softwood and also has other applications. Hardwood is often used for e.g. window frames and other outdoor applications and softwood as construction timber where there is less to little moisture. After all, wood can deform greatly when in contact with high moisture levels. Wood shows a wide range of mechanical properties due to its anatomical structure and natural growth. Modern standards therefore calculate with the 5% undercut values for strength checks in UGT and with the average values for deformation-related checks BGT. For sawn timber, a distinction is made between C and D strength classes where C classes are meant for softwood and D classes for hardwood (Munck et al., 2011). Classes for engineered timber also exits but these are not relevant for this research project. The characteristic values of wood are shown in figure 2.4.

Shrinkage and swelling

Because wood consists of hollow cells on an anatomical level, moisture plays an important role. With long-term exposure to moisture, wood shrinks and expands. As the wood fibres are softened, it loses its strength and can develop cracking, mould or wood rot. The material also loses its strength at internal moisture levels above 20% (Munck et al., 2011). Besides changes in moisture, wood is also very sensitive to changes in temperature, which can cause it to expand or shrink. In addition, there is also the creep behavior of wood. Prolonged loading causes wood to creep, resulting in additional deformation. The strength of wood decreases when long term loads are applied. To account for all these effects in structural verifications wood is classified in climate classes. These classes provide reduction factors for specific moisture scenario's and load-durations. In short, wood is an active material that depends on many external influences. Shrinkage and expansion differ according to fibre direction and are very important to take into account in the design. When using waste wood, it is possible that different types of wood are combined with each other, either because the distinction is not made or because there is not enough supply. Especially when combining different wood species, this can cause unwanted problems because each wood species has different shrinkage and creep factors.

Design strategies for wood

The rich history, deep cultural roots and positive ecological characteristics of wood make it a fascinating material. With the current environmental crisis it cannot be considered as an outdated material with inferior characteristics any longer. Especially due to rapid technological advances in design, simulation and fabrication it can be regarded as the most promising

EN 338 - Table 1 - Strength classes - Characteristic Values

Wood Class	Strength Properties (in N/mm²)						Stiffness properties (in kN/mm²)				Density (in kg/m³)	Density (in kg/m³)
	Bending	Tension Paralel	Tension Perpendicular	Compression Parallel	Compression Perpendicular	Shear	Mean modulus of elasticity parallel	5% modulus of elasticity parallel	Mean modulus of elasticity perpendicular	Mean shear modulus		
	$f_{m,k}$	$f_{t,0,k}$	$f_{t,90,k}$	$f_{c,0,k}$	$f_{c,90,k}$	$f_{v,k}$	$E_{0,mean}$	$E_{0,05}$	$E_{90,mean}$	G_{mean}	ρ_k	ρ_{mean}
C14	14	8	0.4	16	2	3	7	4.7	0.23	0.44	290	350
C16	16	10	0.4	17	2.2	3.2	8	5.4	0.27	0.5	310	370
C18	18	11	0.4	18	2.2	3.4	9	6	0.3	0.56	320	380
C20	20	12	0.4	19	2.3	3.6	9.5	6.4	0.32	0.59	330	390
C22	22	13	0.4	20	2.4	3.8	10	6.7	0.33	0.63	340	410
C24	24	14	0.4	21	2.5	4	11	7.4	0.37	0.69	350	420
C27	27	16	0.4	22	2.6	4	11.5	7.7	0.38	0.72	370	450
C30	30	18	0.4	23	2.7	4	12	8	0.4	0.75	380	460
C35	35	21	0.4	25	2.8	4	13	8.7	0.43	0.81	400	480
C40	40	24	0.4	26	2.9	4	14	9.4	0.47	0.88	420	500
C45	45	27	0.4	27	3.1	4	15	10	0.5	0.94	440	520
C50	50	30	0.4	29	3.2	4	16	10.7	0.53	1	460	550
D18	18	11	0.6	18	7.5	3.4	9.5	8	0.63	0.59	475	570
D24	24	14	0.6	21	7.8	4	10	8.5	0.67	0.62	485	580
D30	30	18	0.6	23	8	4	11	9.2	0.73	0.69	530	640
D35	35	21	0.6	25	8.1	4	12	10.1	0.8	0.75	540	650
D40	40	24	0.6	26	8.3	4	13	10.9	0.86	0.81	550	660
D50	50	30	0.6	29	9.3	4	14	11.8	0.93	0.88	620	750
D60	60	36	0.6	32	10.5	4.5	17	14.3	1.13	1.06	700	840
D70	70	42	0.6	34	13.5	5	20	16.8	1.33	1.25	900	1080

Figure 2.4: characteristic properties of sawn timber (EN 338)

construction material for the future (Menges et al, 2016). Designers and engineers can tap into two design approaches that have emerged in timber engineering. One encompasses the orthotropic and directional characteristics of the wood and employs it in the construction logic, thus using the wood in its most natural efficient way. Corresponding to a quote from Deleuze and Guattari (1987) *"It is a question of surrendering to the wood, then following where it leads by connecting operations to a materiality, instead of imposing a form upon a matter"* The other approach revolves around making the material homogeneous through engineered wood products. These are products like CLT (cross laminated timber) in which fibre orientations are manipulated to make the material more stiff. Fabrication developments made it much easier to break-up a tree into required parts, rendering fast possibilities in timber engineering. Neither of the two approaches is better than the other, engineered products are needed to enlarge the applications for timber but are making recycling more difficult which results in a larger embodied carbon (Menges et al, 2016). The first approach would therefore be more ecological friendly but has often lower structural capacity and is constrained to the dimensions of a tree, except when implemented with a discret esystem.

2.2 Environmental impact of wood

Ecological properties of wood

Bertin et al. (2022) states that 50% of a buildings embodied carbon is the result of its load-bearing structure. Thus a real impact can be made by ensuring a environmental low straining material. Material choice can have a high influence on the embodied carbon (Fang et al., 2023). Wood has the potential the substantially lower the embodied carbon, optimising the construction not only on efficiency and strength but also by matching reused parts could enhance the environmental performance of a building greatly.

Nowadays wood is mainly considered a fully renewable resource with unrivalled potential environmental advantages. The reason for this has to do with the previous life of the used wood. As a tree grows it only needs solar energy and water to produce wood (the tissue of a tree). During its life it converts carbon dioxide into oxygen by photosyntheses, cleansing the air and rendering its ecological footprint positive (Menges et al, 2016). However a tree needs time to grown. As more timber structures are built to lower carbon emissions, deforestation speeds up. The most used construction type of wood is pine. Pine trees grow relativity quick compared to other tree types and need around 25 - 30 years until harvesting can be started (Burger & Zipper, 2009). Thus the need for constructions which can use salvaged wood would be much better for the forestry industry and can help lower demand for new wood and thereby also the price for wood, making it more attractive for construction.

According to Gordon (2012) the production of a wooden panel requires 500 times less energy than a steel panel at similar compressive strengths. Even with consideration of the production process, its embodied carbon can be stay very low. Therefore its is not necessary to optimise a timber construction based on its Green House Gasses (GHG) as using reused pieces is always better than using new pieces in contrast to steel where often the most environmental design is a mix between new and reused parts due to high remanufacturing emissions.

End-of-life of timber

There are a few end-of-life scenario's when considering timber:

- Landfill
- Down-cycling
- Incineration
- Re-manufacture for reuse
- Direct reuse

It is known that wood has the potential to have a very low embodied carbon, however a general bad end-of-life scenario is restricting this. Current new developed engineered products which use glue to assemble and strengthen the material prevent the possibility of recycling. Resulting in higher amount of wood harvested and a higher embodied carbon, the same applies for non engineered wood. According to Morris et al. (2021) engineered wood is predominately incinerated for energy production, and only in some cases it ends up in landfills or is down-cycled. This in contrast with steel that has a recycling rate of 94% due to its recoverable nature. A shift in the design paradigm is required from a landfill and down-cycling end-of-life scenario's to a remanufacturing and reuse scenario. To decarbonise Kunic & Naboni. (2023a) state three points to rethink in the design process: (1) stress-driven material allocation, (2) tectonic flexibility like disassembly, re-use, and adaptability and (3) data-driven design workflows that can accommodate stock-constrained design and allow for material traceability.

2.3 Discrete systems

According to the Cambridge dictionary the definition of the term discrete is: *clearly separate or different in shape or form.* In other words, a system that composes of clearly separate components that can vary in shape or form.

Discrete systems can be seen anywhere in different kind of forms but in this research the focus will be on timber elements. The technique of connecting smaller timber pieces to create a bigger whole has been around since the beginning of construction. Transportation and fabrications constraints have always limited construction from creating structures out of single big pieces throughout history. However, the complexity can vary quite much. A distinction can be made between low and high density systems. Low density systems are as simple as connecting structural elements to create a better and stronger element. High density discrete systems can be classified as systems which consists of many smaller pieces which on their own cannot fulfill an architectural or structural purpose.

These types of systems became possible through great technological advances in digital fabrication, which gave the opportunity to rethink the relationship between individual elements and a whole structure. Discrete design, as this is generally called, is also known as particlised design or as a branch of parametric design (Chen et al., 2021). It is a design language in which the emphasis is almost completely on the design of the individual parts rather than on the overall design. That means that the design process starts with the design of a single part, which can be altered in shape and size if needed. This part is plotted over a given boundary surface which articulates in a final geometry (Chen et al., 2021). In discrete design the individual parts are therefore in precedence and contribute together to the greater whole with equal importance. This gives discrete design on the one hand great tectonic flexibility and geometric freedom, but on the other hand it adds severely to the complexity of such a structure. This correlates with the works of Retsin (2019) and he adds that the design intent is never only based on a single discrete part but follows out of possible relations and compositions of parts. A strange way of designing in which neither a part nor a complete assembly, a whole, is predefined. The parts will influence each other during the design process and are not just simple linear compositions, They embody the final geometry and cannot be seen as mere subdivisions of the lager composition (Retsin, 2019). Such a way of designing can reformulate the entire concept of how traditional elements are perceived. The fixed implication of a traditional beam, column, slab or wall can fade within the structure itself, becoming unnoticeable intertwined with each other. In that way of thinking, a full portal frame for example, can be generated in one discrete structure in which the end or the beginning of the "beam" and the "column" cannot be distinguished, see figure 2.5.

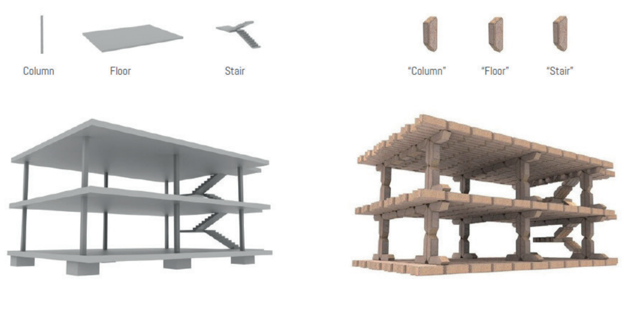


Figure 2.5: Tectonic freedom of discrete elements adapted from, (Ivo Tedbury, Semblr, 2017)

Sánchez (2017) found by observing the chemical process of a water cycle that atoms and the network which they create can be seen as a discrete system, but only on a microscopic level. The structure of the atom can connect to other atoms which together form a whole system. He observed that actively designing is not, in rule, necessary to create a structure with a high degree of complexity and order. But that spontaneous emerges of order exist within nature itself. By extrapolating this experiment to a discrete system four concepts can be identified: parts, links, patterns and commons.

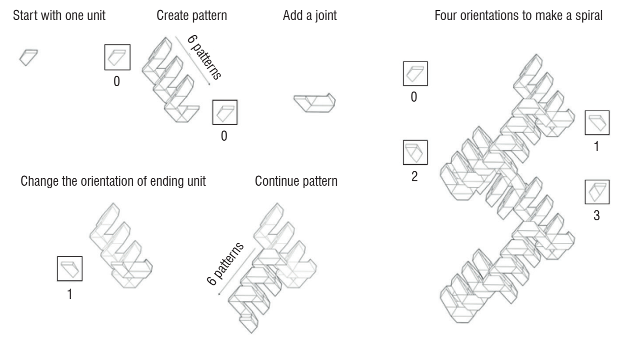


Figure 2.6: Discrete design process (Sánchez, 2017)

Parts embody physical individual elements, ready to be combined. The way these parts are bonded and behave together are the links. Patterns can be formed by structuring these two concepts into a composition. Any combination of individual parts can also evolve into a bigger building block which can form other types of structures. A design is thus not defined by creating new parts but through a combinatorial process which structures patterns. The last concept, commons embodies the notion that an abundant stock has to be available to try out, discard and in the end create these patterns. Figure 2.6 depicts a discrete design process based on Sánchez concepts. This research will approach all four parts and aims to contribute to a new form of architecture which can help and improve the current environmental challenges the world faces. A lot of research already has been done on the first three approaches, this research aims to provide a more detailed overview of the fourth concept, commons.

2.4 Discretization methods

Compositions can be generated in different ways. The main principal is to deconstruct a form into smaller little elements that take up space in a 3D coordinate system, called discretization. There are many discretization methods, the most general used is the voxel-based method which discretizes a form into voxels, cubes. A fast and flexible approach which allows the user to generate designs quickly by simple inputs (Kunic & Naboni, 2023b) and can also been seen in both (Naboni & Kunic, 2019) and (Kunic, Naboni, et al., 2021).

According to Xiao et al. (2020) and Hung et al. (2021) there are generally two possible approaches to generate a discrete structure: the top-down and bottom-up method.

Top-down

In the top-down method a predefined form is discretized in infinity smaller objects like voxels, figure 2.7. The resolution of the voxels can be set by the designer, the smaller the voxels the more accurate the composition but the more heavy the computational time. This method is somewhat in contrast with the previous stated idea about discrete systems in which the final geometry in unknown at the start of the design. However, this method does give the designer a more firm control of the borders of the final geometry. In combination with structural optimization, this can lead to interesting designs in which form follows force. By combining adjacent voxels to a preferred length that correlates with elements out of a stock database a structurally optimized stock constrained design can be achieved.

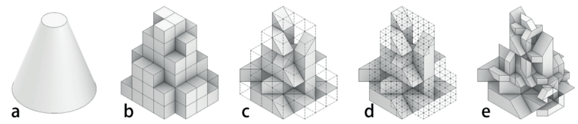


Figure 2.7: Top-down voxelization process using a cylinder as an example (Xiao et al., 2020)

Bottom-Up

In the bottom-up method a predefined set of rules is used to generate a geometry in compliance to set boundary conditions. An algorithm uses these rules and conditions as input and creates a composition which is best suited according to the objective. The input in this particulair a discretization method can be a surface or a line over which the algorithm can create compositions

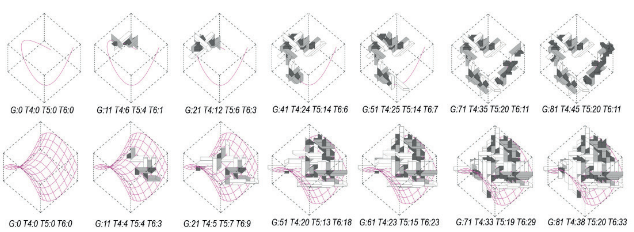


Figure 2.8: Bottom-up process in which voxels are composed by "line" and "surface" (Xiao et al., 2020).

with voxels or directly with pre-set stock elements, see figure 2.8. This method fits more to the ideology of a discrete system defined by Restin (2019) and Chen et al. (2021) and could be very interesting in relation to structural optimisation. Principal stress-lines could be used as input for the algorithm to create a composition along. Principal stress-lines are interesting because they show the natural flow of forces of an applied load within a structure. These lines therefore represent directly the desired continuity of solid material and possible places where material can be left out. In other words, it is a direct representation of the design domain (Tam & Mueller, 2015).

Computational tool for discretization

To accommodate discrete design Rossi & Tessmann (2018) created the Wasp plug-in for visual programming environment Grasshopper. Wasp offers a way to discretize a geometry in pre-designed parts and provides local constraints for joining the parts. In other words, with wasp a designer can select faces that are allowed to be connected with each other, but other constraints can also be incorporated like exclusion zones where no parts are allowed. Previously such constraints had to be defined by set rules for the algorithm. This way design and assembly are closely intertwined.

2.5 Combinatorial design

Referring to Sánches (2020) when designing a discrete system, a designer's focus has to shift at some point in the design process from designing individual objects to designing a generating system that is able of generating many objects and combining them into a functional composition.

Combinatorial design can be viewed in different ways. Terzidis (2015) sees this type of design as purely computational, each variable embodies a degree of freedom and can be classified and catalogued in order to be optimally placed in a specific place of a system. This way of thinking rejects completely that natural intuition and design experience can contribute to the most efficient design. Terzidis states that experience and intuition are part of a trail and error process until at a random point a valid solution arises, but that this can never be labelled as the best solution. His view supports a fully algorithmic solution based design, and favours the name permutation design. Sánches (2020) contradicts this view on combinatorial design, and envisions this way of design as a system that embodies an open-end relation between its parts. A system that can accommodate different functions and performances at different scales of requirement at different times. Such systems cannot be optimised as it has to be able to change over time, be malleable in a such a way that broken parts can be replaced or the system can be reconfigured as the requirement changes. It would be impossible to optimise because the openness implies that there is no optimum solution. Best solution algorithms can usually optimise only on one task and one scenario, with each added scenario and task the solution space will grow exponentially (Sánchez 202). The view of Sánchez enables the designer to choose which solution best fits the requirements at that time and uses combinatorial design as a tool to aid the designer in coming to this choice. This view fits best with the vision and the design objective of this project.

Due to environmental challenges there is rising need of architectural and structural systems that can be shared and reconfigured in other buildings to improve their lifespan, otherwise known as tectonic flexibility (Kunic & Naboni, 2023a). This helps in reducing the embodied carbon of buildings, closes the linear life cycle of materials and makes it necessary for building components to be designed for disassembly. Repurposing a used discrete system can prove to be difficult due to the nature of combinatorial design in which there are usually many unique parts destined for specific places. On the other hand can combinatorial design help aid the circular economy because it repurposes smaller otherwise unusable pieces of wood. To help achieve reusability Rossi & Tessmann, (2018) suggest to design identical parts that combined can form the whole geometry. This way it is more easy to add or remove elements if reconfiguration is desired. However, for a stock-constrained design this is not possible and a new suggestion has to be made to ensure reusability.

By globally analysing excising combinatorial systems a few points can highlighted. Existing systems can be classified in roughly three categories: load-bearing systems, self-supporting systems and conceptual architectural compositions. A small sample of the inventory is shown and categorized on page 20. Notable is that there are a lot of conceptual systems which is no surprise considering that this type of architecture is still relatively new and currently being developed. These are systems that embody an idea of a discrete way of architecture but are/ cannot be constructed, or do not have a clear function. Also a lot of self-supporting systems can be found. These are systems that can support themselves and have a clear function within the built environment but cannot support other elements. These systems cannot be used as load-bearing members because they lack of the structural component. Recently a research group in Denmark started multiple works into discrete load-bearing elements which can be used to actually construct buildings. These types of structures are not seen around anywhere else much and thus it can be concluded that most systems now have an architectural expression and lack the structural or load-bearing component to realise complete buildings.

2.6 Prior structural discrete developments

The Create group of the SDU (Southern Denmark University) is a research group committed to investigate the future of architectural and constructional design by exploring novel architectural ideas which are in close relation to the field of digital and material technologies. In 2019 this group started research into discrete timber load-bearing elements with in total four successful projects:

1. Topologically optimized bridge structure, 2019, figure 2.9
2. Reversible timber beam, 2020, figure 2.10
3. ReconWood 01 & 02 prototype, 2022(3), figure 2.11
4. ReconWood 03 structural slab, 2023, figure 2.12

Each of these projects is centred and adjusted around the theme of robotic assembly which requires a self aligning system in the geometry of the parts. The group started out with the design of a bridge structure generated and optimized out of small timber pieces (Naboni & Kunic, 2019). In 2020 this project was improved and redeveloped into a timber beam. The shape and size of the parts of this beam are made to fit for robotic assembly, so that the elements are automatically aligned and easy to grab by a robotic arm. A computational workflow in visual programming environment ‘Grasshopper’ with ‘millipede’ and ‘monolith’ plug-in was made to generate this structure which composes of four main phases. 1) Voxel based design, definition of boundary conditions and structural optimization. 2) Design of a kit of parts and creating a digital twin. 3) Conversion of structural data into a structural composition. The discretization is done by running a topological optimization on a given volume. The resulting geometry is discretized in voxels which are then connected and converted to the designed parts given as input in phase 2. In this project the Top-down discretization method is used. 4) Robotic assembly simulation (Kunic, Naboni, et al., 2021).

Distinctive of this structure is its layered composition. This type of compositions makes the structure more fire resilient than reciprocal structures that have open cavities between its parts. Results showed however that this beam lacked shear capacity because the elements were not working as a whole. This proved to be the most influential failure mechanism which introduced a new research project. In 2021 and 2022 a lot of research was dedicated to creating a connection

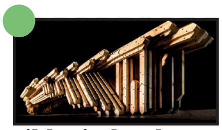
Timber and discrete systems

Highly Structural

Load-bearing element

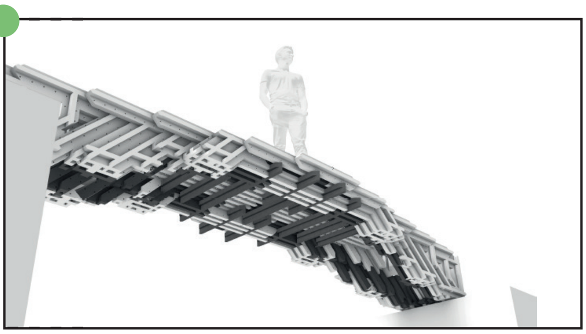
Self-supporting systems

Conceptual systems



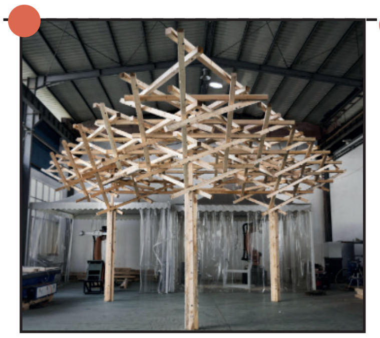
Reversible timber beam
SDU Create group, Denmark, 2021

Recon timber slab
SDU Create group, Denmark, 2023



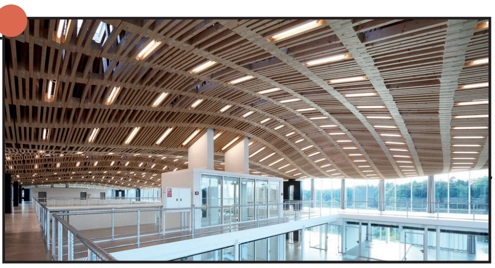
Topology optimized bridge
SDU Create group, Denmark, 2019

Coeda House
Kengo Kuma, Japan, 2017



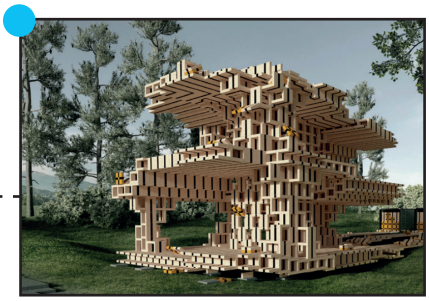
Reconfigurable modular timber grid (RMTG)
Hao Hua et al, China, 2022

Skilled-in Office
Studio RAP, Netherlands, 2017

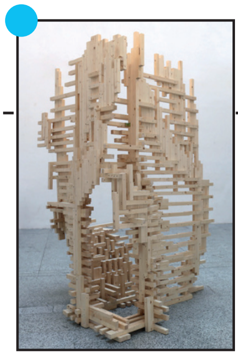


The Sequential Roof
Gramazio Kohler Research, Switzerland, 2016

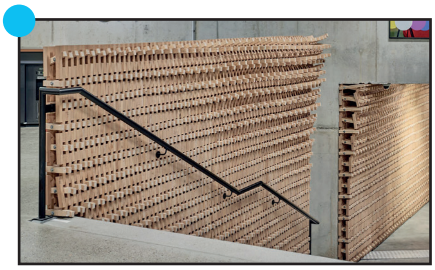
The Tallinn pavilion
Gilles Retsin, London, 2017



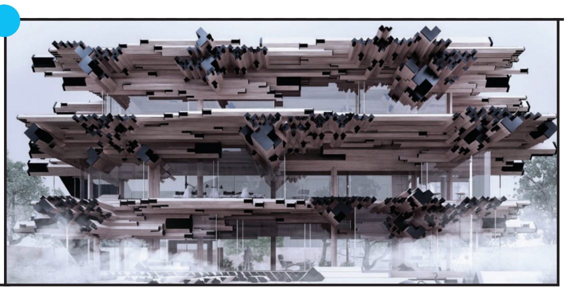
Pizza Robot
Gilles Retsin, 2018



Styx
AA Visiting School, Switzerland, 2018



Circular Experience
Studio Rap, The Netherlands, 2019



Diamonds House
Gilles Retsin, 2015

Plexus
Studio Symbiosis, India, 2021

Highly architectural

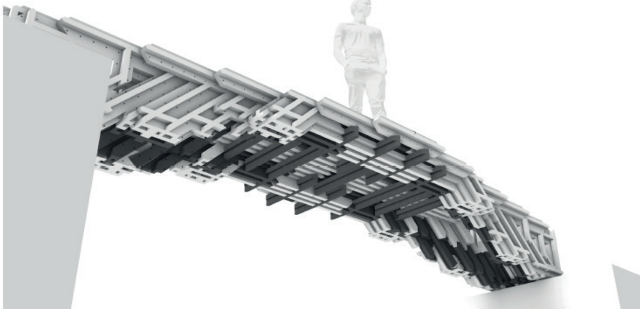


Figure 2.9: Topologically optimized bridge (Naboni & Kunic, 2019).

Figure 2.10: Reversible timber beam (Kunic, Naboni, et al., 2021).

Figure 2.11: Reconfigurable timber v1.0 & v2.0 architectural composition (Kunic & Naboni, 2023b).

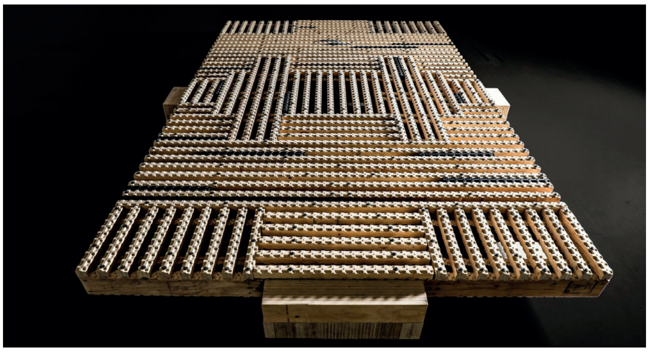


Figure 2.12: Reconfigurable timber v3.0 structural slab (Kunic & Naboni, 2023a).

type that was able to increase the shear capacity (J. L. Hansen et al., 2023), (Kunic, Kramberger, et al., 2021) and (S. G. Hansen et al., 2021). In 2023 the results of this new research were bundled in a new structural project: the reconfigurable structural slab (Kunic & Naboni, 2023a), and with success. Within this project the parts require limited amounts of milling for self-alignment and provide a good shear connection and flow of forces between elements. The slab is designed as a orthogonal reciprocal structure which can be defined as a grid of linear components in which each component is supported and supports its neighbouring components. Such structures are very light weight and structurally efficient but lack fire resistance because it is not a massive layered composition in which the outer parts form a charring layer and protect the inner parts (Munck et al., 2011).

This current research project aims to build further upon the previous research of the SDU to create a similar computational workflow, but which is also linked to an available stock and stream of waste wood. Furthermore, it wants to add by investigating the structural limits and reconfigurability of such members.

2.7 Connection technology

As previously stated, by Chen et al. (2021), Retsin (2019) and Sánchez (2017) the parts within a discrete system are the most important and should get the main focus in the design process. However, this is put in question by Xiao et al. (2020) who argues that the links, or more commonly defined as connections or joints , have to be considered at least as equally important. Especially in large scale structural components the member's capacity as a whole depends on both the structural capacity and stiffness of the parts as much as on the capacity of the link between them. Joints can add severely to the complexity and cost of the design and require a comprehensive design approach. A great variety of joints will decrease the flexibility of the system. On the other hand simple joints ensure greater flexibility and reconfigurability options. This also prompts the question of what an optimal ratio of links in a design is and thus indirectly what is the optimal length of a part? Longer parts equals to less links but also a loss in flexibility. Difference in length of discretized components allow for variable structural resolution of the frame (Kunic & Naboni, 2023a).

With the main advantage of a discrete system being its flexibility (Xiao et al. 2020), this design method offers the opportunity of designing the connection type to be able to adapt to different scenario's throughout time. This means that the system has to be able to be reconfigured. To highlight this potential, this research project irreversible connections like glue and nails are therefore not suitable as they strongly limit the flexibility (S. G. Hansen et al., 2021). Besides being reconfigurable the connection also has to be able to resist shear forces and have moment capacity. Bolt, dowel or mortise and tenon joints can all facilitate these requirements.

Wooden connections like finger joints or mortise and tenon connections can be very strong if interlocked tightly, the downside with these type of connections encompasses an intensive manufacturing process and structural high brittleness as the load response is predominately ruled by the shear strength of the timber itself, see figure 2.13-B (S. G. Hansen et al., 2021).

Steel connections by bolts lack in strength compared to full timber connections due to peak axial stress in the timber around the bolt, see figure 2.13-A. Furthermore, in this type of connection a bolt is mainly subjected to bending which is less favourable with steels in relation to tension in which it performs best. However such a connection behaves very ductile due to the plasticity of steel. This ductile effect first causes large deformation until failure. However in a structure with many connections large displacements are especially problematic (S. G. Hansen et al., 2021).

Hybrid types of connections in which both steel and wood are utilized to its optimal characteristics could pose a good alternative to form a high strength but ductile joint. Hanson from the SDU create group started a research into such a connection in 2021 with the main principal idea to add shear keys to the face of the wood by milling with CNC machines. These keys can transfer the shear forces to other parts and activate the discrete composition to work as a whole with very little deformation. This allows the bolt to be predominantly subjected to pure tension as the shear keys take the bending and friction, the bolt only keeps the faces locked together. This does require the bolt to be installed in an oversized pre-drilled hole, otherwise the deflection will still cause bending moments in the bolt. If the shear keys fail brittle, the bolt will take over and deform plasticity, making the hybrid connection both ductile and strong, see figure 2.13-C. Increased stiffness in the joints is extremely beneficial as the global stiffness of a structure with a lot of connections

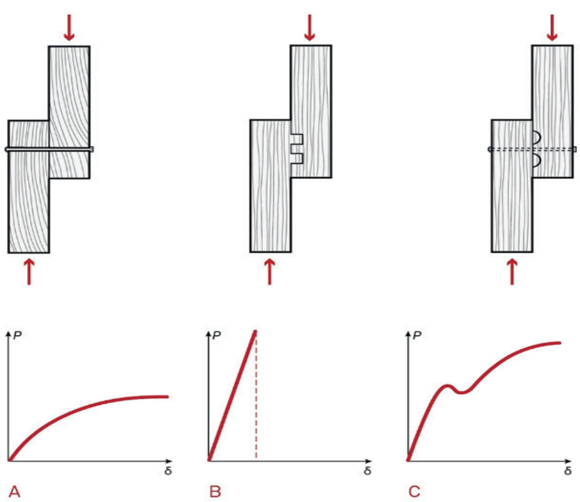


Figure 2.13: Conceptual schematisation of three connection types and their loading response. A - traditional bolted connection; B - finger joint; C - steel timber hybrid shear connection (S. G. Hansen et al., 2021)

is governed by the stiffness of the connection itself (J. L. Hansen et al., 2023). Moreover, this adds the function of self-alignment what makes assembly more convenient. Such a connection type can be seen as a face-to-face connection which Xiao et al. (2020) advocates for when designing structural components. As a large contact area can provide substantial structural strength and stiffness to withstand a certain loading composed of self-weight and external force (Xiao et al. 2020).

S. G. Hansen et al. (2021) observed two types of failure mechanisms which are illustrated in figure 2.8.

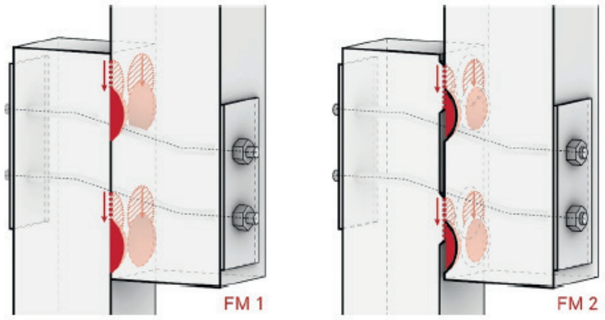


Figure 2.14: schematisation of two occurring failure modes (S. G. Hansen et al., 2021)

In main failure mode of shear keys, FM1, the shear keys are cut of at the base. This effect simple to determine by expression:

$$V_{\text{shear},1} = A_{\text{shear,key}} n f_v$$

In which $A_{shear,key}$ is the area of one shear key at the base, n the number of shear keys and f_v the timbers shear strength. The other failure mode, FM 2 is more complicated to determine and can easily be ruled out by making the shear keys longer or by increasing the area. The disadvantage of this connection is the extensive milling process each parts has to undergo which also adds to a lot of waste. A limitation in the design that was further improved in the previously mentioned project Recon timber slab. Here the shear keys are milled as three dimensional male-female joints as can be seen in figure 2.14. This new connection allows for limited processing but only provides shear resistance

Figure 2.15: three dimensional male-female shear connection Kunic, A., & Naboni, R. (2023b)

in a two dimensional system by stacking layers. This way all the parts are equal and can be easily changed within the configuration what improves the flexibility and reusability greatly. This system might be hard to implement for a system in which members can also be placed in angles and are massively layered instead of stacked. It shares the same failure mode as the former connection in which the shear key is crushed.

Studio Rap has used a similar hybrid connection in the 'circular pavilion' but instead of milling the shear keys in the timber components, holes are drilled in the components in which dowels are placed. These dowels interlock in the neighbouring piece and act as a shear key. The whole composition in held together with steel bolts as can be seen on figure 2.16. The advantage with this type of connection is an even less intensive manufacturing process, but the reusability lowers as the dowels can swell and lock the structure. Moreover, this also reduces the flexibility of a part as it has a specific place in the design.

Figure 2.16: connection from studio Rap with dowels as shear key (Studio RAP, 2019)

Paula (2023) tried to develop this shear-key-based bolted joint further by intertwining the shear key in the bolt head itself, see figure 2.17. Making each element identical and therefore in every direction flexible and reconfigurable, even in a massively layered design in contrast to the Recon wood slab. However, this connection does compromises on cost and assembly rate in spite of flexibility and the ability of full robotic assembly.

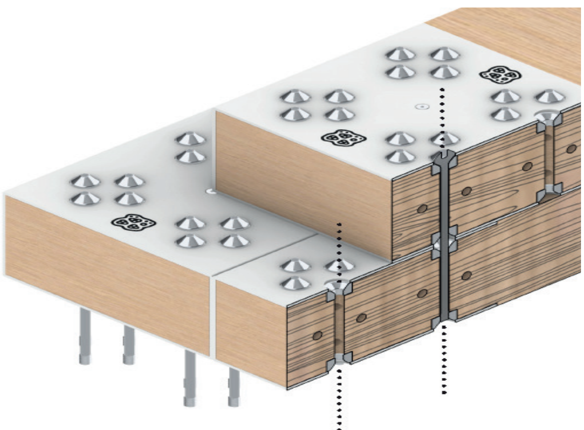


Figure 2.17: Connection in which the shear key is combined in the bolt head (Paula, 2023)

In Circular pavilion Eindhoven another approach can be identified. The pavilion was made using borrowed material which had to be returned undamaged. This required a connection method which does not use, glue, or any kind of drilling, screwing and milling. Engineers came up with an innovative design bringing pieces of timber together, connecting them with metal- and lashing straps creating a structural system with a high load capacity, see figure 2.18.

Figure 2.18: circular discrete system for peoples Pavilion in Eindhoven, using tension straps (Arup, 2017)

There are many more connection typologies and the above mentioned showcase a small sample of possible solutions. The best connection type depends on the required design criteria and has to be assessed for each case individually.

2.8 Timber and fire

When subjected to temperatures exceeding 300°C, timber undergoes combustion. The timber's inherent defensive mechanism involves the development of a char layer that progressively extends towards the wood's centre, diminishing its effective cross-sectional area and strength. The protective char layer formed on the material's exterior, serving to decelerate the temperature rise within the unaffected core. This is a process that gradually progresses and that can be easily calculated for solid timber. Laminated timber or layered assemblies, which comprises of thin laminates or layers, exhibits different behaviour. In the case of layered timber, the charring of the other layer results in detachment by melting of bolts, screws or glue, contributing additional fuel to the fire. This, in turn, increases the time it takes for the structure to burn away completely a lot. To counteract this effect, the steel used in connections has to be protected in the wood itself and if applicable heat resistant wood has to be used as outlined by, Munck (2011), and Borgström and Fröbel (2019). This explains why reciprocal assemblies are not performing well against fire. The parts are not protecting the others and therefore have very little burning time. For the main load bearing structure a massively layered composition seems to make more sense as this has to be able to resist fire for 30 – 120 min in compliance to the Eurocode and the user function.

The (NEN-EN 1995-1-2+C2, 2011) presents

two distinct methods for determining the burning rate: the reduced cross-section method and the reduced properties method. In the Netherlands, only the reduced cross-section method is used, see figure 2.19. This particular approach operates under the assumption that the material's strength and stiffness remain consistent with the original design values. To address this assumption, the cross-section undergoes reduction in relation to timber penetration, progressively diminishing the surface area (Munck et al., 2011).

For structural fire design the load may be reduced according to the following load combination for extra ordinary actions as stated in Eurocode as:

$$\sum_{j>1} G_{k,j} + P + A_d + (\psi_{1,1}\ or\ \psi_{2,1}) + Q_{k,1} + \sum_{j>1} \psi_{2,1} * Q_{k,1}$$

Besides the reduction of the load another beneficial effect can be considered: the maximum allowable stress is increased. The normal strength of wood is based on the lowest 5% of tested specimens. With fire the lowest 20% of tested specimens may be used, resulting in a multiplication factor of 1.15 on the normal strength of wood.

The effective degression of the timber by charring of the wood can be calculated with followings expressions:

$$d_{ef} = d_{char,n} + k_0 k_d$$

$$d_{char,n} = \beta_i * t$$

In which t is the past time in minutes, b a factor that considers the effect of one dimensional charring (b=0.65) of multiple dimensional burning (b=0.7), k_0 a factor which manipulates the starting burning rate util the char layer is formed at 21 min and d_0 the start value of 7mm.

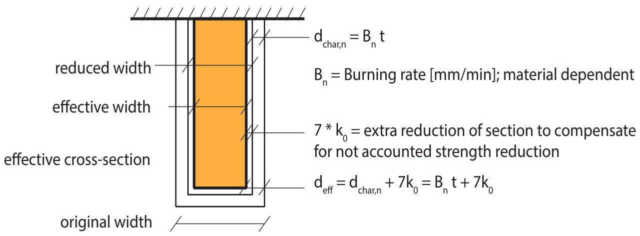


Figure: 2.19: reduced cross-section method (Munck et al., 2011)

2.9 Constraints of discrete systems

Xiao et al. (2020) calls for several constraints within the design of a discrete system. Fabrication limitations are the most influential to be considered and constrain the design through limiting fabrication methods like milling, cutting or printing for timber designs. These methods themselves are again constrained by a certain complexity for manufacturing and assembly, the degree of freedom in composition and further economical and time related factors. Other constraints are related to the structural capacity of the parts and the material characteristics. To find a suitable way for fabrication Xiao et al. (2020) has put these factors into a multi criteria analysis (MCA). In such a scheme these factors are ranked by importance and scored based of different designs. The design with the highest score would than prove to be the best option. He states that the structural capacity and the degree of freedom in composition are the most important factors. The strength of the parts determine the upper limit of the whole composition and therefore the possible structural functions. The degree of freedom in composition is the basic and principal advantage of a discrete system, the two aspects will be translated into design criteria.

2.10 Boundary conditions for design

Timber is an orthotropic material which means that it has different properties in different directions. Along the fibre direction timber is most strong. Orientation in relation to loading is very important to take into account in the design because of severe drops in strength if loaded in 45 ° or 90 °.

Generally there are two discretization methods, the top-down and the bottom-up method. The bottom-up method seems best suited for this project but is more difficult to achieve.

Discrete design is a new way of designing in which the designer focus more on the individual parts and joints than on the final outcome. A discrete system composes of parts, links and patterns. Discrete design has the main advantage that it can be very flexible and be easily fit to a design objective. This flexibility has to be taken in the design, so that such a system can fit more than one scenario, even after being in use. That means that the composition has to be adaptable, this requires from of reversible connections.

Hybrid connections are most suited as it can accommodate both ductility and high strength. However a case can also be made to design two types of connections. Wood for parts in compressive zones and hybrid in tension zones.

A massively layered composition is needed to accommodate fire resilience. The most influential failure mode in a discrete system is local deflection in the joints. This is usually caused by a lack of shear capacity. The joint therefore needs to accommodate for enough shear capacity to provide stiffens to the system. This can be done either by shear keys or strong interlocking wood connections.

Combinatorial design can help aid the circular economy because it repurposes smaller otherwise unusable pieces of wood. Identical parts that combined can form the whole geometry can help achieve reusability , but add in remanufacturing costs and time.

Flexibility and structural strength of the parts can be viewed as most important criteria to focus on in the design. Other criteria are: remanufacturing complexity, assembly complexity and economic and time related factors.

3. Waste wood market in the Netherlands

Because the waste wood industry differs quite much in each country it is hard to get a clear picture of the global market. For that reason this study focusses only on the waste wood market in The Netherlands. This chapter aims to get a clear picture of the waste wood market in The Netherlands in order to determine what types of waste streams, amounts, dimensions and sections of wood are available to reuse. It will further highlight the main problem stated in chapter 1 and provide a more defined scope of the project. This analysis is essential in order to develop a program that can generate a discrete timer element from reclaimed timber.

Figure 3.1: impression of wood scrapyard (Bruggen & Zwaag, 2017)

3.1 Waste collection process

In the Netherlands three main stakeholders organize the waste wood market and are building and maintaining the impressive scrapyard of around 1.7 million tons nationally. A demolisher, a collector and a processor. Figure 3.1 gives an impression of scrapyards of wood.

The collection process in the building industry often starts with a demolisher. This party is hired to tear buildings down at their end of life. After this a collection company is hired which sorts and collects the wood and delivers it to a processing facility. Often the two parties are combined in the Netherlands. For the construction sector specificity a demolisher can also sort the waste wood and transport it to the processor. For waste produced outside the construction sector a processing company can also collect and sort the wood. The processing facility sorts the wood into different classes and depending on this classification the wood is either shredded for energy production or for the production of engineered wooden panels and boards.

3.2 Wood classification

Firstly, the difference between two frequently used terms in this research: wood and timber. Wood is a building material for all non-structural purposes. Timber is a structural material used in construction. This chapter addresses the term wood and encompasses all types of wood for each kind of waste stream. Three categories are used to classify waste wood: A, B, and C-wood.

A-classified wood

A-class wood is the most clean and easy to recycle type of wood. This type is defined by its characteristics that it is untreated, unpainted and free of glue. A-wood can be anything up to pallets, pruning wood, fruit cases and dust or shreds from milling. 80% of A-wood in the Netherlands composes of packaging mostly encompassing pallets. This industry could prove to be a valuable waste stream for the generation of a discrete structural element. According to a study of the waste wood market in the Netherlands of Bruggen & Zwaag (2017), most of the A-wood ends up being mixed with B-type of wood and therefore indirectly becomes B-type wood.

B-classified wood

B-class wood is non-preserved wood that has been glued or painted or cannot be classified as either A- or C-wood. This wood stream can be further divided into solid wood and non-solid wood. Solid wood can be identified as a painted beam, cladding or a roof fascia. Non-solid wood encompasses all the wood containing adhesives like particle boards (Bruggen & Zwaag, 2017). For this study only the solid wood can be used for reuse. The structural properties and quality of the glue as well as the thickenss of board products are factors that withhold this from being salvaged.

C-classified wood

C-class wood is preserved wood. Preserving wood is a process to protect the fibres against fire, termites and fungi but also helps in slowing degradation due to climatic and environmental conditions. Preserving is more commonly known as impregnation which is a process in which the wood is treated with heavy chemicals. Due to this process C-wood is the most unsustainable type of wood and cannot be easily recycled. In the Netherlands this type of wood is prohibited from being incinerated for bio-energy due to lack of proper facilities. Therefore the C class wood is transported to Germany where proper facilities can further process the wood (Bruggen & Zwaag, 2017). Especially this type of wood can benefit from direct reuse as it cannot be recycled as of yet.

3.3 Excising stockpile

Commissioned by the central government, TAUW bv conducted a study of the waste wood market in the Netherlands in 2017 (Bruggen &Zwaag, 2017). They conducted several interviews with stakeholders to obtain data. This research identifies current bottlenecks and it also statistically analyses the waste wood market. The values from this study are based on documented key figures for the year 2017. Unfortunately, no new research has been started as of 2023.

92.5% (1610 kton) of the 1.741 million tonnes of waste wood that was in circulation in 2017 came from production, while 7.5% (131 kton) came through

Waste wood market in the Netherlands 2017 [in kton]

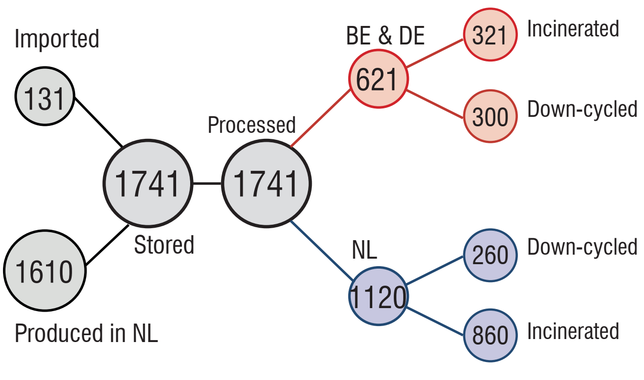


Figure 3.2: Wood market in the Netherlands 2017, adapted from (Bruggen & Zwaag, 2017)

importation. The wood originated from Norway, Germany, Belgium, and the United Kingdom. Waste wood is often imported due to the transient nature of this niche market. Taxes in other countries can make it more attractive to sell their wood to the Dutch market. Also processing limitations can be a reason to import waste wood from other countries. Bio-energy plants in the Netherlands do not have the capacity to burn al the scrap wood for energy, for that reason a large quantity is also exported. Figure 3.2 visualises what happened to the total amount of wood in 2017. Figure 3.5 is more detailed and also shows the share of each wood class in relation to the total amount of collected and processed wood. It is notable that C-wood is produced the least and that all the C-wood is also exported to Germany. This is because the Netherlands lack the facilities to properly dispose this type of wood.

In 2017 half of the wood is incinerated and only 15% is down-cycled. The other 35% is exported to be incinerated for energy or to be down-cycled in other countries. This export is the result of insufficient capacity of the power and board making industry. The surplus is shredded into small pieces to make transportation more efficient and is then exported. This shows that the need for a more efficient wood market is essential. Non-solid B-wood has the largest share of 51%. This type of wood is not usable for this study because it consists of particles and glue. It is also the wood that is used for the energy production. Combined A-wood and solid B-wood encompasses 42% of the total collected wood. Both types can be viable waste streams for this project, especially the B-wood solid class because it is certain that this waste stream consist of solid usable wood. The A-wood class could also compose of saw dust and shreds. That means that each year a potential 370 kton B-wood that now is shredded could be redistributed for direct reuse. This would also ensure a much lower surplus and thus much lower emissions for the export industry. C-wood could also have the potential to be used for this project but is contaminated wood. Therefore it could be perfect to reuse again because incineration of this type of wood is particular harmful for the environment. However much is unknown of this waste stream, what kind of chemicals are used and if it is safe and responsible to reuse. Figure 3.3 visualises the waste wood market in the Netherlands and summarizes the aforementioned findings.

Because the research from TAUW bv from 2017 is somewhat outdated, data from the CBS is used to check if the trend of the wood market still complies to the results found in 2017 (Centraal Bureau voor de Statistiek, 2023). Figure 3.4 shows the harvesting, import and export of biomass wood. The harvesting relates closely to the waste wood production because a large share is used as biomass. The production of biomass wood is from 2015 to 2021 very constant and

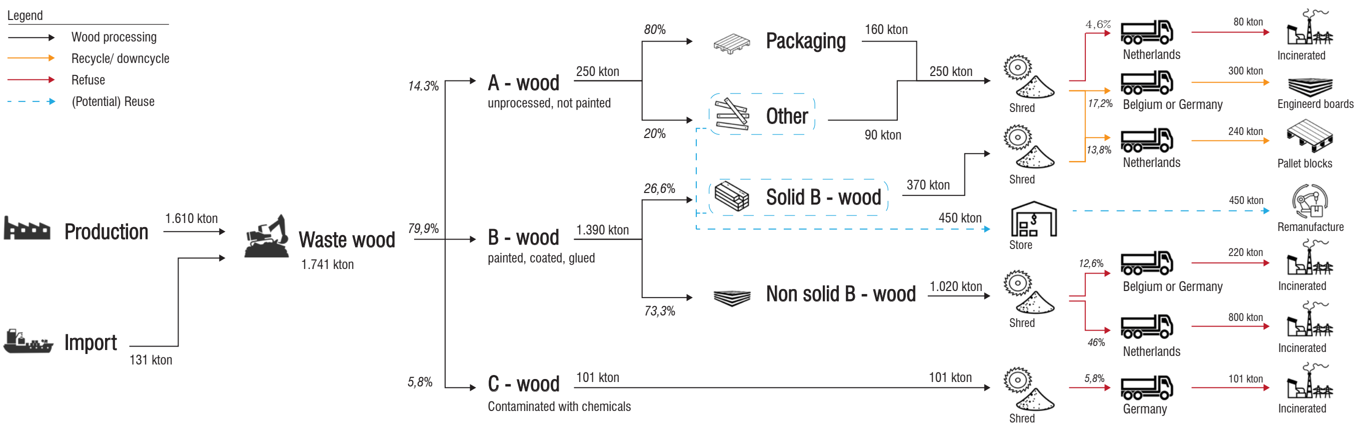


Figure 3.3: Estimation of the processed waste wood in 2017 in million kg, adapted from (Eijk, 2021)

shows that the waste wood market is most likely not changed much. As the export of biomass has risen, the capacity and efficiently of the recycling facilities in the Netherlands would most likely not have increased or improved. Both arguments are validating the older study of Bruggen & Zwaag (2017) up to 2021.

Harvesting, import and export of wood intended for biomass

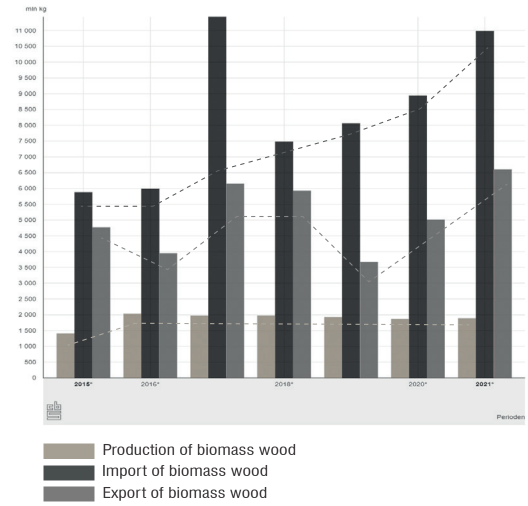


Figure 3.4: Harvesting, import and export of wood intended for biomass (Centraal Bureau voor de Statistiek, 2023)

3.4 Potential waste streams

As the study of Bruggen & Zwaag (2017) provides a lot of insight into the waste wood market, it fails in providing detailed information of consistency of the scrap wood. Mantje (2023) conducted a study in relation with the waste wood market. Through interviews with stakeholders she discovered a general consistency in the scrap:

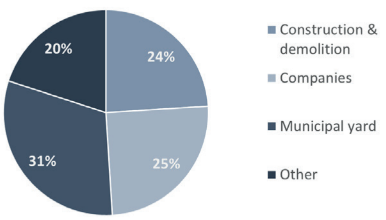


Figure 3.5: Division of waste wood adapted from (Mantje, 2023)

In figure 3.5 a division of collected waste wood is shown. Wood from construction & demolition seems a good waste stream to use as it often consists of highly standardised elements. Moreover, this wood type is often suited for construction as it has been previously used in a load-bearing way. An estimation was made by Mantje (2023) of the consistency of wood in this category, also visualised in figure 3.6:

- Beams (19%)
- Boards & planks (20%)
- Door & Window frames (17%)
- Doors (2%)
- Wall & other framework (22%)
- Other (20%)

While this breakdown of solid waste wood provides a much better insight in the current stockpile there is no national database in which these parts are documented. Furthermore, Common dimensions and cross-sections are also still unknown. But as the construction sector is responsible a large amount of waste wood a lot of standardised dimensions can be expected due to high standardisations in available cross-sections as stated in de NEN 5499. This standardisation is helpful in making a realistic database which can be used for teh design tool.

Besides the study of Mantje (2023), Eijk (2021) interviewed a delegate from 'Bloem gebruikte bouwmaterialen'. Eijk concluded that in 2020 around 4000m of wood was collected by Bloem gebruikte bouwmaterialen with lengths varying from 3 up to 5.1m. This collected wood had cross-sections varying from 50x150, 65x165, 70x 195, 75x210 and 75x220mm, and identifies as common purlin sections. The interview also concluded that wood specifically collected for reuse has lengths from 3 to 5 meters. Anything below 3 meters would not be profitable to collect for a stock buyer an anything larger than 5 meters would result in problems in relation to transportation. This study will especially have use for short pieces under 3 meters due to the greater tectonic flexibility that small pieces provide. This project thus has potential to open up a whole new recycling market in the Netherlands.

3.5 Bottlenecks

The study of Bruggen & Zwaag (2017) was especially focussed on mapping and highlighting bottlenecks in the Dutch waste wood market. A overview of the most important bottlenecks is be considered here.

Capacity

For waste wood, the Netherlands currently has a recycling capacity of 260 kton for A wood and/or solid B wood. This capacity is entirely filled with Dutch waste wood. This ensures that a large proportion of potentially good reusable wood is exported and/or burnt. The wood that is recycled is often shredded and used for board material. This is a waste because once the wood is pressed into a plate or pallet block, it can never be reused because of the glue. The government should support and encourage new recycling initiatives to increase capacity. Especially important is that the wood is not just used for down-cycling but that there are separate facilities that sort, store, renew and reissue the wood.

Database

To reuse wood properly, a public database of available wood needs to be created. This can be organised by timber buyers or by processing facilities. With such a database, designers can already take into account what is available at the front end of the design. This will greatly help the design method of 'designing with what is already manufactured'.

Sorting

Sorting A-rated wood or solid B-rated wood from the mixed (A) B-rated wood stream is labour-intensive and requires significant investment (e.g. in automation) to be carried out cost-effectively. It is not worthwhile for these companies to upgrade waste wood facilities to the required specifications for recycling without a strong demand. Companies will settle sooner for incineration as final processing for the waste wood. An estimation is made that this bottleneck leads to 10-20% less recycling of A-rated wood and solid B-rated wood. Mandatory separate collection (and possibly mandatory recycling) of, for example, A-rated wood and/or solid B-rated wood may offer a solution. When recyclable waste wood streams have already been delivered separately, it is cheaper for a collector or processor to upgrade the material to the specifications required for recycling.

Subsidization

There is serious competition for waste wood from energy plants. These are willing to match (and even exceed) the rates of chipboard manufacturers or compost producers for this material. This makes bio energy-plants a serious competitor to recyclers. Subsidisations on A-wood procurement to boost "renewable energy" generation works against recycling, and the government should focus more on subsidizing recycling.

3.6 Conclusion of wood market

This chapter sought to shed light on the Dutch waste wood market. A few notable discoveries were made. There is a very large waste stream of wood in the Netherlands. A potential stream for direct reuse of around 450 kilo ton is produced each year. Now almost all of this wood is shredded and burned or down-cycled.

Three types of wood can be identified: A-, B and C-wood. For this project especially A or B wood could be used. C wood is contaminated with preserving chemicals what withholds it from reuse. The amount of usable A-wood is hard to determine as a lot of it consists of prunings, pallets and other packaging. Moreover, a lot of A wood is mixed in the B-wood which can give a distorted image. Of all the usable wood around 24% is produced by the construction industry. As this is a waste stream close to a designer and composes of a lot of standardised cross-sections this seems a good waste stream to tap into. Through the interviews from Mantje (2023) the consistency of this 24% is broken down into a few categories. These can be used to create a database which can be used further on in the project as a constraint for the optimization. The two biggest categories, beams and framework, are used to create a database for the design of the computational tool.

There is still almost no direct recycling in the wood market. In almost all cases, the remaining wood is shredded for down-cycling to sheet material or used as 'renewable' fuel for power generation. The wood that does get reused are mainly large pieces between three and five metres. Pieces under three metres in length are not profitable for a buyer to resell because there is little or no demand for them. A discrete structural element consisting of small pieces so that topological optimisation is easy to carry out can thus open up a new market, increasing the value of small remaining pieces of wood and making direct reuse attractive.

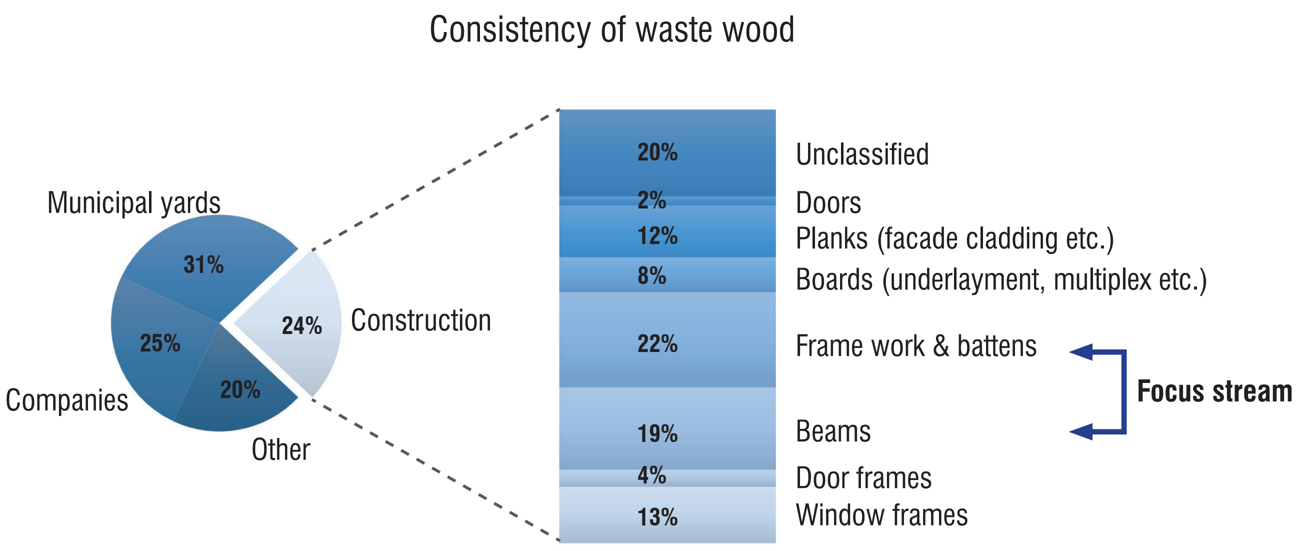


Figure 3.6: Consistency of waste wood, adapted from (Mantje, 2023)

4. Reusing wood as timber

This chapter aims to provide a workflow on how waste wood could potentially be reused for a discrete timber element. As concluded in chapter three, wood is still more often down-cycled and incinerated than reused. How would such a process look like and what needs to change in order to apply direct reuse on a discrete timber element?

4.1 Reuse Process

Traditionally, strength properties have been assigned to wood based on visual aspects. NEN 5499 allows sawn European softwood to be divided into four quality classes, T0 - T3, solely on the basis of visual aspects. These quality classes are directly linked to the established strength classes C14-C30, see figure 4.1.

NEN 5499	EN 338
T_0	C14
T_1	C18
T_2	C24
T_3	C30

Figure 4.1: visual grading categories (Munck et al., 2011)

Nowadays, thanks to technological advances, non-destructive machine strength grading can also be used to categorise wood pieces with more precision, resulting in more available wood classes. Both the volumetric mass and elasticity modules have a correlation with strength and can be used to predict the final design strength and quality of the wood. However, there is a large margin of error if only one aspect is tested. This can be reduced by also considering other indicators such as visual aspects, tassels, discolouration and thread drift. Different types of equipment can be used during the grading process such as X-ray measurements, laser scans, weight tables and high resolution cameras (Munck et al., 2011). These methods can be combined in one process with robotic sorting and machine learning to separate wood types so that A wood does not get mixed with B wood. This can be a cheap and efficient way to collect wood and create a database that designers are able to use.

This process can start at the waste collectors, by combining robotic sorting with machine learning techniques, precise databases can be formed and wood can be catalogued. Incoming wood can be sorted on properties like:

- Wood type (soft or hardwood)
- Treatment (painted or clean)
- Cross-section
- Length
- Structural integrity
- Volume
- Mass
- Density
- Visual quality

When these properties are know a sorting machine can label the pieces so they can be retrieved later and places them in the correct storage area. The properties can than be send to a SQL database which can be nationally accessed. Another scenario could be that each waste wood dealer creates their own database. In the database designers and other interested parties can filter for the required wood and link it to their designs. This way the database is be able to communicate with the design tool. Such a database is essential for designers to start working with salvaged timber and make this new way of design more attractive and accessible. The focus of this research project is not on the sorting and scanning process that has to happen on the front end, but instead on creating a potential design tool that can be linked to a future database. The focus area of this research project is visualised in figure 4.2.

4.2 Implementation proposal

While this research project does not focus on the front end of the circular loop a few ideas are provided on how such a circular process could look like. Eijk (2021) identified two potential scenario's through interviews with waste collecting companies. The first strategy is based on an already ongoing trend at the demolishers

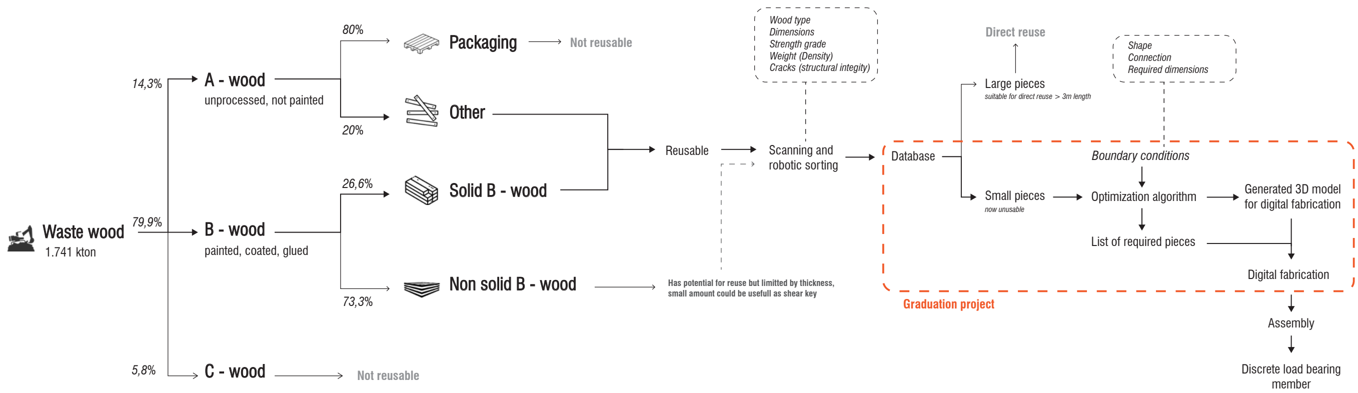


Figure 4.2: Processing workflow, adapted from (Eijk, 2021)

operating field. Eijk found that demolisher's are already taking initiatives in dismantling buildings and selling the parts to vendors and other interested parties rather than only tearing it down. Some demolishing companies even have their own store on company terrain from which collected building elements are sold. When waste becomes valuable enough to collect and sell, a more active role for demolishing companies can be expected and this scenario could expand even further. More demolishers will expand their operating field to compete in this new market which focuses on dismantling and selling the parts to stock-buyers rather than only demolishing buildings, which can already be seen on a small scale. Stock-buyers can than catalogue the collected elements and create a national database. Or if even taken further, the demolisher can also expand its role entirely and become the collector, storer and seller without the extra party.

The second scenario Eijk came up with for wood collection is that the processing companies that are now only sorting and shredding the wood are going to see potential in the reuse market. These collecting and processing companies could greatly contribute to the circular economy by expanding their operating field to not only processing but by adding storage and remanufacturing possibilities. Such parties are often already fitted with the right infrastructure to sort the incoming waste wood. By investing in new equipment that can scan, label and store the wood a national database can be created, designers can start using this in their designs and a new business case opens up. The circular economy can be boosted even more if these parties not only focus on storing and documenting but by also adding a remanufacturing facility that can turn waste wood into products. This can make using reused materials more attractive for all kinds of parties and consumers, not only designers. The infrastructure for logistics and transportation is already there, what makes these parties best suited for this job.

4.3 Dummy database

For the purpose of generating a structural element out of a varying stock, a fake database, also called dummy database, was made that could be linked to the computational tool. This database will simulate a potential future salvaged timber database in a format that a designer could expect to get after the waste wood is scanned and sorted. The database is made in a SQL format because this type of database is able to organize and filter large datasets more easily and quick than for example excel which can easily become slow and chaotic with complicated and large datasets. Moreover, the SQL database can be directly linked in the visual programming environment Grasshopper without needing file paths that commonly limit accessibility of

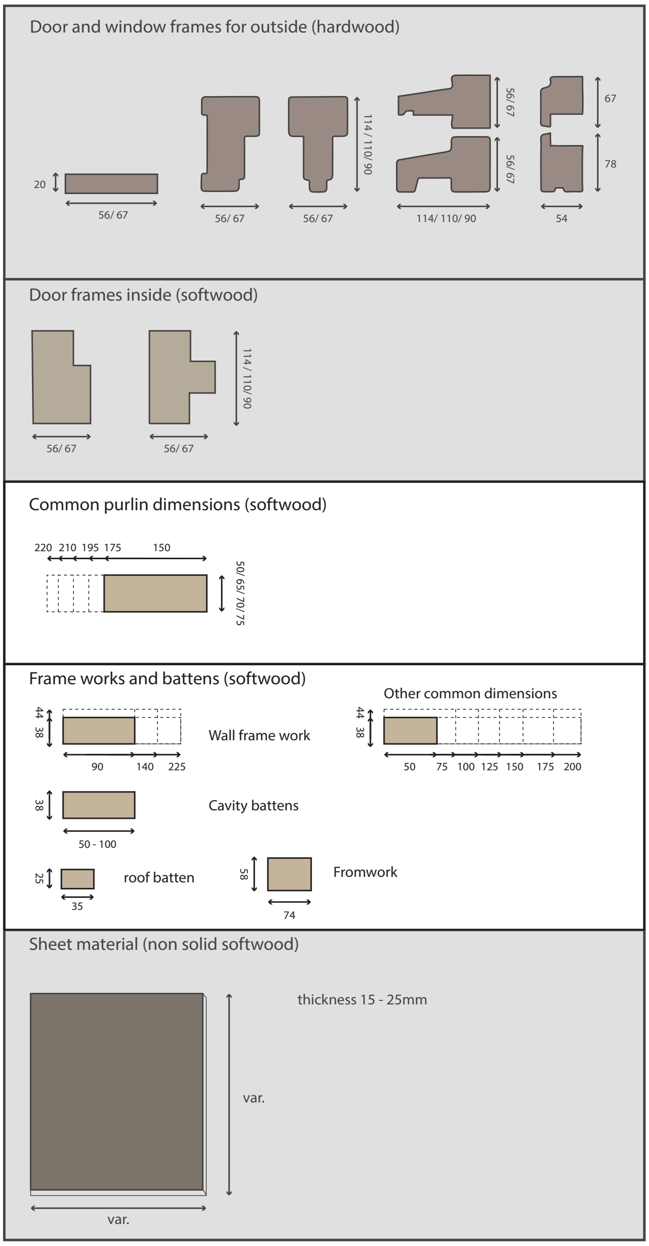


Figure 4.3: Cross-sections to be included in the database

the tool and are thus very useful for a designer.

In this project the database is made in the PostgresSQL but any kind of SQL language is suitable. Python is used to generate data that simulates incoming waste wood from the construction & demolishing sector, aforementioned in chapter three.

The database will contain common cross-sections that can be found in the construction sector like wall and other frame work, cavity battens and purlins, as these categories hold the largest share in the incoming waste. The dimensions of these cross-sections correspond with the common timber trade sizes of the NEN 5499. Figure 4.3 shows an example of cross-sections that are generated by a Python script and exported to the SQL database. The grey areas show other common found waste wood sections from the construction industry but are not considered in this research due to added complexity in geometry or

id_wood_piece integer	length_wood_mm integer	width_wood_mm integer	depth_wood_mm integer	strength_grade text	area_mm2 integer	Wxx_mm3 integer	Wyy_mm3 integer	Ixx_mm4 integer	Iyy_mm4 integer	painted boolean
1	570	50	210	C24	10500	87500	367500	2187500	38587498	false
2	270	65	220	C30	14300	154917	524333	5034791	57676664	false
3	580	75	220	C24	16500	206250	605000	7734375	66549997	true
4	720	70	210	C24	14700	171500	514500	6002500	54022498	true
5	540	65	150	C30	9750	105625	243750	3432812	18281249	false
6	620	50	220	C18	11000	91667	403333	2291667	44366665	false
7	850	70	165	C18	11550	134750	317625	4716250	26204061	false
8	590	50	220	C30	11000	91667	403333	2291667	44366665	true
9	750	65	150	C18	9750	105625	243750	3432812	18281249	true
10	540	65	165	C18	10725	116188	294938	3776094	24332343	false
11	410	50	165	C24	8250	68750	226875	1718750	18717187	false
12	270	50	195	C24	9750	81250	316875	2031250	30895311	true
13	570	50	220	C24	11000	91667	403333	2291667	44366665	false
14	750	75	220	C18	16500	206250	605000	7734375	66549997	true
15	630	75	195	C14	14625	182813	475313	6855468	46342967	false
16	610	65	220	C18	14300	154917	524333	5034791	57676664	false
17	860	75	220	C14	16500	206250	605000	7734375	66549997	false

Figure 4.4: Example of SQL database

because they do not fit the required thickness. In figure 4.4 an example of the SQL database is visualized. The length, width, depth, structural integrity and the visual condition are generated by a Python code using the random library. The width and depth are set according to the cross-section in figure 4.3. The length is constrained to 200mm - 1000mm. Pieces smaller than 200mm are viewed as too small to be used for this project. Pieces larger than 1000mm could potentially still be useful but as this project tries to find a new function for smaller scrap pieces and to comply to the wood cascading principals of the circular economy the maximum length is constraint to 1000mm. This database only contains these lengths, in a real database also other lengths will be included. In the SQL database the user can make quarries. These are actions to filter and analyse data. The user can easily find en select required data, making it perfect for a waste wood database. The structural integrity is based on the idea that the wood is weighed, visually inspected and scanned to determine the structural class. It could be that wood that was once C24, has degraded to a lesser C14 class. The values in this database represent the values after the scanning and sorting. C24 wood is used most often in the construction sector, so it is assumed that a lot of wood has degraded so C14, C18 and C24 classes. These are evenly distributed in the database. Higher classes are not taken into account. The data is generated in Grasshopper and then send to SQL to save the data. In Grasshopper the distribution of strength grades can be adjusted so that the design can be tested with different kind of stock varieties. Weight and mass can be expected to also be present in a real database. But as these properties dependent on each other and the dimensions of the wood, it cannot be randomly generated and thus is not considered in this dummy database. The area, moment of resistance and moment of inertia are generated columns. That means that these columns are linked with expressions to the other columns. This database shows how a national database could be set-up and what a designer could expect to get as input for a design. A real database would be much more extensive and also contain vendors, locations, other material properties or defects. With the Python extension of Postgres, Psycopg2, the dataset can be imported in Grasshopper.

5. Optimization

In this chapter the concept optimization in introduced. The research project aims to design a discrete structural system, linked to an excising stock. This objective concerns two main optimization problems: structural optimization and stock constrained optimization. The structure will be optimized to the acting forces and configured out of the available stock. This chapter highlights the general theory of optimization, discusses optimizations techniques and provides an overview of usable programs, boundary conditions and objectives.

5.1 General theory on optimization

Optimization is a concept that is know in every branch and relates to maximizing or minimizing a certain objective by adjusting a variable within a set of constraints that defines if the solution could be possible and valid. Optimization algorithms are used to incrementally improve the objective in order to find the optimal solution by making lots of iterations. Reaching a true optimal design can be difficult as often an improvement in one field results in a degression in another. Within the field of optimization a concept of 'Pareto-efficiency' is often used to determine the best solution. This is a state in the design in which no change could lead to improvement of an aspect without losing something on another aspect. The other possibility is to focuses the optimization on one single aspect and accept loses in other aspects. According to Kochenderfer & Wheeler (2019) a general formulation for the optimization of a problem can be expressed as:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned}$$

A problem always contains a variable, $\mathbf{x}$, which can influence the objective, f with a constrained region $\mathcal{X}$. An example of constraints within the structural optimization are mass, displacement, emissions of GHG, strength or stiffness. A lot of optimization formulations exist and are made to fit specific problems, but they can all be rewritten from this basic formulation.

Within the branch of structural optimizations before the time of computers commonly made by creating physical models to find optimal forms like hanging models to find a perfect compression only structure or to do destructive experiments and tests. Nowadays structural optimization is commonly done by numerical analysis methods like the dynamic equilibrium, geometric stiffness and the stiffens matrix methods. Such methods are widely used and solved by finite element models (F.E.M.). These are models which discretize structures into smaller hexahedral or tetrahedral elements to avoid harp edges and otherwise resulting singularities. The smaller the elements the more precise the results are but respectively also the slower the solver becomes. A lot of F.E.M. software exists like Karamba3D which is a plug-in for Grasshoppers visual programming environment. This is a tool which can quickly solve partial differential equations to find stresses and deformations. This software can be combined with existing optimization algorithms, plug-ins, to create a parametrically optimal design. However, this software lacks computing power to perform detailed analysis and is therefore mostly useful for early design stages. Sofistik is a more powerful F.E.M software which can also be linked to Grasshopper and used for more detailed analyses (Sofistik) & (Karamba3D). Anemone, a plug-in for the Grasshopper environment, can be used to create data loops to help find optimal results. Anemone can prove to be useful because most of the optimization software and plug-ins lack the option to start an optimization sequence which feeds information back in the script.

5.2 Structural optimization

The final topology of a structure influences a structure considerably and has a major impact on the visual appearance, embodied carbon, structural performance and the constructibility (Tam & Mueller, 2015). Therefore structural optimization is an important tool in order to create efficient and elegant structures. Structural optimization problems can be divided into three categories, shape, size and topology.

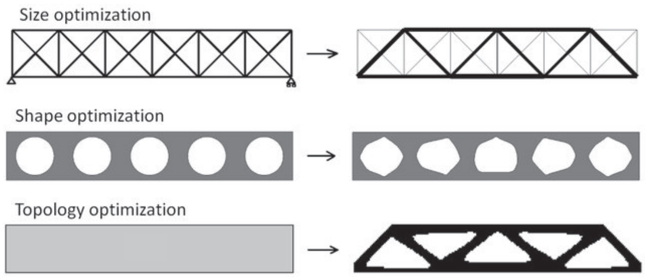


Figure 5.1: structural optimization types (Mozumder, C. K., 2010)

Shape optimization

Shape optimization is a problem formulated by the concept of form-finding. An optimization in this category can lead to a naturally efficient structure in tensile or compressive forces but never in both. An example for forming-finding optimization is the finding the optimal the optimal angle to place cables for bridges in. This optimization method is not further explored in this research.

Topological optimization

Topological optimization is a problem formulated by the spatial order and connectivity of a domain. This type of optimisation permits a domain, which can be

2D or 3D, to have zero thickness in several areas. The number and shapes of the holes are free to take any shape. Therefore it is the broadest form of optimization and it combines the other two optimization types. It is very closely related to shape and size optimization because an area is often cut out till the point that only the required structural material remains shaping the area in the best possible shape and size. This ensures optimal structural systems as all the unnecessary material is removed. The downside of this optimization method is that after the optimization no over-capacity is included and the loading scenario can never change again.

In this research topology optimization will be one of the structural optimization methods next to the size optimization. The topological optimized geometry is the input for the discretization after which a size optimization can be run. Topological optimization methods can be distinguished in roughly two focus area's: A structural based optimization, otherwise know as the optimal criteria method, and a more rational based optimization, also know as the heuristic method. The heuristic method is based on intuition and experience of the designer. Such methods are therefore not guaranteed optimal and often not used structurally. With heuristic methods often patterns in geometries are optimized and objectives like similarity, and least amount of distortions are pursued. This research will focus on the structural based optimization area that uses fixed criteria to determine the optimal solution.

Size optimization

Size optimization is a problem formulated by the concept of member sizes. In this type of optimization the structure itself, the domain, and the forces are known. A size optimization searches for optimal member sizes to use in a structure. An example of this type of optimization is the search for optimal profile sizes of a truss-structure.

Within this research size optimization is one of the main topics and focuses on a stock-constrained design. Such a way of designing requires a structural topology that is designed in order to make best use of the available elements in stock (Warmuth et al., 2021). That means that the objective is not focused on the cross-sections itself, but to place existing stock on the right place in the design according to given boundaries and conditions.

Already a substantial amount of research has been put into this domain, but has not yet been exhausted (Tomczak et al., 2023). Three different kinds of approaches for stock-constrained design can be identified in the existing literature.

The first being a infinite stock of a few types of standardized elements that can fill the design space. This

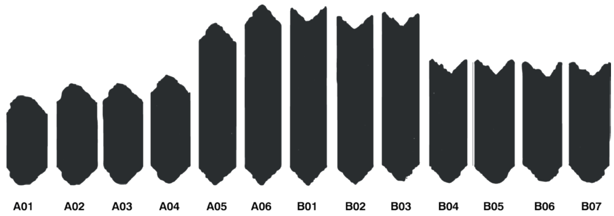


Figure 5.2: Size optimization with a kit of parts (adapted from, (Kunic, et al. 2021).

is the same principle that can be found in the reversible beam for the SDU (Kunic, Naboni, et al., 2021) which uses a kit of parts as available stock. These parts can be new but can also be viewed as salvaged timber that is remanufactured into standardized elements. The advantage is that this way a system can become uniform and more flexible for later configurations as parts are easily replaced or added. However, as the elements are cut to size there is also more waste, and this will results in more joints globally as the elements will all be of limited size to accommodate flexibility in the system. In other words, on a place were one longer beam would be able to fit this approach uses multiple smaller ones.

The second approach being a finite stock which is sufficient and used to create a design as can be seen in: (Parigi, 2021), (Hung et al., 2021) and (Brütting et al., 2021). A more extensive stock will therefore result in a larger design space and more possible configurations.

Figure 5.3: Size optimization with a highly variable stock

In this approach the objective can be to reuse the available stock directly if possible or to cut members and putting the cut-offs back in the stock database. This reduces remanufacturing costs and time but requires more extensive computational work in the design phase and extensive coordination during the building phase as each element fits only on one place. Furthermore, another added value is that each elements in the design can be of optimal length for that location which will result in fewer joints, respectively lower costs, and less degrees of freedom thus improved stiffness. This approach fits more in the circular economy as it beholds to one of its principals of wood cascading. Which is a concept in which timber is kept as large as possible in every reuse step because the bigger the elements the more future reuse possibilities it can accommodate.

The last identified approach being a finite stock that is not sufficient and which needs to be combined with new elements to fill the gaps and form a hybrid structure and can be seen in: (Tomczak et al., 2023), (Warmuth et al., 2021) and (Brütting et al., 2020). For elements like steel which are not abundantly available and in which a hybrid solution often can be more economical and environmental positive this approach seems the most realistic. But looking at timber, which has an abundance of waste material available which can easily be cut into the right size or be linked by extra connections this approach seems not necessary. Moreover, using salvaged timber is always more environmentally positive than using new timber which is not the case with materials like steel which have high emissions when remanufactured. Approach one or two seems more suitable for a discrete timber system and are the ones that will be focused on in this research project. A case can be made for the first approach that only elements within a certain range are remanufactured so that bigger pieces can be used for other reuse projects. This view fits within the wood cascading ideology which promotes keeping wood as large as possible to accommodate future reuse. Testing has to show what is the influence of the number of global joints and if there is and optimum element size.

5.3 Structural optimization methods

In this research project three optimization methods are considered. Two traditional methods known as: "Homogenization", "ground structure" and a new alternative method: "principal stress line" as visualized in figure 5.4.

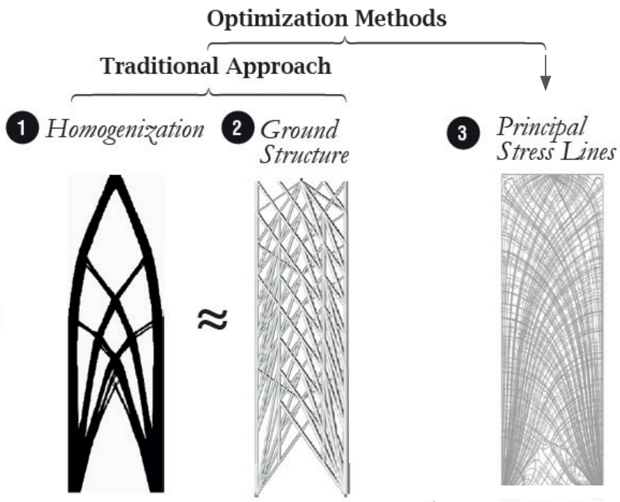


Figure 5.4: considered structural optimization methods (Tam & Mueller, 2015).

Traditional optimization

Within the structural based topological optimization focus area the traditionally used methods are "homogenization" and "ground structure" as depicted in figure 5.5. Within the ground structure method a volume is completely filled with bars and nodes, after which optimization starts by excluding bars in which the forces and stresses are negligible or small until

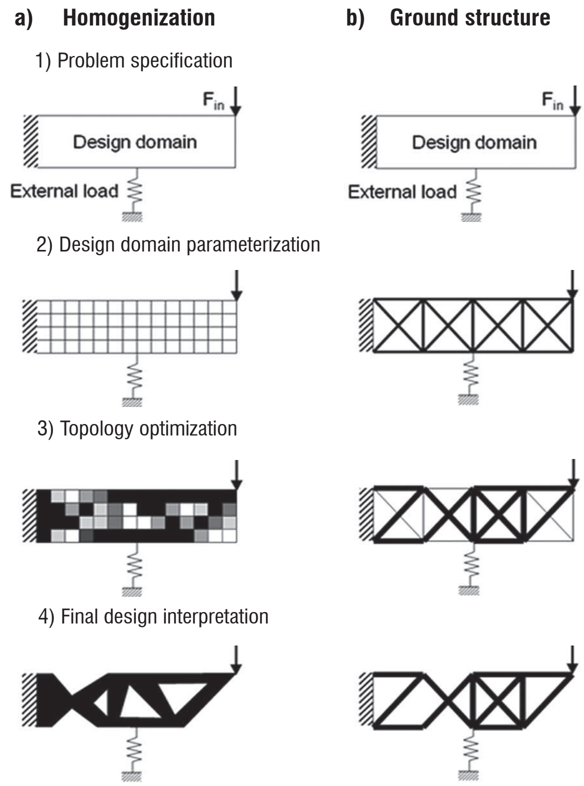


Figure 5.5: Typical synthesis procedures for compliant mechanisms using homogenization or ground structure, adapted from (Lu and Kota, 2006)

a pre-set percentage of mass reduction is achieved. Homogenization is an optimization method where a volume is converted to finite elements that can be either a void or have mass. This way the topology problem is transformed into a shape and size optimization problem in which the size of the void and the overall shape will determine the final structure. This method generates a porous structure by removing all unnecessary mass until also a certain objective is achieved, resulting often in a shape that comes close to the principal stress lines (Tam & Mueller, 2015). These methods are not without fault, and are often afflicted by issues that render results unusable like discontinuities in material, also known as 'islanding', point flexure's in which elements are connected to single points and gray areas in which variables take on values which cannot be interpreted correctly, see figure 5.6. Such issues can result in geometries that are not feasible to create and which require additional input of the designer. However it is difficult for designers to control the generated results. Errors can be resolved by adding filters, using

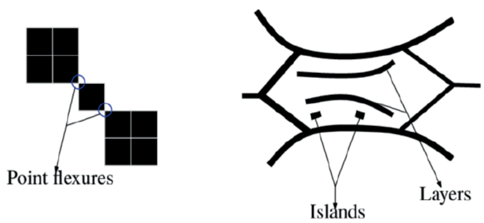


Figure 5.6: Optimization issues, on the left point flexure's, on the right 'islanding', adapted from (Kumar, 2016)

more sophisticated and extensive algorithms or by reformulating the optimization objective (Lu and Kota, 2006), (Kumar, 2016) and (Li en Chen, 2010).

When considering to combine structural optimization with stock-constrained optimization, these two methods align with the previously mentioned top-down discretization approach in which a given design domain is discretized.

For homogenization the two optimizations processes are separated. The size optimization begins after the geometry of the structural optimization is found. A fitting workflow could be to first structurally optimize a volume, which will be used as design domain for the size optimization. Excising plug-ins for Grasshopper like Millipede or Topos can accommodate such optimizations. The resultant volume can then be discretized again into small voxels. Lastly, a stock can be assigned by combining voxels to fit a certain length that is imputed by the salvaged wood database.

For the ground structure method a more efficient workflow can be sought after in which the two optimizations are combined in one program. The design domain could be randomly filled with bars that are available within the database, minimising cut-off. By adding constraints to avoid 'islanding' and infeasible designs an algorithm can start excluding unnecessary bars. This can all be done with the plug-in Karmaba3D inside Grasshopper and a matching algorithm. The downside of this approach is that the design space is first filled with bars which does not guaranty a global optimum. The matching process is taken out of the optimization and the algorithm cannot determine the best place or fibre orientation for a certain piece of wood because these are already placed. But, in such a way stock and structural optimization are completely intertwined.

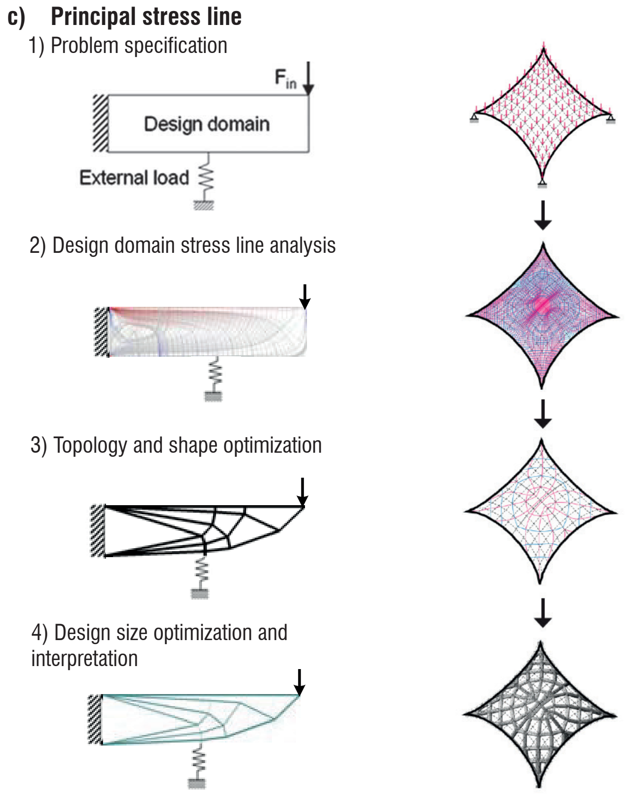


Figure 5.7: stress line inspired topology, adapted from (Tam & Mueller, 2015) with on the right a shell structure and (Chen and Li, 2010) with on the left a beam structure.

Principal stress line optimization

Another, relativity new, approach can be found in (Tam & Mueller, 2015). Here the principal stress lines are used to create an efficient structure. It is a more direct approach in contrast to the traditional approaches in which the final optimized result usually also resembles closely to these lines. Principal stress line optimization can be traced back though history. One of the most influential designer that implemented this approach was Nervi, his structures were often influenced by the concept of force flow and its architecture is still highly praised for it. This optimization method is especially suited for continuous materials like concrete or steel and less for timber as this is an orthotropic material and has manufacturing limits for free form structures.

Principal stress lines are pairs of orthogonal lines that illustrate the paths taken by internal forces in response to an applied load. These lines inherently convey the optimal structural topology and represent optimal routes for material continuity. Therefore they can resolve already the common issue in the traditional methods, ‘islanding’. Principal stress lines represent a striking regularity and order and are not affected by changes to material properties or scaling of applied forces. Li en Chen (2010) sees this approach as much simpler and faster than the more well-known traditional methods aforementioned. In this method initial stress-lines that follow out of a F.E.M. analysis are used to find the optimal structural geometry. These lines are very dense, imprecise and cannot be directly used as input, therefore they are used to obtain structural data. This data forms the base of a new seeding plan for the generation of new high resolution stress-lines that can be materialized to produce a structure. The seeding plan is made by interpolating the obtained stress trajectories and can be altered to fit the designers constraints, see also figure 5.7. Hence, a topological optimization algorithm is avoided with this method and the focus is on the interpolation and finding the right lines.

With this approach, the issues encountered in traditional methods, such as discontinuity of material and grey areas, do not exist. However, According to Tam and Muller (2015) this method has some issues itself which are related to limitations in the stress line analysis of current softwares like low stress line resolution, poor stress direction interpolation and discontinuities in stress lines. Even so, with a simple algorithm from Chen and Li (2006) a new set of stress lines can be generated by interpolation, which empowers the designer with some control of the subdivision of the stress lines and thus the geometry of the final structure. This accommodates an active and configurable design process in which the designer can strongly participate by adding constraints. The lines are not directly materialized and can also act as a reference allowing for potential discrete elements to be placed along.

This approach allows for a previously mentioned bottom-up discretization method. In such a discretization method a line or surface is used as reference alongside of which a composition can be created. An important difference with the other two traditional methods which use a top-down method. This method allows for stock to be directly placed alongside the created reference line and eliminates the need for a discretization process in voxels or other type of elements. By combining these principal stress lines with the force flow lines and local stress values , the required thickness of timber can be acquired. This information can be fed into a matching algorithm which can start create compositions.

5.4 Discrete timber optimization

Structural optimization for discrete timber is relatively new and as aforementioned executed a couple of times by de SDU. In chapter one already a brief outline of the workflow that the SDU uses is given. This paragraph seeks to provide key attributes specifically needed for the optimization of discrete timber systems.

The optimization workflow used in the projects of the SDU is quite consistent. Both in (Naboni & Kunic, 2019), (Kunic, Naboni, et al., 2021) and a new unpublished work from Jensen et al. (2023), the Re-voxlam truss. The general optimization workflow of these projects consist of a voxel based discretization method combined with homogenization optimization and principal stress line analysis. The Re-voxlam project is a bachelors project from students of the SDU. This project sought, similar to this research, to create a discrete system using waste wood. Different from this research is that the parts are laminated with glue and are therefore not reversible. Moreover, the parts also have a high remanufacturing process with as it seems large cutting losses due to fibre orientation manipulation. The optimization workflow of the Re-voxlam is shown in figure 5.8. By homogenization a volume is optimized and discretized in voxels, then structural attributes required for timber design are found by stress line analysis and assigned to these voxels. Lastly, the voxels can be converted to a pre-designed kit of parts.

In order to translate digital information to physical building blocks it is imperative to collect and assign suitable attributes to the model. In (Naboni & Kunic, 2019) four fundamental tectonic characteristics for discrete timber design optimization are given. This combinatorial matrix contains: Material strength, orientation, connectivity and assembly direction. These attributes, have to be embedded in the digital data, and are vital for efficient and effective optimization of discrete timber systems.

Material strength - Wood is a material which can vary a lot in strength per piece, varying pieces best placed in compliance with the occurring stress to

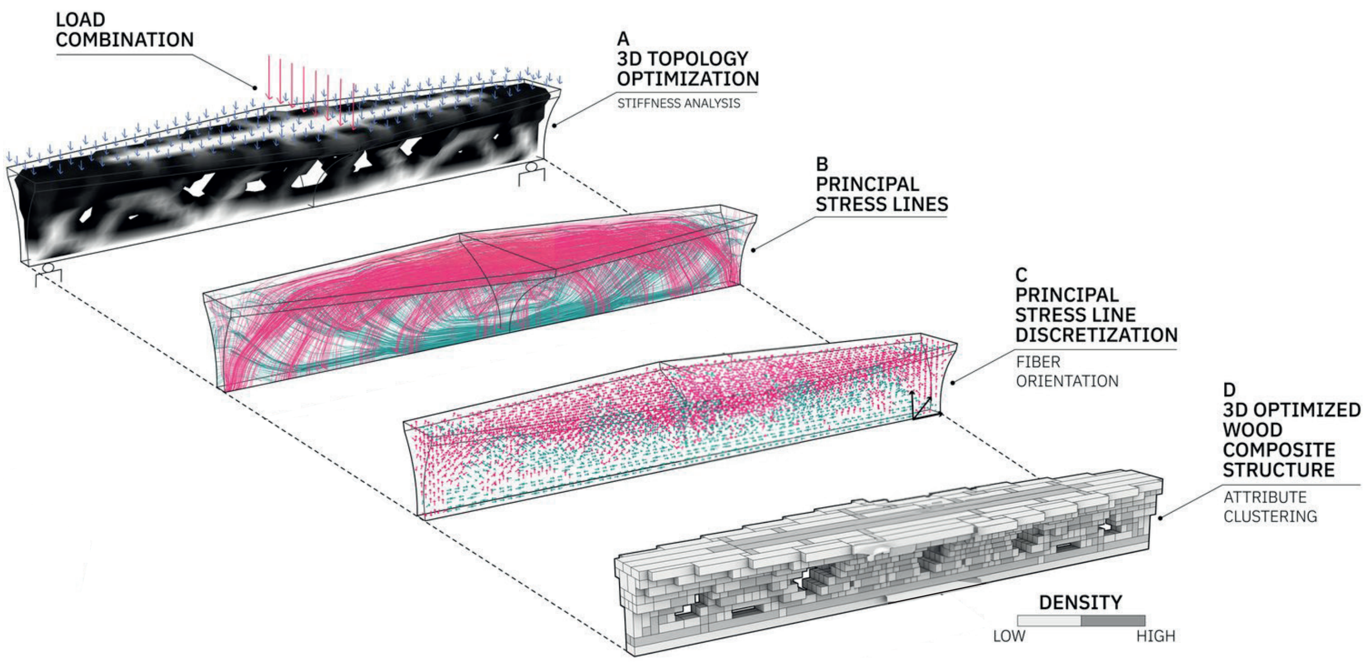


Figure 5.8: Discrete timber optimization process combining homogenization and principal stress lines (Jensen et al., 2023c).

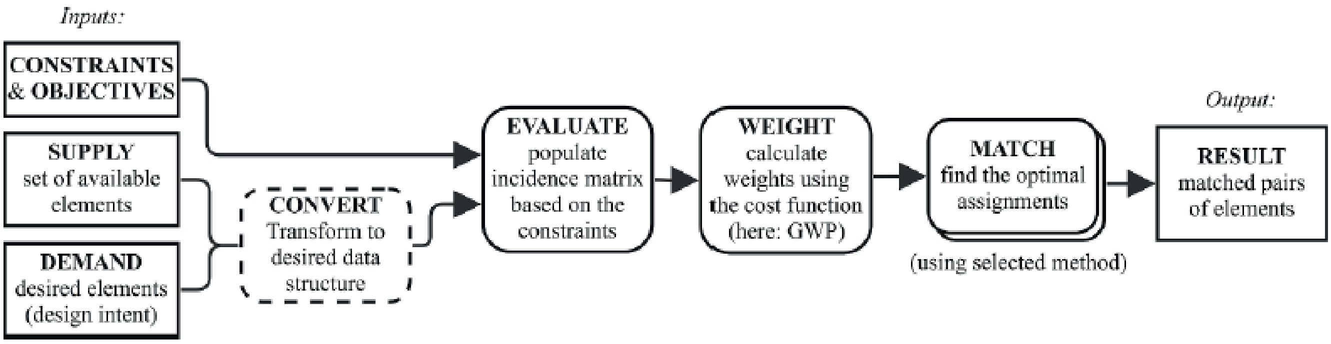


Figure 5.9: A matching algorithm's workflow (Tomczak et al. 2023).

increase their efficiency, improve overall weight, lower material use and reduce cost. Thus high performing material should be placed in zones where maximum tensile and compressive strength occurs.

Orientation - Due to woods orthotropic nature its properties vary in each direction. Wood loaded at 45° or even 90° has far less capacity than when loaded at 0° as aforementioned in chapter one. Controlling the orientation of the timber pieces is essential to create an efficient structure. This can be done by aligning the wooden pieces to the force trajectories obtained by stress line analysis.

Connectivity - Especially a discrete system's mechanical behaviour is dependent on the interconnectivity between its parts. Connectivity coincides preferably with the force flow induced by external loading conditions. Connectivity can be sought in three coordinates, X, Y and Z. Constraining connectivity conditions in the matching program can ensure overall mechanical capacity and reduce computational time.

Assembly - The final design has to be able to be assembled and reconfigured and the parts have to be able to accommodate this by self-alignment or by constraining connectivity in a specific axis.

5.5 Optimization algorithms

Optimization objective

To be able to select an algorithm, first the objective must be clear of what needs to be optimized. The structural optimization is done with excising software and the plug-in Karamba3D inside the Grasshopper programming environment so for this part there is no need to create an finite element program. For the optimal matching of a stock there is, however still not a suitable commercial software available. Phoenix 3D is the first plug-in in Grasshopper that allows for a stock-constrained design. But alas, is this software only suitable for truss structures and does not provide a way for layered assembly or a firm control over the design. Thus for this objective a workflow has to be set up with a suitable algorithm that can solve the matching problem. In mathematics, various algorithms are available to solve and optimize such problems, each with their own level of complexity and advantages (Huang et al. 2021). A brief overview of the most seen algorithms will be given.

The algorithm should be able to minimize necessary cut-offs of the available parts while generating possible geometric layouts with a non standard set of elements. This can be done by either a single objective, thus focusing only on a optimal place in the design related on length of the stock elements, but also in a multi objective form in which other aspects like structural lightness, different strength grades of the elements or the optimal combination of different wood types can be considered. Such a task can take days or even weeks if done manually and automating this process is therefore an important step for stock-constrained design (Tomczak et al. 2023). Tomczak created a general workflow of matching algorithms shown in figure 5.9.

Existing matching algorithms

In existing literature a number of algorithms can be found. The most common seen algorithms are the Greedy search algorithm in (Huang et al. 2021) and the Mixed Integer Linear Programming (MILP) is used in Bruting et al. (2020, 2021) and Tomczak et al (2023). is a heuristic type of algorithm that is fast and simple to implement but does not guarantee a global optimum. It always searches for a local optimum solution but without a global view. In other words, it cannot reverse its previously made decision, even if it turns out to be wrong. Mixed Integer Linear Programming (MILP), another available algorithm to solve matching problems. Such an algorithm is more complex than a greedy search and therefore much slower. However this type of algorithm always searches for a global optimum. MILP algorithms work by adding a cost function in the problem assignment. Each variable is scored on how well it fits within the set constraints. The solution with the best cost function is considered the optimal solution. Tomczak et al (2023) found that MILP does find a global optimal solution compared to the greedy search, but that in small projects the differences are negligible.

Tomczak sees potential obstruction of creativity and exploration of possibilities when problems become bigger due to long computation time and advocates for a fast algorithmic approach that can be used efficiently to iterate through multiple design options even it does not result in a global optimum. Greedy search algorithms will result in more cut-off waste compared to more thorough algorithms but elements with a surplus length can be cut and put back into the database to be matched again making it an interesting option considering its speed (Tomczak et al. 2023).

More unknown algorithms like the Hungarian algorithm (Huang et al. 20121) and SPEA-II (Parigi, 2021) are currently tested for stock-constrained optimization with promising results, but not further investigated.

Knowledge gap

The aforementioned algorithms are used in research projects that focus on creating reciprocal structures. This requires different constraints and objectives than a massively layered assembly. In reciprocal systems the only constraint is the length which can be easily manipulated by cutting the piece or moving the nodes that represent the end points of the beam. This creates longer or shorter pieces effectively only considering one dimension. In massively layered elements this is not the case. To ensure global stiffness and accommodate for connections the boundaries of the pieces cannot overlap in the {X}, {Y} or {Z} direction making the optimization problem more complex and three-dimensional. Fast computational performance is key especially when considering three dimensions which can result in large computational time limiting the functionally and design flexibility. Existing algorithms for bin-packing could be used to create massively layered compositions. These are algorithms that try to fit as many items in a bin as possible and often used to efficiently pack a pallet with items.

Another approach could be to use the aforementioned algorithms to solve combinatorial problems. These are problems that involve a finite set of elements with the goal to find the best arrangement that minimizes cut-of waste. Combinatorial computation can result in large computational time, when analysing a large stock. To reduce computational time a Dynamic Programming Table method can also be considered. This is a method that solves the global problem by breaking it down in smaller sub-problems. A table is constructed in which the solutions to these sub-problems are stored. This process is called memoization and significantly reduces the time complexity of the algorithm (BasuMalick, 2024). The key advantage over a greedy search or MILP is that this method avoids redundant computations reducing computational time while still reaching a global optimum. It provides a balance between efficiency and accuracy which can outperform the greedy search and MILP methods especially when using a large stock - which is the case in this project.

5.6 Discrete optimization conclusion

Three structural optimization methods were identified: size, shape and topology. Size and Topology optimization are applicable for this research project and will be used for the design.

Within the size optimization two relevant approaches were found. The first being a kit of parts which requires intensive remanufacturing but does allow for more flexibility and future adaptation. The other being a more direct reuse in which cutting off pieces is limited which reduces remanufacturing costs, material waste and future flexibility, but requires intensive computational planning and design.

Within the topological optimization three structural optimization methods have been identified.

The first being homogenization, which can be executed with plug-ins in Grasshopper like Millipede or Topos. This method removes material were forces are too small, generateing the optimal geometry based either on maximising stiffness, strength or minimizing volume. This method allows for a voxel based discretization of the structural geometry to start the matching process. In this method topology optimization and size optimization are separated and size optimization starts after the topology optimization is finished. For the size optimization a separate matching algorithm has to be written.

The second method is called ground structure. This method fills a predefined design space with bars and starts optimization by exuding bars until a either stress, stiffness or volume based objective is met. With this approach the matching problem is intertwined with the structural optmisation. A design space is filled with elements from a stock, a designer can influence where in the colume what type of elements are placed by creating zones. However, placement cannot be done based on finite element analysis of the structure and rests on standadised and predetermined rules.

The last method is the principal stress method. This method uses the principal stress lines in a structure to create a optimised topology. These stress lines contain useful information such as force flow trajectory and stress values. Either a top-down of bottom-up discretization method can be used. Thus either the structural geometry can be discretized into voxel, or the lines can be uses as reference for an algorithm to place stock along. This method gives the designer more control on the final geometry but is relatively new in discrete timber design.

Overall, the ground structure method seems logical as stock is directly placed but does not ensure a global optimum considering strength grade optimisation. Homogenisation has proved to be successful by the SDU but the final geometry is hard to influence for a designer and often paired with issues. The principal stress line method on its own seems, perhaps, not most suitable for timber design as the output does not fit to the characteristics of wood. However, the principal stress line could prove valuable to determine high stress areas. Combining the ground structure method with the principal stress line can accommodate for good performing strength grade optimisation.

For this research project, either a combination between principal stress line method and the ground structure method seems the most promising. For the reason that the stress lines can enhance the ground structure method with information on force trajectory and high and low stress area's to accommodate for strength grade optimisation.

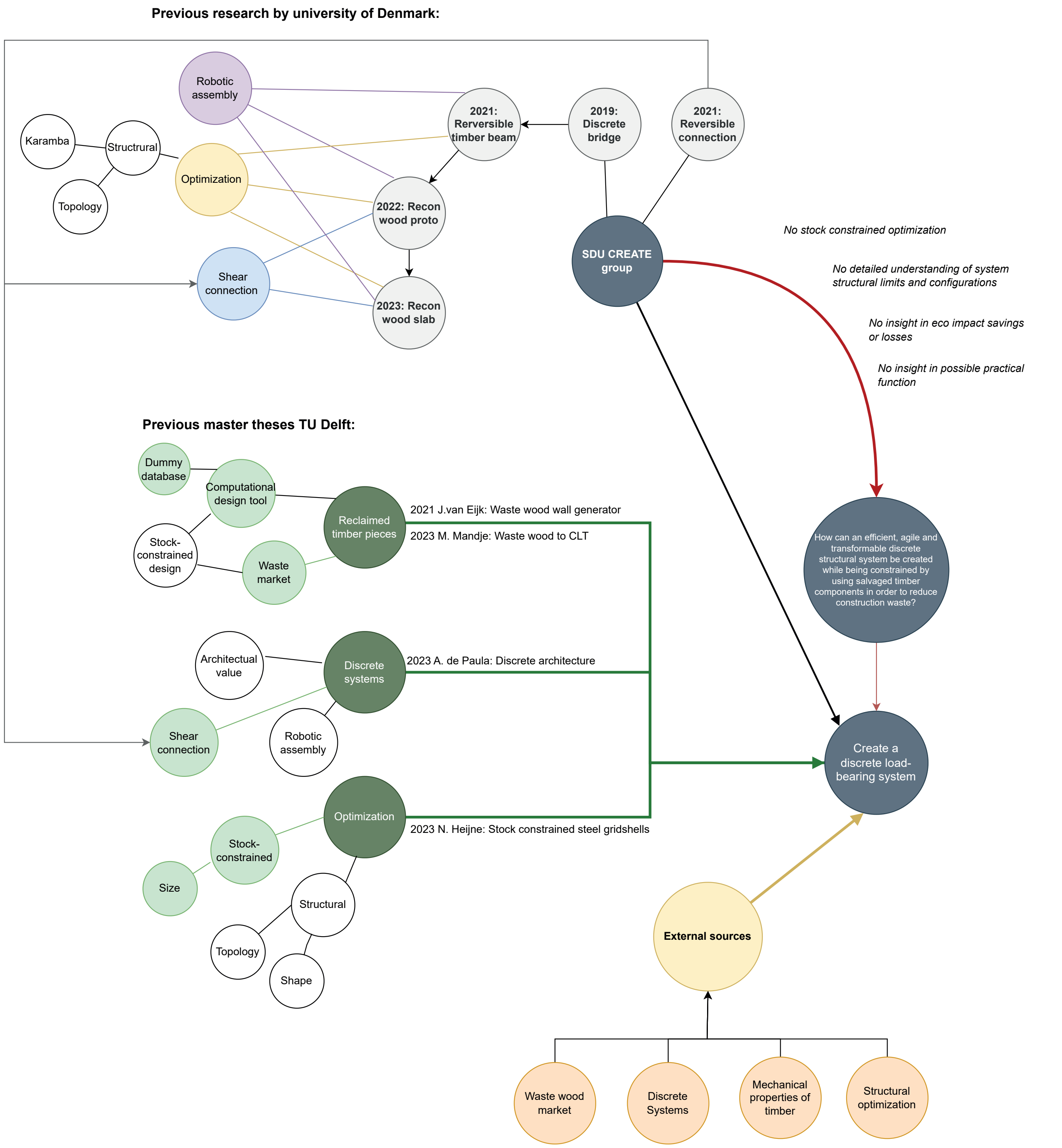
For discrete timber design it is important to focus on four attributes to achieve an efficient structure: varying material strength, fibre orientation, connectivity and assembly. These tributes ask for a multi-objective problem formulation.

II Development Phase

Knowledge gap

Through the literature review a gap in current knowledge and applications of discrete timber could be identified. Currently, stock constrained design is not yet seen often. Only a few projects have been found implementing stock constrained design on small scale reciprocal structures. The SDU has also shown progress in layered design but either not with the implementation of stock-constrained design or within the circular philosophy of using reversible joints, modular parts, minimizing processing effort and cut-off waste to accommodate future reuse. In the figure below a scheme is made which aims to highlight these knowledge gaps. A lot is know and tried but all in separate projects. This research project aims to combine previous works and results into a new research project, focussing in massively layered discrete timer load-bearing structures adholding to circular principles of minimizing waste, and accommodating future reuse. The befitting design vision is formulated as:

"To design a generative reconfigurable structural member out of available waste wood to support a efficient, circular and transformable form of architecture without limiting spatial flexibility and future adaptation."



6. Structural design

6.1 Design objective

To highlight the full potential of discrete timber for structural applications this research project will focus on developing a tool, able to generate portal frames using available reclaimed wood. In other words, this project seeks to develop a parametric portal frame generator and design a new type of circular timber structural system. The approach, challenges, gains and shortcomings found during this process will be discussed in the following chapters.

A portal frame is a typology which is essentially a structural frame consisting of two columns, uprights, and one or two beams, joists. The frame is often rigid, meaning that it is stable by moment connections between the elements. The structural typology of portal frames is chosen because this is an area which could benefit greatly from topology optimization due to large spans. Moreover, portal frames offer a lot of design options and applications. Structurally the typology requires analysing both vertically and horizontally loaded elements, and make together with the moment connections for an interesting challenge for discrete timber design. Architecturally discrete optimised portal frames can prove to be an interesting feature to. As each composition is unique a new type of architecture timber is introduced which is hopefully able to reflect the previous stated design vision: *"a generative reconfigurable structural system created from available waste wood that supports a efficient, circular and transformable form of architecture without limiting spatial flexibility and future adaptation."*

Based on the literature review the best fitting optimisation approach for this design objective is the ground structure method. This is an optimisation method that completely fills a design space with bars and iteratively removes the unnecessary bars. This method is explained more in-depth in chapter 5. The workflow is illustrated in figure 6.1, Design spaces i.e. the columns and the beams, which together form the portal frame, are iteratively filled with pieces from the inventory. This requires a program that is able to communicate with the database. An optimization loop

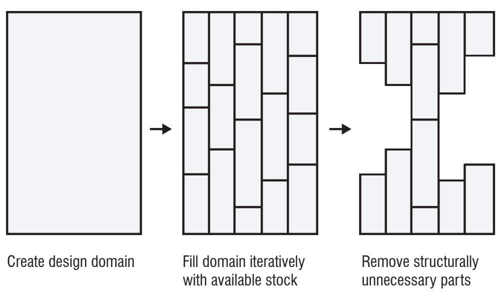


Figure 6.1: workflow of optimization process

will remove all pieces structurally inactive or not required. The final geometry consists of parts which have to be connected to create a global stiff system. The design of the parts and the connections and the structural theory of the system are discussed in this chapter. In chapter 7 the optimisation process and algorithm are discussed.

6.2 Design criteria

The design assignment is formed based on aforementioned discussed literature. The system is designed according to six hard design criteria and one hard optimization criteria. The design criteria are: massive layering, reversible joints, ductile system, unique parts, minimisation of fabrication effort and tectonic flexibility, see figure 6.2.

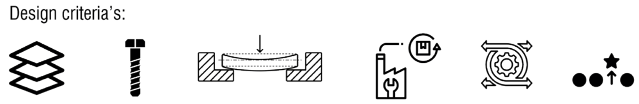


Figure 6.2: design criteria, from left to right: Massive layering, reversible joints, ductile system, minimize re-fabrication, tectonic flexibility and use of unique parts.

These design criteria should ensure a circular system which is focused on the theme of future reusability allowing salvaged timber to get a second, third and perhaps even fourth life. The optimisation criteria is centred around strength grade matching. This means that besides the matching of the unique parts in a design space aiming to minimize cutting, the placement strategy also includes the strength of the part. In chapter 2 and 5 this was found to be an imporant optimisation criteria of timber as its quaility can vary much. In other words: higher strength wood should be placed in area's with higher stress and vice versa. Initially the orientation of wood was also included in the optimisation design criteria, but could not be implemented due to time constraints.

6.3 Local geometry (parts)

Previously discussed literature concluded that the composition type and geometry of the parts has a large effect on the global stiffness, local joints and circularity of the system. Based on the literature three potential composition types were identified for a massively layered assembly, and depicted in figure 6.3.

Shear keys can be milled into the parts to create interlocking connections with a running type of aggregate. This would result in very strong and overall global stiff composition, but would require a lot of processing of the wood, reducing large parts of the cross-sections and the future reuse potential. A

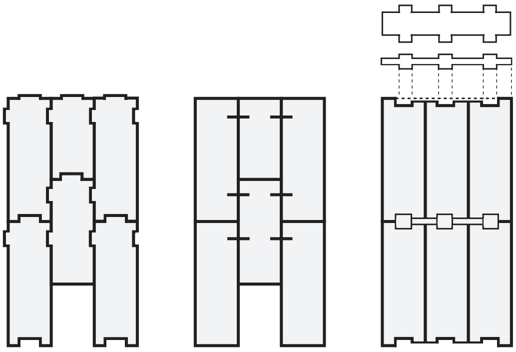


Figure 6.3: Aggregate possibilities from left to right: running aggregate with extensive processed shear keys, running aggregate with connectors, uniform aggregate with engineered board shear keys.

combination of dowels on the long faces and milling on the short faces could ensure a modular part that can be reused for similar projects. Fasting of the parts would be most efficient with glue but to accommodate circularity external fasteners are required.

Another approach is to create a running type of aggregate with parts connected by pens. Each pen can be placed on a modular distance, another method is to use standardised holes to accommodate future reuse more effectivly. This type would also require additional external fasting to prevent a horizontal hinge form forming, enhancing the global stiffness and creating a more ductile system.

The last approach is to create an uniform structured aggregate in which parts are connected by engineering board sheets. Milled slots on the top and bottom of the sections allow for tight connections. This would however require a lot of the same cross-sections and allow for little diversity in the stock. Milling is reduced to a minimum and parts can be turned into a modular system to accommodate future reuse more easily. The uniform aggregate however would result in a less stiff composition, and additional fastening would be required to enhance the global stiffness.

Based on the design criteria the second approach, using a running aggregate, seems best fitted. Key arguments are the minimal fabrication effort, future reuse possibilities through modular design and the potential implementation of a highly diverse stock. Global stiffness has to be obtained by a befitting connection type.

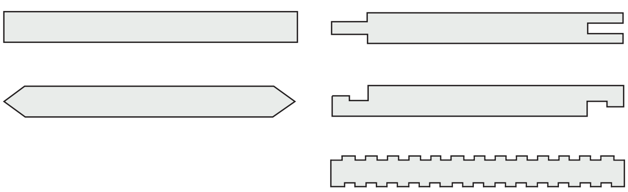


Figure 6.4: Potential geometric shapes for the timber parts

Parts can take various geometries to accommodate better interaction between elements as depicted in figure 6.4. Arrow headed shapes can help in alignment during assembly and expansion of the wood. Finger joints can help in transferring loads between parts longitudinally. The choice was made to reduce fabrication effort to a minimum and use a rectangular shape so loads can be transferred longitudinally between parts by either compression or pen connection.

6.4 Local connections (links)

The local joints determine the overall global stiffness of the discrete system but also the degree of potential future reuse of the parts. Designing a reversible joint that can provide enough stiffness and accommodate future reuse as much as possible is therefore an important part of the system. There are many types of connections, and the running aggregate allows for screw or pen connections, where a pen can be either a timber dowel, or a steel bolt. Glued connections are not considered as they cannot facilitate disassembly. The joints have to provide stiffness and ductility and withhold a horizontal hinge from forming. Figure 6.5 illustrates the considered connection types for this type of aggregate. In total three kind of connections are considered, steel connections, hybrid dowel connections and dowels - timber connections.

Screws and bolts both have the ability to ensure a tight fit between the parts, compressing the members faces by a tensile force. This ensures that a percentage of the section is activated and helps in distributing the load throughout the entire cross-section. In other words, not only the connector is loaded and can transfer loads from one member to the other but the faces of the parts can also transfer loads through the generated friction from the pressure of the connector. Composite behaviour ensures a higher global stiffness and provides some form of ductility. However, using steel connectors would require a large amount of fasteners as each element in the system has to be connected and compressed, enlarging its environmental impact. Moreover, using bolts results in random bore holes in the parts diminishing future reuse.

Instead of steel connectors also timber dowels can be used. Dowels show however the opposite behaviour compared to a steel connector. Using dowels perpendicular to the applied load results in effectively only loading the dowels as the faces of the parts are not compressed. With a very tight fit some composite behaviour can be expected but not to the extent of steel connections. This is because almost no pressure is put on the composition, resulting in little aid from friction between the elements. Therefore using solely dowels will result in brittle failure and lower global stiffness.

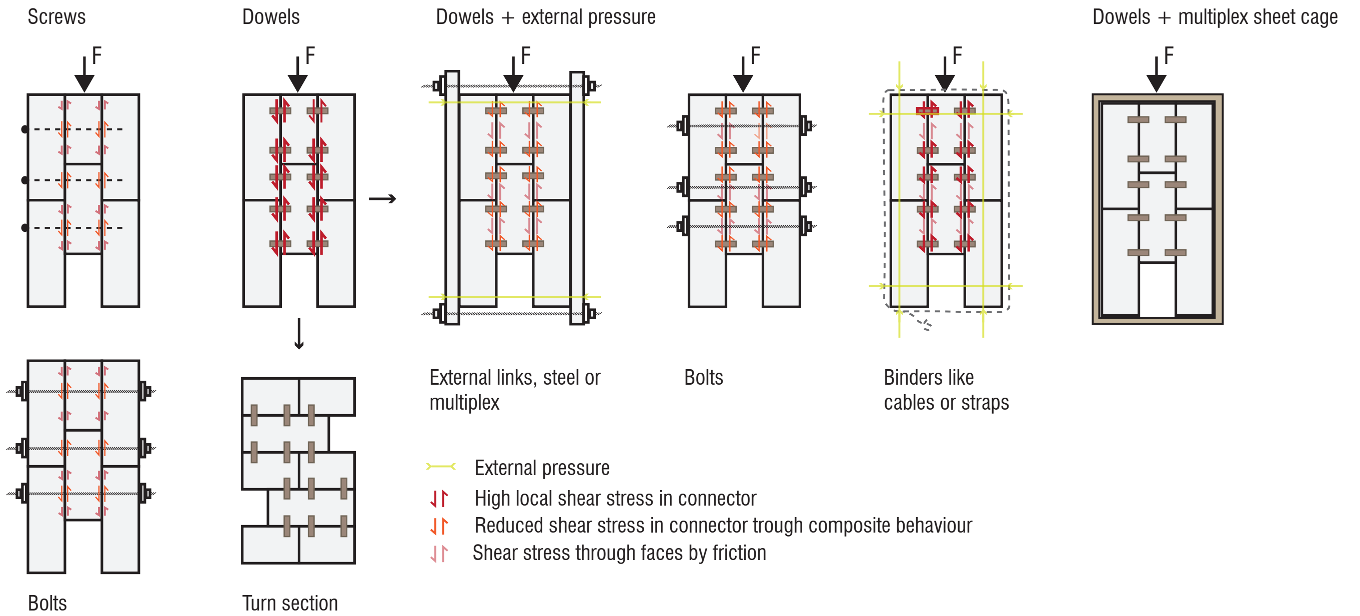


Figure 6.5: Potential connection types for adaptable and future proof discrete timber design.

When using dowel laminated timber (DLT) in practise, this effect is usually avoided by turning the section 90 degrees, placing the dowels parallel to the load. This way gravity provides composite behaviour. For linear elements like beams and columns turning the section is not very efficient and reduces optimisation possibilities as fewer parts can be removed.

Another method to create composite behaviour using dowels is by applying external pressure on the section so that the faces of the parts can be activated to transfer loads throughout the assembly. This can be done with numerous approaches like external links, bolts or by creating a binding system with a cable, sheet or rope like material. Each application ensures pressure on the assembly and a frictional force between the parts. Moreover, using external links results in lower overall needed fasteners and minimizes random holes in the timber parts. This, in turn, aids the future reuse potential of parts compared to using bolts. External links can be made either from thin plated steel or thick plywood. Solid timber will become to large and stocky to resist the pressure form deforming the wood. For a binding system either a flax fibre, steel cable, steel pallet binder strips or simple ratchet straps can be used. The downside of using narrow and small binders like metal strips is that the timber is easily crushed by local peak stresses. Furthermore, pressure from binders is coming from the top and bottom and sides, putting vertically also more stress on the dowels enacting a new failure mechanism of sliding.

The last identified connection type is a combination of dowels and a plywood engineered boards, i.e. an external cage. This cage locks the parts together, removing the risk of a horizontal hinge from forming and providing high stiffness and some ductility. The downside of this connection type is that the optimised structure is completely hidden, it adds a lot of weight to the structure and requires a lot of screws to fix the cage to the structure. Parts will end up with lots of bore holes, diminishing future reuse. On the other hand, engineered boards are the largest category of waste wood and thus abundant. Encasing the structure enhances its fire resistance and in put in contrast some holes can be made to showcase parts of the optimised structure for architectural purposes or non at all.

Decision making

To aid the decision making process a Multi Criteria Analysis (MCA) was done. This is an analysis that ranks options, in this case connection types, based on weighted criteria. The MCA is enclosed in annex A. Six criteria are used to rank the connections: *Architectural impact, Potential future reuse, Global stiffness, Ductility, Environmental impact and fire resistance* with respective weight factors: *0.05, 0.3, 0.3, 0.10, 0.20 and 0.05*. Each connection is ranked with a score 1 - 4, 4 being the best outcome and 1 the worst. Crucial criteria are *potential future reuse* and *global stiffness*, high scores in these criteria results in overall high ranking. These themes are most important as they reflect the hard design criteria. Strength of the connection is important for the system to work structurally and circularity is very important to accommodate future reuse and lower virgin material consumption.

The connection type with the highest ranking is a combination of dowels and timber external links and has an overall score of 3.1. The option with steel external links reached an overall score of 3.05. Both options reach similar scores and can both be viewed as viable options. Decisive arguments for the linking system are little affliction to the timber pieces and keeping potential future reuse high. Large pressure can result in high friction, good composite behaviour and high global stiffness of the system. Due to timber pressure distributors (links) low amounts of steel are needed reducing the environmental impact. This project will continue developing the hybrid dowel connection with timber links.

6.5 Structural principle of the joint

The chosen connection type relies on timber dowels and on friction between the parts created from horizontal compressive forces. These forces are generated from external plywood links that clamp the pieces together. In figure 6.6 the structural principle is illustrated. On the left a potential optimised cross-section is shown. Dowels are used to connect the pieces however due to vertical loading only the dowels can be used to transfer forces to consecutive parts. This connection type is called a section without composite behaviour, the parts in the section do not work together and effectively only the dowels are loaded, governing the global stiffness of the system. This is a low strength, brittle and ineffective connection.

On the right the same cross-section is shown but with external plywood links which are tightened by steel bolts. These external links are clamping the parts by compressional forces acting perpendicular on the parts. This compression generates a frictional resistance in the opposite direction, effectively utilising some composite behaviour. The composite behaviour reduces dependence on the dowel connection as the faces of the parts are also able to transfer loads to consecutive parts. The amount of composite behaviour is dependent on the force perpendicular to the section which in turn is restricted to crushing strength and bending resistance of the plywood.

Because the section can be highly irregular, a spacing block is added between the plywood so the structure is able to clamp the entire section. The lower this spacing block is in respect of the sectional height, the lower the bending moment in the plywood, resulting in higher friction resistance and composite behaviour. However, large compressive forces on this irregular geometry can also cause unintentional internal stresses and failure mechanisms like sliding, putting even more stress on the dowels. Another mechanism that could occur is that certain pieces start bending, effectively pulling out the dowels and counteracting the desired behaviour. This connection is still highly theoretical and requires mechanical testing to get an understanding of the internal stress distribution and the amount of composite behaviour that can be generated without causing counteracting effects. The assumption is made

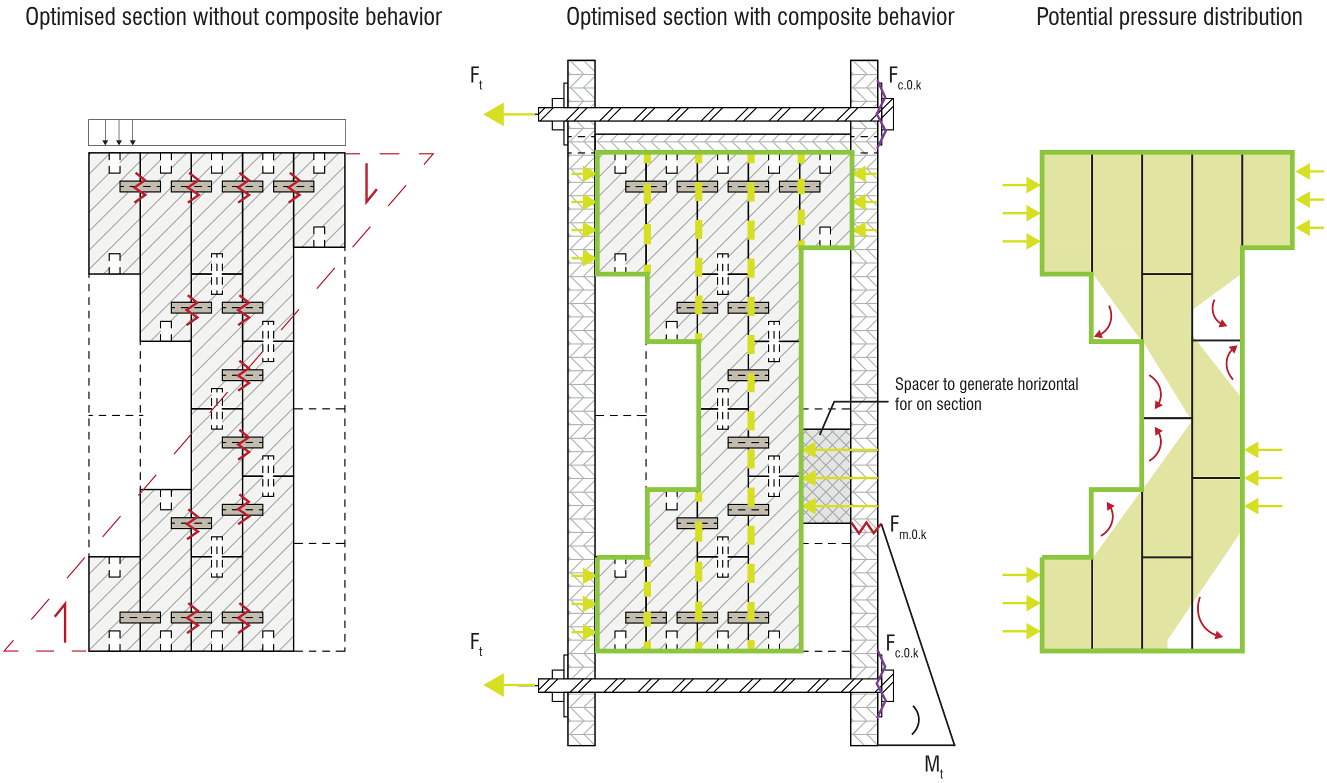


Figure 6.6: Structural principle of optimised section and theoretical joint. Left a potential geometry of an optimised section with only dowels an no composite behaviour. Right the section with external links to generate friction and composite behaviour.

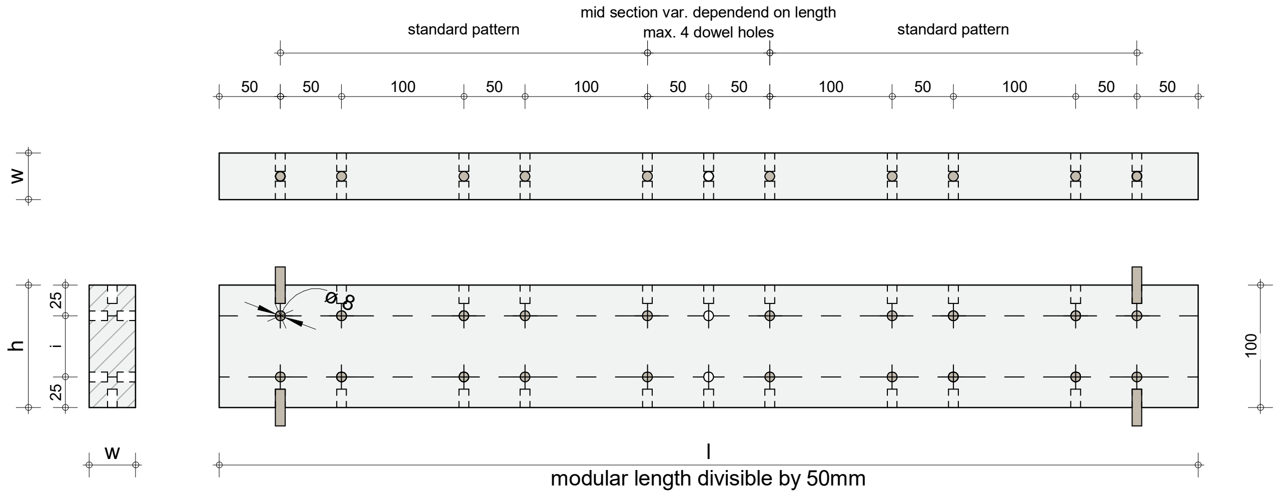


Figure 6.7: The modular part of the system

that the force does not have be very large to generate composite behaviour and that the bolts should not be overly tightened.

6.6 The modular part

This project aims to use reclaimed pieces of wood to aid the circular economy and ultimately reduce virgin material use and waste. The current approach includes an algorithmic matching process which allocates pieces within a design space. The selected pieces have to be processed and remanufactured to fit the requirements of the selected connection type. Every modification to a piece of wood reduces its future reuse potential and will ultimately not result in a reduction of construction waste but instead, in a delay of waste. To effectively reduce waste and close the life-cycle a system has to be developed that accommodates for future reuse. That means that modifications to the wood have to be limited to a minimum. Another aspect that can aid the circularity of the structural system is making the pieces within the system modular. This paragraph will

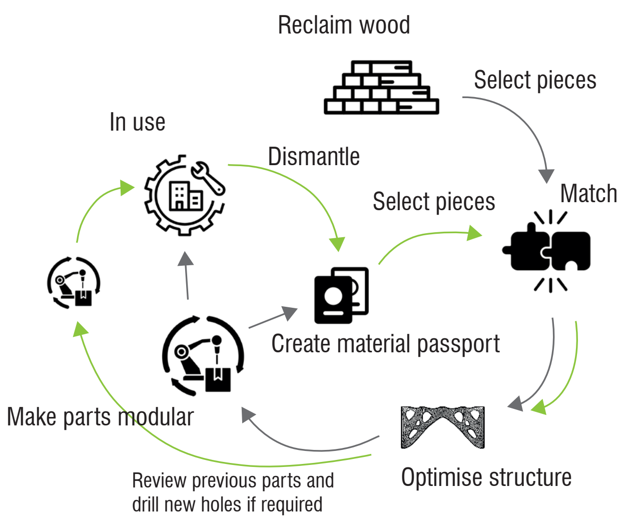


Figure 6.8: Circular loop by modular system

highlight the future reuse potential.

Circulair vision

Creating a modular sytem often ensures that future reuse for similar projects is possible without diminishing the value, i.e down-cycling, of the parts within the system and so creating a circular loop. This project also aims to use modularity to aid the circularity of the system. The length of the parts are set to fit within a system divisible by 50mm with the assumption that a low amount of reclaimed pieces will have sizes cut to a measure dividable by 10mm. Parts in the inventory which do not comply to this size or pieces selected by the algorithm causing excess length in a column or beam element are cut to the nearest modular size. This means that the dimensions of the portal frame are also constrained to this modular size. The width and height of the pieces are variable but based on commonly seen sections used in construction discussed in chapter 4. Selected parts for a project can be removed from the original waste wood database and put in a new database specifically intended for this system. This database serves as a material passport for each project, and contains information like: Status, Project name, Project location, Issue date, Original seller, Properties of the pieces, and in which element of the project the pieces are enclosed. The material passport makes future reuse more convenient and enables designers to close the life cycle of timber, figure 6.8.

Dowel spacing

Dowels spacing in the length direction of the part is essential to the modularity of the part. Figure 6.9 illustrates variable dowel arrangements. A pattern of 50mm is able to accommodate dowels in each type of position within the structure but has the downside of excessive drilling. A dowel spacing of 100mm would be more fitting but results in some cases that dowels not

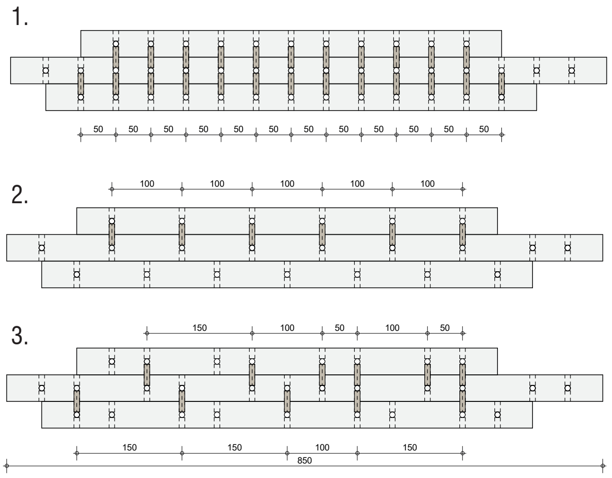


Figure 6.9: Top view of timber parts with possible dowel spacing arrangements. 1, within the modular size of 50mm, 2, using 100mm spacing, 3, using a pattern that can accommodate both sizes.

overlap. To reduce excessive drilling but accommodate a modular system a pattern is created which can, in any configuration, accommodate a dowel connection every 150mm and often a connection every 50mm or 100mm. The final pattern is shown in figure 6.7. Each piece starts with two holes every 50mm, follwed by a space of 100mm and then again two holes 50mm apart. The mid section is variable and has to employ holes in a range of 1 - 4 dowels every 50mm. This pattern ensures connectivity, modularity and reuseability of the parts within the system.

Dowel edge distance

The modular size of the pieces ensures that holes for the dowel connection can be standardised as much as possible, meaning that if a portal is dismantled, pieces can be reused in other projects more easily using this system. Dowel size has a large influence on the minimum cross-sectional height due to the required edge spacing. Before the matching process can start a dowel size has to be chosen. The selected connection type is theoretical and encompasses a still unknown structural performance. Mechanical testing is required

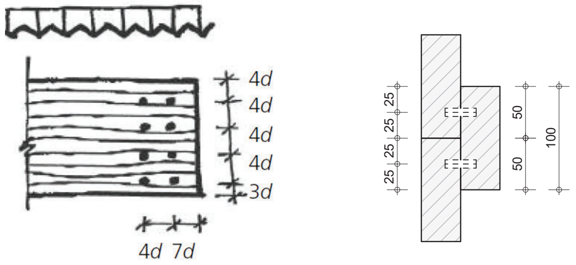


Figure 6.10: Minimum edge spacing dowels > 6mm and minimum cross-sectional height

to test the performance and potential gain from the frictional force on the stress in the dowels before a dowels size can be selected with certainty. For this project a dowel size of 8mm is selected but can be altered if future mechanical testing shows that the capacity is insufficient. Complying to minimum dowel spacing as stated in Eurocode 5 and illustrated in figure 6.10. 8mm dowels result in a minimum distance to the edge of 32mm and exemption of 24mm on the bottom. Because the dowels are not fully loaded due to the added friction by the external links, the lower bound of 24mm is taken as minimum dowel distance. This results in a overlap between each piece of at least 50mm to fit two dowels vertically and sets the minimum cross-sectional height of the pieces at 100mm. Smaller sections can still be used at places where only one dowel is required. Like the start or end of a row in the aggregate as illustrated in figure 6.11.

The drilling holes are initialised in perspective of each odd layer in the aggregate and then projected to the even layers. This can result in placement of dowels outside of the modular system size as is also shown in figure 6.10. This is a limitation of the current system and of using a dowel type of connection. However, this does not have to reduce the future reuse potential. New holes can be drilled at positions marked with a star in figure 6.10 or the matching program should include the original configuration as a constrained in a future project. Dowels that are allready in a used

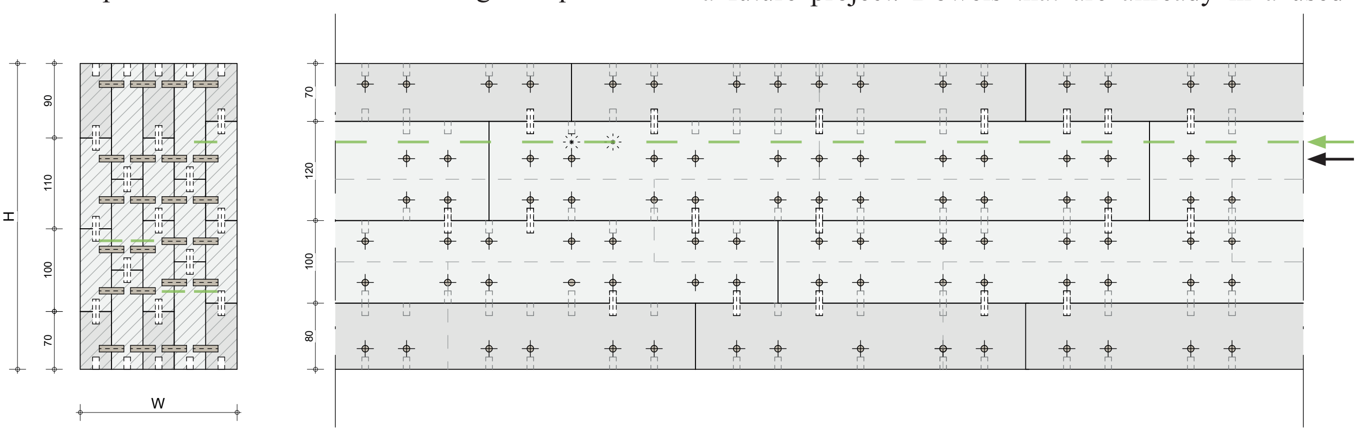


Figure 6.11: Principle of connecting parts within the system, minimum overlap of 50mm needed for dowel connection, in light gray pieces equal or larger than 100mm and in dark gray pieces smaller than 100 mm which can only be placed on at the end or start of a row within the aggregate

pieces but not rquired in a new comosition can also be mechanically be sheared off, leaving on half in side the wood resulting in a whole piece again.

6.7 Global connections

This section will discuss and illustrate a set of details highlighting the global connections within the system. Figure 6.12 visualises half a portal frame and shows the location of the global connections. Detail one shows the connection between two beams if the portal has a kink. If the portal is linear, detail 1 is non existent. Detail 2 shows the moment connection between the column and the beam and detail 3 the connection of the column to

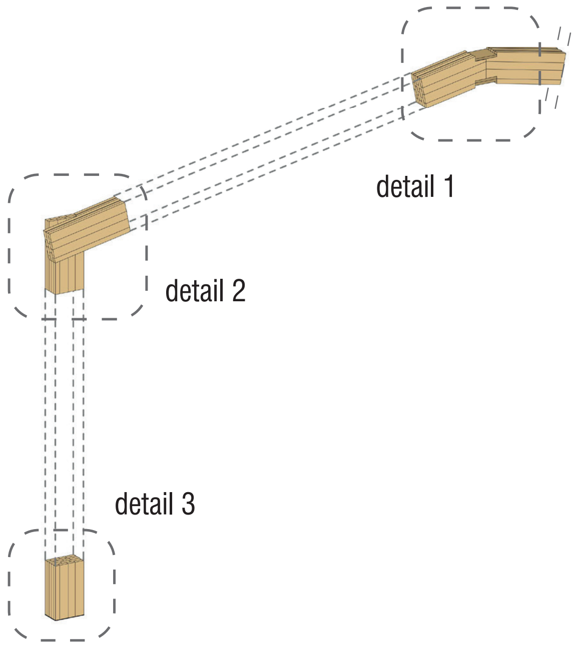


Figure 6.12: global connection overview in portal frame

the foundation.

Detail 1: Crown connection

For the connection between the beams at the 'crown' of the structure, two plywood panels are used to make a stiff joint. This type of connection was found to be most suitable because it is not affected by the aggregation of the beams. If stock is limited or extremely diverse the two beam elements could have different layer compositions, making finger joint connections, which would be more graceful, unsuitable. The connection is illustrated in figure 6.13. This type of connection is often seen in construction with connecting CLT floor panels. Plywood is a engineered board product, which therefore can be very stiff and strong. A quick check to determine the thickness of the plywood board can be done by converting the acting moment to a normal force by:

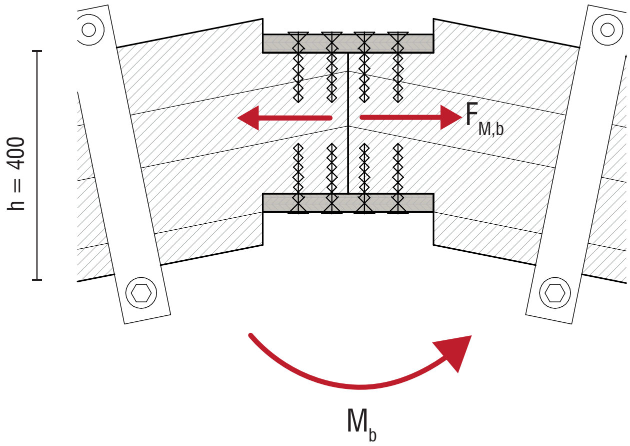


Figure 6.13: Crown connection

$$F_b = \frac{M_b}{h} \qquad eq.6.1$$

In which h is the height of the beam. The minimum tensile strength of plywood is +/- 27mpa. A beam of 180mm thickness would be able to resist a maximum acting moment of 42kNm with two plywood boards of 22mm. This is without considering safety factors but still substantial enough to consider the connection a viable solution. The downside of this connection type is that the future reuse potential of the wooden pieces around the connection is diminished.

Detail 2: Moment connection

The connection between the column and beam is essential for the portal frame to work because in this point moments have to transferred. For the same reason as the crown connection, finger joints between the columns and beam are not feasible because of potential varying layer thickness. Therefore the elements are tapered at the end. Two tapered ending can be connected to form a whole by steel bolts. This method is not affected by layer thickness and applicable for all configurations. This moment connection does diminishes the future reuse potential of the parts in that area but is required to take moments.

The connection is illustrated in figure 6.14. To create a structural model it is essential to determine the rotational stiffness of the connection as this highly influences the global stiffness of the portal frame. The rotational stiffness of the connection can be calculated with the following expression and is depended on both the stiffness of the beam and the connection:

$$\frac{1}{K_{r,A}} = \frac{1}{K_{r,u,connection}} + \frac{1}{K_{r,beam}} \qquad eq.6.2$$

The rotational stiffness of the beam can be determined with the length and stiffness of the beam by the relation between the angular rotation of the beam and the moment in the connection. This relation can be determined with a basic rule from mechanics:

$$M_a = K_{r,beam} * \varphi_a \qquad eq.6.3$$

Considering the beam as a cantilever with a uniform load the equation can be rewritten as:

$$K_{r,beam} = \frac{6EI_{beam}}{l} \qquad eq.6.4$$

The rational stiffness can be approximated when considering that the deformation of the connection is dictated by the bedding stiffness of the wood. The bedding stiffness provides the relation between the force and the displacement of an infinite stiff, circular pen, transferring the force to a hole in the wood. This pen is displaced on distance U. See figure 6.14, this relation is assumed to be linear and can be expressed as:

$$F_m = K_s * U \qquad eq.6.5$$

Here K_s reflects the bedding stiffness of the wood which can be determined with the mean density of the wood (p_{mean}) and the nominal thickness of the bolts (d_{nom})by:

$$K_s = \frac{\rho_{mean}^{1.5} d_{nom}}{23} \qquad eq.6.6$$

With this relation the rotational stiffness of the moment transferring connection can be determined based on equating the following basic formulas. The moment causes a force F_m on the n amount of bolts.

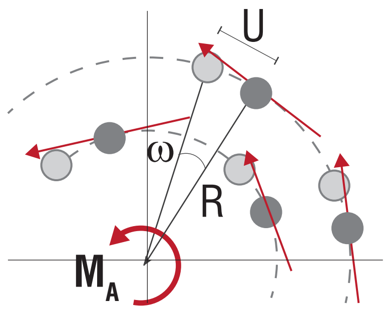
$$M = n * F_m * R \qquad eq.6.7$$

Due to force F_m every bolt will be subjected to a displacement in the tangential direction on the imaginary circle on distance U. The translation U, can be expressed as:

$$U = \omega * R \qquad eq.6.8$$

Equating equations 6.3, 6.5, 6.7 and 6.8 results in:

$$M = K_r * \omega = n * K_s * \omega * R^2$$
$$K_r = K_s * n * R^2 \qquad eq.6.9$$



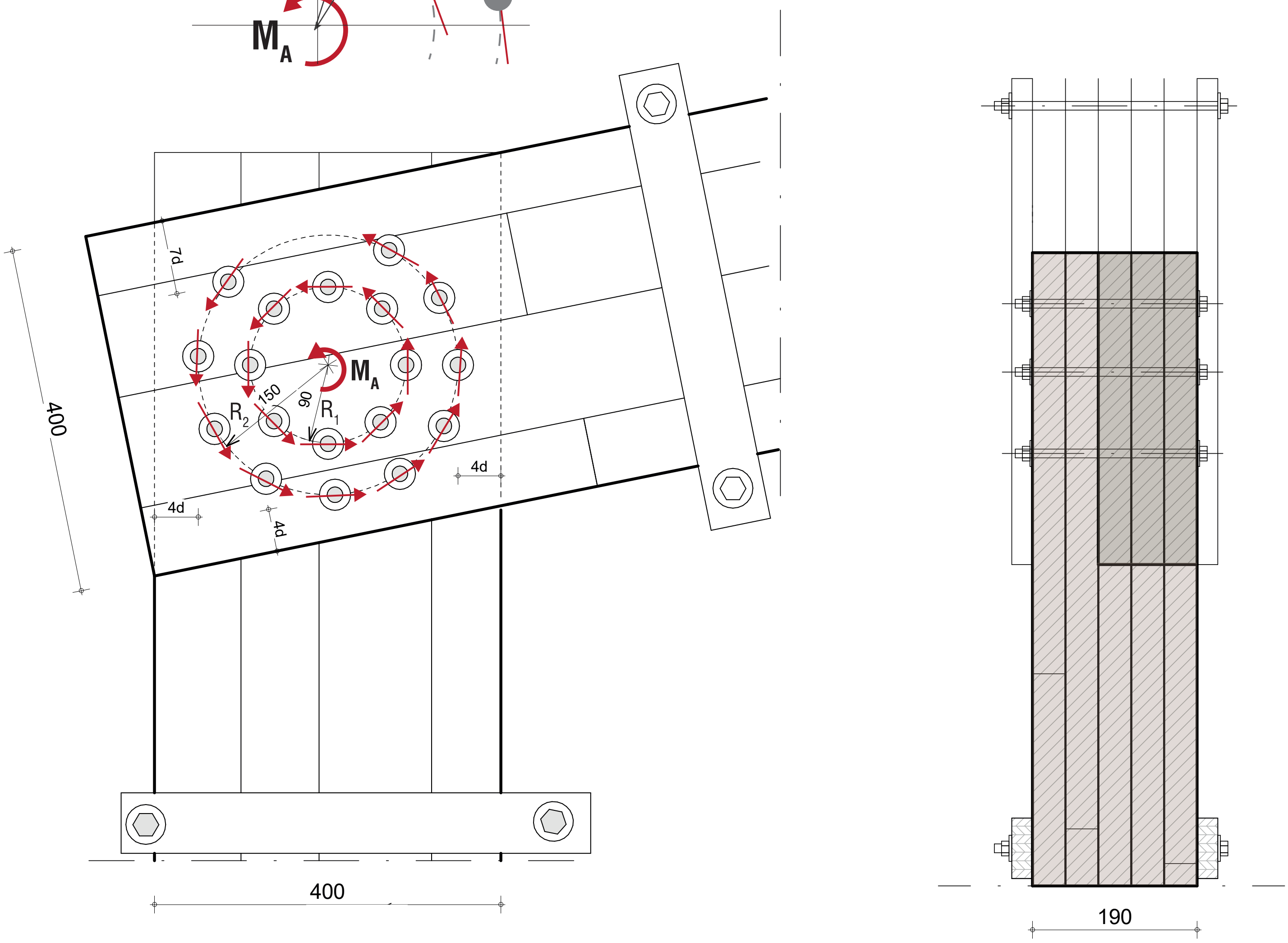


Figure 6.14: Moment connection of portal, highlighted with the structural theory of the derivation of rotational stiffness for a circular pin pattern

Because of minimum edge distances, not all both are placed on the same radius. That means that the polar moment of inertia is in case of this particular connection:

$$I_{polar} = n_1 * R_1^2 + n_2 * R_2^2 \qquad eq.6.10$$

For calculation of force distribution in ultimate limit state, a joint stiffness of 2/3 of the elastic stiffness should be taken. This reduced stiffness can be expressed as:

$$K_u = \frac{2}{3} K_{ser} \qquad eq.6.11$$

The rotational stiffness of the whole connection is thus:

$$K_{r,esr} = K_s * \ n_1 * R_1^2 + n_2 * R_2^2 \qquad eq.6.12$$

and:

$$K_{r,u,connection=\frac{2}{3}K_{r,ser}} \qquad eq.6.13$$

Noticeable is that the final rotational stiffness is highly depended on the rotational stiffness of the connection itself. Applying this theory and assuming that the wood used in the connection is at least of C18 quality gives for a the connection shown in figure 6.14 a rotational stiffness $K_{r,e}$ of 1.083kNm/rad. This value is considered to be quite low representing a moment connection and would cause, in this particular case, a deflection of 100mm at mid-span considering a total span of 7 meters. To put in perspective, moment connections of 7.000kNm/rad for timber joist can be found in Munck et al., (2011). By testing the structural model it was found that a rotational stiffness of at least over 4.000kN/rad is required to create a stiff connection with acceptable deflections. This would allow for a deformation of approximately 24.4mm. Lower rotational stiffness that 4.000kNm/rad will cause deflections to become normative rather than the utilisation of the wood. For this reason timber portal frames with moment connection are often tapered, and fitted with cross-section height of +800mm at the connection to create a larger arm that is able to take moments more efficiently (Munck et al., 2011), figure 6.15. Traditionally timber joist are connected by either bolts or finger joints, both requiring a large cross-sectional area. The large minimum edge and end distances are constraining the amount of bolts that can be used in smaller sections. Either a higher cross-section can be implement or the connection can be made more stiff to resist moments using different methods.

Moment connection alternatives

A few alternative connection types can be selected to improve the rotational stiffness of the connection and are depicted in figure 6.16 and 6.17.

As external links are already used to enhance stiffness of the discrete elements they can be extended to enhance the stiffness of the connection. Figure 6.16 shows that by extending the outer link a more rigid connection can be made. This connection enacts a triangular force transference, resulting in a highly efficient and stiff connection. This connection type already fits within the existing system however, it lowers the future reuse potential of more pieces as the plywood sheet has to be fully bolted through the timber

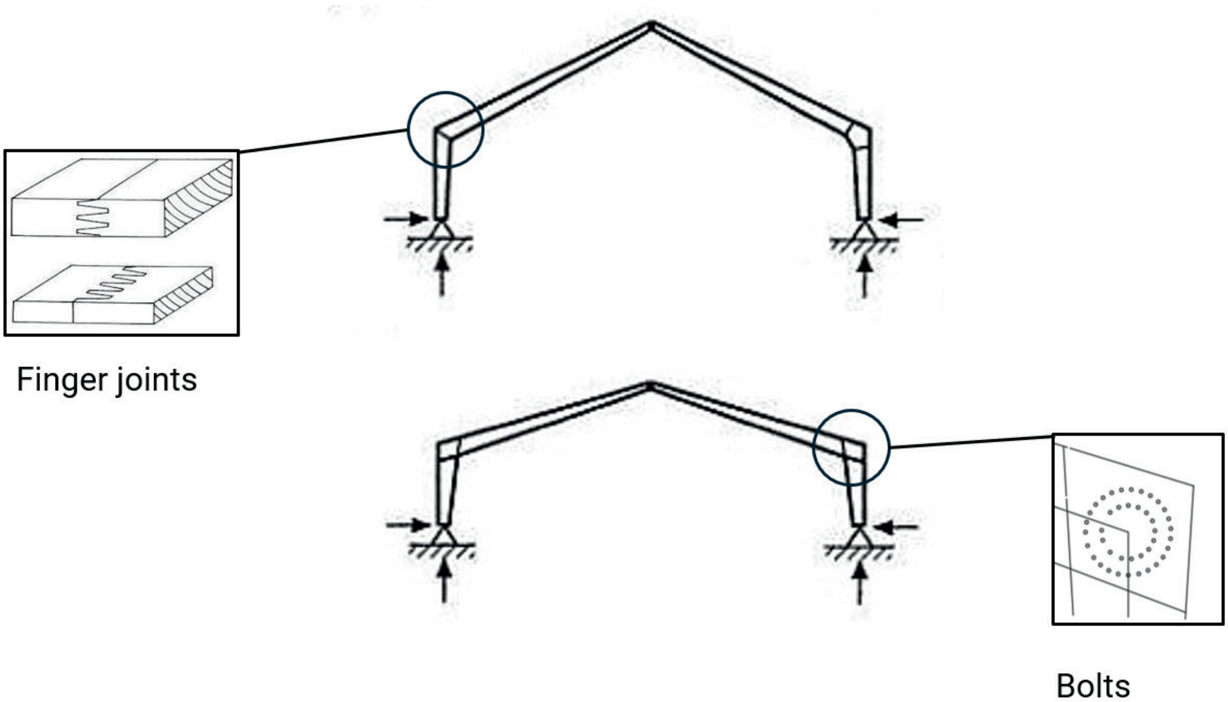


Figure 6.15: Traditional timber joist moment connections and geometry, adapted from Munck et al., (2011)

joists and it is highly influential on the architecture.

The second alternative is to enlarge the cross-section in order to fit the required amount of bolts. As the topology of the portal frame is being optimised there is no need to create a tapered geometry as the ineffective parts will be removed. However, this would still lead to a higher material consumption, which is discussed in more detail in chapter 8.9, and is therefore not the most favourable alternative.

The last connection alternative is to introduce the use of steel equipment to create a strong and stiff connection. Figure 6.17 shows some of the most common steel-timber moment connections. All the connection types are either using large screws or steel threaded bars. The main principal is to tighten these bars, compressing the timber joists and putting the steel in tension, effectively loading each material to is most favourable force. Effective methods for a massive piece of solid wood however when using a discretized cross-section composed of smaller pieces the bars and screws become less effective due to the seams in the section. Moreover, this method adds to the steel consumption and lowers the future reuse potential of the pieces but ensures a smaller cross-section. In Scheibmair (2012) a new type of moment connection can be found which is based on the same principal as connection A and D in figure 6.18 but without inserting the steel bars in the wooden joist itself. The steel is put on the outside of the timber, accommodating for a discrete system and ensuring a high future reuse potential of the pieces as the pieces are minimally affected. This new connection type seems to be the most viable solution as it does not enlarge material consumption, affects the architecture minimally, uses a minimal amount of steel and keeps the future reuse potential of the pieces high. The rotational stiffens for this connection type is not calculated, but in Scheibmair (2012) a rotational stiffness of + 50.000kNm/rad was achieved, with sizes and loads three times as high as in this research project, validating the potential of this system. The connection form Scheibmair requires embedded shear dowels to resist longitudinal shearing and a plumb cut on from the horizontal member of the portal frame as can be seen in figure 6.18. The plumb cut ensure that the horizontal member is not displaced when tightening the connection. This reduces the future reuse potential of the pieces a lot and is therefore unfavourable but

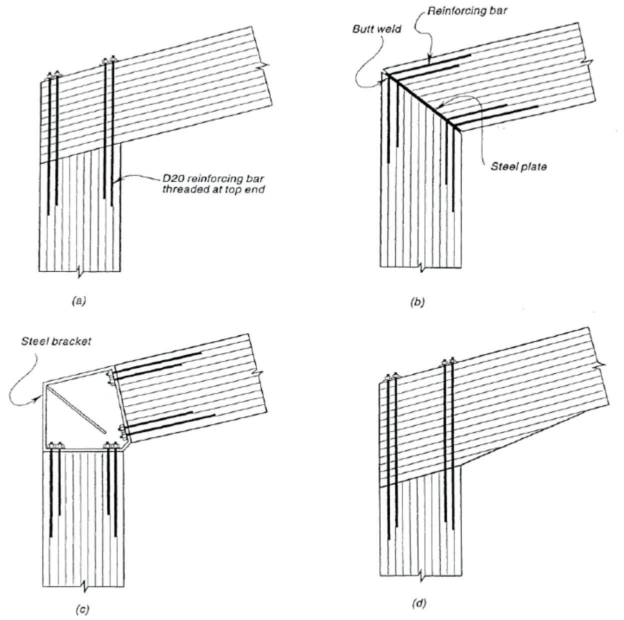


Figure 6.17: Steel-timber moment connection alternatives (Buchanan, 1993)

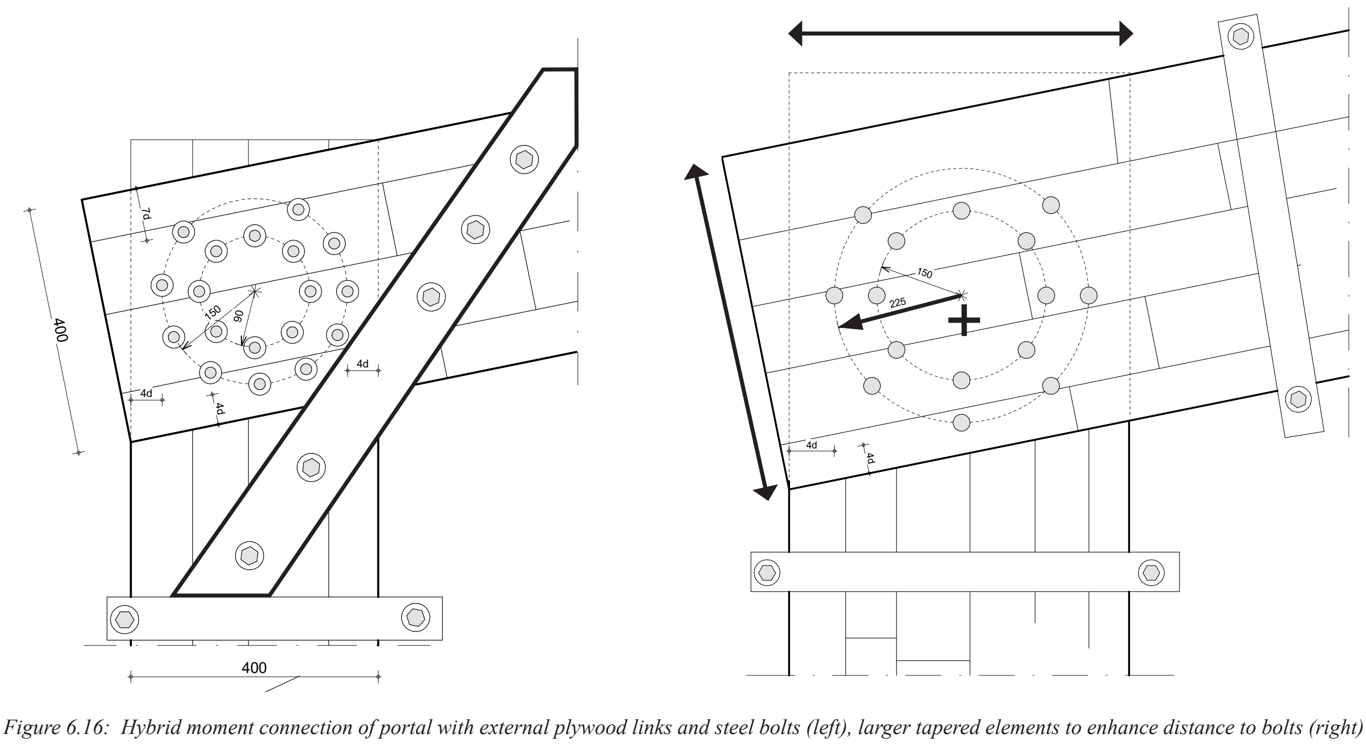


Figure 6.16: Hybrid moment connection of portal with external plywood links and steel bolts (left), larger tapered elements to enhance distance to bolts (right)

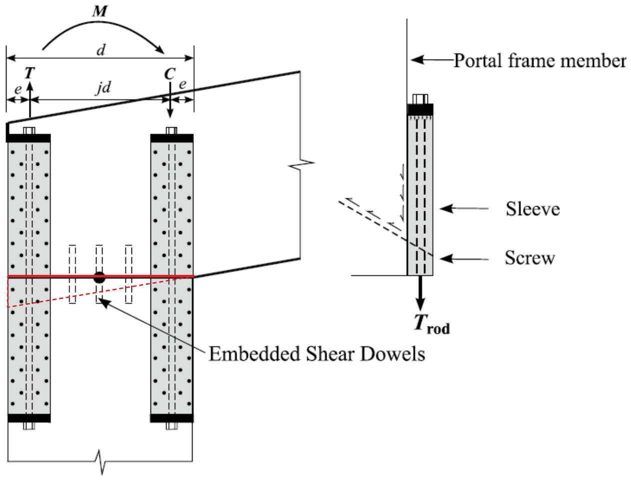


Figure 6.18: External steel-timber moment connection, from (Scheibmair, 2012). In red the lost wood due to cutting.

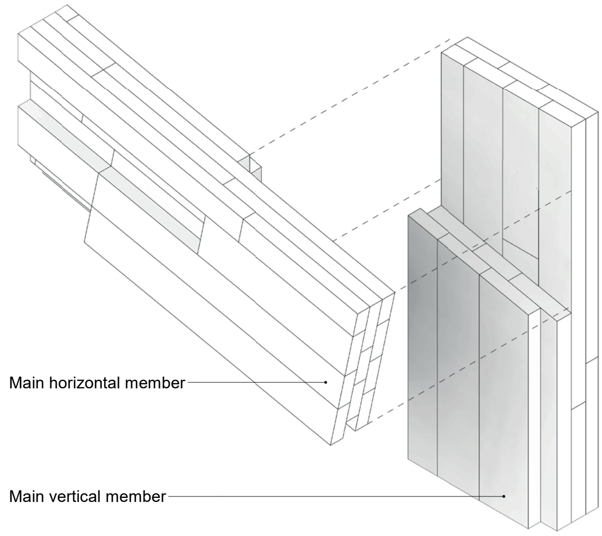


Figure 6.19: Tapered members for longitudinal shear resistance

can be avoided by tapering the ends of the members, ensuring that the member cannot be sheared as depicted in figure 6.19. This requires the connection to be placed in two directions on parallel to the vertical member and the other parallel to the horizontal member, both depicted in figure 6.20. Using the connection in two directions could potentially counteract its efficiency if one of the two direction is tightened first locking it in one position and restricting the other connection to be tightened correctly. It is therefore essential to tighten each side simultaneously, reducing internal stresses and displacements. The connection relies mostly on the tensile strength of the steel rods. By tightening the rods the sleeve in which the rods is placed is compressed with externally high forces. To prevent crushing of the wood this sleeve has to be fitted with steel bearing plates a both the top and bottom. The sleeves are fitted to the portal frame members by a large amount of self -tapping screws, inserted diagonally to prevent the screws form exactly running through the segment seam of the pieces and with the added benefit of performing diagonally more optimal in shear. The wood type should be a harder type of wood to ensure it cannot be crushed. By tightening the rods a compressive force is put on the faces on the timber members which, together with the steel that is in tension, forms the connection that able to resist large bending moments. The connection is worked out in detail in figures 6.21 and 6.22.

For the structural model a rotational stiffness of 7.000kNm/rad is assumed and used to analyse the structural performance of the structural system.

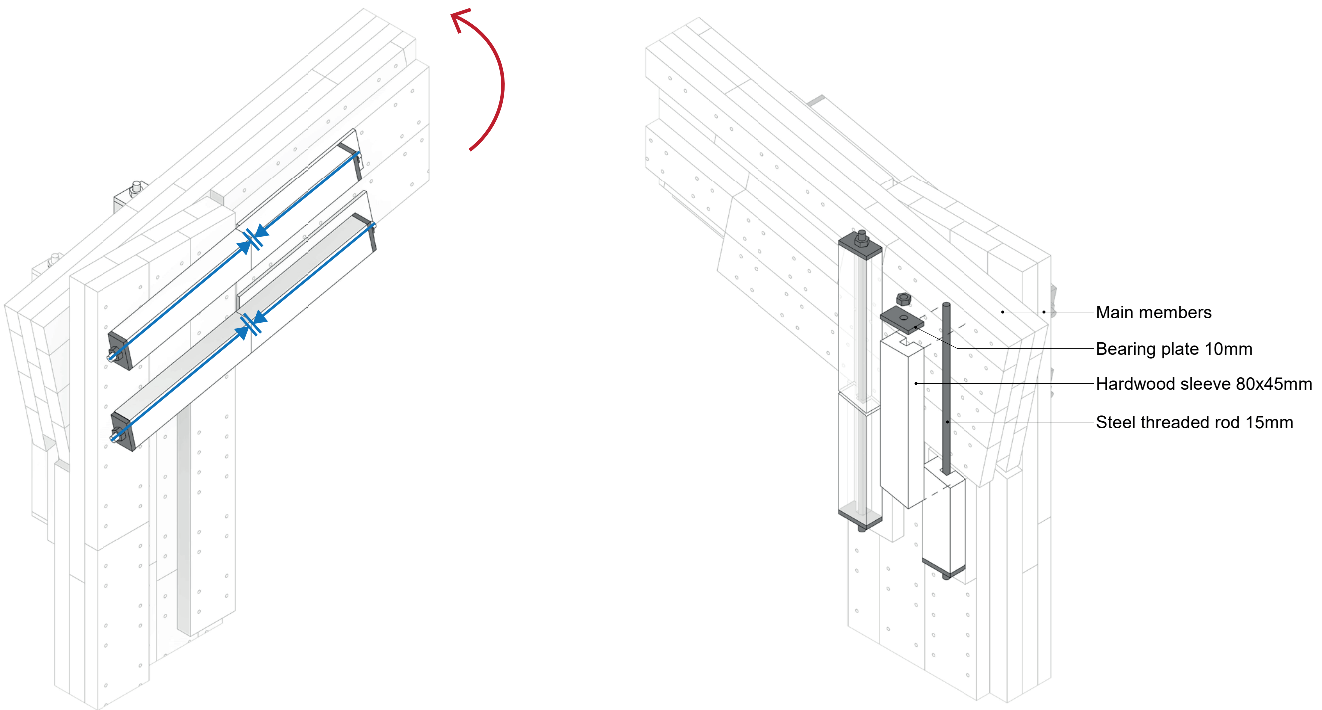


Figure 6.20: External steel-timber moment connection with the discrete timber design and topological optimised members

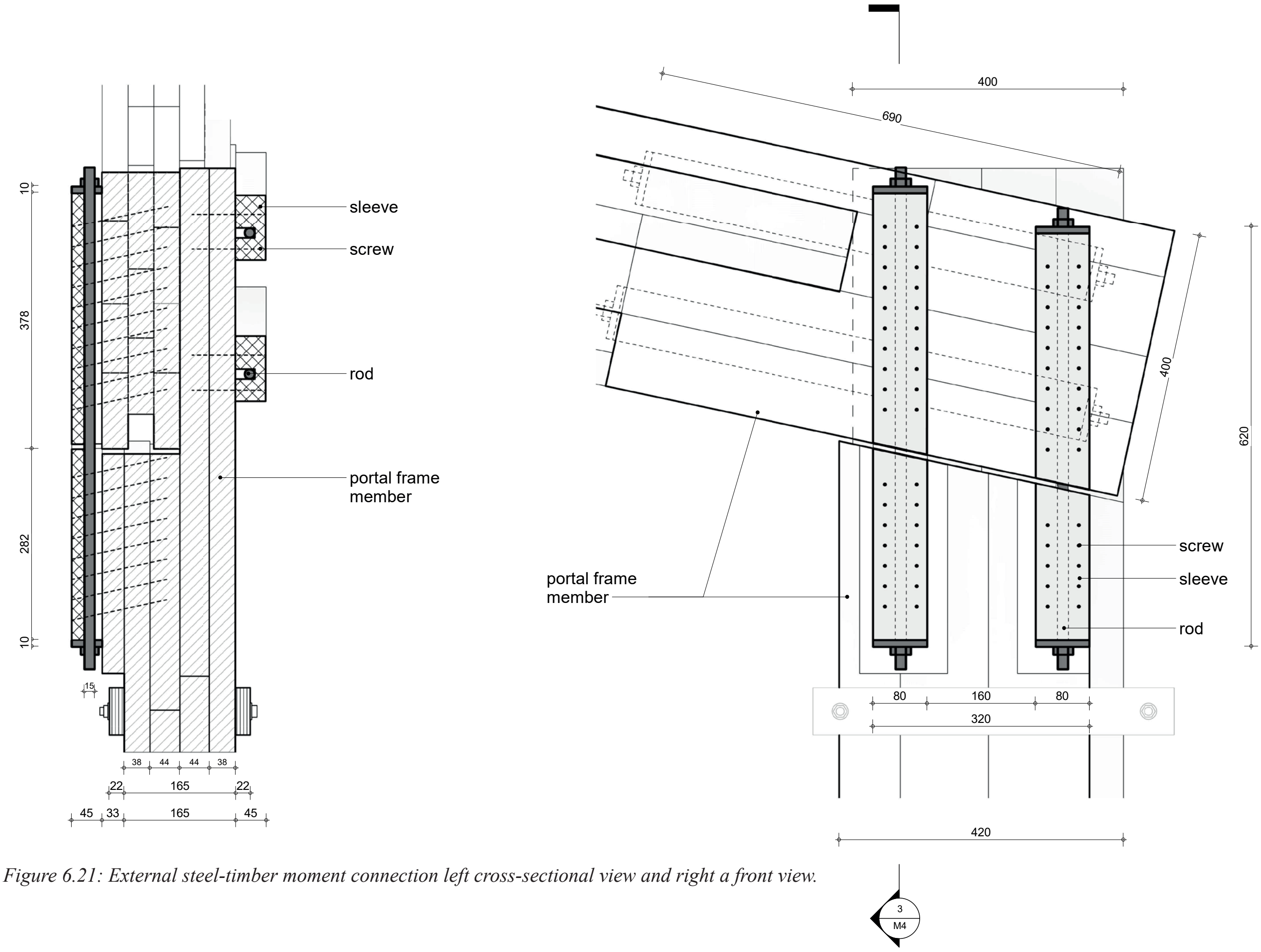


Figure 6.21: External steel-timber moment connection left cross-sectional view and right a front view.

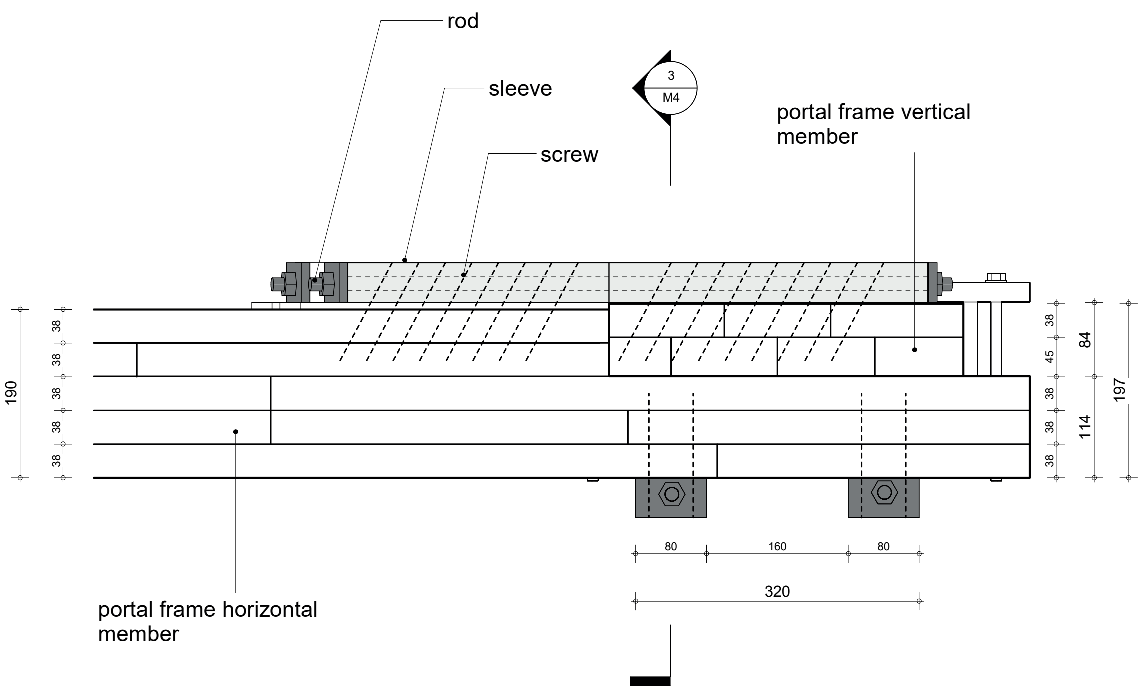


Figure 6.22: External steel-timber moment connection top view

Figure 6.23:mock-up of moment connection

Figure 6.24:mock-up of moment connection

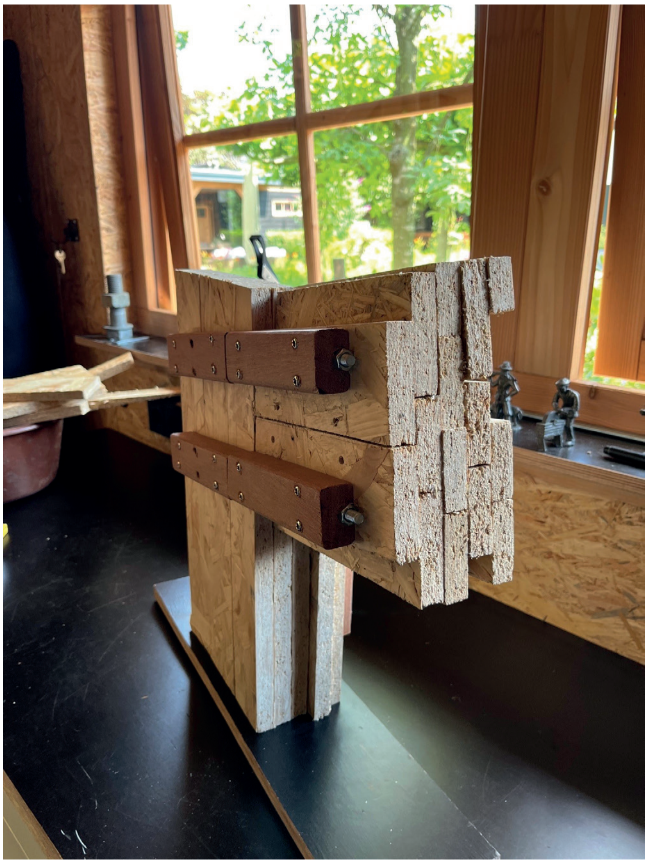


Figure 6.24:mock-up of moment connection

Figure 6.25:mock-up of moment connection

Detail 3: Support connections

Figure 6.23 and 6.24 show two possible support connections. These connection can be either pinned or rigid. The middle layer starts 200mm after the other layer to accommodate for a steel tube. This can be used to in the frame. With a rigid support a steel tubular shoe can be put around the column which can take the moments.

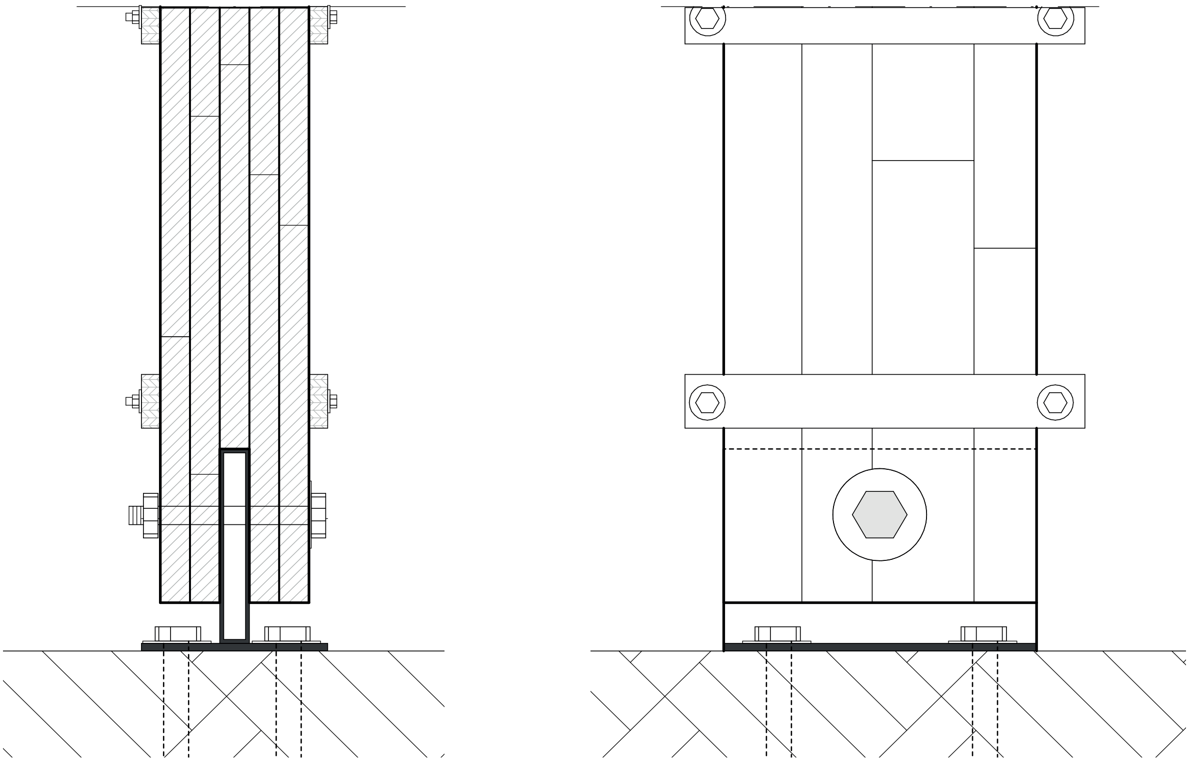


Figure 6.26: Pinned support

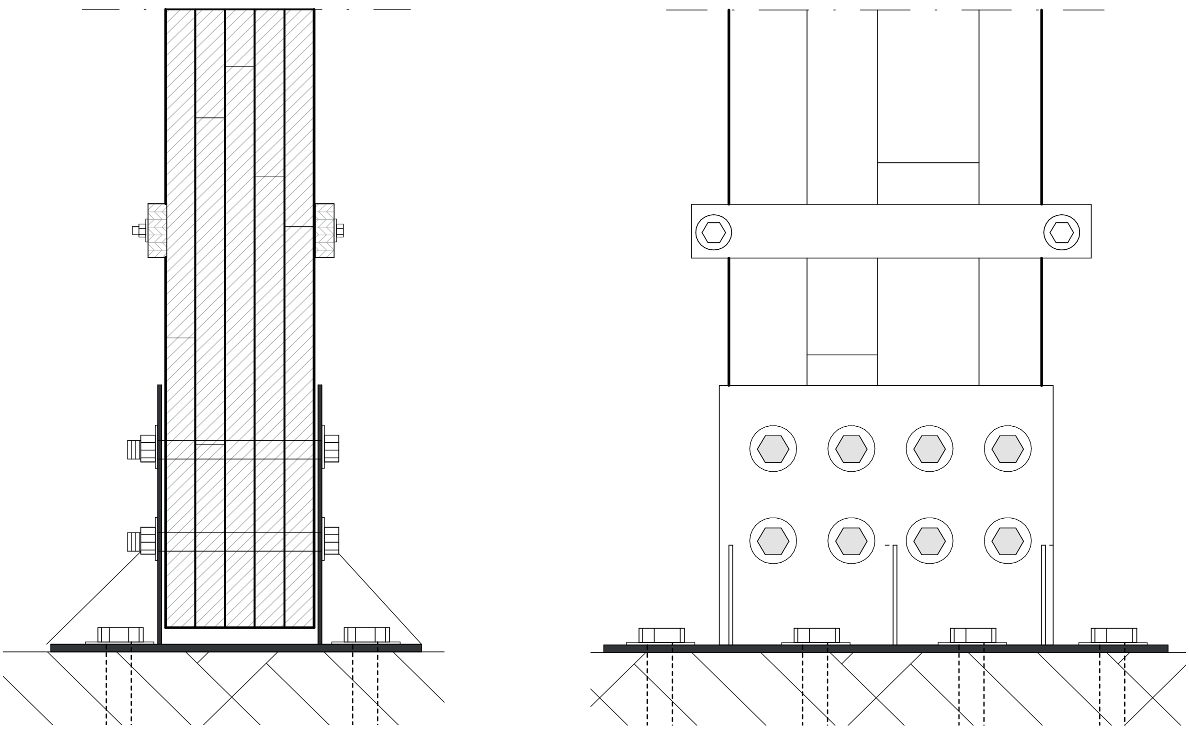


Figure 6.27: Rigid support

7. Algorithmic Development

This section will discuss the development of the matching algorithm that is able to generate portal frames from salvaged wood. Dynamic programming is used to solve the combinatorial matching problem and provides a global optimal solution. The algorithm is written in IronPython, the built-in python coding function in Grashopper. Karamba 3D is used for the structural analysis and optimisation. The algorithm developed on a HP laptop containing an Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz processor and 16GB of installed RAM. Limiting the solution space and inventory per portal allowed the algorithm to be run the optimization problem within an acceptable time span on this laptop. The matching problem takes about 20 sec to run. The topology optimization another 6 min. per portal frame.

7.1 Three-Dimensional packing problem

The initial approach was to consider the matching problem as a 3D packing problem. Algorithms for this type of problem already exist and are often used in the supply chain and loading field to load pallets and trucks efficiently. In bin packing, an algorithm tries to fill a space as efficiently as possible with as few bins as possible. The bin packing algorithms found were complex and all used multiple different libraries. This limited the usability of the algorithms such that adaptation and rewriting was not an option. An attempt was made to write a 3D bin-packing algorithm but without success. As shown in figure 7.1 the algorithm did fill the design space but could poorly keep track of the available space. With each block placed, the design space was divided into smaller boxes, resulting in a pattern of small remaining spaces where nothing else would fit. Moreover, placement of pieces was hard to control and was done in such an uniform way that assembly would be impossible and resulted in high computation time. A 2D implementation of bin-packing was also tried but resulted in similar problems. The plug-in Wasp that accommodates discrete design from Rossi & Tessmann (2018) was also tested but provided little control over the final composition and resulted in similar issues as that were seen with the bin packing algorithm where available space is not correctly tracked.

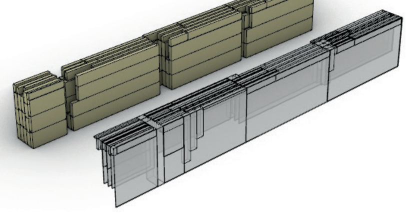


Figure 7.1: Result of 3D bin-packing algorithm. On the left the final composition. On the right the available space that the algorithm sees.

7.2 One-Dimensional matching problem

The unsuccessful implementation of the bin packing problem required a new approach. To get firm control on the geometry and keep the algorithm readable for designers, an one-dimensional approach was initialized. In this approach the three-dimensional matching problem is solved sequentially, one dimension at a time. One-dimensional problem solving reduced overall complexity and provided overall control on the final composition. For each dimension the problem is dissected in a combinatorial solution domain which is easy to solve, but if left unconstrained can result in large computational time. The combinatorial problems can be solved by any type of algorithm. This research compared two methods, the greedy search and the dynamic programming, both constrained to the same inventory and initialized with the same objective: minimize remaining space in the x(length)-direction. Figure 7.2 shows the result. The greedy search method allocates the largest parts first as this contributes most to reaching the required length as quickly as possible, a local optimum. This however, as expected, does not result in a global optimum and on average leads to a filling rate of 80 - 90%. More often than not, it requires cutting of pieces to get a uniform composition. The dynamic programming in contrast tries to find a global solution and uses a balanced mix between small and large pieces to reach the required length almost always perfectly without requiring cutting of the pieces. Moreover, on average the dynamic programming method was faster in producing solutions then the greedy search method and is not much more complex to implement. For that reason it was decided to continue developing the algorithm using the dynamic arrangement approach.

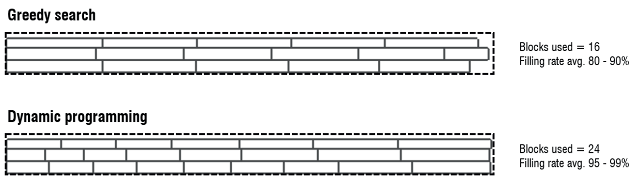


Figure 7.2: Result of piece placement only considering the X-direction without any constraints, Greedy search method compared with a Dynamic programming method.

7.3 Overview of the algorithm

In figure 7.3 a global overview of the developed algorithm can be found. This flowchart showcases the overall working of the constructed program. The programme is structured in five processes. The first three processes are run together in one loop. In the first process, initialisation and boundary conditions, the designer has to enter parameters and load and selected data of the salvaged wood components to start the programme. The second process is creating the composition. Here, the ground structure optimisation method is started. The different volumes representing the roof joists and columns are filled with pieces of wood from the database. After this, pre-processing begins. This involves creating virtual connections between the placed pieces and putting the elements in the right place. Lastly, a finite element model is set up in Karamba3D for stress analysis in the structures components and end the first loop. The designer can view and analyse the results, and if found satisfactory, the optimisation loop can be started. In this loop, pieces of wood that are not essential to the structure are removed, reducing mass and material use. After each piece is removed, the structural model is recalculated in Karamba3D and a new piece is selected for removal. When no more pieces can be removed, the loop stops and post-processing begins. In this phase, the designer can generate slices at certain points in the structure and view the stiffness and remaining profiles. These slices can be used for detailed calculations as buckling and tilting and exported to Excel. At the same time, the designer can retrieve 2D drawings that include the required boreholes for the dowels for each element and display the element IDs for each layer. If the design is satisfactory, the designer can remove the used pieces form the database and send them to a new database, if not, parameters can be readjustment and the process can be run again. It is also possible to iterate through multiple iterations of the optimization loop and select an option in which fewer parts are removed than necessary. In total the whole process takes about 5 - 10 minutes depending on the size of the portal and inventory.

Used plug-ins and programs

The complete program is made within the visual programming environment of Grasshopper. Several plug-ins have been used in the computational model:

- Python: the majority of the program is written in the built in python component of Grasshopper. This uses the IronPython coding language.
- GHPython: this is a interpreter for Python due to which it is possible to use external Python libraries in IronPython.
- Karamba 3D: is used to perform finite element analysis on the structure to determine internal stresses.
- Anemone: is a plug-in that enables Grasshopper to create loops. It is utilised to loop the output of the finite element analysis, i.e. the element with the lowest stress, back into the program to be removed form the compostion.

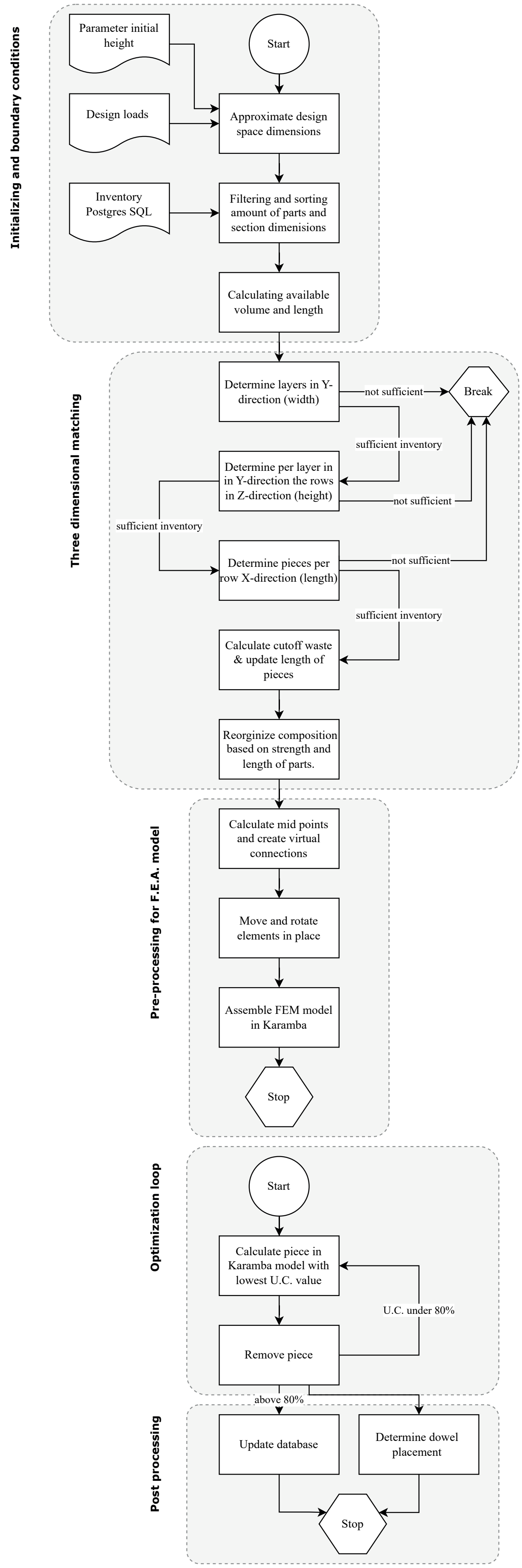


Figure 7.3: Global flowchart of developed algorithm

7.4 Initialization and Boundary conditions

To start the program, The designer is required to first deliver some input. The designer has control on the height of the left and right side of the portal, the span and mid-point position in Z- and X-direction, see figure 7.4. These five parameters provide the flexibility to construct almost every portal and can be extended to the needs and requirements as necessary. The two support conditions of the column can be set as required. The model assumes moment connections between the columns and roof joists.

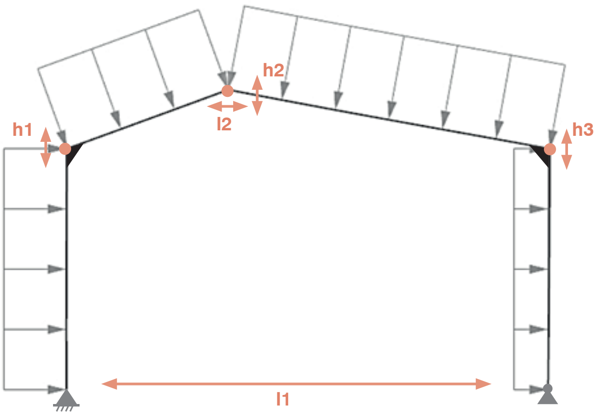


Figure7.4: Geometric parameters

The programs expects several types of loads including: dead load of the roof, the live load and the wind suction and pressure considered from both sides. The are three result-cases used in the program:

- RC1: Deadload & liveload
- RC2: Deadload & live load & wind pressure and suction positive
- RC3: Deadload & live load & wind pressure and suction negative

The stresses from these loads are calculated for each piece and the maximum value out one of these three result-cases is used to determine the piece with the lowest value. It is expected that the designer determines the key combination factor and use the associated loads.

Following, the designer has to input an initial height for the cross-sections. The program then calculates the expected required width of the cross-section based on the loads and the initial set height. The designer can adjust the parameters again until the initial dimensions are satisfactory.

The program provides an estimate number of parts required to create the portal frame. The designer has to import the preferred database from a selected wood-dealer and filter on the desired length, width and height that sections are allowed to take in the composition. The program gives a option to analyse the database and filter on, length, width, height, strength and visual state. The inventory should be reduced to the number of advised parts to reduce computational time. A set of 1.000 parts is still within reasonable computational time but less is advised. The stock is based on the database shown in chapter 3.

Lastly the derringer has to input the rotational stiffness of the moment connection which can be calculated through the method in chapter 6, as default the rotational stiffness is set on 7.000Knm/rad

Inventory processing

The pieces are sorted based on the width of the cross-sections available in the stock and put into separate dictionaries. Dictionaries are a data types in Python that allow for storing data and fast access of a filtered set. Data can be accessed with a key. The keys are in this case the variable sectional widths. Calling a key means calling on a sorted part of the dataset, i.e. calling on all parts contain with the same width, see figure 7.5. Through this data type the matching of inventory can be executed efficiently. The total volume for each dictionary is calculated. For each dictionary the total length of each unique cross-sectional height is calculated.

7.5 Three-dimensional matching

This part of the code aims to fill the design spaces, i.e. the initial volumes of the four elements of the portal,

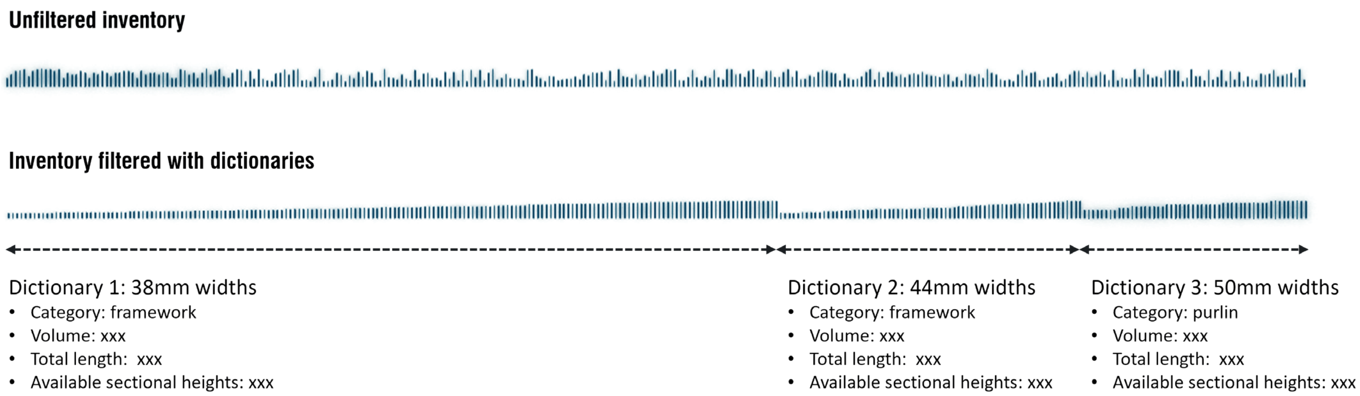


Figure 7.5: Processing inventory with dictionaries in Python

as efficiently as possible using pieces of wood from the database. The matching problem is dissected into three functions, one for each dimension, solving sub-problems sequentially of each dimension results in solving the global three-dimensional problem. First the volume is horizontally split in layers. Each layer is again split vertically in rows and each row is dynamically filled with wooden pieces available in the stock.

Local Y-directional combinatorial problem

The first sub-problem aims to find the most efficient combination of layers in the Y-direction of the volume, the width, of the volume. In other words, the number and width of each sub-layer. This is achieved based on the available volume of each dictionary. The volume is used to calculate the maximum amount of times a dictionary can be used to create a layer. The program is inherently greedy and tries to use the least amount of pieces to get to the desired width. This research project focusses on two waste streams, frame work, sectional widths < 44mm and purlins, sectional widths > 44mm. Without constraints this will result in compositions mainly constructed of purlin sections. These sections are large, with a minimum height of 150mm and therefore resulting in a coarse level of detail (LOD). An unfavourable effect for the topological optimization as there is not a lot of pieces can be removed without damaging the structural integrity, especially for a smaller element. To ensure that cross-sections are not solely built of large waste wood pieces originally used as purlins, the program uses standardised cross-sections which sets a maximum on the amount of purlin sections that can be used. Standardised cross-sections are illustrated in figure 7.6.

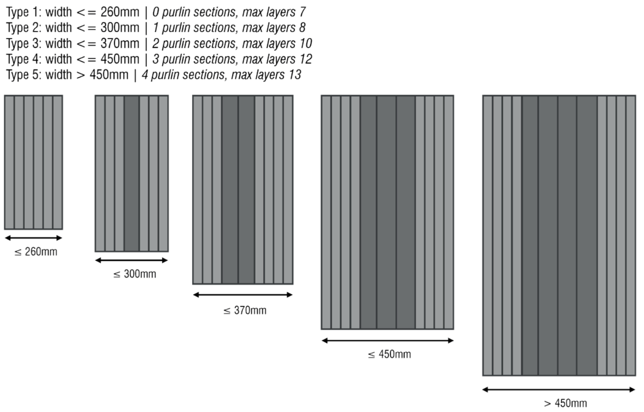


Figure 7.6 Cross-section types, dark gray purlin sections and in light gray the framework sections.

Next, all possible combinations are generated, constrained to the minimum and maximum amount of expected layers befitting to the cross-section type. If not constrained, this will result in an exponential increase of computational time as the combination length increases. The combinations that minimizes remaining space are selected and expressed as:

$$\textbf{Find}: \textit{Optimal width distibution}$$

$$\textbf{Objective}: \ \textit{Minimize} \ \left| \sum_{i=1}^{m} width_i - original\ width \right|$$

$$\textbf{Subject to}: \textit{Maximum width}: \sum_{i=1}^{m} width_i \leq original\ width$$

$$\textit{Inventory}: used\ section \leq \max usage$$

$$used\ prulin\ sections \leq \max used\ purlin\ sections$$

$$max\ usage_i = \min\left(\left\lfloor \frac{0.95 * volume_i}{width_i * length * height} \right\rfloor, \max used\ purlin\ sections \right)$$

The program is developed that the biggest section is always put in the middle to accommodate for a detailed topological optimization. Lastly, the y-coordinates of the layers are calculated.

Local Z-directional combinatorial problem

The second sub-problem aims to find the optimal combination of rows in the Z-direction, the height of the volume, for each layer found in the generated width distribution of the volume. With the available length per unique cross-sectional height per dictionary this combinatorial problem is solved with a similar approach seen earlier in the Y-direction. The algorithm tries to find combinations of heights to fill the layers. Also here the program is inherently greedy. To limit the use of large sections and make the compositions as diverse as possible a prioritization is given to longer combinations and to combinations with the most unique values. Crucial in this combinatorial problem is again to limit the solution space. The maximum and minimum combination lengths are calculated based on the available inventory and set as boundaries for the solution space to reduce computational time. The objective of the program is to find a combination that minimizes remaining space while prioritizing the longest and most diverse combination which can be expressed as:

$$\textbf{Find}: \textit{height distribution for each layer in every volume}$$

$$\textbf{Objective}: \textit{Minimize} \sum_{i=1}^{N_b} \sum_{j=1}^{N_l} \left(R_{i,j} + Z \right) - \mu * D$$

$$\textbf{Subject to}: \textit{Inventory}: \sum_{k=1}^{N_s} \left(inventory(h) - volume\ length(i) \right) \leq inventory_{i,j}(h)$$

$$\textit{Max height}: \sum_{k=1}^{N_s} height(k) \leq volume\ height(i)$$

$$\textit{Clashes}: \left| z_{i,j} - z_{i,j-1} \right| > 0.04 \quad \forall i,j$$

$$\textit{Variable startpoint}: Z = \sum_{i=1}^{N_b} \sum_{j=1}^{N_l} \delta_{i,j} * z'_{i,j}$$

μ is a weight representing the importance of maximizing diversity in combinations
δi,j is a binary variable indicating whether incrementation is made for layer j in volume i,
z'i,j is the incrementation for layer j in volume i.

The program works sequentially through each volume and each layer. That means that the last volume can result in leftover sections and be structurally less efficient. Prioritizing the longest combination ensures for the first couple of volumes, pending the contents of the stock, a cross-section which is diverse enough to accommodate a detailed topological optimization but can results in the last couple of volumes to only consist out of large left over sections. In this research project is effect is accepted because the roof joists of the portal are aggregated first which are the elements in which topological optimization can provide the best result for reducing mass. A method to avoid this behaviour is to dynamically reduced the stock per volume to ensure that each section has the same amount of large and small pieces.

The function is constrained by four variables: the available inventory, the height of the volume, the minimum required overlap of 50mm to accommodate dowels, and a variable start point when the stock is not diverse enough. Most important during this process is that rows in consecutive layers do not overlap and have a offset off at least 50mm. This makes the composition structurally sound and is needed for assembly. Sections smaller than 100mm provide too little room for dowel connections and therefore cannot be used inside the composition. To maximize the use of all types of cross-sections, the algorithm is allowed to place these section as first or last item in the composition because on these places only one dowel is required. The program checks for each part in the combination if it satisfies the required overlap by cross-checking it with all parts in the previous layer with exemption for the last part to accommodate for a flat loading area. If no combination can be found due to clashes between rows, the startpoint of the consecutive layer is increased with z1. By default this is 30mm and is then increased by 20mm until the maximum incrementation of 100mm is reached. If a section smaller then 100mm is placed in the previous layer than the z1 is 20mm. With each incrementation the total height of the volume (Z) is reduced. If an incrementation is required in a layer the minimum combination length is reduced once per layer by 1. Figure 7.7 highlights these constraints.

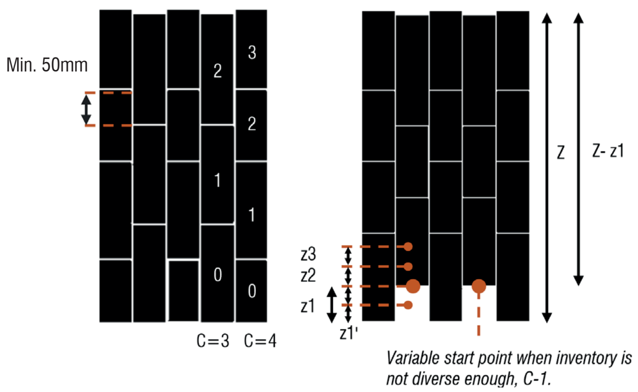


Figure 7.7: Z-directional constraints.

Global X-directional matching problem

Constraining the two dimensions (Y, Z) allows for the main problem to be solved. Dynamic programming (DP) is used to select and optimize the arrangement of wooden pieces. To account for the natural variety of strength grades it is important to place lesser quality wood on places with lower stress and higher quality wood on places with higher stress. Additionally, longer pieces are more suitable to place at locations where the least optimization is expected and smaller pieces on the remaining places so the optimization can be executed more accurately. Based on the normative principal stresses distribution the volume can be divided into two type of regions. High strength regions (HSR) and Low strength regions (LSR). see figure 7.8. For rows that are located inside the HSR a priority is set on pieces with higher strengths and larger lengths. For rows in LSR a priority is set on lower strength grade and shorter pieces in the middle and higher and longer strength grade pieces at both ends.

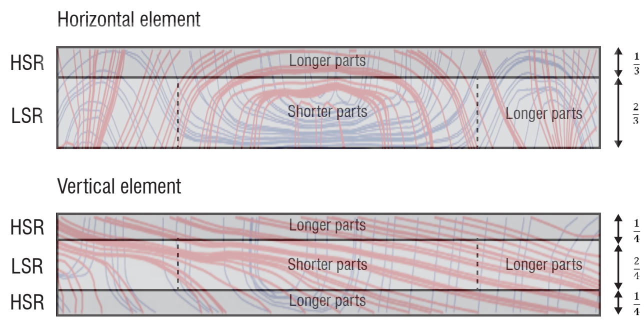


Figure 7.8: Strength regions for strength grade optimization.

The optimization process concerning dynamic programming consists of three steps: creating a table and filling it iteratively, find the appropriate target, and backtrack through the table to find the solution to the original problem. A DP table is a two-dimensional matrix where each cell corresponds to a sub problem. In this case the sub problems are defined as the attainable length with various combinations of pieces. The global problem to be solved is which combination of pieces most effectively can reach the desired length while adholding to the strength region constraints. This objective can be expressed as:

$$\textbf{Find}: Most\ efficient\ combination\ of\ pieces$$

$$\textbf{Objective}: Maximize\ Total\ length = \sum_{i,j,k} x_{ijk} * Length_k$$

$$\textbf{Subject to}:\ Minimum\ Length: \sum_{k} x_{ijk} * Length_k \geq Length_{volume}$$

$$Inventory: \sum_{i,j} x_{ijk} \leq 1$$

$$x_{ijk} = 1\ if\ block\ k\ is\ selected\ for\ layer\ j\ in\ volume\ i, otherwise\ x_{ijk} = 0$$

The columns in this DP matrix reflect the maximum attainable length. The rows represent the number of pieces that are used. The number of rows correspond to the total number of pieces in the stock. The number of columns correspond to the desired length. For example, a desirable length of 500cm and 150 pieces, would correspond to matrix of 500 columns and 150 rows. The value in each cell represents the maximum length that can be reached with the available pieces. The target is the length that needs to be reached, by default the bottom-right cell, but any cell can be selected.

After the table is constructed the backtracking process starts. This process is used to determine which pieces were used in the optimal combination, target combination, selected in the DP table. The algorithm iterates from the target cell upwards to 0, 0. Each cell is compared to the same cell in the row above, if the cells are not equal it means that the piece representing the row was used to achieve the cumulative length represented in the cell. If it is equal its moves within the same row to the left until it finds the first value that is not equal. After a piece has been identified the position in the matrix is updated by subtracting the length of the piece from the current position in the DP table. The update reflects moving backward in the X-direction to the position before the current piece was placed and continues until all pieces are identified. This ensures that pieces cannot overlap each other and that there are no gaps in the final composition. This process is illustrated in figure 7.9.

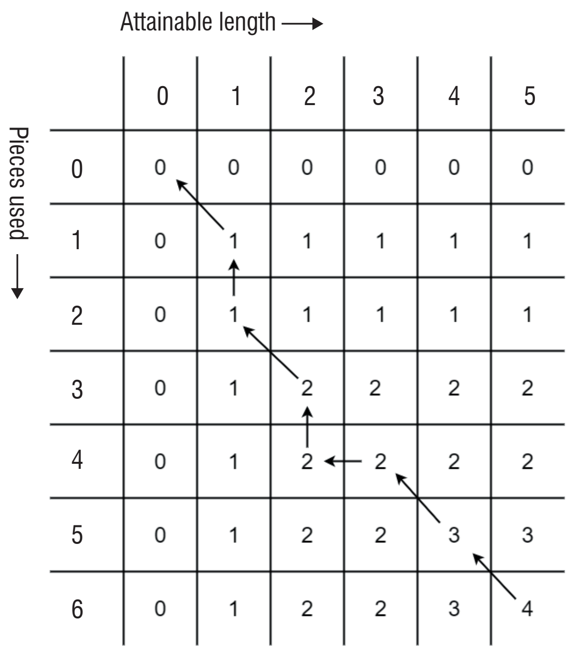


Figure 7.9: Scheme of the process of a dynamic programming table.

In the developed algorithm the DP table is constructed for a larger length: a length margin of +0.8m is added. The target length will therefore not be the default bottom right cell but the first cell equal or larger to the length of the volume to ensure that the length is always reached. If the desired length cannot be reached exactly the excess length can be cut. When the leftover piece has substantial length the remaining cut piece it can be placed back into the inventory. The program identifies pieces cut under 300mm length as waste, everything above is considered reusable and put back in the database. The dynamic arrangement is inherently greedy, meaning that it starts with large pieces and ends with small pieces. Excess length is therefore always cut from the first piece in the layer to ensure that small pieces are not cut even smaller. If the excess length is larger than 500mm the program cuts 300 mm of the first piece and 200 mm of the second in the row to ensure that the length of pieces is not reduced excessively and still one of the two parts can be reused.

Variable start and end aggregations

The program handles three types of connections which are illustrated in figure 7.10. The volumes representing the columns of the portal are fit on one side with connection type 1, accommodating a hinge support, and on the other side with connection type 3, accommodating a rigid support. The volumes representing the beams are fit on one side with connection type 2 and on the other side with connection type 3, except if there is only one beam, then both ends are fit with connection type 3. These types are build in the dynamic arrangement process by altering the required length and start points of the layers. This reduces cutting waste to make the connections. The connections are previously worked out in chapter 6.

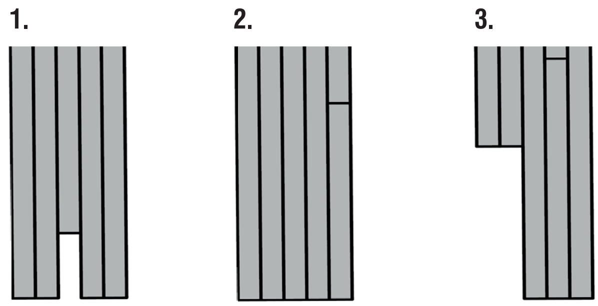


Figure 7.10: Default end types of volumes to accommodate connectivity between the elements

Lastly, when all the blocks are selected and 'cut', the algorithm reorganises the arrangement of pieces in the volume to the accommodate the conditions from the strength regions, effectively optimizing the volume to the quality of the wood. With the positions of the parts fixed, the X-coordinates of each piece can be retrieved.

7.6 Pre-processing

When the program has made the arrangements of pieces for each volume the pre-processing phase starts. This phase translates the created arrangements to a structural model in Karamba 3D, a finite element tool for Grasshopper. Karamaba 3D only allows for beam or shell analysis and cannot support the 3D arrangements directly. The parts, therefore have to be converted to lines and points and given a cross-section and material type. As the beam model does not allow to model the dowel connections as they are in reality, virtual connections between the elements have to be created. These connections simulate the connectivity between the consecutive parts and allow for transference of forces. This ensures that the model can simulate composite behaviour between the elements, see figure 7.11. Each part is discretized in two half's to be able to create three connections in total. Two at the ends and one in the middle. The program thus calculates for each half of the parts the stresses. Discretizing the parts into even smaller sections will improve accuracy, especially for connectivity, but also enlarges computational time. In this case speed is chosen over accuracy. This paragraph will highlight how connectivity between parts and between elements is made in a beam model.

Connectivity between parts

As aforementioned, each parts has three connection points, one representing the start of the part, one the middle and one the end. The program searches for each point in a layer to the closest point in the consecutive layer horizontally and vertically twice. This process

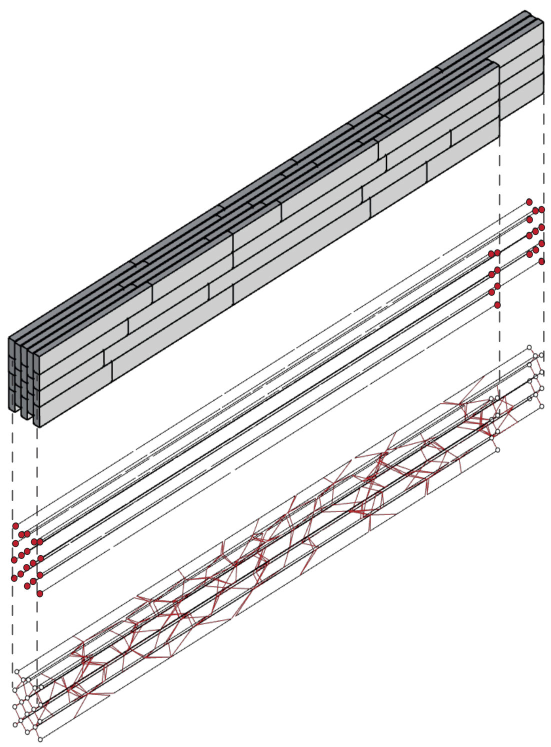


Figure 7.11: Preprocessing technique: Finding centrelines and support points of each piece and creating virtual connections between centrelines.

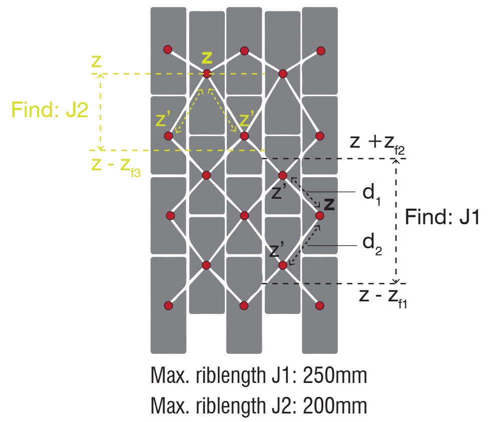


Figure 7.12: Process of creating virtual joints with befitting constraints

is visualised in figure 7.12. Horizontally the program calculates the euclidean distance to each point in the consecutive layer and selects the point closest. This point has to fit within two constraints: The z-coordinate should be in range of $+Z_{f1}$ and - Z_{f2}. Which are set on 80 mm and 130mm respetivly to prevent vertical impossible connections even if proven to be the closest. The maximum length of the connection cannot exceed 250mm. This function is executed form layer 0 - n and from layer n - 0.

Because the network is three-dimensional some essential connections could be skipped by these constraints. The program therefore also searches vertically for joints. The constraint for this function states the closest point must have a z coordinate lower than Z - Z_{f3}, which is set on 130mm. A global constraint checks if joints between start- and endpoints can be made. For joints between a start and endpoints: [x-coordinate of {S} + 100mm < x-coordinate {E}], for joints between and end and startpoint this is reverse [x-coordinate {E} - 100mm > x-coordinate {S}]. The 100mm ensures that connections satisfy the longitudinal overlap of 100mm. This results in a network which is illustrated in figure 7.11 and capable of transferring loads through the sections.

Stiffness of virtual joints

These vertical joints cannot simply be given a cross-section and material as they have to represent the stiffness generated from the dowels and the external links. The connections should be given a modified stiffness representing an reasonable global stiffens, or in other words an expected amount of composite behaviour in the sections. As the devised structural system performance is unknown the Youngs and shear moduli of the connections can be modified to represent a certain global stiffness based on O'Ceallaigh et all. (2022). This research conducted mechanical testing of DLT with compressed timber dowels oriented parallel to the load, which showed a difference of 60 - 80% in stiffness compared to Glulam. Glulam can be considered to show 100% composite behaviour and is therefore used as a baseline. This test set-up has the dowels parallel to the load, enabling force transfer by the faces of the pieces and not only the dowels which is not the same as the structural concept used in this research project. However, the external links should generate similar behaviour, and therefore it is assumed in this research that the global stiffness of the system should be in range 60 - 80% lower than compared to a similar section from Glulam timber. The deflection is calculated for a Glulam timber portal in Karamba3D and compared to the deflection of the discrete system. The Youngs and shear moduli of the connections are modified to fit within the set range on 15.000mpa and 10.000mpa respectively. In the next chapter (9) algorithmic performance tests will conclude is there is a correlation between size of the cross-section, span and the required stiffness of the joints.

Connectivity between aggregates

The last step in the pre-processing phase is to rotate and move each element in the correct position and assemble the portal frame. This requires making virtual connections between elements representing the beam and columns of the portal frame. Figure 7.14 illustrates these virtual connections. The parts at the end of each element are isolated and used to set both the support and the connectivity conditions between elements. The support conditions can be easily set to rigid or hinged by changing the degrees of freedom accordingly. The connections between the elements are made by isolating all the points of the parts belonging to the connection.

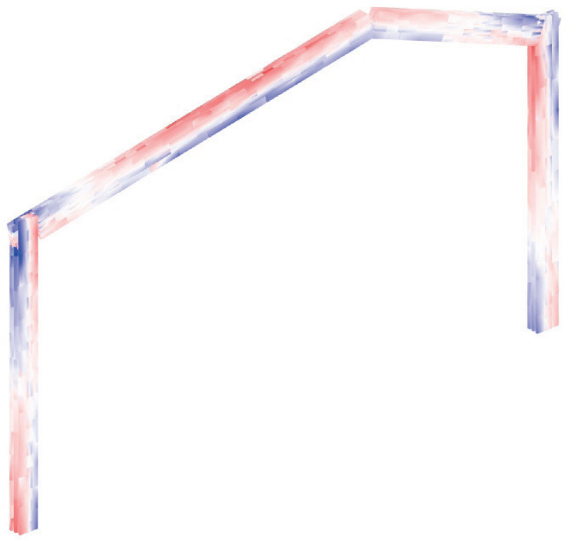


Figure 7.13: Assembled Karamba3D model showing composite behaviour between all discrete parts.

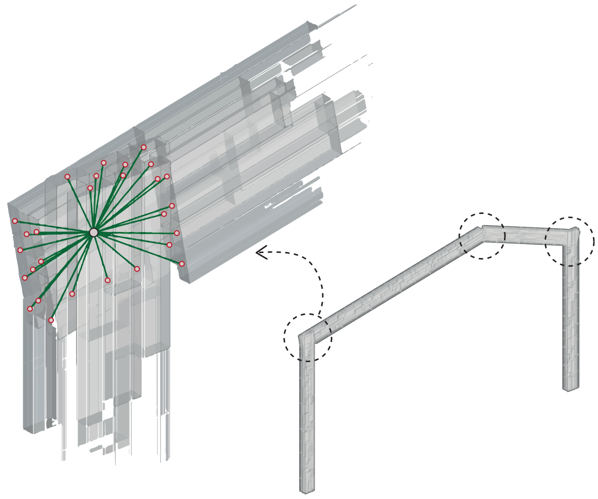


Figure 7.14: Virtual structural knot connecting the elements with a given representative rotational stiffness

The centroid of all these points is calculated and used to create a virtual structural knot. This knot is given a representative rotational and translational stiffness to simulate the expected deformation. Rotational stiffens can be calculated by the method given in chapter 6.

Lastly, the structural model can be calculated in Karamba 3D and analysed. In figure 7.13 a composition is illustrated, the figure illustrates a clear composite behaviour between the parts with distinctive compressive and tensile regions.

7.7 Optimization loop

The optimisation process has to be started manually after the designer is satisfied by with the aggregation and its performance found in the structural model. For each part the highest utilisation value considering all the result-cases is calculated. This ensures that the wind load is considered from both sides and that both columns are optimised to handle wind form both directions.

The used optimisation type is a stress based optimisation. The objective in a stress based optimisation is to maximize the amount of stress in a structure. Other optimisation types like compliance based optimisations are not possible using Karamba3D as the algorithm only supports an ESO type of method. Karamba3D calculates the utilisation rate, a value that represents the percentage of stress in the part in perspective to the maximum allowable stress. However, each parts is discretized in two halfs resulting in two utilisation rates. The average utilisation factor of each part is calculated and used to determine which piece can be removed most effectively. The part paired with its virtual connections showing the lowest utilisation are removed from the composition and the stresses in the structure are recalculated. A priority is given to pieces with a larger distance to the centre to prevent a tube from forming which otherwise could hinder the removal of pieces due to arising connectivity issues. This is an iterative process and continues until a maximum utilisation factor in the system of 85% is reached. The loop saves each iteration and the designer can easily iterate through previous versions. A restriction is set on parts which are not allowed to be removed and shown in figure 7.15. Parts around the global connections cannot be removed because these are required to be able to construct the connections. The loading area is also constrained from being removed to accommodate a bearing space for the roof structure.

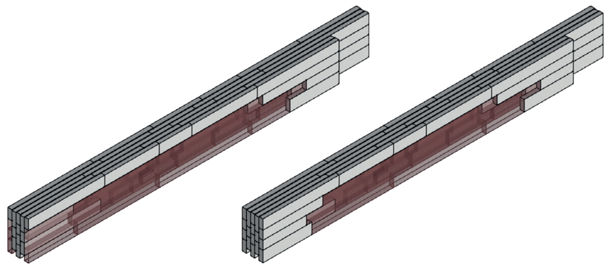


Figure 7.15: Parts restricted from being removed. in red parts allowed to be removed in grey parts that are constricted form being removed. Left a beam element and right a column.

7.8 Post processing

In the last phase the designer can retrieve more detailed information concerning the optimised structure. The post processing has three actions available for designers:

1. Create cuts through the sections to obtain more detailed information on specific locations.
2. Produce 2D drawings of locations for drilling holes.
3. Update the inventory and create a material passport for the elements.

Cuts can be made in the structure to retrieve the geometry of the section, geometric properties, utilisation and deflection in the cut part. The designer also has the ability to create 2D drawings of the layers and the required dowel holes for each part. Lastly the designer can, if the design is satisfactory, remove the parts used in the composition out of the wood database and create a material passport by importing these parts with additional information on the project into an the new database to accommodate future reuse. Figure 7.16 shows an potential result of a optimised portal frame created with the algorithm. In the next chapter the performance of the algorithm is tested.

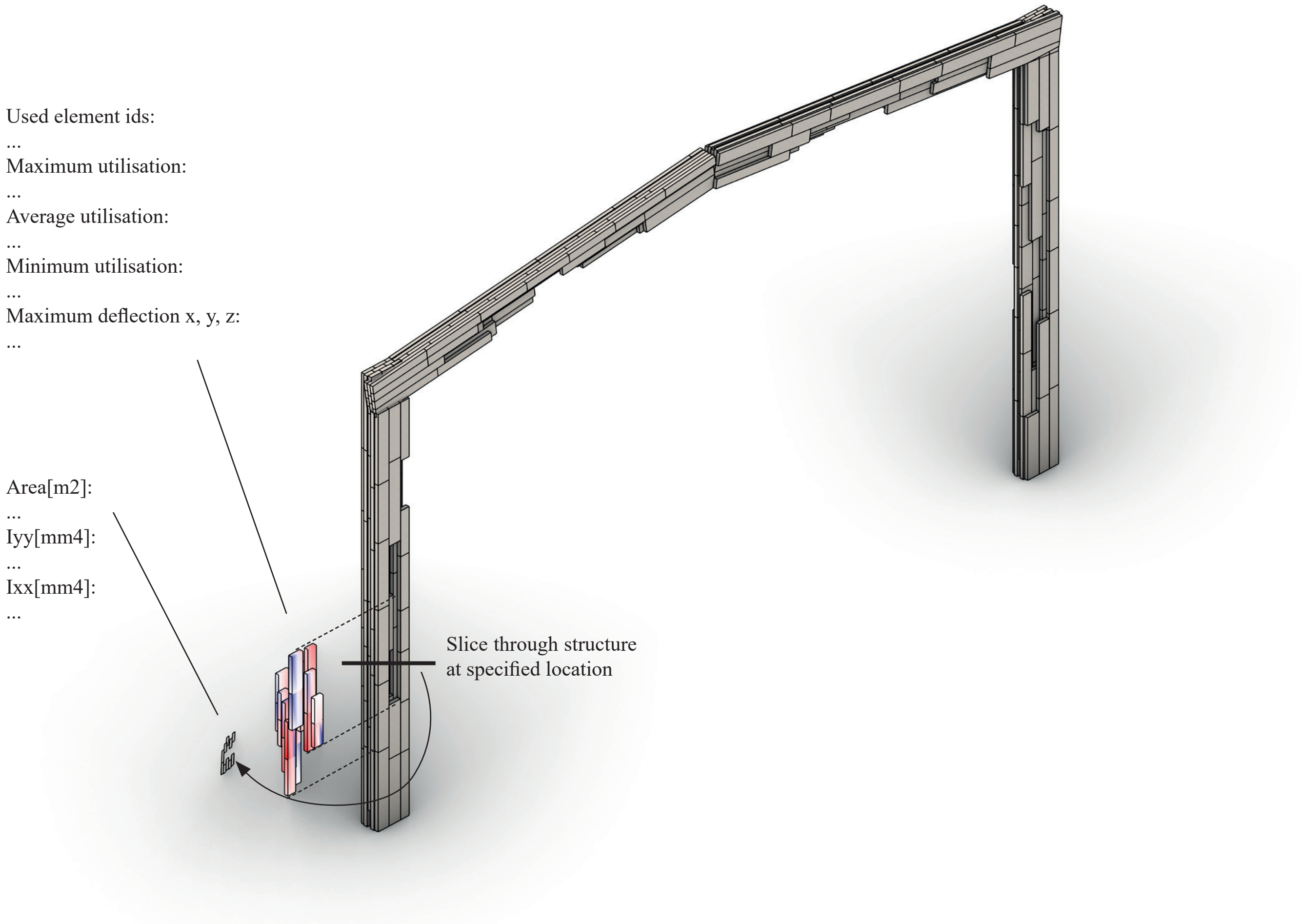


Figure 7. 16: Post processing analysis. Cut can be made through the structure and geometric and structural information be retrieved.

III Test Phase

8. Algorithmic and structural performance

This chapter will discuss the structural performance of the system and test the flexibility, speed and limitations of the developed algorithm. The performance is tested with a simple structural frame spanning 7 meters, a bay size of 3.6m and loaded with characteristic loads for dead load of a roof structure (LC1), wind (LC2) and snow (LC3), see figure 8.1. The load combination key used is in compliance with NEN-EN 1990+A1+A1/C2/NB and expressed as: (LC1 * 1.2) + (LC2 * 1.5) + (LC3 * 0.9). Structurally the performance of the system is tested on the following topics: Size influence of the cross-section types, the gain in performance through strength grade optimisation and the influence of the modified stiffness of the joints. The optimisation process is tested by checking the constraints on the utilisation. Lastly, the algorithm is also tested with varying inventory scenario's.

8.1 Algorithmic output

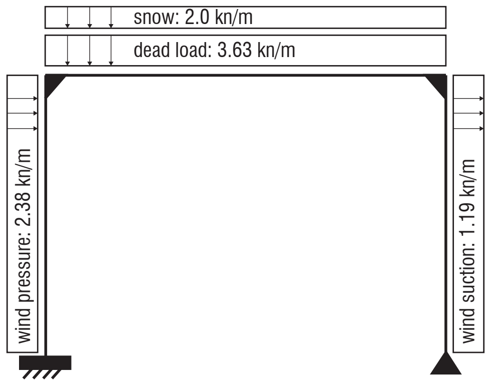


Figure 8.1: Input loads and key forces acting on structure.

Initialisation

The resultant maximum bending moments are illustrated in figure 8.2. The algorithm uses these bending moments to determine the initial dimensions of design spaces, i.e. volumes, for each element within the portal frame. The example below is created for a portal frame with an cross-section that is initially optimised to an utilisation of 76.9%. Results from the optimisation are found to be highly variable and depending on strongly dependent on the configuration. Further in this chapter the effects of different aggregations is discussed. The portal frame consist of three elements: two columns and one linear beam. The initial dimensions where outputted and described according to the loads and dimensions of the portal in figure 8.1 as:

- Element A: 160x400mm x 5 meters
- Element B: 200x400mm x 7 meters
- Element C: 160x400mm x 5 meters

Element A has a noticeable larger cross-section compared to the other two elements, this is due to the set rigid support condition. This element is expected

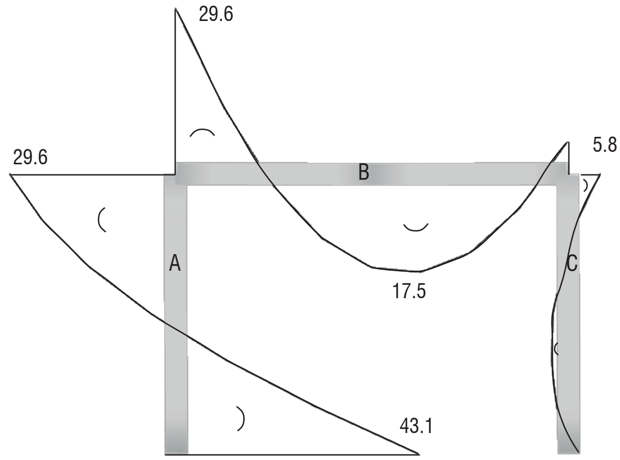


Figure 8.2: Resultant bending moments

to take moments in the foundation and thus requires a larger cross-section.

Aggregation

The aggregation is made with a scenario of abundant and highly diverse stock Cross-sections can vary in width between 38mm and 44mm, heights are ranging from 70mm - 150mm in steps of 10mm and lengths from 300 - 1000mm in steps of 50mm. The strength grade distribution of the stock is set on C14: 30%, C18: 30%, C24: 35%, and C30: 5%. A random sample of a 1.000 parts is taken.

The aggregation is made with a prioritization on selecting the most diverse layer combination. In total 410 of the 1.000 parts were used to create the three elements. The algorithm was able to find combinations of parts resulting in a 100% filling rate. In other words,

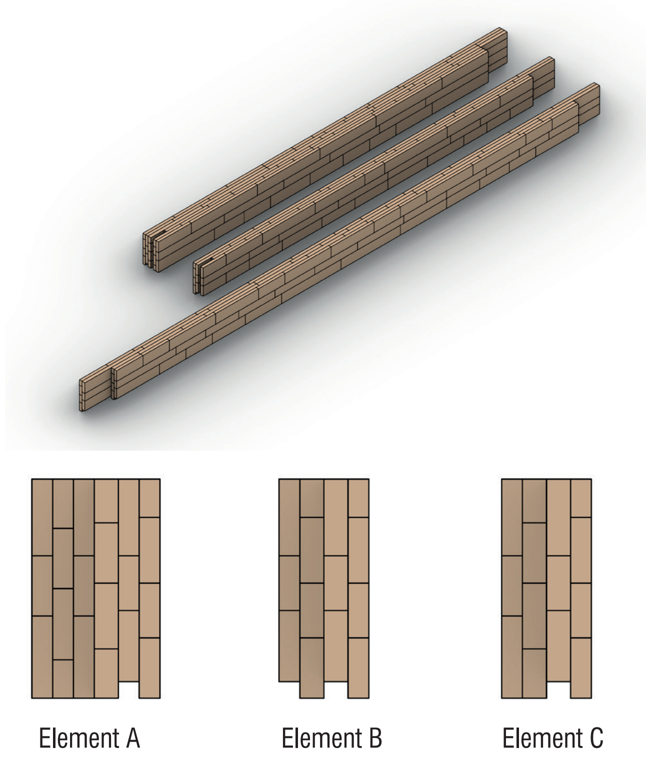
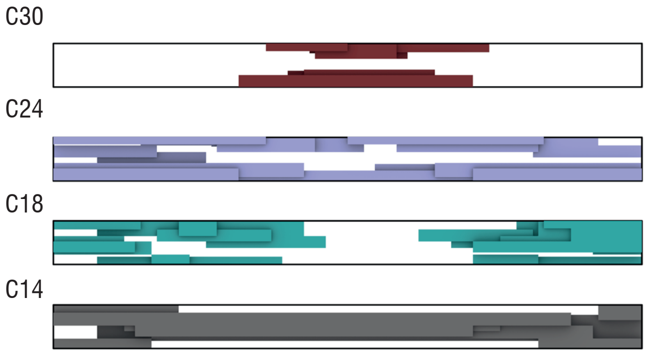


Figure 8.3: Aggregation results.

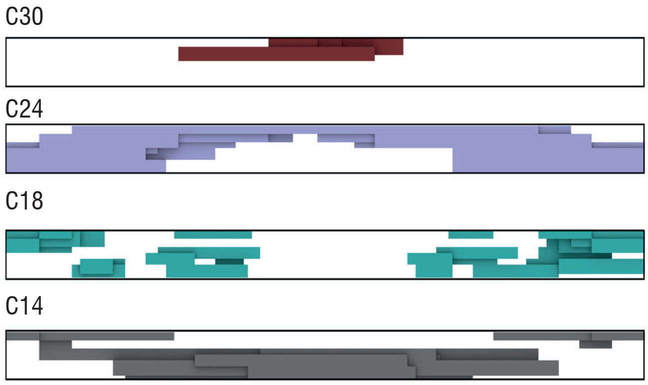
the aggregated elements can be made without cutting any of the parts, resulting in zero cut-off waste. The aggregated elements are illustrated in figure 8.3. The random taken sample from the database reflects the distribution of strengths. All three elements have a similar distribution of strength grade compared to their respective volume. An abundant and diverse inventory will therefore result in elements of equal quality. The overlap of 50mm is guaranteed in each aggregation and parts smaller than 100mm are only placed at the start or the end of a layer to accommodate dowel placement. Incrementations of maximum 30mm had to be made to ensure this overlap. This shows that the implemented constraints all perform as expected.

Figure 8.4 illustrates the strength division within the elements. The algorithm was able to successfully create an aggregation considering the strength optimisation criteria set in chapter 7. The beam element is divided into two regions, a high strength region, 1/3 of the section height and a low strength region, 2/3 of section height. As clearly depicted in the figure 8.4 element B, the higher strength grade parts are placed in on the sides and on top in the middle, following the compressive stress line. The lower strength grade parts fill up the remaining spaces as set in the strength region constraint from chapter 7. This division was chosen because it simulates the principal stress-lines, placing lesser parts at places with lower stress levels. For column type of elements the strength grade division is different and also shown in figure 8.4 elements A and C. This optimisation was also successful considering the set constraints. Different arrangements and the influence on the overall strength of the system will be tested in chapter 8.3 and 4.

Figure 8.5 shows the length distribution in the beam element. The algorithm was also able to create a composition beholding to this prioritisation. The contours of this prioritisation are visible, but it also shows elements placed outside of the prioritised area's. Smaller elements are also placed at the supports and longer elements are also found in the middle of the low strength region. This is a limitation of the stock constrained design. The algorithm prioritises minimizing waste over all other constraints what affects the length constraint the most. Therefore aggregations cannot, in each case, comply to each constraint. The implemented stock constrained design strategy is beholden to the available elements in the inventory and must therefore prioritise placement over other optimisation constraints to minimize cutting waste and further down-cycling of timber parts.



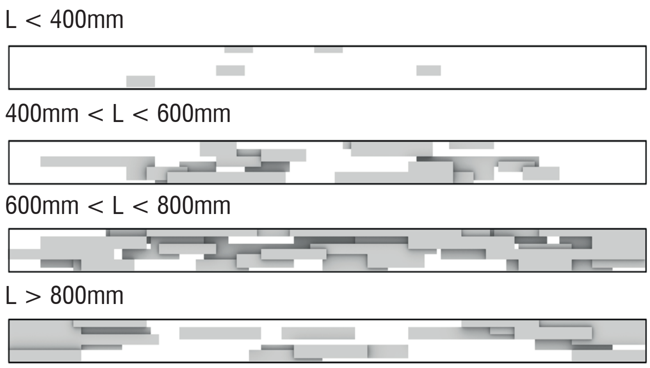
Strength grade distribution in column element (A, C)



strength grade distribution in beam element (B)

Strength grade	Used parts				
	Element A	Element B	Element C	total	percentage
C14	51	59	30	**140**	*34%*
C18	33	33	24	**90**	*22%*
C24	46	68	38	**152**	*37%*
C30	14	9	5	**28**	*7%*
total	**144**	**169**	**97**	**410**	*100%*

Figure 8.4: Strength grade distribution in elements A, B and C.



Used parts				
Long: *L >= 850mm*	27	35	37	99
Medium long: *600 <= L < 850mm*	70	52	37	159
Medium: *400 < L < 600mm*	33	35	17	85
Short: *L <= 400mm*	39	22	6	67
total				**410**

Figure 8.5: Length distribution in elements A, B and C.

Structural performance

The combined aggregation results in a structural system of 474kg, with a maximum utilisation factor of 76.9% and a maximum deflection in the z-direction of 15.8mm. A rotational stiffness of 7.000kNm/rad is applied for the two connections between the elements. The virtual connections between the parts are set with a Youngs module of 15.000mpa and a shear module of 10.000mpa. These values provide enough stiffness to ensure the target difference of 50 - 80 % in deflection between the discrete system and the initialised massive Glulam portal as aforementioned in chapter 7.

Optimising this aggregation results in a final composition of 385kg with a maximum utilisation factor of 83.7% and a maximum deflection in the z-direction of 21.6mm. Through the optimisation a removal of 85 parts was possible, creating a structurally efficient structure with 325 parts in total. The final optimised discrete system is illustrated in figure 8.7. In total 41 slices were made through the structure to be able to study the cross-sections, deflections and utilisation of the parts more detailed. Each slice returns essential geometric information for structural post-optimisation validations like buckling. Figure 8.6 illustrates the deflections and utilisation for each slice before and after the optimisation. Noticeable is that the deflections become larger but do not change unexpectedly except for the deflection in the y-direction. Due to removal of parts, torsional forces can more easily affected the structure. The external links and roof structure will help in stabilising the beam from rotating but lateral torsional buckling should be taken into account with thin and shallow sections. The utilisation becomes larger due to the optimisation, but also changes shape. This can be seen more clearly when projected on the structure as in figure 8.8. Notable is that the elements are utilised more uniformly after the optimisation, especially the beam section profits form the optimisation. Although there are a few peak stresses in this section, there is still a clear linear trend. Peak stresses occur when the

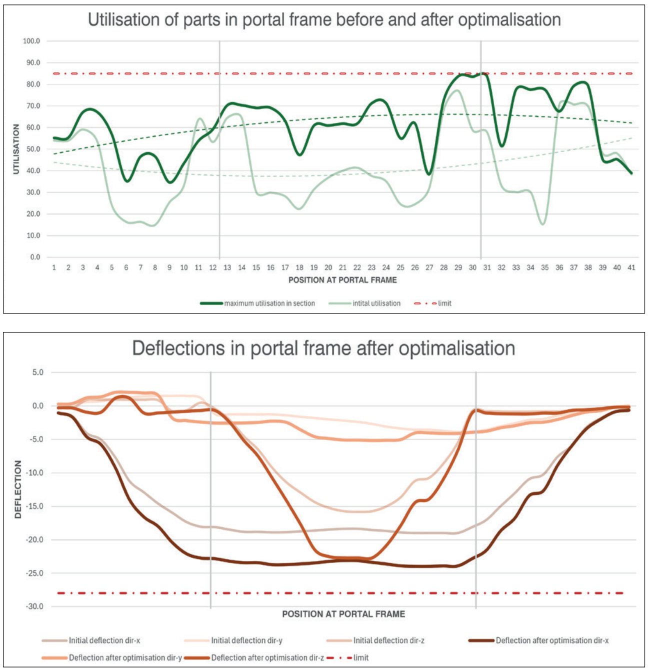


Figure 8.6: Results form optimisation process

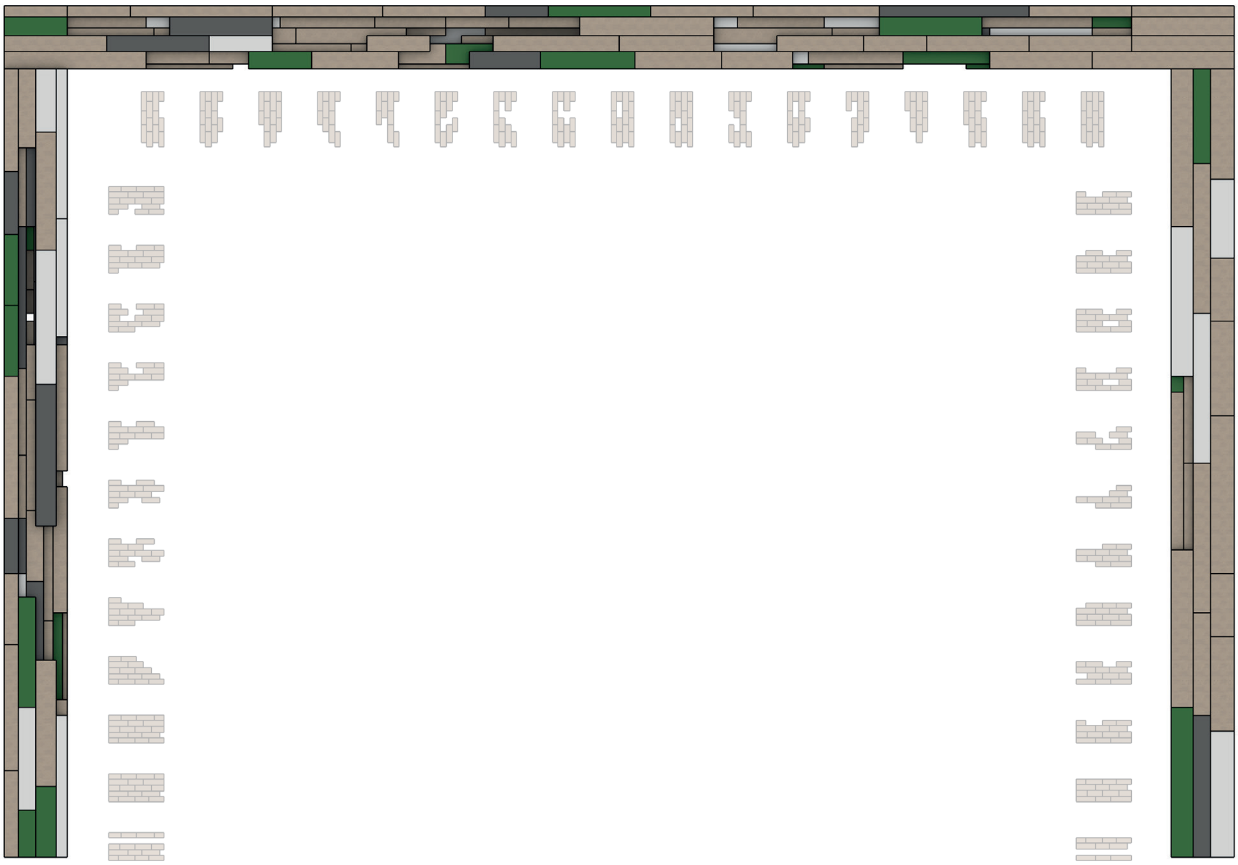


Figure 8.7: Front view of optimised structure, with cross-sectional slices

optimisation program removed the last possible piece. Each additional removal of pieces would result in global instability and therefore an full uniform utilisation cannot be reached. It is also noticeable that most parts are taken form the middle of the beam and on places where the moment line reaches 0. This behaviour falls within the expected results as decreasing weight is most effective in the middle to lower the bending moment and at places with moments nearing zero equal lowers stresses. Overall, it can be concluded that the developed algorithm works well. A viable aggregation was made and shows results within the expected solution space. In the following parts of this chapter variables in the algorithm will be added and tested in order to analyse the algorithms functioning and principles of discrete design.

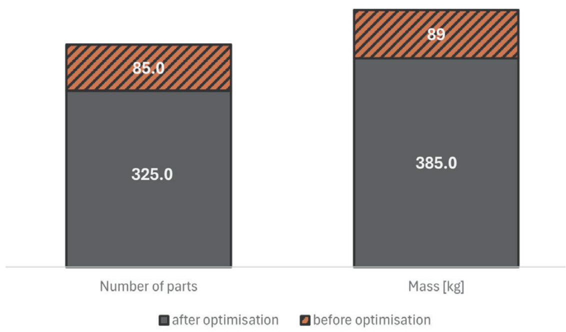
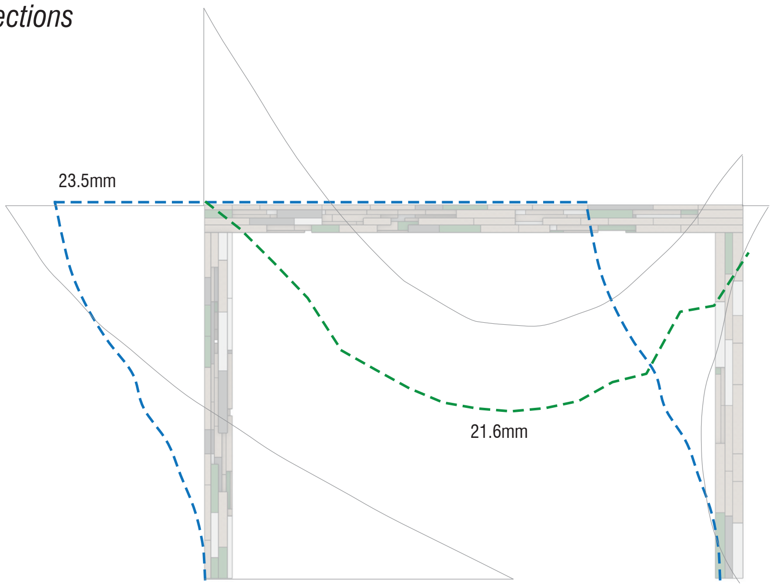


Figure 8.9: Optimisation results

Resultant deflections



Resultant utilisation distribution

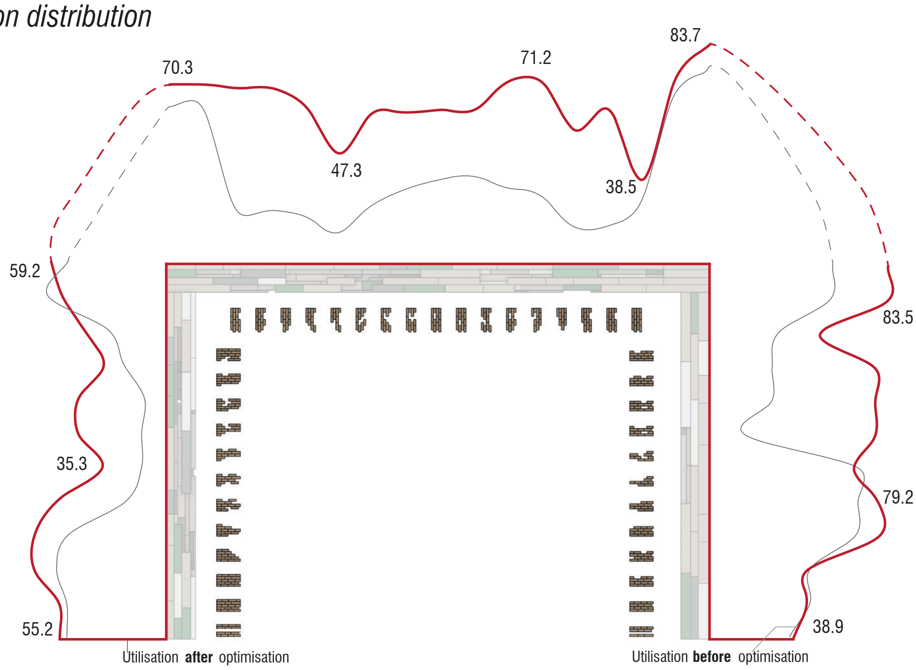


Figure 8.8: Front view of optimised structure, with cross-sectional slices, deformations and utilisation across the structure.

8.2 Optimisation approach

The optimisation is executed based on the utilisation factor of the elements. This optimisation method removes the element with the lowest utilisation until the set objective is reached. As aforementioned in chapter 7, the parts are discretized into two halfs in order to create a structural model, see figure 8.10.

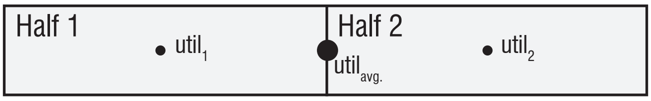


Figure 8.10: Discretized part with possible optimisation approaches

This indicates that the utilisation of a part cannot be represented by a single value. Either the average, minimum or maximum of the two halfs can be used to determine which piece has to be removed. The above result is generated with the average value of the piece. The maximum utilisation value is used to determine if the objective threshold of 85% is reached and the loop should be broken. Figure 8.11 shows the influence of various optimisation approaches. Removing the

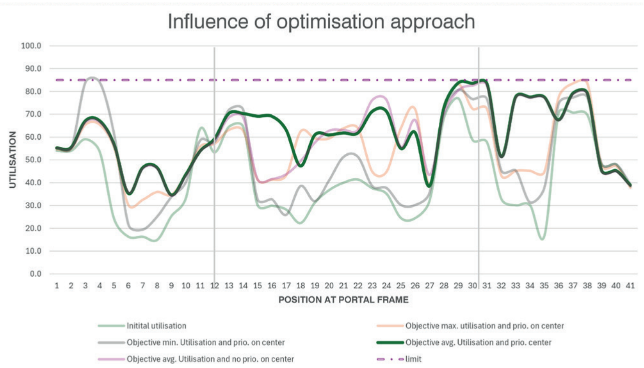


Figure 8.11: Results of various optimisation approaches

piece with the minimum utilisation factor results unexpectedly, in this case, in the worst result, and was able to remove the least amount of parts and has no clear uniform utilisation distribution. Using the average utilisation ensures much better results, maximizing the utilisation the most uniformly across all elements in the structure. Using the maximum utilisation of each part results in a similar result as the average factor, but does not ensure a uniform distribution across all elements and removes fewer parts. In contrast, with different aggregations results of the minimum value achieving the best result have also been observed, which means that the approach is also dependent on the quality of the composition. Overall, using the maximum and minimum could result in slightly better results in some cases but in majority of the time cannot guaranty a global optimum. Using the average utilisation value is more stable and always provides similar results. Using an extreme value could, in some cases, lead to a better global solution but cannot be guaranteed.

If utilisation values are the nearly identical, the priority of the elements is another variable in the optimisation approach. A priority can be set to remove pieces farther or closer to the centre first if there are elements with similar utilisation factors. Removing parts that are further from the centre would result in an I-beam shape, which could reduce connectivity issues. The middle parts of the composition have the ability to connect more parts with each other; by removing these, the structure would be left with fewer connections, which would lead to a lower mass reduction and global stiffness. Results with and without the prioritisation are similar, as seen in Figure 8.11. In general, structures with centre prioritisation are utilised more uniform. The outcome of this prioritisation is depicted in Figure 8.12.

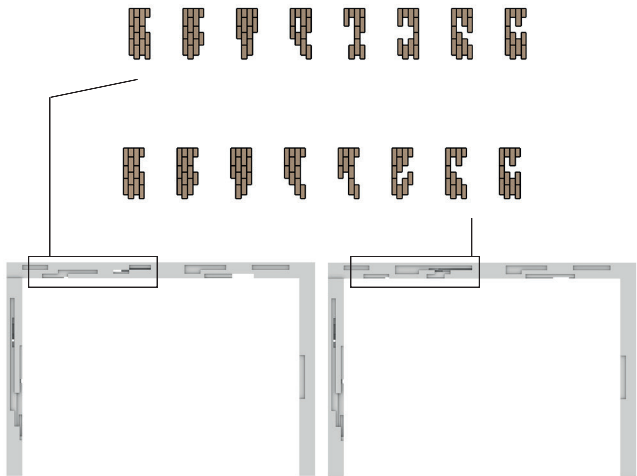


Figure 8.12: Difference in results with (right) and without (left) prioritisation on the final composition.

8.3 Virtual joint modelling

The virtual joints between the parts as highlighted in chapter 7 can be modelled according to two principals, One method is to model the connections as springs with a respective rotational and translational stiffness to simulate dowel action. Another method is to model the connections as beam elements with a modified Youngs module to simulate the dowel connection. Both methods were tested, and showed different structural behaviour. Figure 8.13 shows the stress distribution in the structure for both approaches. On the left the spring method is shown. This image clearly shows that the springs do not allow for a composite behaviour within the beam. Each element has a local compressive and

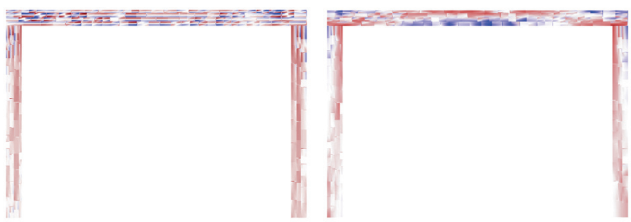


Figure 8.13: Stress distribution in structure with spring connections (left) and beam element (right)

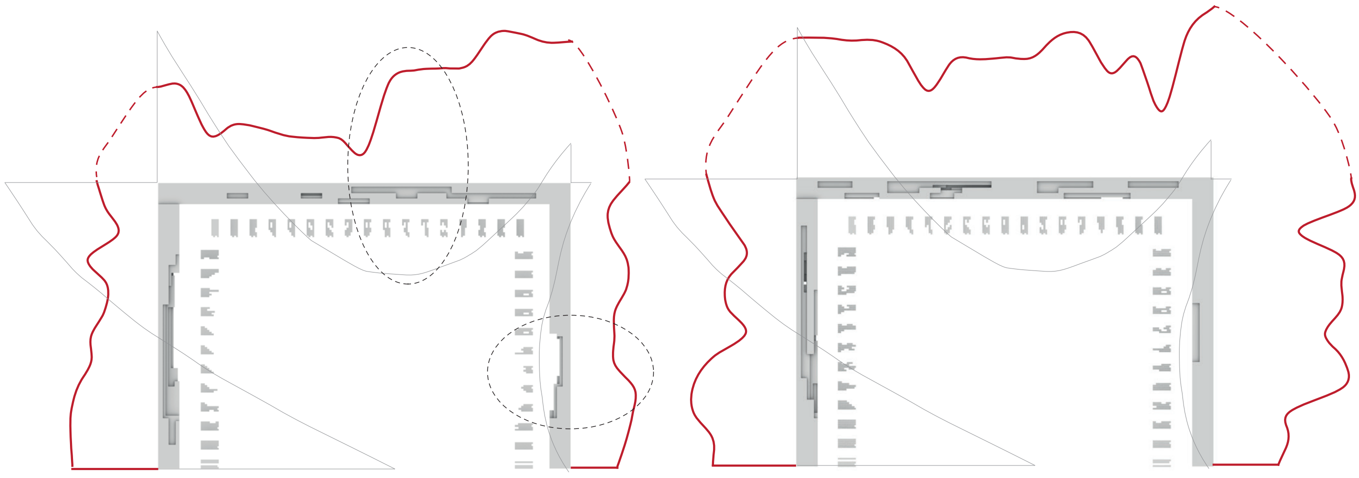


Figure 8.14: Utilisation distribution after optimisation for both cases of modelling the connections. On the left the spring method and on the right the beam method.

tensile area instead of complying to the expected global stress distribution. On the right the beam approach is shown, modelling with beams allows for composite behaviour within the beam and better transfer of forces. The amount of composite behaviour can be influenced by modifying the Youngs module. In both methods the stiffness values are adjusted to reach a global difference in deflection of -60% in comparison to a Glulam beam. The optimisation is run for both cases and the result is plotted in figure 8.14. Results are varying a lot, the spring method removes most of the parts on the places where the bending moments are largest. Moreover, the column on the right is reduced to a maximum and still the utilisation remains the lowest comparing the whole structure. On the other hand the beam method leaves most of the column in tact and removes the most parts at the position were the moment approximates zero. This method provides better and more realistic results. The strange behaviour found using the spring method could be caused by the rotational stiffness of the connections. An to high stiffness could result in a to stiffens composition restriction any deformation. However, allowing for complete rotational stiffness does not improve the outcome. Another explanation could be that Karmaba3D calculates deformation with springs locally and not globally, resulting in a structural model that cannot reflect the real deformation and

Moddified stiffenss: E 15.000mpa, G 10.000mpa			
Span [m]	height [mm]	width [mm]	Deflection comparison to Glulam C24 [%]
5	350	160	106%
5	400	150	106%
6	400	150	56%
7	350	190	75%
7	400	160	68%
7	450	160	81%
8	350	210	67%
8	400	180	52%
8	450	160	93%
9	400	200	65%
9	450	170	133%
Average			82%

Figure 8.15: Length and strength distribution in composition without optimisation

stresses. The beam method was proven to give more viable results within the expected solution domain and is used further in the algorithm.

The default modified stiffness is set on 15.000mpa and 10.000mpa for respectively Youngs and shear module. For spans form 5 to 9 meters and different cross-sections tests showed that the deflection compared to Gluam is high versatile, see figure 8.15. Values ranging form 52% to 133% were observed. All deflections reside above the minimum range of 50% and some proved more unfavourable. Noticeable is that smaller spans have a high deflection difference. This might be explained through considering that, when using the same stock, more pars are available for elements with a smaller span than for elements with a larger span. Because the algorithm is essentially always greedy, longer elements are given precedence over smaller ones. Since the model only provides three connecting points for each element, fewer connections can be made with longer parts. As a result of this programme limitation, a composition with numerous longer parts may have a lower global stiffness. As more connections can be made in the model, a composition with a high degree of diversity may be more rigid. In reality the opposite would be expected.

More connections would result in more hinges in the system making it more unstable. This hypnotises is further researched in the next paragraph. A limitation of the program is that currently, for each composition the stiffness has to be checked and if needed readjusted. This method could be providing only optimistic results and in reality it could be found that different composition, with either more short or longer elements are simple less stiff than others. Mechanical testing has to conclude the validity of this method. A representative stiffness is essential for viable results and highly controls the final outcome further development should be done to make the program more stable.

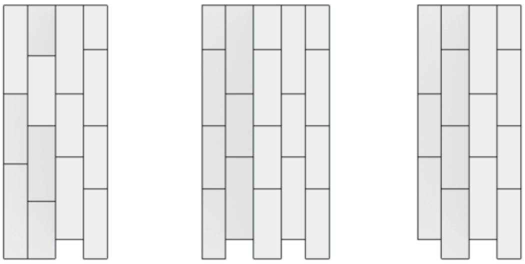


Figure 8.16: Used cross-sections for testing length influence

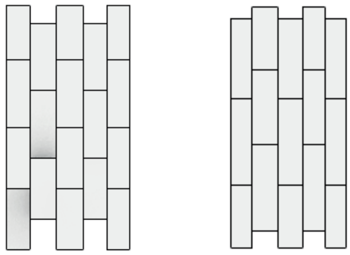


Figure 8.17: Used cross-sections for testing size influence, left an aggregation restricted to pieces < 110mm and right restricted to piece > 110mm

8.4 Size influence

This paragraph will investigate the influence of the size of the parts on the global stiffness and optimisation performance of the structural system. Both the influence of the diversity in height of the parts as the length of the parts is investigated. The tests are executed without taking varying strength grade into account. All parts are set on strength grade C24 and the stock alone is adjusted. Loads, span and global cross-sections are all set as in paragraph 8.1 to ensure a viable comparison.

Length influence

A composition of parts with lengths ranging from 700 - 1000mm showed an initial deformation of 18mm (91% difference compared to Glulam portal) and maximum utilisation value of 66.5%. In total 325 parts were required to create the composition which is illustrated in figure 8.16. In comparison, an composition only using parts with lengths ranging from 300 - 600mm required 601 parts. An identical composition could be created as illustrated in figure 8.16. This aggregation has a initial deformation of 23.8mm (153% difference compared to Glulam portal) an maximum utilisation factor of 110%. This result effectively disproves the previous stated hypothesis that longer elements reduce the global stiffness as a result of fewer virtual connections.

The opposite has proven to be true: smaller element make the system more prone to larger global deflections and less stiff. This complies with the expected realistic behaviour of a discrete system. The amount of joints per part is not manipulating the results and the program simulates the behaviour well. Large differences in global stiffness between aggregates could therefore be explained to an assembly of a composition with many small elements.

Further mechanical tests should conclude if the large difference can be viewed as a realistic result or that the stiffness should be readjusted each time and better method for modelling the joints has to be devised.

Cross-sectional height influence

Two compositions with restriction of different heights were made to analyse the influence on cross-sectional height on the structural performance of the system. Both composition types are illustrated in figure 8.17.

Stock restricted on parts with heights lower or equal to 110mm results in an aggregation with an initial maximum deformation of 17.9mm (89% difference compared to a Glulam portal), a mass of 474kg, a maximum utilisation of 82.7% and a total of 450 parts. After optimising the structure, 124 parts could be removed, reducing the mass to 360kg. The final

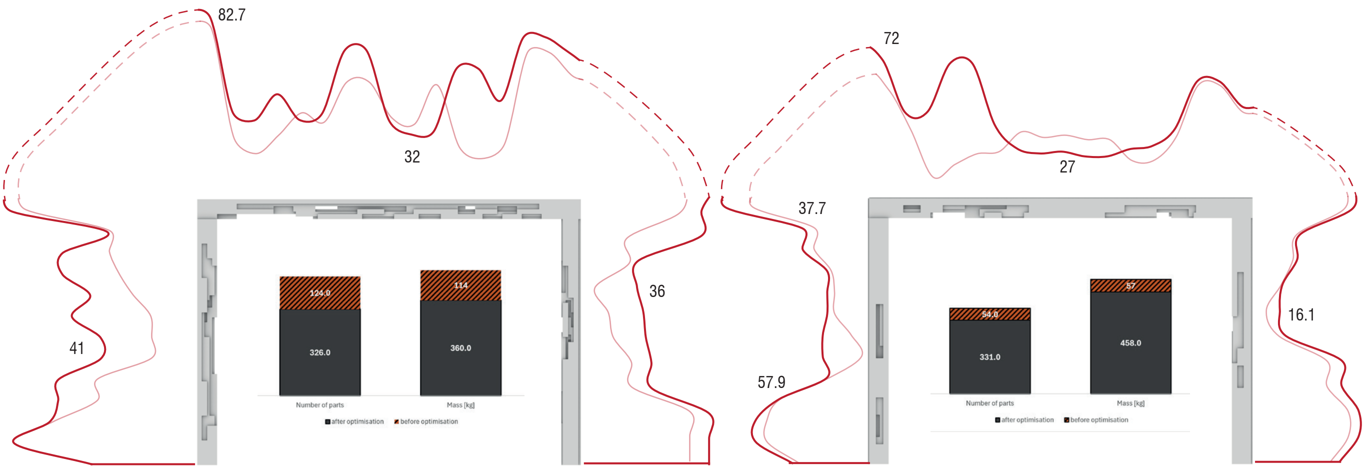


Figure 8.18: Used cross-sections for testing size influence, on the left using small parts, on the right using large parts.

structure is illustrated on the left in figure 8.18 and has a maximum deformation of 24.1mm and a maximum utilisation of 82%.

Stock restricted on parts with heights greater than 110mm results in an aggregation with an initial maximum deformation of 15.6mm (68% difference compared to a Glulam portal), a mass of 515kg, a maximum utilisation of 61.4% and a total of 385 parts. After optimising the structure 54 parts could be removed, reducing the mass to 458kg. The final structure is illustrated on the right in figure 8.18 and has a maximum deformation of 20.5mm and a maximum utilisation of 72%. Figure 8.18 shows the utilisation lines of both compositions. These results conclude that compositions with a higher share of large pieces will result in a more stiff structure. Initial deformation and utilisation are far lower compared to compositions a larger share of small pieces. However, noticeable is that through optimisation the structure with smaller parts becomes more efficient than the structure with larger parts. Smaller parts accommodate more connections through the system providing the advantage that more parts can be removed resulting in a structure with lower mass. The utilisation line in figure 8.18 shows clearly that the structure with a higher share of large parts has not reached its maximum utilisation yet. Structurally more parts could be removed but connectivity issues withhold this.

Concluding: smaller pieces add more degrees of freedom in the structure, resulting in larger deformation and lower global stiffness, see also figure 8.20. Larger parts result in higher initial stiffness however, smaller pieces have the advantage that the optimisation can be executed more effectively resulting in a final structure that is utilised more uniformly an efficient. In contrast, optimising the structure with a higher share of large pieces results in premature end of the loop due to connectivity issues. A diverse aggregation will provide the best results combining the flexibility for optimisation of the smaller parts with the higher stiffness for the larger pieces.

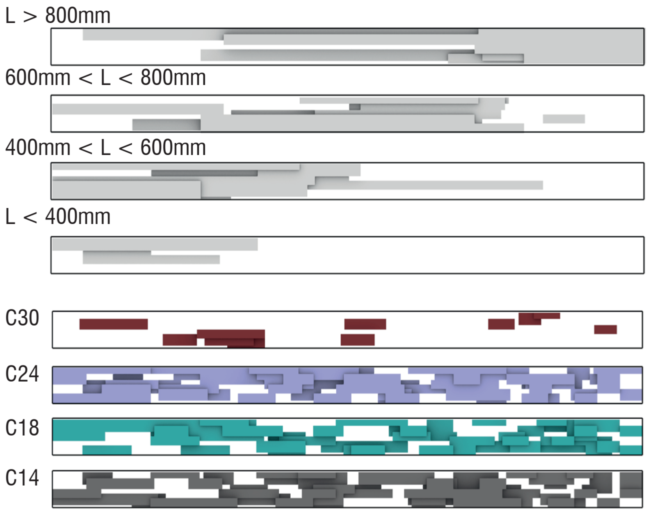


Figure 8.19: Length and strength distribution in composition without optimisation

8.5 Strength grade influence

In this section the influence on the algorithmic and structural performance concerning the strength grade optimisation is discussed and analysed.

Strength grade regions

As aforementioned, the strength grade distribution within the aggregation is optimised based on the principal stress-lines. Higher strength grade parts are placed in area's accompanying higher stress levels. In this section strength regions are altered to see the effect on the structural system. The length prioritisation of each region is kept constant.

Figure 8.19 shows how the algorithm would place the parts if the algorithm would not include the strength grade optimisation. All parts are placed randomly, and all the longer parts will be placed on one side of the aggregation as the algorithm start with placing longer parts and ends with smaller parts. This composition would result in a initial utilisation of 97.8% and a maximum deflection of 24mm. Exactly

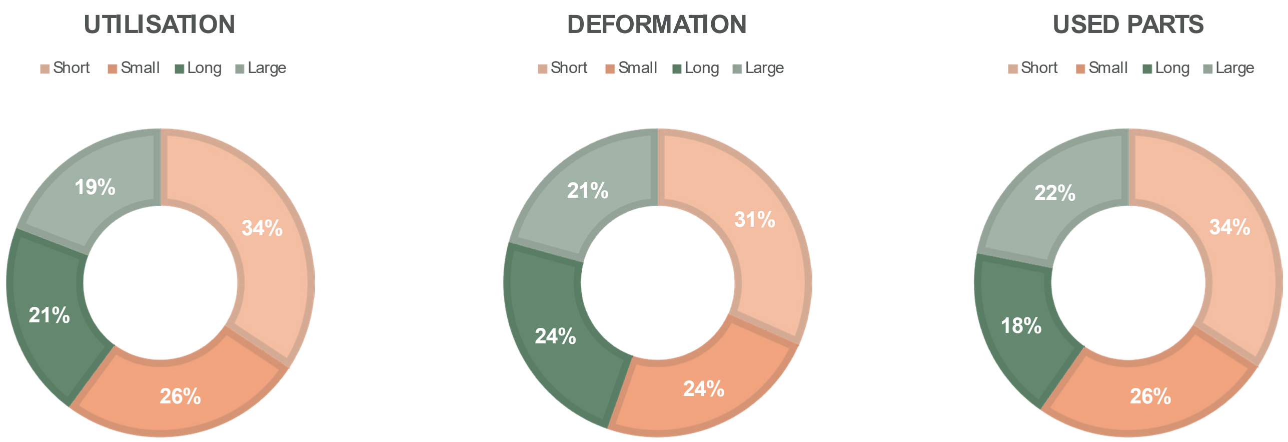


Figure 8.20:Size influence of the pieces on the structural performance

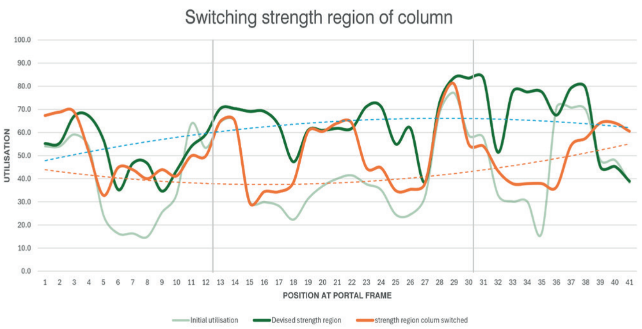


Figure 8.21: Utilisation lines from structural system for switched strength region of column.

the same stock, loads and dimensions were used as described in paragraph 8.1. Comparing results shows an increase of 20.9% of initial utilisation and an increase of 34.2% in deflection. Swapping the high and low strength regions of the beam result in an initial utilisation of 121.9%, proving that this region is defined well in the optimisation. Swapping the strength regions for the columns from high strength grade on the sides to the centre shows an initial utilisation of 75.3% and deflection of 15.4mm. Both caused a small decrease in comparison to the set regions in chapter 7. Running the optimisation resulted in the removal of 63 parts and a total mass reduction of 64kg to a total of 408kg. This result is worse compared to the initial strength region in which 85 parts could be removed. Figure 8.21 shows the utilisation of the structure compared with the default strength region. It shows that the columns are not utilised efficiently. This could be due to a larger share of high strength pieces with a low utilisation which will be removed first. Removing pieces in the middle first can cause connectivity issues, and a premature end of the loop. The default set strength regions are providing the best results.

Strength grade distribution

This section will investigate the influence of the strength distribution on the final optimised structure. Figure 8.22 shows two compositions which are both generated with identical methods and variables but different stocks. Composition 1 is the composition from section 8.1. The strength grades for this varying composition were manually modified to C24 the see the effect. Stronger wood should result in a better final optimised structure. This is however not the case for this composition. A similar amount of pieces could be removed from both structures. This result was unexpected as stronger wood should result in a more optimal structure. Moreover, the structure showed a low overall utilisation plotted in figure 8.24. A second composition was tested with a the same stock, but a

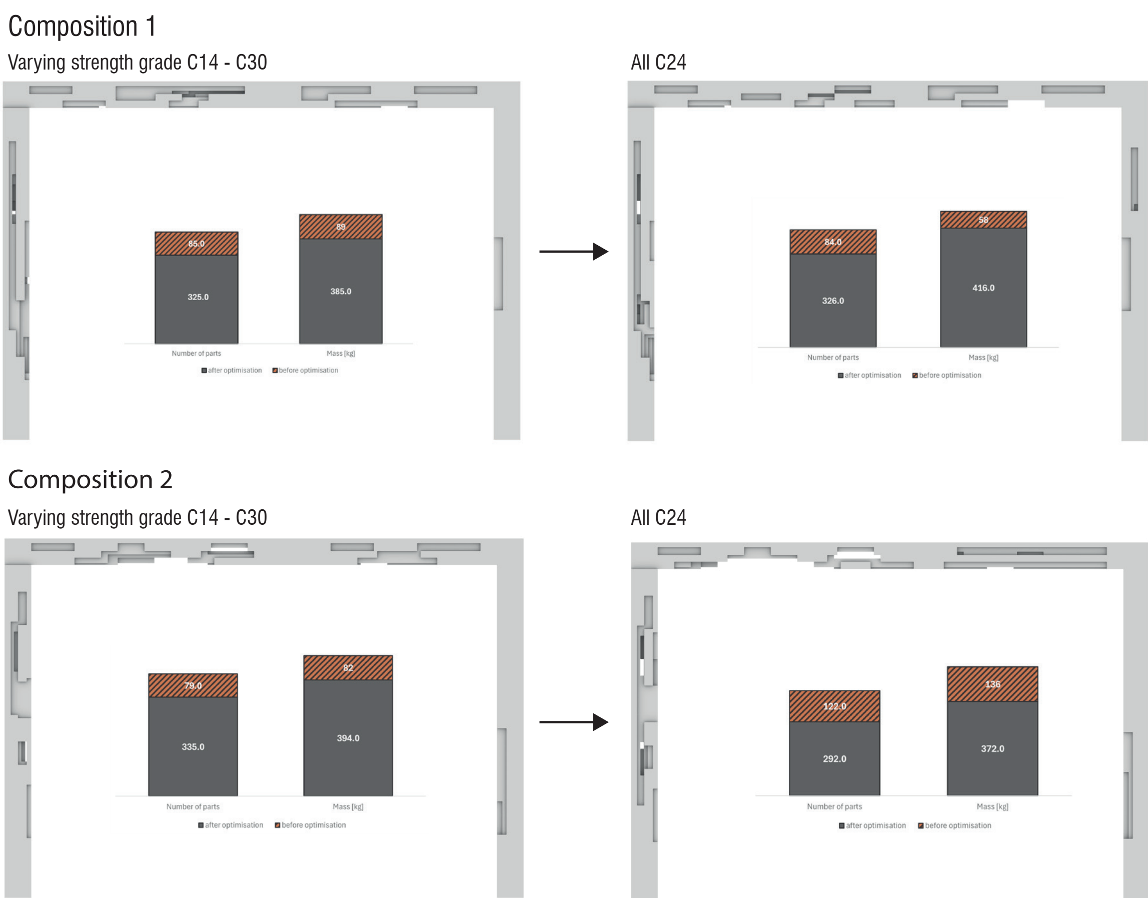


Figure 8.22: Comparison on optimisation results of two different compositions

different random sample to gain more insight in the program's working. The second composition with varying strength grades was able to remove 97 parts. With a manually modified strength grade of all parts to C24, it resulted in the removal of 122 parts with a final mass of 372kg. Figure 8.22 shows for both the varying strength grade and the uniform strength grade a similar contour, but due to higher quality of wood the structure could be optimised further. This composition shows the expected behaviour while using almost exactly the same amount of long and short pieces and the same percentage of strength grades and lengths as can be seen in figure 8.24. This shows that the final result highly depends on the composition rather than on the strength grade of the wood. It is likely that the place of the parts in the composition plays a role in this behaviour as this is the only variable that is different between the compositions. A potential explanation for the behaviour seen in composition 1 could be that the composition has overlapping parts in the x-direction on critical places for the optimisation, where the moment is zero for example. The program is now built that parts cannot be joint if there is an overlap of less than 100mm. However, it can still place parts that do not have this overlap. Initially such overlap is compensated by other parts but if a large amount of pieces is removed it could cause major connectivity issues as depicted in figure 8.25. Composition 1 initially has a good load path. When removing one piece only one load path remains making the structure weak and forcing all the loads thorough a small amount of pieces in contrast to a composition that does have enough overlap to make connections everywhere. Due to time constraints this flaw could not be developed further and is a limitation of the program. Further development can make the program far more stable by accounting for this behaviour. As of now some compositions can prove to be not efficient, therefore it could be valuable to try different compositions by altering loaded stock when running the program.

Strength grade	Used parts				
	Element A	Element B	Element C	total	percentage
C14	41	59	27	**127**	*31%*
C18	36	36	26	**98**	*24%*
C24	54	61	43	**158**	*38%*
C30	17	9	5	**31**	*7%*
total	**148**	**165**	**101**	**414**	*100%*

	Used parts			
Long	28	31	38	97
Medium long	74	51	31	156
Medium long	26	40	22	88
Short	37	26	10	73
total				414

Figure 8.23: Used parts in composition 2

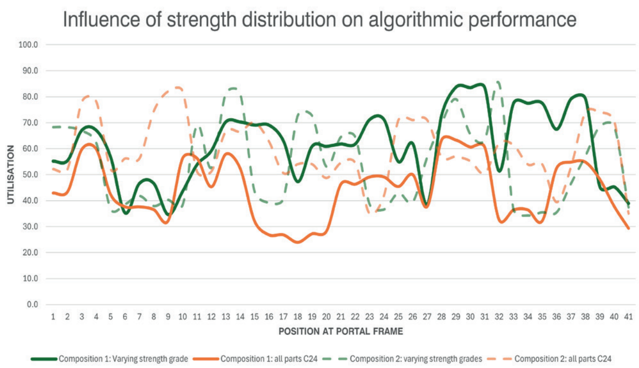


Figure 8.24: Utilisation values for each tested composition

The portal frame of the second composition is shown more detailed in figure 8.26. This reaches a better global optimal solution in comparison with the first composition seen in section 8.1. The cross-sections framed in the black rectangle were selected for a 1:2 mock-up as this is the most critical part of the structure. The mock-up will be used to check if the algorithm creates stiff structures and to make sure the parts can be connected. The mock-up is shown and discussed in 8.6.

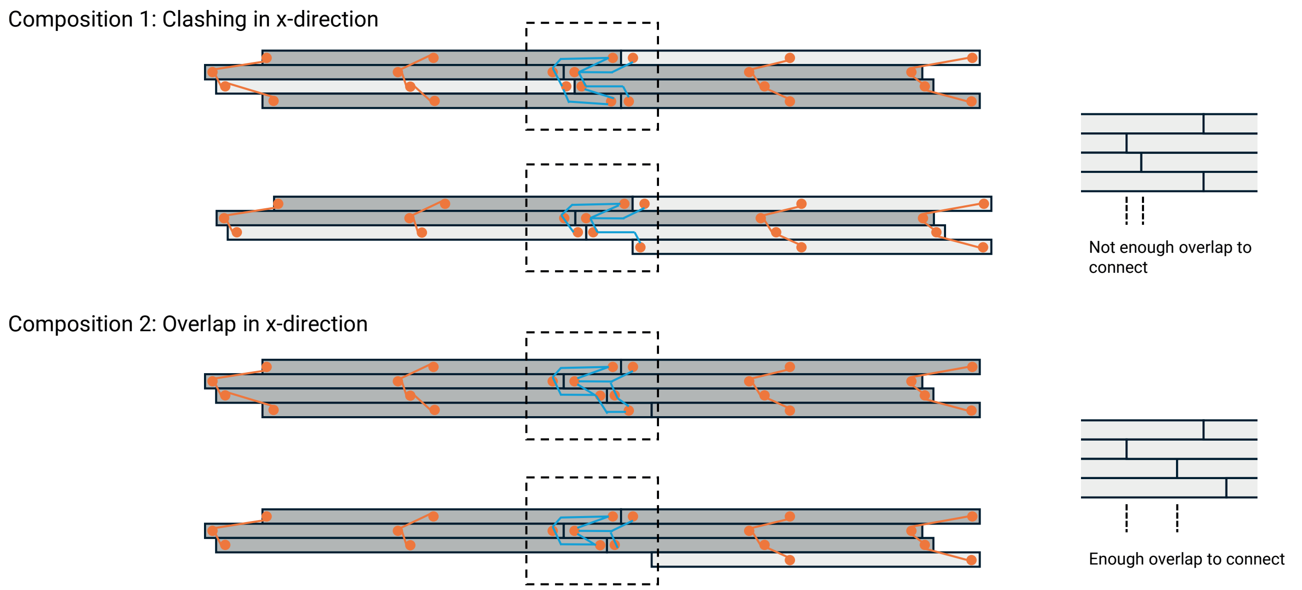
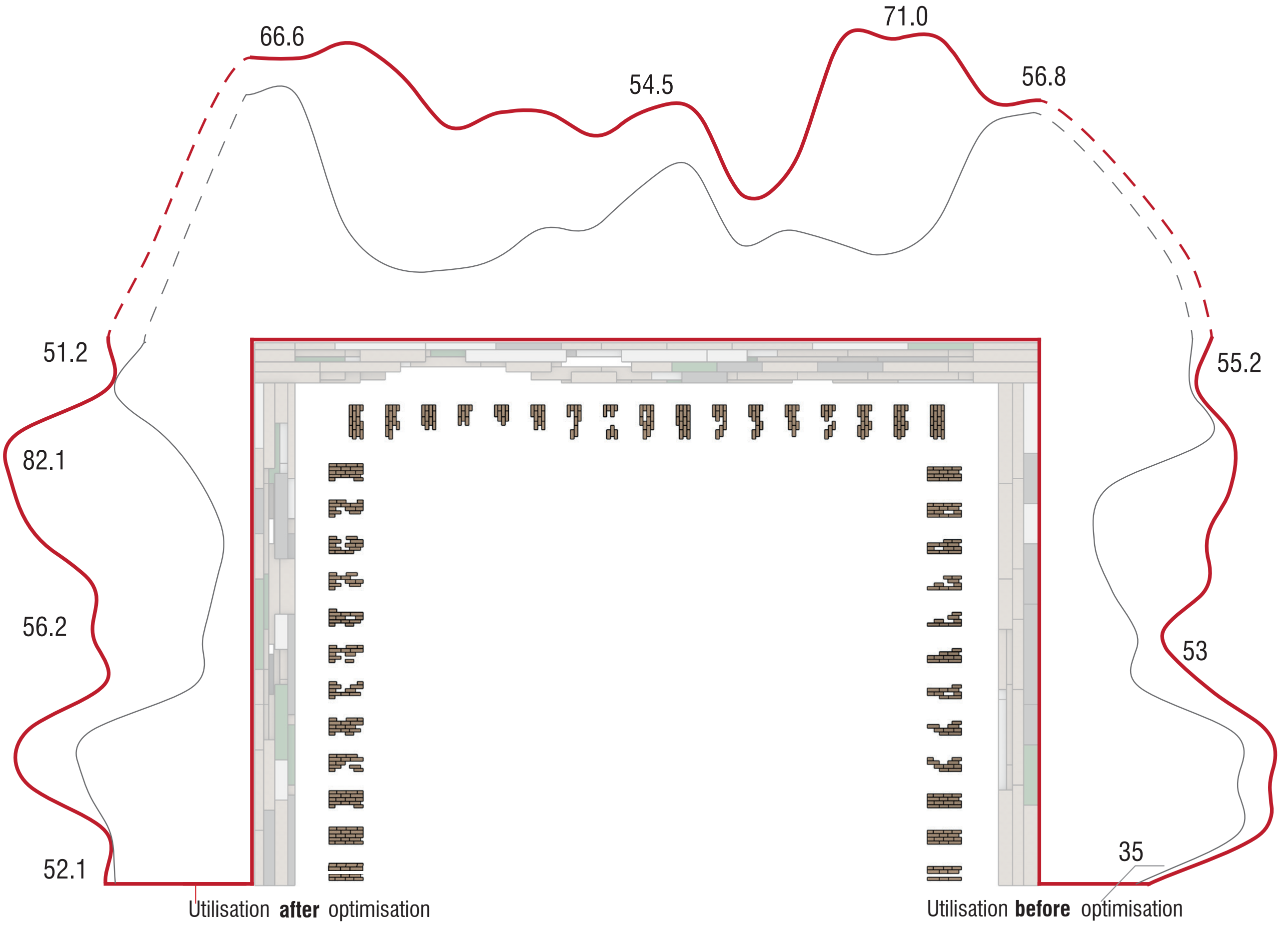


Figure 8.25: Top view of aggregation visualising the current flaw concerning connectivity issues in the model causing 'accidental' good or bad compositions



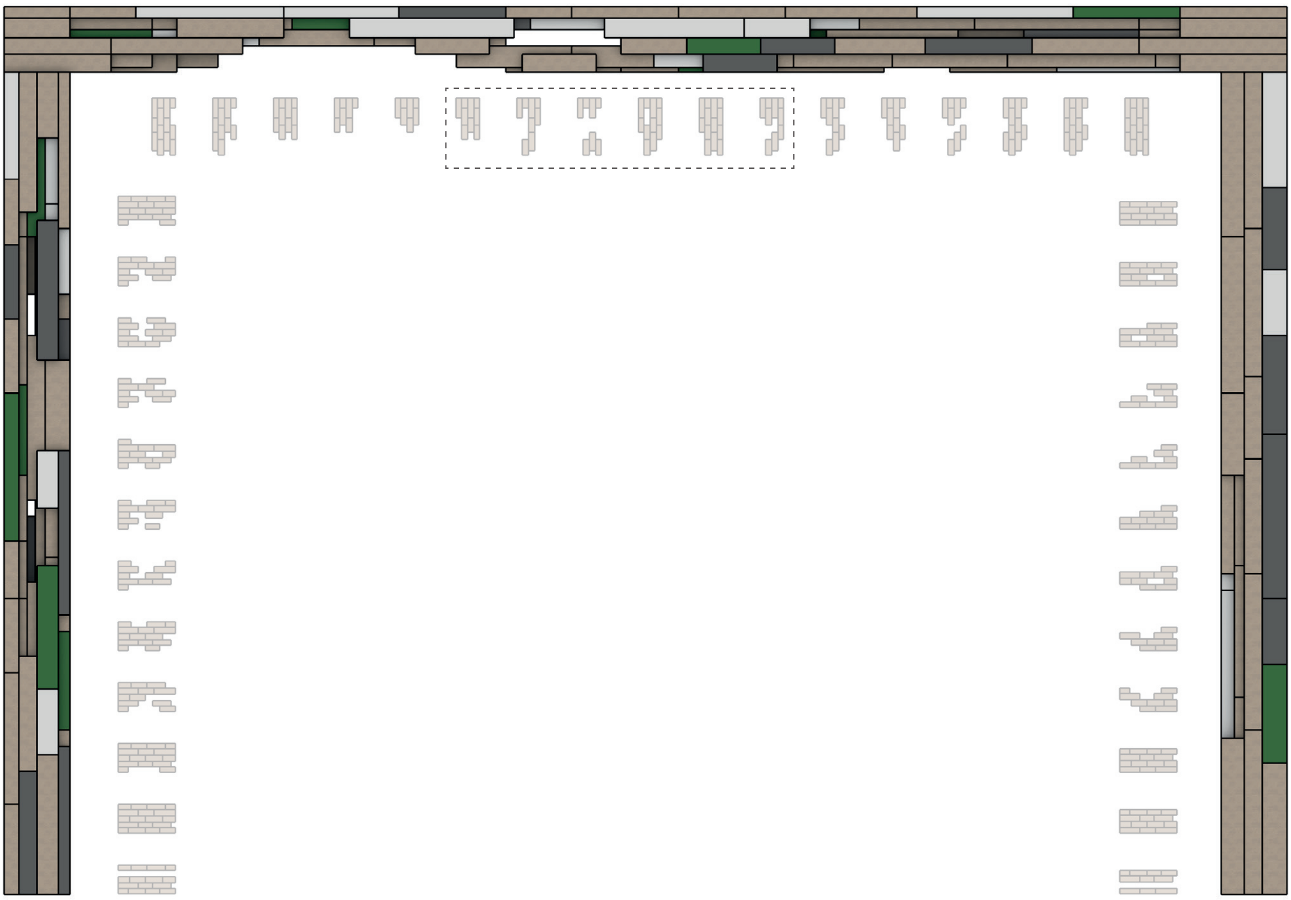


Figure 8.26: Utilisation values for each tested composition

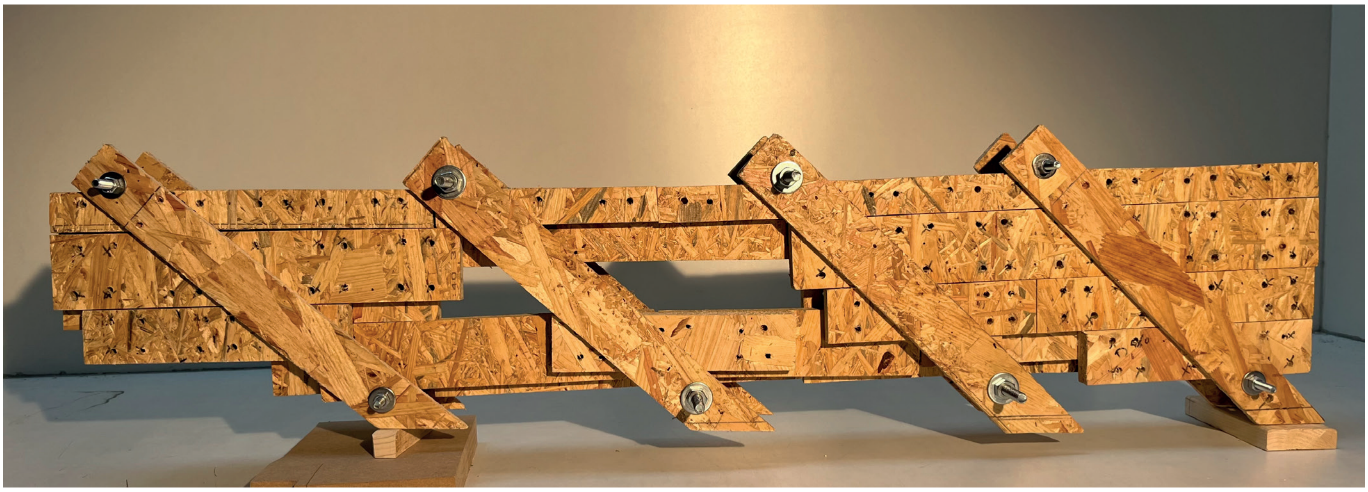


Figure 8.27: Front view of build mock-up

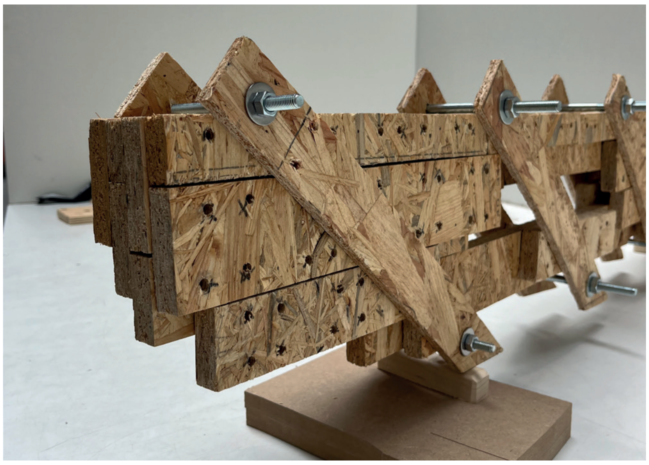


Figure 8.28: Mock-up highlighting clamping system

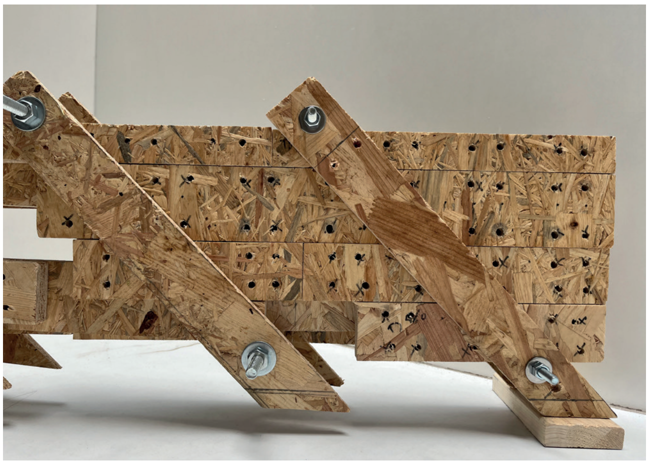


Figure 8.29: Mock-up highlighting drilling pattern

Figure 8.30: Side view of build mock-up

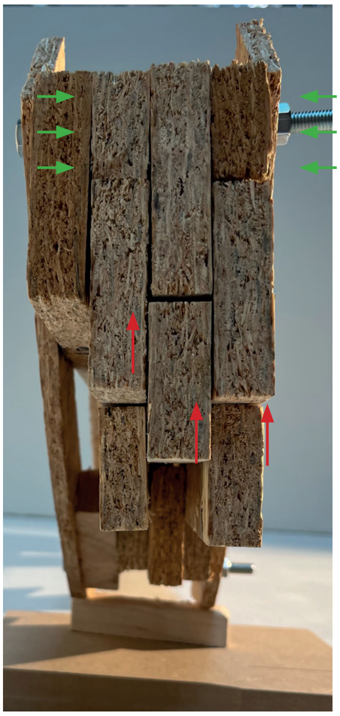


Figure 8.31: Side view of build mock-up

8.6 Mock-up

Building the mock-up was good exercise to check the devised system in practise. The built cross-section entails six different sectional heights ranging from 70 - 140mm and 10 varying lengths between 350 - 1050mm.The model is built in scale 1:2 with OSB of 20mm and dowels of 6mm. The external links provide a lot of stiffness and make the beam feel as a composite piece. Without the links the beams had a lot of rotational freedom, as each dowel is essentially a small hinge. Now in total around 1/3 of the dowels that would have to be used in real are used in the model, which was in this case 100 dowels. It could be due to the amount of dowels that there was still as a lot of rotational freedom without the links or due to the oversized holes from double drilling.

On the image on the left it is visible that with the clamps not all pieces are compressed, this could be caused by due to low bending stiffness of the thin sheeted OSB links and downscaling of the project. Bolts cannot be tightened to much, while this results is less likely with more stiff links, it to exaggerate the effect, a lower compressive force on the middle parts. Not all pieces will be compressed and therefore 100%

Figure 8.32: Close up of build section highlighting deformed clamp and the spacing block

Figure 8.33: Close up of build section highlighting optimised structure and versatile composition

Figure 8.34: close up on optimised parts showcasing the link and spacing block

composite behaviour cannot be expected.

Within the system there is little to no room for tolerances. Dowel holes have to line up exactly right which is hard by hand, especially considering the use of salvaged wood which could be deformed a little already. For the model a mold was used so each dowel hole would line up correctly. Robotic assembly and drilling is essential to make this system effective. Further research should be conducted to investigate a suitable assembly process in form of digital assembly with robots or other techniques to make this system less labour intensive and less sensitive to errors for manual drilling.

In total 4 external links were needed for the model which translates to one link each 500mm. Which is substantial. In the design phase a distance of 1000mm was intended. In retrospect, The links are adding a lot of material, it feels like the material saved by the optimisation is put back in the form of links. An increase in the minimum length of the pieces would substantially lower the amount of links enhancing the systems efficiency.

Unintentional bending is seen in the links due to downscaling. Plywood links of 22mm should provide better results and less deformation. The OSB used had a thin lacq layer which does not simulate the friction between the elements well but still resulted in a stiff structure. Overall the system works, the algorithm made a beam which has enough overlap to be connected and there are no real fragile points in the model.

8.7 Matching performance

In this section the algorithm is tested on performance considering the filling rate. The filling rate is defined as the number of pieces that can be placed without the need of cutting pieces. If all pieces can be placed without cutting, the algorithm has achieved a filling rate of 100%. The algorithm can either cut pieces for waste, cuts < than 300mm, or cut pieces for reuse, cuts > 300mm. The left over parts of the cut pieces for reuse can be put back into the stock to be used again. The minimum remaining length a piece that has to be cut is 300mm. If a row cannot be filled by only cutting one piece a second piece is cut. The algorithm is built so that it always cuts from the largest piece, maximizing future reuse potential of pieces an minimizing unnecessary pieces from being created. The stock is tested with different scenario encompassing scenarios with abundant inventory to limited stock. Filling rates and used parts are based on aggregations with the dimensions of the portal frame found in section 8.1.

Scenario 1

This scenario describes an abundant stock, tested twice with lengths varying from 300 - 1000mm and 500 - 1000mm. Lengths in the stock are dividable by 50 and 100mm. With this scenario the algorithm is always able to find combinations resulting in a filling rate of 100%.

Scenario 2

This scenario describes a limited stock, tested twice with lengths varying form 300 - 1000mm and 500 - 1000mm. Lengths in the stock are only dividable by 100mm. This time the algorithm requires cutting of a maximum of two parts, with a maximum cut of 200mm and thus both intended for waste.

Scenario 3

This scenarios describes both a limited and abundant stock, tested for lengths varying from 800 - 1000mm, only dividable by 100mm. The abundant stock required cutting of three parts, two of which are intended for waste and one for reuse. Limiting the stock the algorithm requires cutting 11 parts. The larger portion of the cut parts, seven pieces, are intended for reuse, four left over pieces are too small to be reused again and have to be discarded. The longest cut was 400mm and the shortest remaining piece had a length of 300mm and is within the set tolerance.

Scenario 4

The last test scenario describes an extreme case in which only pieces with lengths varying from 900 - 1000mm, dividable by both 50 and 100mm are selected. With abundant stock loaded, the algorithm requires cutting of 18 pieces. This means that in each row of each element parts have to be cut. 10 pieces are cut for waste and 8 parts can be put back into the stock. The smallest part in the aggregation is 600mm. Using a limited inventory requires cutting of 23 parts, reflecting almost two pieces in each row. In this case only 8 pieces are cut for waste and 15 pieces have left over lengths large enough to be reused and put back in the inventory. The smallest part in the aggregation is in this case 500mm. Figure 8.35 shows the overall results in a graph.

Concluding: The dynamic programming works very efficient. With a diverse and abundant stock it can with create aggregations which require no cutting of pieces, keeping the future reuse potential high and re-manufacturing costs low. When the inventory gets constrained more the algorithm is still able to create aggregations minimizing the amount of parts that have to be cut to maximum of 8%. The remaining length of pieces in the aggregation that were cut was always more than or equal to 300mm and the algorithm never cuts the pieces smaller. Overall, the algorithm is flexible enough to handle diverse scenario's considering the contents of the stock. It minimizes cutting and ensure that larger cuts can be put back in the inventory to be matched again.

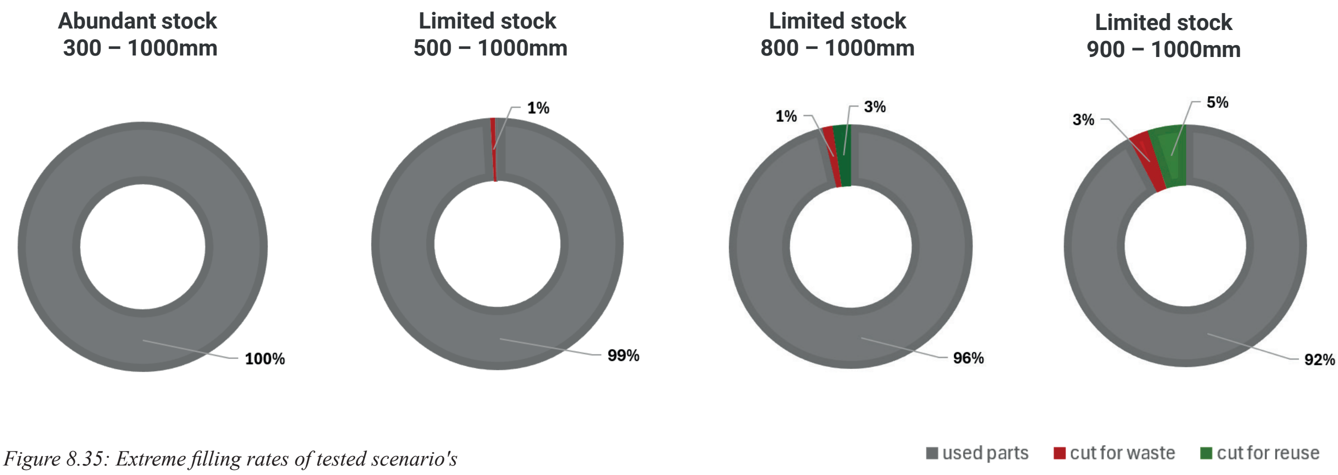


Figure 8.35: Extreme filling rates of tested scenario's

Figure 8.36: Buckling behaviour found in the algorithm, A). Optimised structure without accounting for buckling. B). Buckling factor of 1.1. C). Buckling factor of 1.28.

8.8 Buckling behaviour

The optimisation process is stress driven, meaning that pieces are removed based on the lowest stress-value, maximizing the utilisation of the remaining pieces. This method does not consider buckling, while this is a normative failure mechanism for columns. Columns are subjected to a normal force and a bending moment at the supports and connections. The normal force is usually negligible in low-rise constructions and therefore making the occurring stress not representative for the required cross-section. Karamba3D can account for buckling, however since the model is discrete it does not understand that the combined parts in the columns, together represent a full column. In smaller portal frames buckling did not seem to be an issue at first. But by generating more portal frames it was found that the issue is, as the quality of the portals, dependent on the composition.

Some compositions are not affected by this behaviour at all however, during the generation of a larger portal with larger cross-sections of the parts, 75x200mm, used in the beam element alone the optimisation program showed extreme buckling behaviour as illustrated in figure 8.34. An explanation for this behaviour could be that the large size of the cross-sections in the beam withhold this element from being optimised efficiently, removal of one large section would result in major connectivity issues. This causes the program to fixate on maximizing the utilisation in the columns, showing that buckling is not taken into account. Scenario A in figure 8.36 shows this effect. The column is mid span only affected by compression. Sole compression requires little cross-sectional area and causes low stresses in the wood. Bending due to compression is normative. A buckling factor was multiplied by the utilisation of the columns to create a more realistic composition. Scenario B shows the same column with a buckling factor of 1.1, but still seems to fragile. A buckling factor of 1.28, scenario C, shows a more stable composition and realistic result. In some iterations of smaller portals without large sections this effect was also noticeable but less evident. A buckling factor of 1.1 was found to be better suited for smaller spans, meaning it is dependent on the size. Further work should implement a more realistic buckling check within the program in order to generate viable results. Currently the designer has to manually adjust the buckling factor sometimes is unrealistic slender structures are being generated without the loop exiting.

For the final geometry of the optimised structure in section 8.6 the weakest cross-section is checked on buckling to evaluate the algorithms performance. Compression and buckling are can be integrated in a combined check for timber columns. The check encompasses reducing the design strength of the column by a the instability factor k_c which compensates for buckling and is expressed as:

$$\frac{\sigma_{c,0,d}}{k_c \cdot f_{c,0,d}} \leq 1{,}0 \qquad eq.8.1$$

The stress in the column is determined by:

$$\sigma_{c0d} = \frac{N_{Ed}}{b * h} \qquad eq.8.2$$

The properties of timber are influenced by many factors. The largest factor is the duration of the load. Loads that are permanent have to be checked with a lower design strength. A reduction in the compressive strength has to be made per load duration using formula:

$$f_{c0d} = f_{c0k} * \frac{k_{mod}}{\gamma_m} * k_h \qquad eq.8.3$$

To calculate the instability factor k_c, several values must first be found, starting with the rotational stiffness of the column at its weakest axis:

$$k_r = \frac{3EI}{l_A} \qquad eq.8.4$$

The effective buckling length can be calculated with:

$$l_{ef} = h * \sqrt{\frac{\pi^2 * E * I}{h * k_r}} \qquad eq.8.4$$

The radius of inertia is derived from the inertia and the surface area of the column:

$$i = \sqrt{\frac{I}{A}} \qquad eq.8.5$$

The slenderness of the column can be calculated using expression:

$$\lambda = \frac{l_{ef}}{i} \qquad eq.8.6$$

The normative slenderness from direction y or x should be used in further calculations. This allows the relative slenderness to be determined using expression:

$$\lambda_{rel} = \frac{\lambda}{\pi} * \sqrt{\frac{f_{c0k}}{E_{0.0,05}}} \qquad eq.8.7$$

The factor k is determined using formula:

$$k = 0{,}5 * \left(1 + \beta_c * (\lambda_{rel} - 0{,}3) + \lambda_{rel}{}^2\right) \qquad eq.8.8$$

B_c is for sawn timber as default set on 0.2. With all unknown factors solved, the factor k_c can be calculated and the sections checked, see verification below.

$$k_c = \frac{1}{k + \sqrt{k^2 - \lambda_{rel}{}^2}} \qquad eq.8.9$$

The minimum I_{yy} and I_{xx} of the weakest cross-section of the column are respectively 53.943.663mm4 and 6.221.943mm4. The normal force is maximum 24kN and all the timber parts have strength class C24. Without accounting for the reduction factor K_{mod} and γ_m, using the design strength, the column is in y direction sufficiently stiff and has a unity check of 66%. In x-direction the column would buckle and has an unity check on buckling of 563%. Secondary beams will have to provide additional stiffness, reducing the buckling length of the column. With secondary beams every two meters, the column can comply in x-direction to an unity check of 91%. The external plywood links will also provide more resistance to resist buckling and torsional deformation but these are not taken into account.

This concludes that the added buckling factor of for some composition is not necessary. The designer currently has to implement a factor if the final geometry becomes unrealistic slender. The buckling now is dependent on the quality of the composition. Further development in the program can helping in resulting a more stable program in which the designer has adjust less variables. The method above to check for buckling can be integrated into the program to modify the utilisation value of the columns and generate better results.

8.9 Global size influence

This section evaluates the optimisation process when using a very exaggerated large cross-section and a low initial utilisation. The hypothesis is that the program will create a tapered portal frame, removing as much weight mid-span and leaving as much material as possible at the supports. Using an exaggerated larger cross-section can aid in creating a bolted moment connection with a large enough rotational stiffness between the horizontal and vertical members. Bolted connections require a large section from comply to the minimum edge spacings and to achieve a high rotational stiffness.

Figure 8.37 shows the result with identical span and loads as described in section 8.1. The cross-sectional height is set on 600mm. The figure shows that the optimisation process complies partly to the pre-stated hypothesis. Most parts are removed from the mid-section of the beam however, the portal frame is not tapered as expected. The cross-sectional height of 600mm is still present along the full portal frame, reducing the free space unnecessary as a portal in section 8.1 and 8.5 with a cross-sectional height of 400mm can also comply. The maximum initial utilisation was 31% with a total of 696 parts and a mass of 846 kg. Due

Figure 8.37: Optimised geometry with exaggerated large cross-section using varying strength grades C14 - C30.

Figure 8.38: Optimised geometry with exaggerated large cross-section using varying strength grades C24.

to optimisation 264 pieces were able to be removed, reducing the mass to 554kg. In retrospect with the portal frame of 8.1, this optimised geometry is 169 kg more heavy. This concludes that using exaggerated large cross-section doe not result in a global optimum after optimisation. It is therefore recommended that the initial dimensions of the cross-section are already optimised in range of 70 - 80% of the utilisation.

Figure 8.38 shows the same scenario as aforementioned but with a manually modified strength grade of C24 for all pieces. Here the program does comply to the expected hypothesis. At mid-span the beam is reduced to a height of around 200mm and more material is kept at the moment connections. With this scenarios the program was able to remove 330 pieces and reducing the mass to 506kg. This shows that still the program is unable to achieve a global optimum which was seen in section 8.5. Concluding: the program is not suitable to use with exaggerated cross-sectional heights and therefore not suitable for a bolted moment connection.

Reflecting on the programs performance

Overall, most important the program works, and the method of design can be proven to be effective. There are still a few flaws and most results are highly dependent on the quality of the composition. The composition is currently not constraint form overlapping in the x-direction what is the probable main cause of the various results and quality of the aggregation. Constraining this will likely make the program more stable, allowing more detailed analysis, together with mechanical testing and implementing a buckling check will result in a much more robust and stable program. Currently the designer has to make multiple iterations until by 'accident' a composition with a high quality is created.

9. Case study

In this chapter the tool will be put into practise to bring all acquired results together in one project that serves as a final test for the overall design of the system as well as the performance of the algorithm.

9.1 Design assignment

The newly designed system bespoken in this thesis was given the name: ReSurge timber. To build the optimised timber elements ReSurge timber requires a new headquarters with storage area for the salvaged timber pieces and a workplace to were the elements can be assembled. The headquarters will, of course, be built using the new system to showcase its potential. The building is designed to have one large storage hall with joist spanning directly 10 meters across. The storage hall is 6.5 meters at its highest point and slopes to a minimal height of 5 meters, adequate for large and height storage shelf's. An oblong hall spanning 7 meters will be attached were the discrete elements can be assembled and possible working spaces can be installed. The brief encompasses a building envelope of 130m^2. The facade and external envelop will be designed by an external architectural firm and not included in this report. Figure 9.1 shows the design brief.

The portals are all generated with the bespoken algorithm and optimised to the acting forces, including a roof structure, PV-panels, insulation and secondary beams. The loads are included in annex B. The Python code used for the optimisation can be found in annex C.

The designer has found two local waste wood dealers that are selling parts and both have their on database in a SQL format. The designer first links both databases in one big stock and loads these in the algorithm. Than he can adjust the geometric parameters to fit the requirements and lets the algorithm generate the portals. In total 2.891 parts arts were used for the portals saving 3.220kg of wood from being otherwise discharged. It took some trail and error to acquire good and sound compositions. Buckling behaviour as described in section 8.8 was seen in many portals and the buckling factor had to be fit to tailor each case individually. Furthermore, Different seedings of stock were required to get to good quality compositions. The whole process took about three hours to create good compositions for all the eight portal frames.

The images on the following pages illustrate the final result of the aggregated portals, including a comparison of one discrete portal with an Glulam portal.

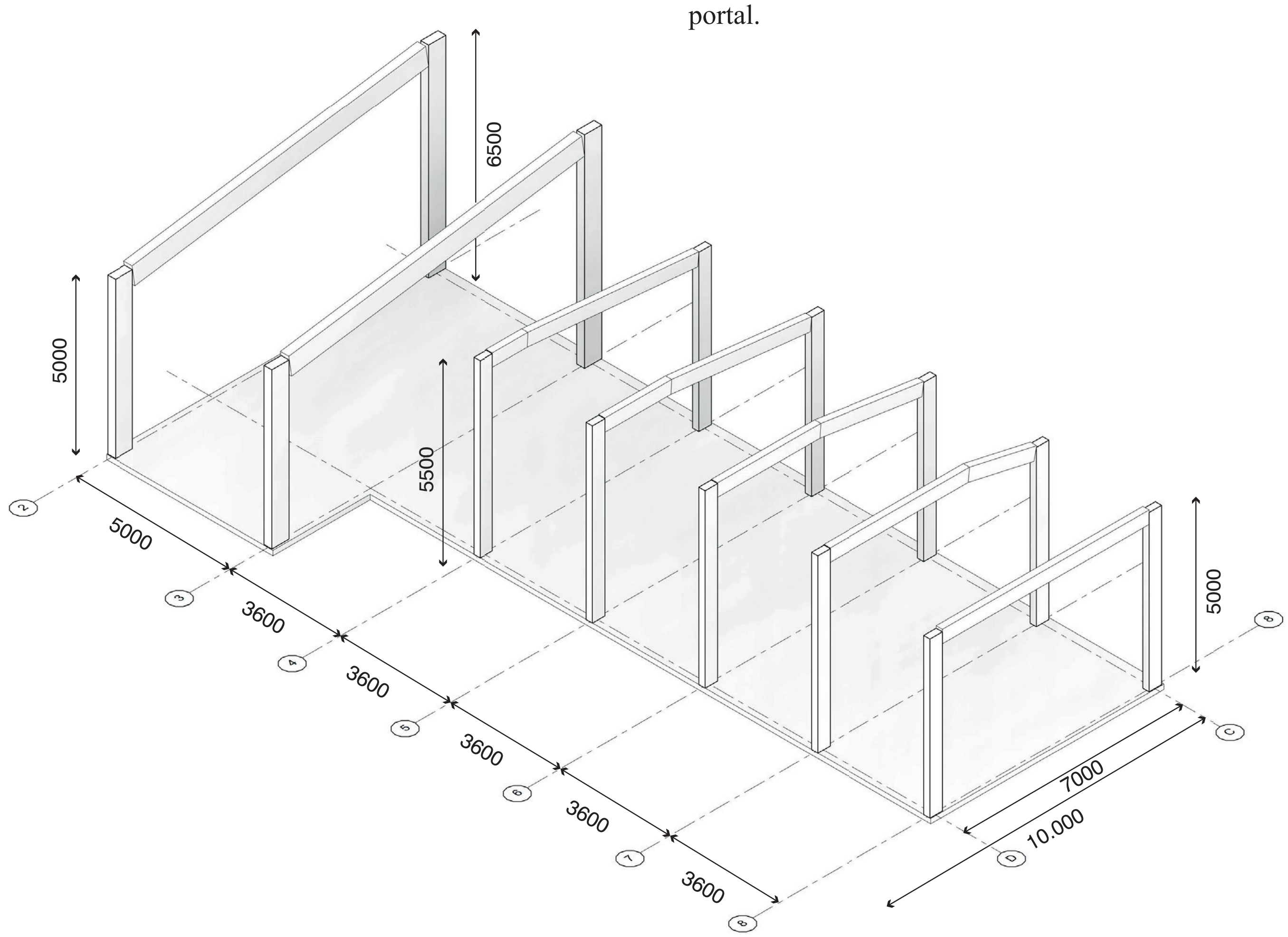


Figure 9.1: Design brief of ReSurge headquarters.

Figure 9.2: Optimised portals produced with the developed algorithm

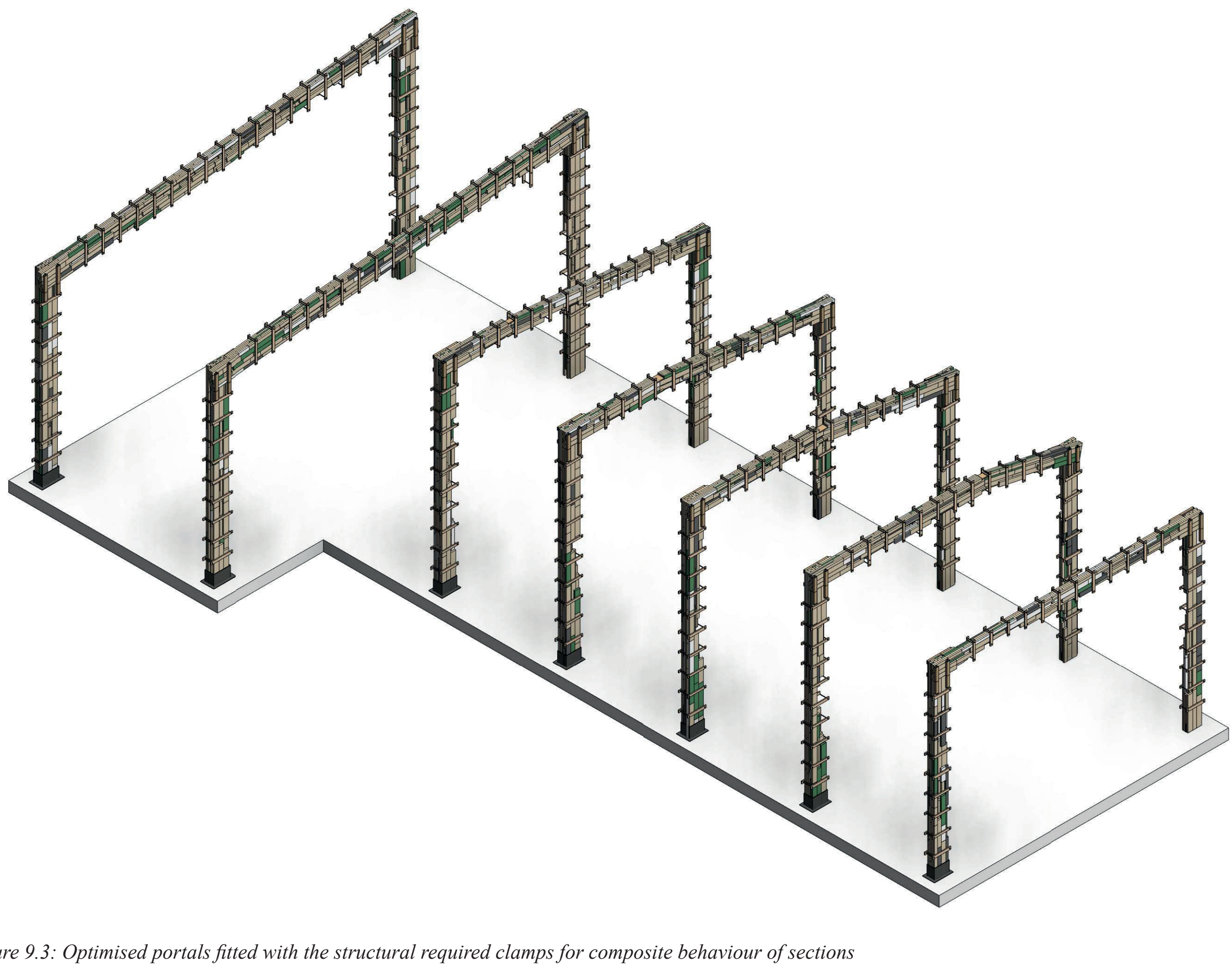


Figure 9.3: Optimised portals fitted with the structural required clamps for composite behaviour of sections

Figure 9.4: Close up on large portal frame highlighting the topological optimised geometry

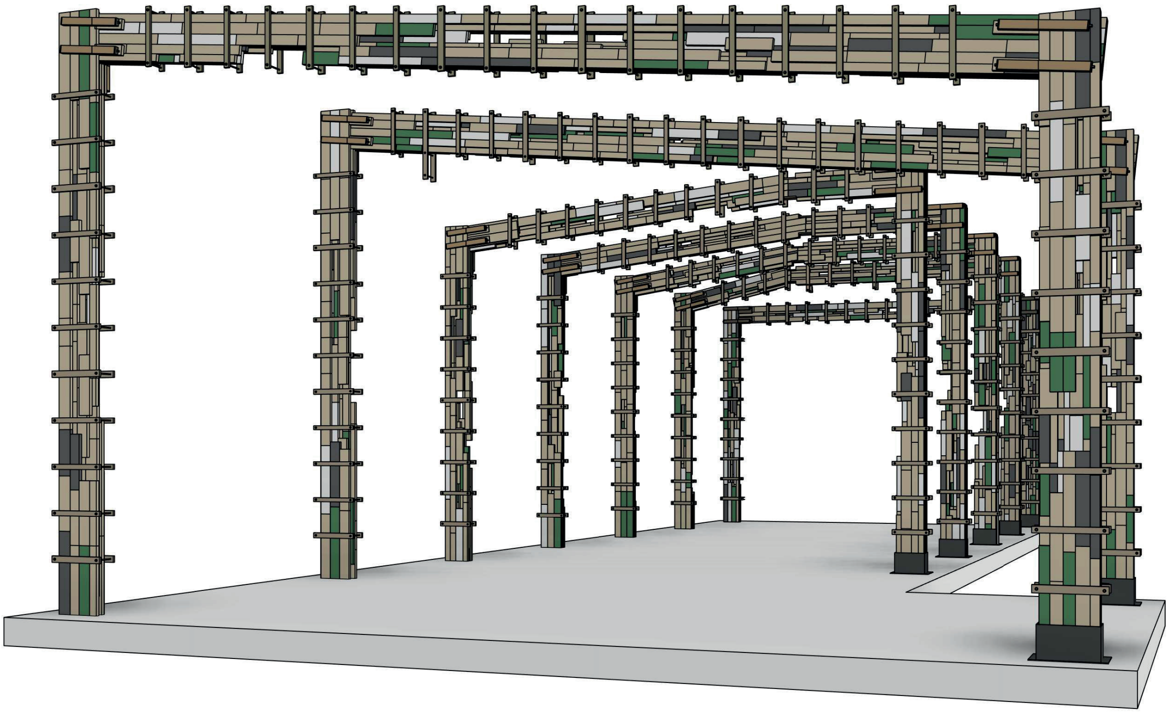


Figure 9.5: Close up on large portal frame highlighting full structural system, links are placed each 500mm

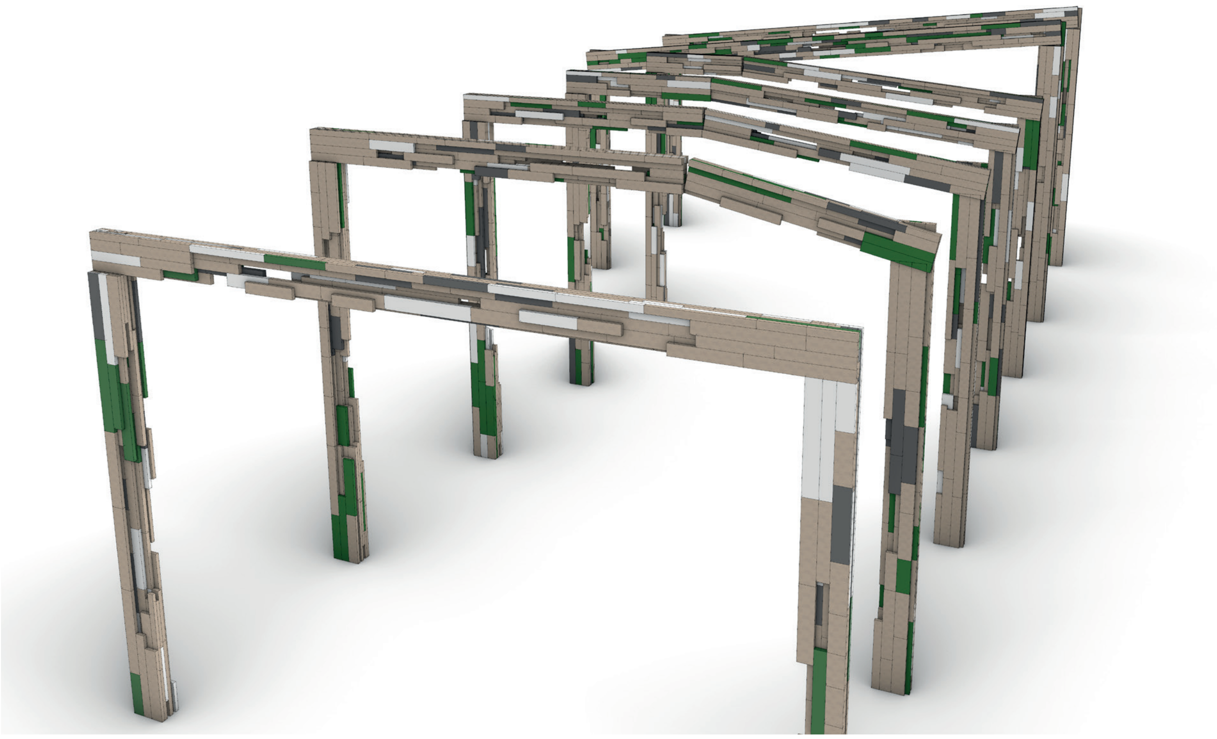


Figure 9.6: Close up on varying roof, highlighting the potential diversity the algorithm can provide

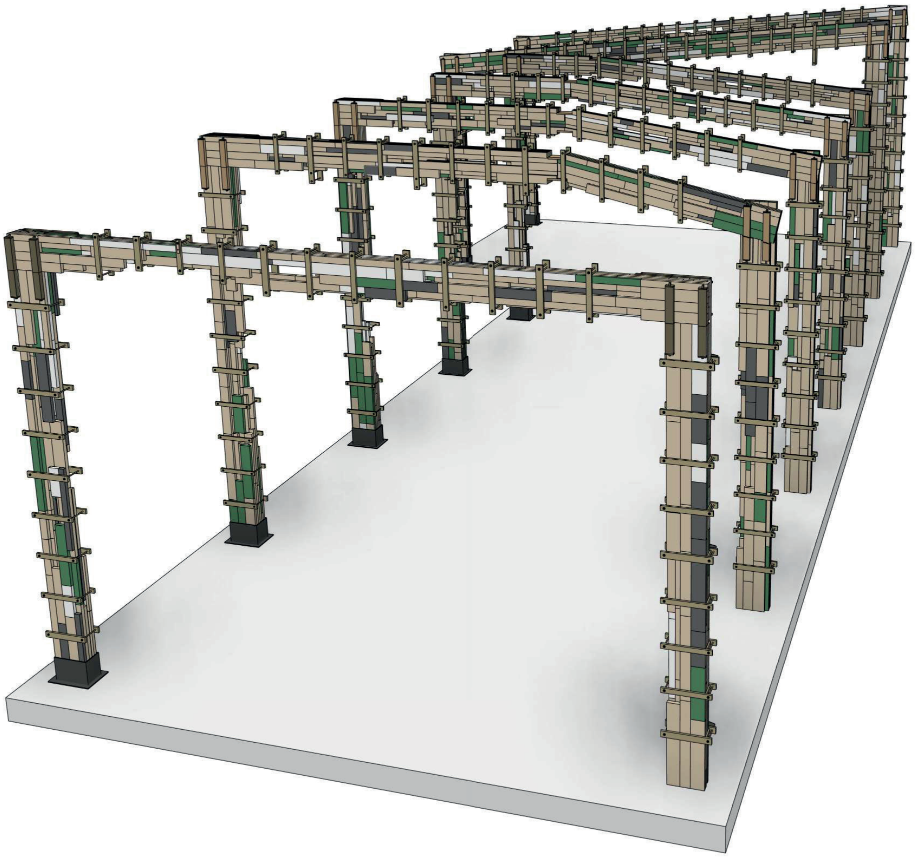


Figure 9.7: Close up on varying roof structure highlighting the full structural system

9.2 Relation to other building elements

Figure 9.8 and 9.9 are illustrating a small horizontal part of one of the optimised portal frames. Figure 9.8 illustrates all the parts involved to create and attach the plywood links. The links are 25mm thick and put with a plywood spacer can be inserted in an oversized milled slot within the links to ensure that the vertical links are always parallel to each other and share the exact same vertical offset from the main member. The plywood links can be fastened with steel bolts and tightened to enact a frictional force between the separate parts. Spacing blocks can be added between

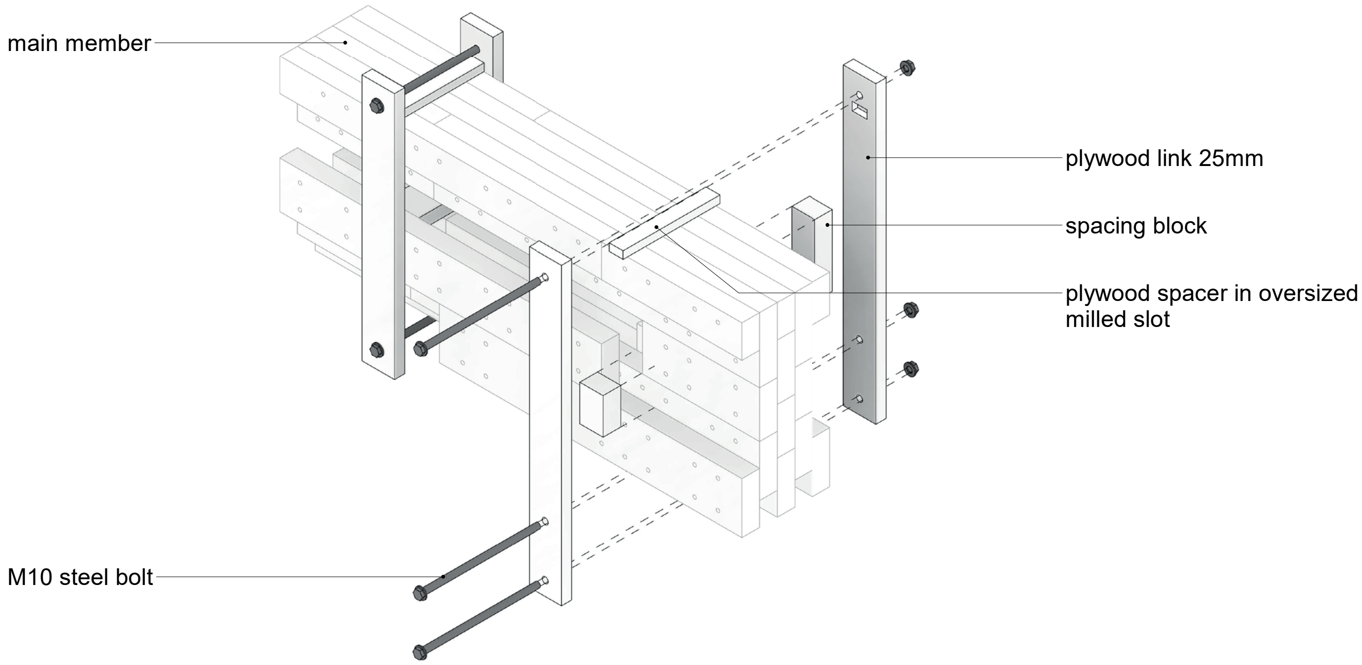


Figure 9.8: Detailed axonometry of clamping structure and its parts

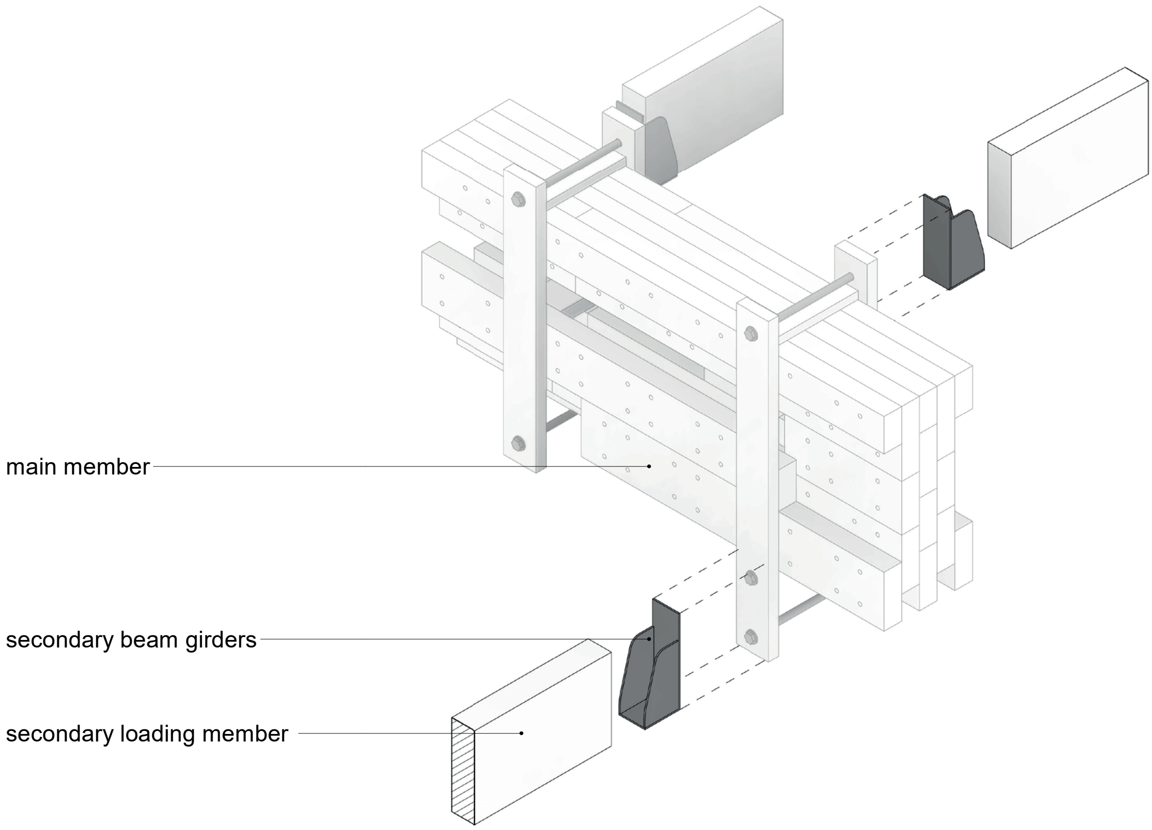


Figure 9.9: Detailed axonometry of secondary load carrying structure

the links and the main member if through optimisation some parts cannot be compressed. Figure 9.9 show a way to connect other building elements to the structural system. By extending the usage of the plywood links it can be ensured that the reclaimed parts used in the main member are not affected outside of the modular modifications. This ensures that the parts in the main member can be reused infinitely. Beam caries can be added to the plywood links in which a secondary timber structure can be placed. This secondary structure can be placed to reach above the links on which roof or facade elements can be attached. This system does restrict bay-sizes lager than 6 meters as this is the maximum length of sawn timber purlins. Further research should conclude if the plywood links are adequately strong to hold the secondary structure and the weight of the roof.

Looking into other extended usages of the links shows that they can also be used to attached cladding. This cladding can serve the purpose of protecting the main member from fire or to cover up the optimised structure for architectural purposes. Easy attachment systems can be designed to fix the cladding by simply sliding or plug the elements on the plywood links. A potential system is illustrated in figure 9.10.

Figure 9.11 and 9.12 are showcasing different attachment options for the facade. The plywood links are attached similarly as aforementioned however, to avoid vertical sliding dowels are also fitted to the links. This is possible as each link is placed at a distance of 500mm and the modular dowel pattern of the pieces is a system dividable by 50mm. The facade can be attached similarly as described earlier with either a secondary construction, beam carriers and prefabricated elements or by attached self-supporting facade elements to the links by l-shaped steel profiles. Both solutions are reducing the buckling length of the column. Other methods can also be used as long as there are stiff elements able to resist buckling at least every two meters. Figure 9.13 and 9.14 are showcasing the complete structural system with the added secondary load-bearing system.

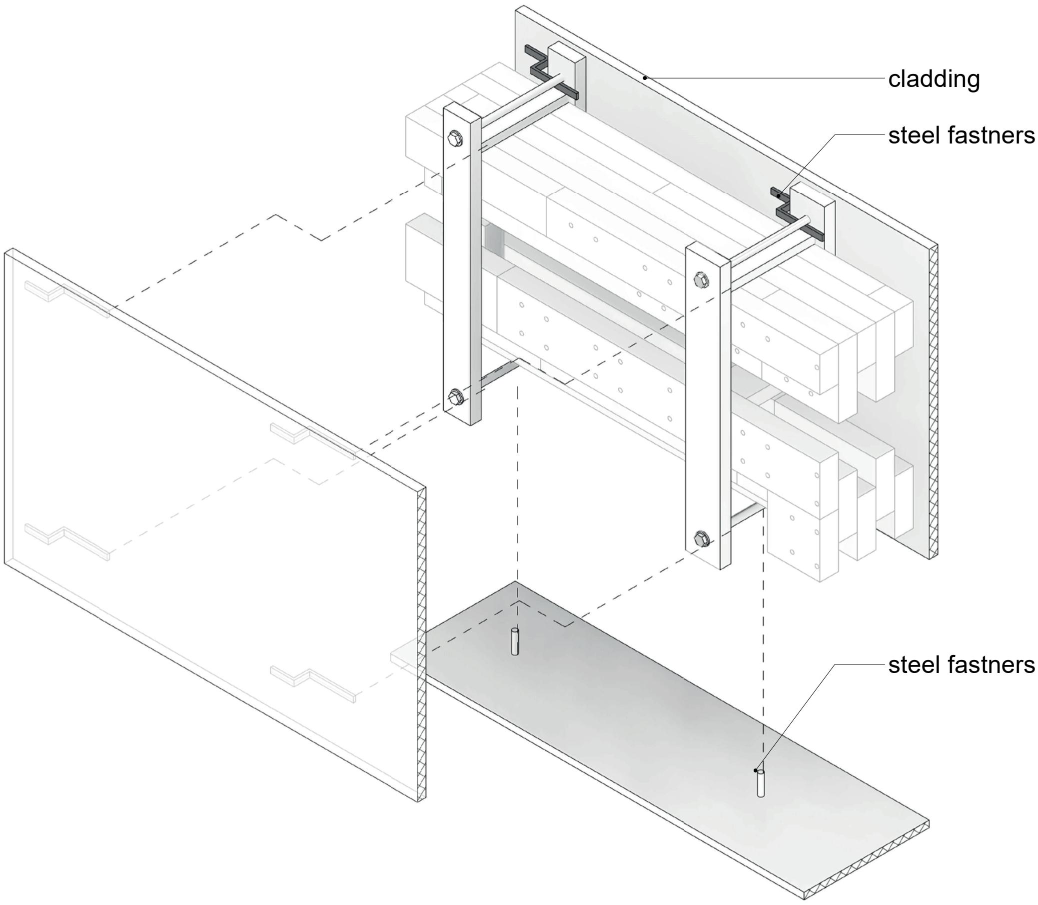


Figure 9.10: Detailed axonometry of potential covering system

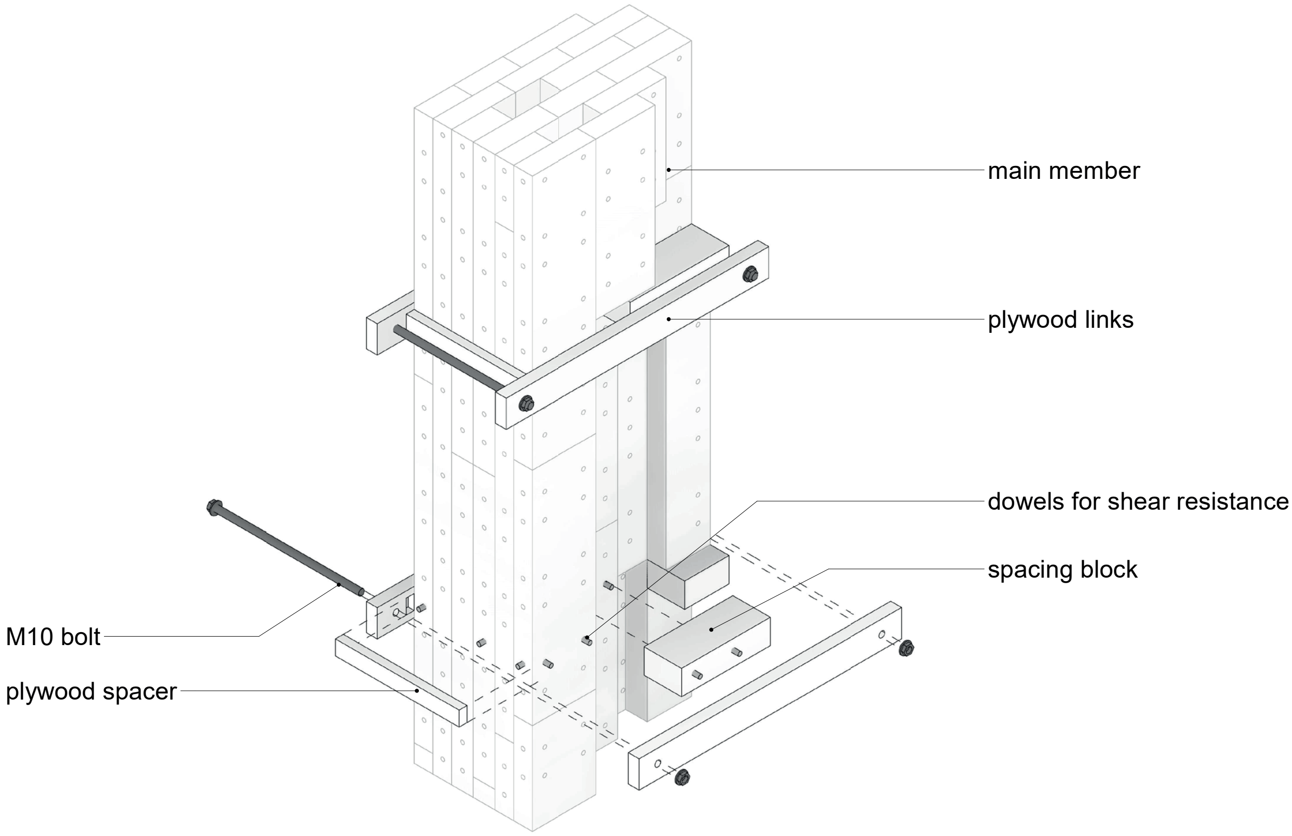


Figure 9.11: Detailed 3D axonometry of vertical clamps

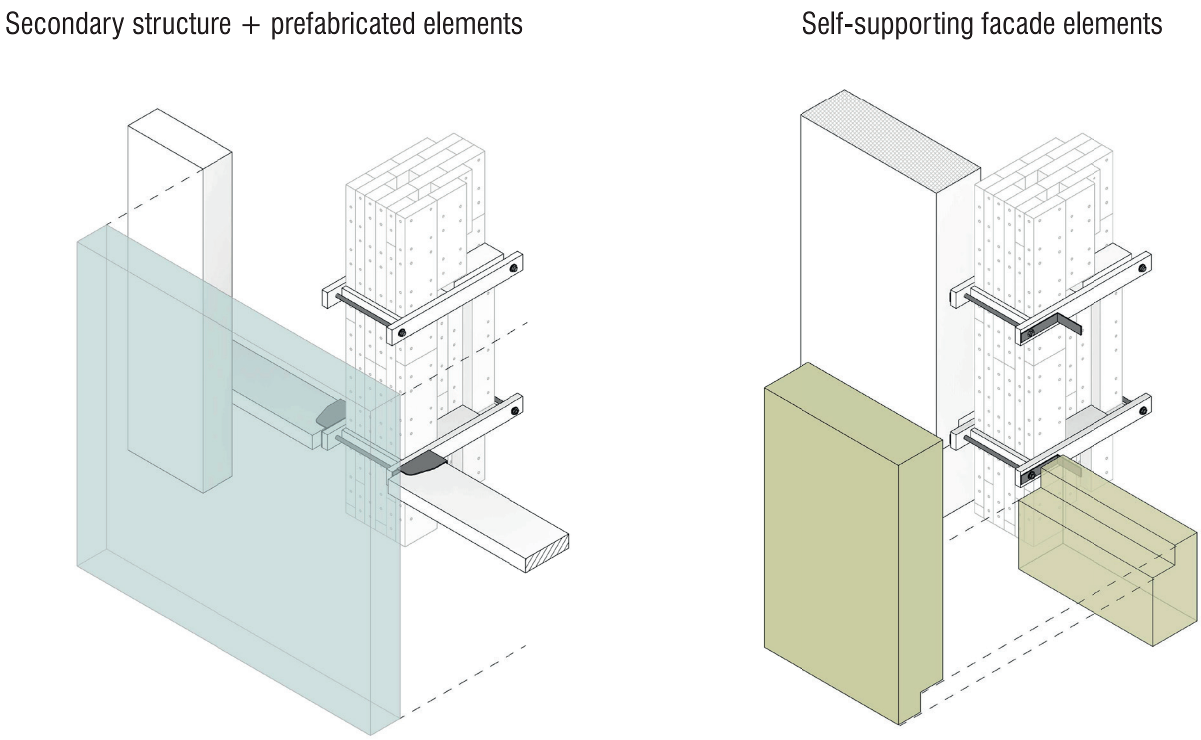


Figure 9.12: Various facade systems that can be combined in with the structural system, left secondary load-bearing structure with prefabricated panels, right self-supporting facade elements.

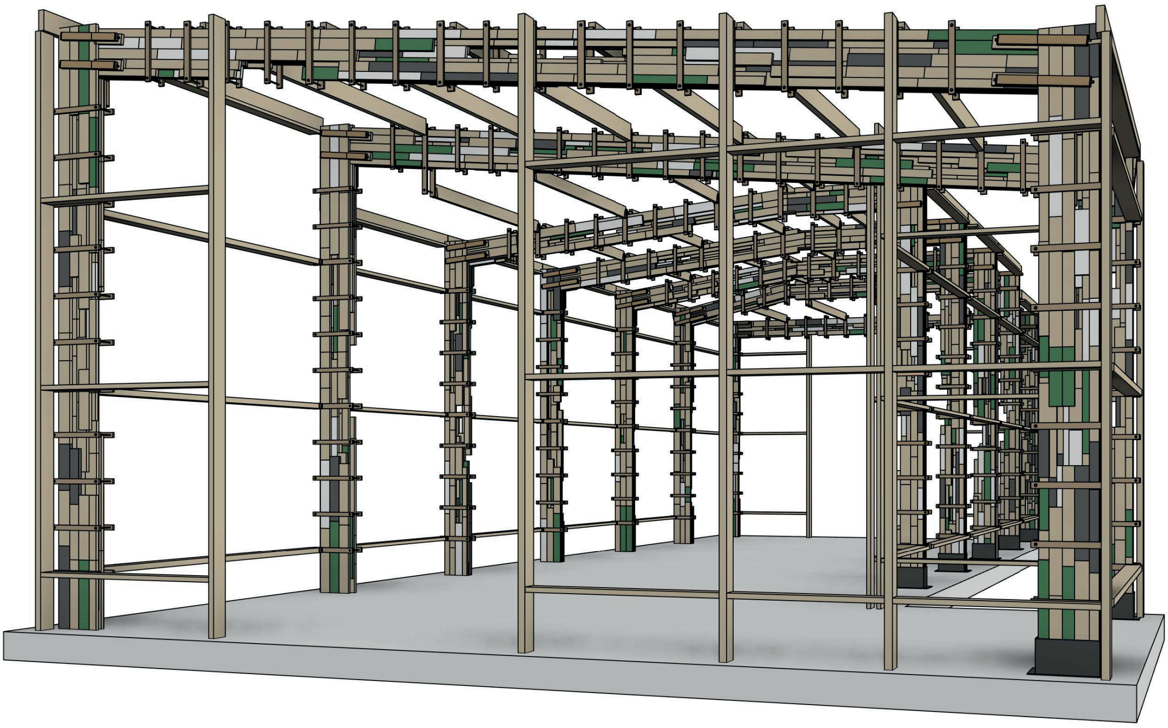


Figure 9.13: Close up on portal frames highlighting the full structural system and secondary structure

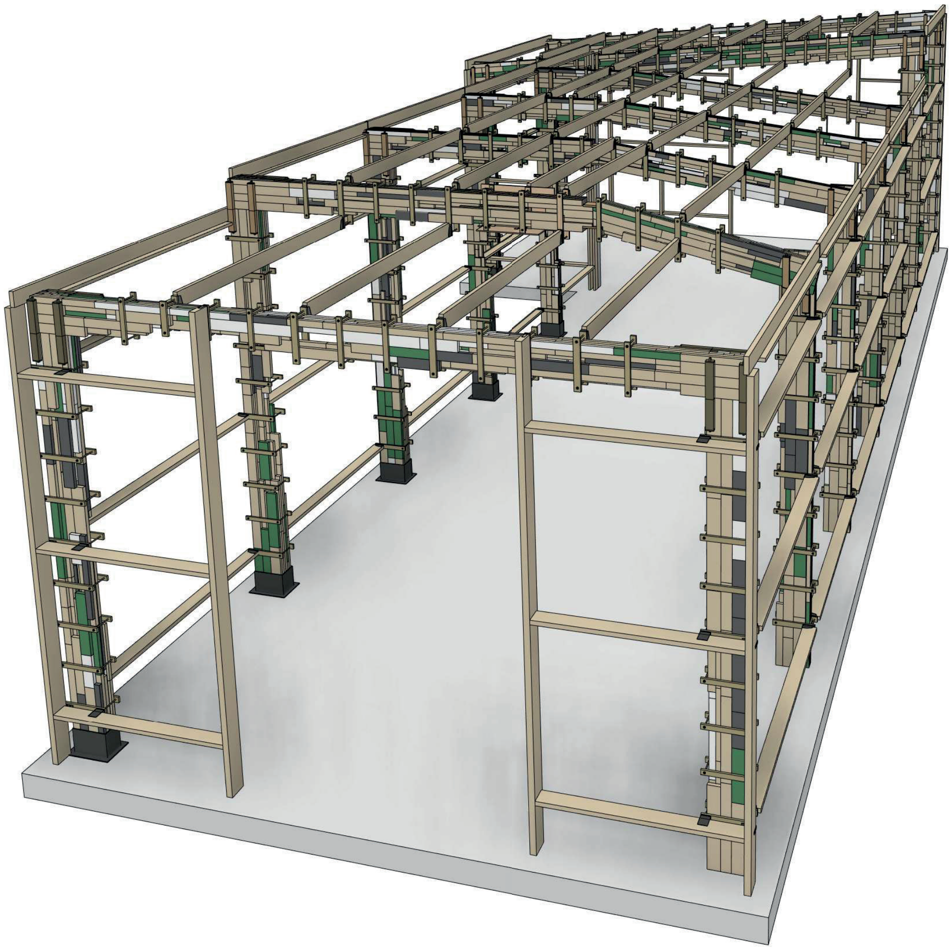


Figure 9.14: Close up on varying roof structure highlighting the full structural system and secondary structure

9.3 Detailed integral analysis

Figure 9.15 and 9.16 are illustrating the optimised portal analysed in chapter 8.5. The portal is in these figures illustrated with the structural plywood clamps and the moment connection. Every 500mm a clamp needs to be installed due to the usage of small pieces. This results in a total of 30 links. More clamps were required than initially thought, affecting the structure's architecture a lot. This is a limitation of the chosen system. Reducing the amount of shorter parts could aid in the reduction of the number of required links.

Before the optimisation process the portal had a mass of 502kg, after optimisation this mass could be reduced to 372kg. In total a reduction of 136kg. The additional weight of the plywood sheets are adding another 38kg. This results in a total weight of the structure of 410kg. The plywood sheets effectively adding approximately 1/3 of the removed material back to make to structure stable.

Putting the discrete portal in perspective with a more traditional system like Glulam shows, that when the same structure is analysed in Karamba3D a Glulam portal requires a cross-section of 160x350mm for a maximum utilisation of 85.9%. A portal with these dimensions approximates a mass of 437kg. Thus in retrospect the discrete system and a traditional system

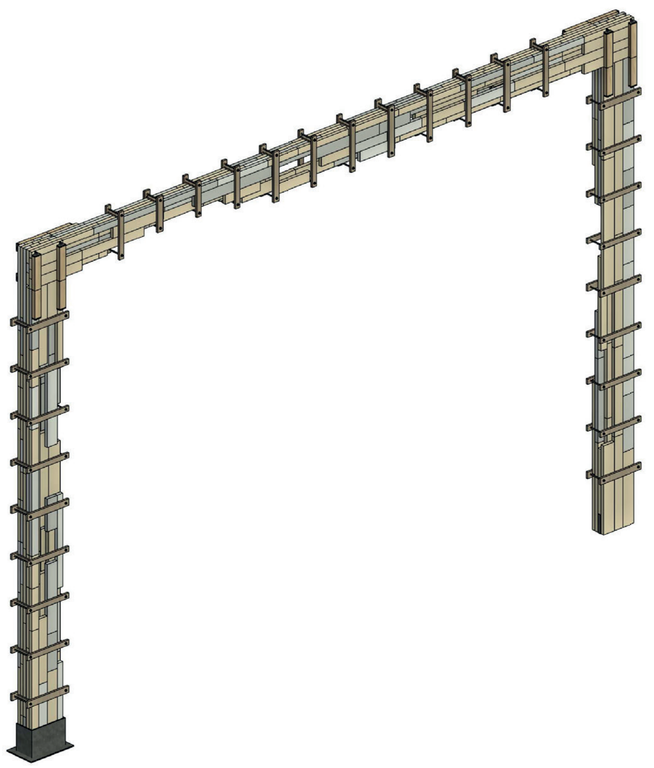


Figure 9.16: 3D view of optimised portal highlight the full structural system

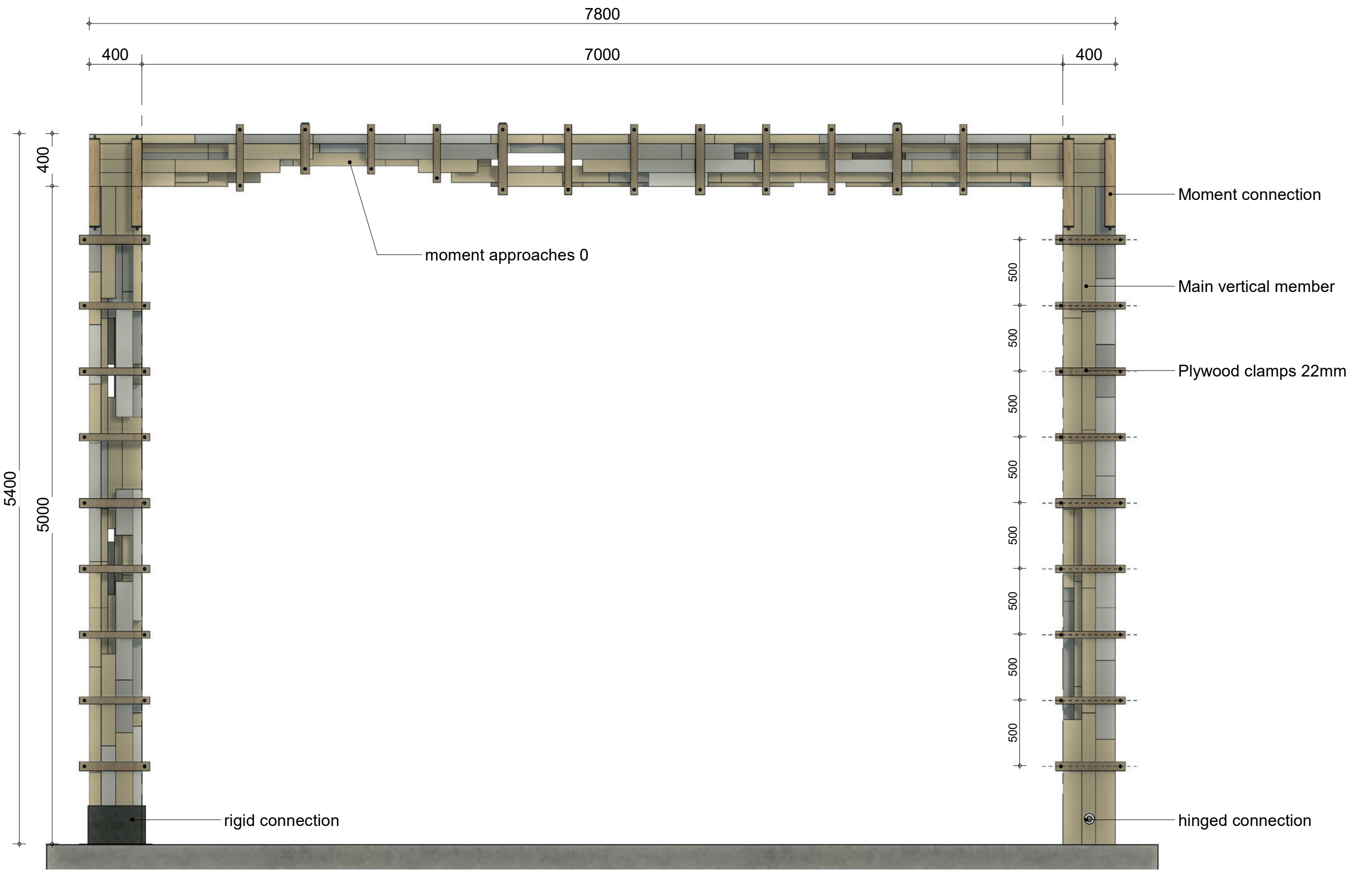


Figure 9.15: Front view of portal frame as also depicted in chapter 8.5 highlighting the full structural system

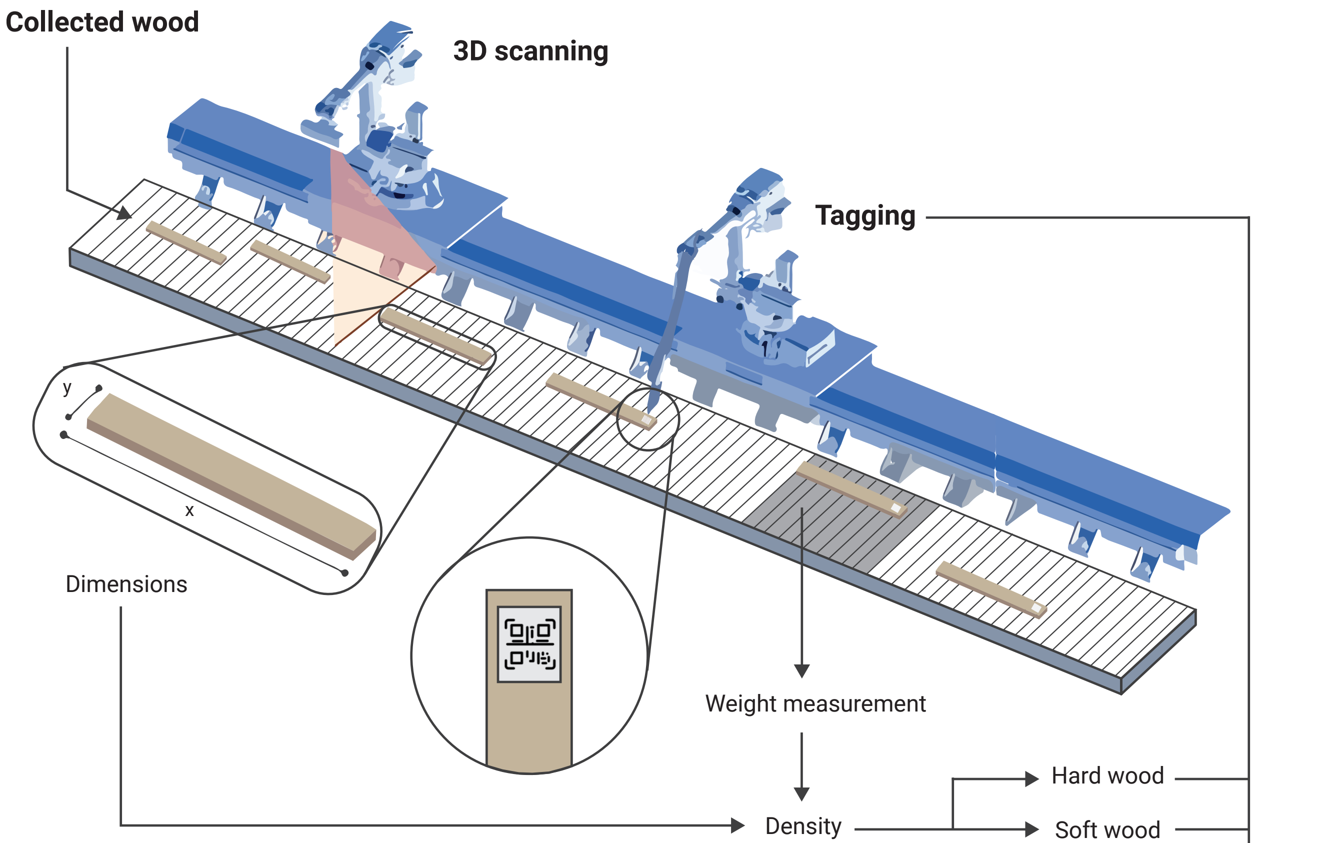
	id_wood_piece [PK] integer	length_mm integer	width_mm integer	depth_mm integer	strength_grade character varying	painted boolean	original_seller character varyin	status character varying	project character varying	usage character varying	element character varying
1045	2651	400	38	90	C30	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1046	2653	300	38	140	C14	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1047	2655	950	38	140	C18	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1048	2658	700	38	130	C14	true	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1049	2662	1000	38	80	C14	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1050	2665	950	38	70	C14	true	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1051	2666	850	38	120	C14	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small B"
1052	2667	350	44	130	C30	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small C"
1053	2668	600	38	130	C14	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1054	2670	850	44	150	C14	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small C"
1055	2671	750	38	150	C18	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1056	2674	550	38	100	C18	true	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"
1057	2676	500	38	140	C30	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small C"
1058	2677	550	38	110	C18	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small C"
1059	2678	800	44	150	C18	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal Small C"
1060	2681	900	38	90	C24	false	"Dealer x"	"Selected for assembly"	"WoodResurge headquarters and storage"	"Portalframe"	"Portal big A"

Figure 9.17: Resurge timber material passport and database for future reuse

would results in roughly the same mass, the one being a bit lighter than the other. The used of reclaimed pieces and the possibility to disassemble the structure do make the discrete portal frame an interesting competitor.

Figure 9.17 shows a portion of the material database, which acts a material passport for this project. All used pieces are send to this database, and each piece is documented with information in which portal they are, in which project and what the status of the project is. After the portals served their life-time the owner can change the status of the pieces to: can be selected for a new project. Then a designer can use these parts in a new project again by loading these pieces in the algorithm as stock. Ideally a designer would want to use as much of the already processed pieces from this database for new project as they require no more processing. If there are to little pieces available a designer can load a mixture of already used pieces from the material database and from a database which contains waste wood that is for sell. This way a circular loop can be sustained. Due to the modularity of the pieces and the attachment of the other building elements to the plywood links the reclaimed matched pieces from the main members are not affected outside of the modular system. This ensures that the pieces can be reused and matched until damaged beyond repair.

The system is completely made of wood, and optimised to the acting forces creating slender elements. Usually wood can be oversized to provide some resistance to fire, this is however not the case for this system. Therefore the applications of this system are limited to buildings which are either completely sprinkle-red or building with only one floor like the showcased hall out of which people can flee quick and easily. For other applications the structure has to be covered to enhance its fire resistance.



9.4 Overview of design and construction process

To be able to built with this system a new type of construction process has to be developed. This section will highlight a potential construction process required for the designed system. The process starts with collecting the wood. Wood collected form municipal yards, construction and industrial waste should be collected and brought to a processing facility. This facility has to be able to scan for dimensions, weigh for density and strength, check for visual defects and tag the wood to be able to sell and put into a database. The most effective way to do this would be by implementing robots. Designer can than access this database which can be either run nationally or locally per processing company. Figure 9.18 shows a schematic overview of the whole process and a more detailed view of the first phase. The university of applied sciences of Amsterdam (HvA) did research to a similar collecting and sorting method which can be consulted for more detailed information on this potential process (HVA Urban Technology, 2024). A designer can than merge multiple databases or link one to

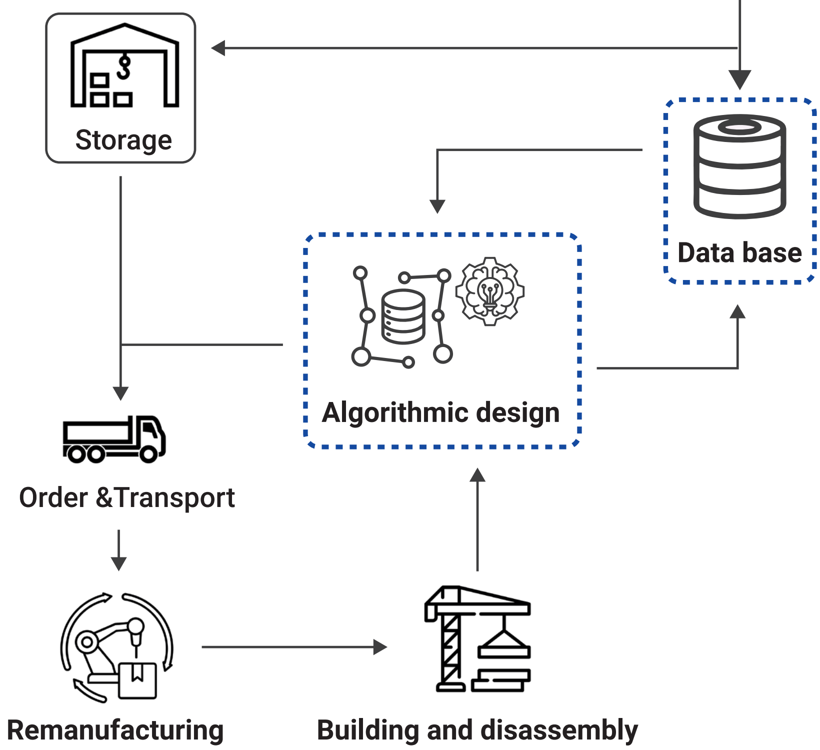


Figure 9.18: Resurge, from garbage to portal frame process scheme, dotted in blue the main focus of this thesis, adapted from (HvA Urban Technology, 2024)

Remanufacturing

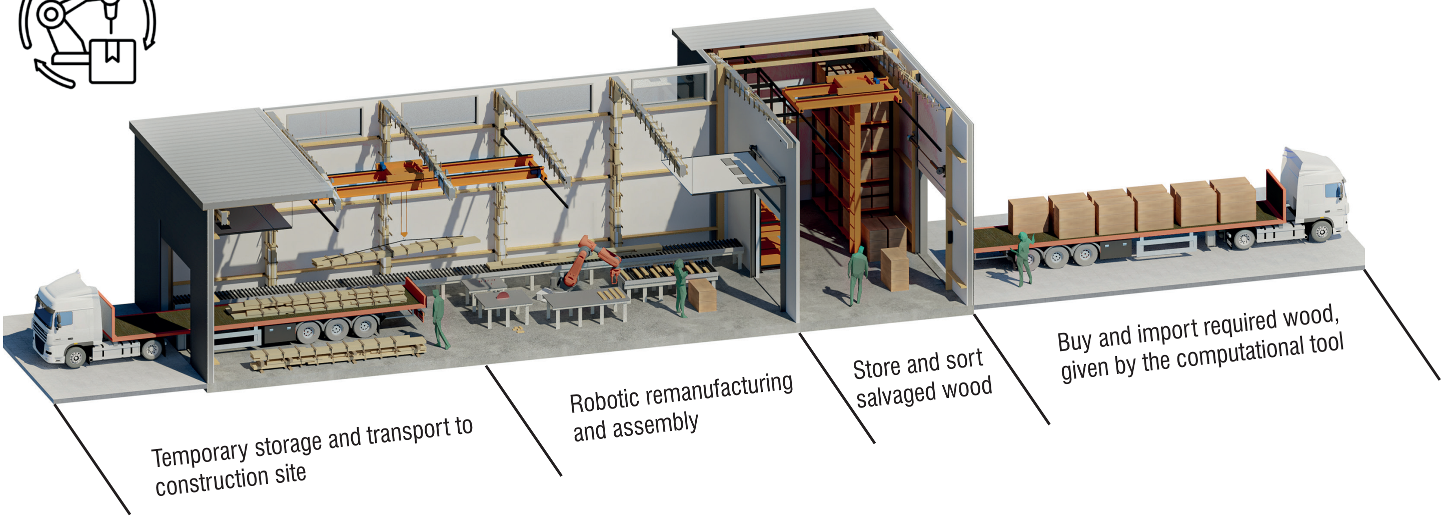


Figure 9.19: Resurge timber assembly and remanufacturing facility overall plan.

the created computational tool and design a optimised structure. The selected parts can than be ordered and shipped to assembly and remanufacturing facility. This facility can be the building designed for the case study and depicted if fully operational in figure 9.19. In this facility the ordered wood can be temporally stored and sorted. Than the sorted wood per element can be put on a treadmill which is controlled by a robot. This robot can cut the pieces, drill the exact dowel holes and than assemble the layered element with extreme high precision. Such a set-up is put in another perspective in figure 9.20 and 9.21. Robotic assembly is required as the system is very labour intensive, complex and precise with little room for error. The assembled elements can be temporarily stored and than put on transport to the construction area.

Figure 9.20: Resurge timber assembly and remanufacturing facility interior.

Figure 9.21: Resurge timber material passport and database for future reuse

Building and disassembly

Figures 9.22 - 9.26 are showing the construction phases and global steps to be taken. Arriving at the construction site the elements can be towed into place (1), fasted with a the devised connection type (2), fitted with the required steel connectors or finishes (3), than the secondary structure can be added (4) and lastly fitted with facade and roof elements.

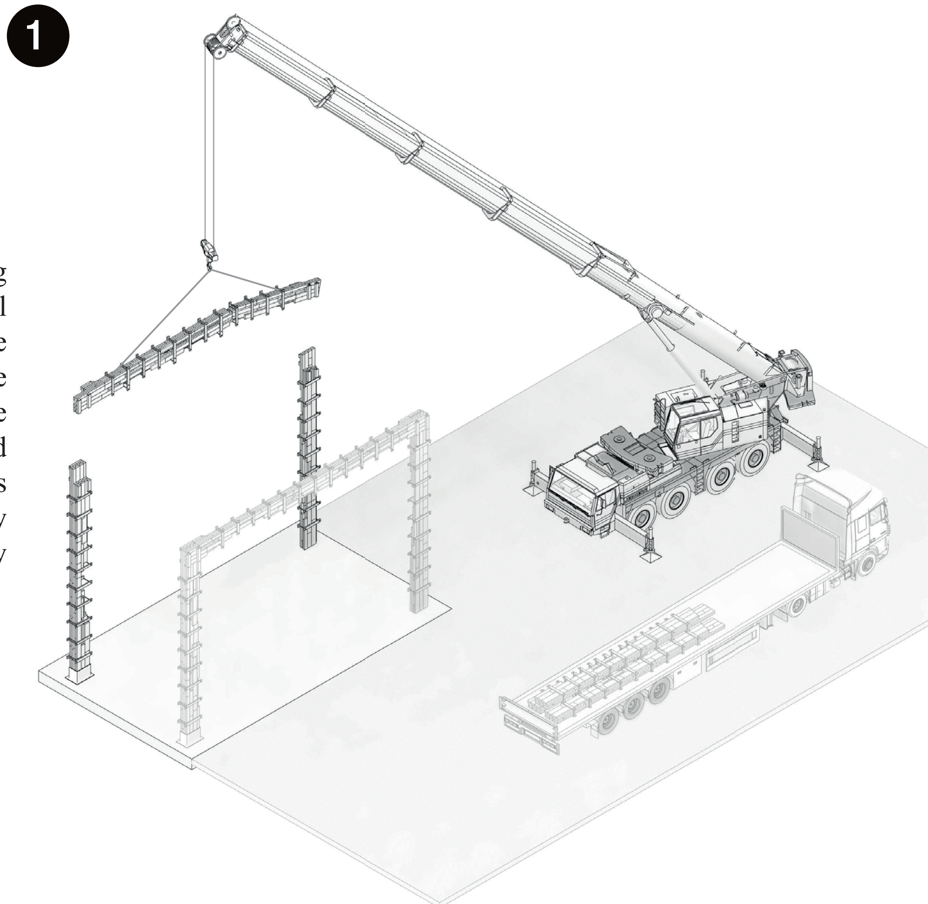


Figure 9.22: Construction phase 1: Towing elements into place

Figure 9.23: Construction phase 2: Fastening the elements in place

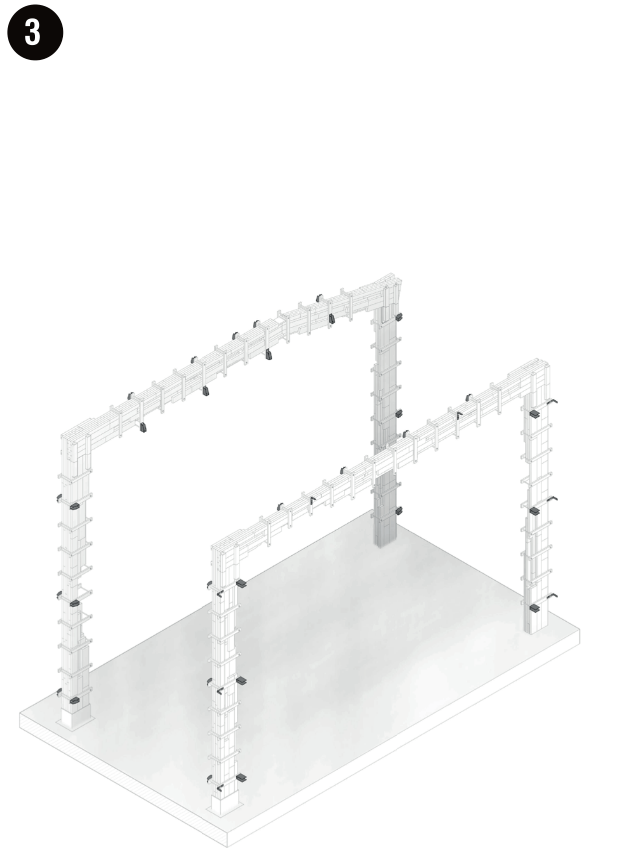


Figure 9.24: Construction phase 3: checking and adding the required steel connectors

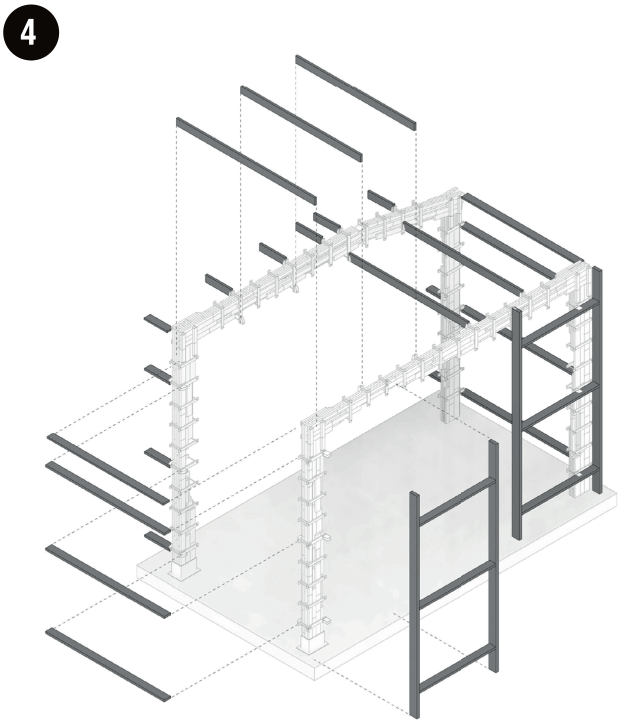


Figure 9.25: Construction phase 4: Installing the secondary structure

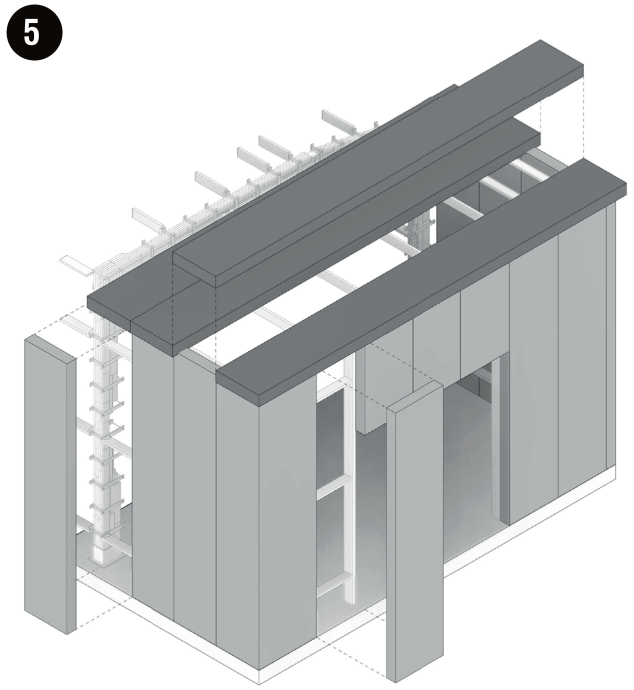


Figure 9.26: Construction phase 5: Installing the facade and roof elements

Figure 9.27: Resurge timber structural system impression

10. Conclusion and Outlook

This final chapter will conclude the research project as outlined in this thesis. First, the obtained results are used to answer the research questions. This will be followed by an overview of the current limitations, recommendations and guidelines for potential future work on this topic.

10.1 Conclusion

The main research question as stated at the start of this research was defined as: *How can programming be utilised to create a discrete structural system using reclaimed timber parts that maximizes efficiency and adaptability but minimizes virgin materials in construction?* This question was split up in a series of sub-questions which have formed the outline of this research.

Which timber waste streams can be identified to be useful for the concept "direct reuse" and what are typical components, lengths, sizes and quality of this available timber?

It is estimated is that annually a timber waste stream of 1.7 million tons of wood is produced in The Netherlands. Around 540 kilo ton of this wood is solid wood with the quality to be reused for construction. Currently this wood is down-cycled and turned into engineered boards or pallet blocks after which its life-cycle ends. Of this 540 kilo tons around 41% can be put in the category beams and framework. Such sections are highly suitable for direct reuse, and are most often seen in widths of 38 and 44mm for framework and 50, 65, 70 and 75mm for beams. Common usable cross-sectional heights are in range of 70mm to 150mm for framework and 150 - 220mm for beams. Lengths are highly variable, however pieces with lengths below 3 meters are currently not selected for reuse as they currently have little applications and are therefore disposed of. Standard construction quality of wood is C24 but with reclaimed timber it is assumed that through loading and moisture the quality can have degraded to a lesser grade such as C14 and C18. This research project is specifically devised to find new applications for reclaimed timber pieces bellow 1.5 meters in length to enable a larger market.

What connection types can be identified to create an adaptable discrete system considering: flexibility, structural performance and manufacturability?

Potential connection types can be categorised into steel, timber and hybrid connections. Steel connections are ductile but not efficient when put in bending. Timber connections are strong but brittle. Hybrid connections are able to combine the two effects into a optimal type of connection. Dowels, shear keys, bolts and screws can be used to join the parts, and can be combined with external links or binders to create a stiff and ductile system. Dowels in combination with external links were selected for this project. Crucial criteria were potential future reuse and minimal fabrication effort of the pieces combined with a potential high global stiffness. Due to the dowels the pieces are affected minimally and can be disassembled and easily reused due to a modular dowel pattern. This method is believed to maximize the future potential reuse of the parts while provide a stiff, ductile, robust and transformable system.

What method for structural optimisation constrained by the fixed dimensions of a stock is best suited for a discrete timber system?

Three optimisation methods were identified: Homogenization, ground structure and the principal stress-line method. The ground structure method can be used most efficiently for discrete timber design as it works with linear elements and can be integrated seamlessly while being constrained by the fixed dimensions of a stock. This method was moreover already proven to work in a previous thesis. In contrast the homogenization and principal stress-line methods were found to be not suitable and often used for castable and solid elements like concrete or steel.

How can the discrete system accommodate future adaptations and requirements?

By using dowels and external plywood links the parts are modified minimally and therefore making the composition/ design suitable for disassembly. The only alteration required are drilling of dowel holes in the reclaimed pieces. These holes are drilled in a modular pattern which ensures that each part becomes modular and therefore exchangeable between similar projects. The developed program allows the user to create a material passport of the used structural element and the parts. This information is saved in a database and allows, for re-matching if the elements requires to be dissembled or altered. The program can be run either with new elements, already used elements or a mix of elements, closing a large parts of the life-cycle of the reclaimed pieces, and minimizing waste.

What are the structural limitations and recommended configurations for such a system?

The system has a lower global stiffness compared to traditional systems like Glulam. It is assumed that the global stiffness of the system is somewhere in range of 50 - 80% less than a similar structure of solid glue laminated timber. This difference is substantial and deflection will become normative when using large spans. Moreover the quality of the system is dependent on the available timber. One portal can easily require 400 parts to be constructed. Not only the size but also the content of the inventory reflects on the quality of the structure. An inventory of solely small parts will result in a less stiff structure. The most stiff structure would be a composition with long pieces and large sections. This however results in a low quality optimisation as these larger parts are more difficult to remove. The optimal composition would be a highly diverse composition with smaller and larger pieces, providing enough stiffness but also leaving the optimisation program room to remove low utilised parts. The moment connection between the column and beam can become normative for the height of the elements if only a pen pattern is used. Buckling is not by default accounted for, some composition of lower quality can result in optimised geometries which are not realistic due to buckling. A buckling factor can be implemented for geometries to counteract his behaviour. A buckling factor of 1.1 was found to be sufficient for smaller portals and 1.28 for larger portals.

How can a program be created that accommodates a stock constrained discrete design method effectively in practise?

A SQL database is used to store the reclaimed timber parts. This database communicates through a Python library, Psycopg2, with a developed computational program in Grasshopper. SQL can be a suitable format for a national waste wood database as it can handle large amounts of data and pass updates in real-time, but any other format can also be chosen. The developed program is built so it iteratively solves one-dimension combinatorial problems to find a global optimal solution. A design space is broken down horizontally in layers corresponding to available widths of the available parts. These layers are broken down again vertically in rows using this time the available heights. Dynamic programming is used to fill each row iteratively with pieces from the inventory. This method uses memoization, a method of storing solutions to sub-problems in a table, effectively avoiding redundant computations which reduces the time complexity of the algorithm significantly while still reaching a global optimum. Pieces are cut if a filling rate of 100% of a row cannot be reached. Cut-offs longer than 300mm are put back in the inventory to be rematched again. The placement of pieces is optimized based on strength quality by placing lesser pieces in areas with lesser stress in accordance with the principal stress-lines. A finite elements model calculates the utilisation of each piece in the aggregated elements and removes the element which is utilised the least to create an efficient structure. The whole process takes round 6 - 10min. Depending on the size of the portal frame. This program can be used by designers, engineers and even contractors to generate discrete structures. The program is able to achieve a filling rate of 100% if stock is abundant but drops to 95 % if stock is extremely constrained

What are the main benefits, challenges and applications of this discrete system in comparison to traditional systems?

The foremost gain of this system over traditional systems is that it uses no virgin material, minimizes waste and its topology is derived from the acting forces resulting in efficient constructions with a low mass. It can outperform existing optimised systems as it is not bound by its initial geometry. The system can be dismantled, reconfigured and reassembled if loads or requirements change or parts can be exchanged between similar projects. On the other hand the system is quite unconventional and could be viewed unsuitable from an architectural point of view. The external links add a lot of material and make installation of external building elements like a roof or face more difficult. On the other hand it helps in reducing the sectional height in order to create a rigid moment connection, enhances the global stiffness of the system greatly, reducing buckling behaviour and allows for a circular system that can be disassembled and reused. The global connections require some parts to be cut or drilled outside of the modular system, diminishing their future reuse potential. These parts can therefore only be reused in the same configuration. Applications are endless. This project focused on portal frames but beams and columns can also be generated separately. Every shape can be made as long as the elements are linear. Even curved structures could be made if the curve are discretized in linear parts.

How can programming be utilised to create a discrete structural system using reclaimed timber parts that maximizes efficiency and adaptability but minimizes the need for virgin materials needed in construction?

Overall, a new structural system has been created by bringing together multiple smaller reclaimed pieces of wood that together form an efficient structural system. The parts are connected by dowels and external plywood links. These links should provide enough friction between the parts to enact a form of composite behaviour and to maximize the future reuse potential of the pieces. The aggregations are made by solving an one-dimensional combinatorial problem aided by dynamic programming to create a highly efficient matching process. The system is efficient as structural lesser parts are placed in areas where stresses are lower. Moreover, the topology of the structure is derived from the acting forces, minimizing its mass and consumed material. The devised algorithm works, but is still highly dependable on the aggregated element. Currently the program is not yet stable enough to always generate and guaranty a viable composition. To generate a viable composition, variables have to be checked and altered to fit to the dimensions of the design space. By trying different combinations of stock an optimal composition can be found. The system is adaptable as it can be disassembled and reconfigured. Due to the modular size and dowel pattern many parts are exchangeable between similar projects, closing the life-cycle of many of the reclaimed timber parts, minimizing waste and virgin resource consumption and aiding the circular economy. The system uses a new design language called discrete design, which may not yet be valued by everyone from an architectural point of view but acts as a prove of concept. Paving the way to a world which ultimately has no more need for virgin wood, bringing deforestation to a minimum.

10.2 Research limitations

This section provides various limitations of this research project regarding encountered challenges, still uncompleted work or work that has to be done to prove the made assumptions of this thesis.

Firstly, all aggregations were made with a generated 'fake' database based on common seen cross-sections in construction. Therefore there has not been a scenario which reflects a true realistic stock of waste wood dimensions. Although the algorithm is flexible enough to process each type of stock and various tests were conducted with alternative scenario's, it could be valuable to cross-check this with real wood vendors and stock-buyer to determine if it would be realistic to expect the used dimensions.

The structural optimisation does not include optimisation of orientation of the wood within the aggregate and is based on linear placement of elements. Therefore, no insight can be provided on the influence of orienting the wood to the angles of the principal stress-lines.

Due to time constraints the program is not constrained on placing parts in the aggregation with overlapping boundaries of parts in consecutive layers in the longitudinal x-direction as explained in section 8.5. Results showed that this could, in some cases, result in a low quality aggregation causing connectivity issues and a premature stop of the optimisation process. This makes the program less reliable and requires checking different compositions to evaluate the best 'accidental' placement of pieces.

Lastly, a new type of structural system was designed which is based on a frictional force between the parts that enact some composite behaviour and ductility. This system is still highly theoretical. The built mock-up showed that the external links enhance the global stiffness significantly and some composite behaviour is generated. Mechanical testing could offer valuable insights in the real structural behaviour and limits of the system which can help in developing the algorithm further.

10.3 Discussion and future recommendations

This thesis took significant steps in the field of discrete design. While stock constrained design is not new and some examples could be found of reciprocal systems, stock constrained design with a massively layered system could not be found anywhere else. Massively layered systems are hard to design when using a stock, as pieces cannot overlap and clash in more than one direction. A lot of challenges had to be overcome to create a stock constrained tool to design an optimise discrete structure form reclaimed timber and there is still a lot of room for improvement. This section will outline in which directions this research can be extended.

Re-evaluation of the algorithm

The developed computational model still has a lot of room for improvement. First, the aforementioned limitation is that not all parts comply to the mandatory overlap in the longitudinal direction of the aggregated elements of 100mm. Acquired results are highly affected by the composition as some composition could 'accidentally' turn out well while others can have lots over overlap. Overlap leads to a lower amount of joints, higher global utilisation and could result in connectivity issues and premature termination during optimisation. Results can therefore be misguiding. The program currently can detect these clashes but not solve them. The clash control should be incorporated into the reorganisation after the dynamic programming has selected pieces. Further development can improve this reorganising process with this constraint to make the program more stable.

Secondly, currently in the structural model it is assumed that the global stiffness of the structure should be in range of 50 - 80% of a Glulam beam with identical dimensions based on the works of O'Ceallaigh et all. (2022). The stiffness of the virtual connections between the parts is modified to accommodate this range. The stiffness of the joint highly affects the results and performance of the model, and if not adjusted right can result in invalid results. Different spans and different compositions result in an alternate global stiffness without a logical pattern. Further development should include a method which stabilises the global stiffness of the structural model, and should devise a method of determining a realistic value for the stiffens of the joints. Mechanical testing could provide valuable insight in this area.

Thirdly, the optimisation is based on an ESO (evolutionary structural optimisation) and constrained to Karmaba3D, Grasshopper and Anemone. This means that only pieces can be removed, but not be put back. BESO (bi-directional evolutionary optimisation) is a method that can removes pieces and redistribute pieces in the design space. It can also replace pieces if removal leads to a worse outcome. By discarding Karamba3D and Grasshopper and creating a F.E.A. in Python a much more sophisticated optimisation can be executed not only based on utilisation but for example maximizing stiffness which is not possible with Karamba3D.

To account for bucking currently a safety factor is multiplied by the utilisation of the columns. This however is not extensively tested and validated. Further research could implement a buckling check within the optimisation loop to make results more viable.

The program currently is not able to generate a stable stream of results. To generate a portal a lot of variables have to be checked and altered. Different compositions of stock have to be tried to find the best portal. Making the program more stable requires more development introducing more constraints to lock variables.

The model is constrained to the computational power of the computer. An inventory of 1.000 parts is still within acceptable computing time, larger inventories and portal frames are increasing time complexity greatly, making the model hard to work with. By implementing dynamic programming also in the first two combinatorial problems time complexity can be reduced significantly, making the model more workable for larger stocks.

Lastly, the current model uses a one-dimensional approach, dividing the design space in combinatorial problems which are solved per direction. This approach works quickly but a valuable improvement would be to create a 3D matching algorithm which can fill a given design volume iteratively, just as existing bin-packing algorithms do. This way parts can also be placed outside of a layer, resulting in a better natural shear resistance creating interesting compositions for further analysis of structural behaviour of discrete assemblies. Moreover, with such a method, rotating parts for orientation optimisation would be more convenient and the program can be extended with constraints like exclusions zones where parts cannot be placed to create a predetermined hole in a beam for air channels or other requirements.

Mechanical testing

The designed method for connecting the timber parts is based on a theoretical structural concept of friction. It is still unknown how much composite behaviour can be expected from the plywood links. It is also unknown which dowel size and distance are best suited. A lower amount of dowels would be beneficial for the future reused potential and speeds up assembly. These aspects can all be tested by mechanical testing of the optimised results from the program and provide value insights on how to make the program and system more reliable.

The joint type chosen can be disputed and currently adds a lot of material to the optimised structure. Revisiting the joint type and re-evaluating all options, including glue, could offer new insights. The research can lastly be extended more in the field of the circularity aspect, testing scenarios and the reconfigurability of the system.

Post-processing and assembly method

The post-processing part of the algorithm could be further extended so that the user can extract 2D drawings of each element with the required dowel holes and identification numbers of each part. This is essential in order to create a working production method. The research scope was fixed on creating a working digital model while potential production techniques were not taken into consideration. Producing the discretized frameworks could be an interesting field further research can build upon.

Lastly, The acquired data and the model after optimisation is currently only exported to Excel. However the composition and 3D model are not saved. After Grasshopper is closed the finite element model and the optimised model are gone. When loading the exact same inventory the same result can be acquired again but further development should look for a way to store the structural model in another file or push the result to a design exploring software so users can easily compare designs and review results.

11. Reflection

1. How is the graduation topic positioned in the studio

The building technology track focuses on teaching students to become sustainable designers being able to bridge the gap between architecture and engineering, using innovative approaches and new technologies. The graduation topic 'Discrete timber' is a fairly new way of designing. It encompasses design with numerous elements, either highly variable or a kit of parts. By bringing these elements together a structure can be created. Discrete timer fits perfectly in the building technology track, as by using an innovative design approach reclaimed pieces can be turned into a new circular and sustainable structure. Innovation, new technologies, structural knowledge, sustainability, and design are all combined into this thesis to create a new type of sustainable structure allowing for a new architectural language.

2. How are research and design related?

The thesis encompasses a broad research aspect. Including a research aspect, a design aspect and a research through design aspect. The literature research has provided the basis on which the algorithm and the design could be developed. The comparison of various joints, optimisation techniques and discrete design can be seen back in the design. The algorithm could be developed based on the found approaches in the literature, and is optimised by research trough design. By each time assessing the results of the algorithm, reflecting back on the design of the structural system an readjusting the algorithm a working tool could be developed. The type of joints, the dowel size and much more has a direct link with the algorithm and cannot be seen apart from each other. The design parts can be seen by how the parts are brought together, a product, in form of the new type of composite cross-section is developed which can be projected on every scale. Also here the design has been through a continues loop of iterations and reflection and research.

3. How did the research approach work out? And did it lead to the results you aimed for?

The research approach worked out well. The literature study has provided all the information on discrete thinking, optimisation and connecting discrete parts required to conduct the research. After the literature review I was able to start building the algorithm without first needing to do more research. I knew what my options were and what I needed to do, this values the implemented approach and method, as sufficient. The building of the algorithm was a painstaking process of trial and error and took a few weeks to really form into something that could be tested and further developed with more detail. The system, and therefore also the algorithm, accompanies a lot of variables what has made it difficult to create a true stable algorithm. In retrospect the initial approach is adjusted once but in the end led to the results I aimed for. Initially I aimed for better results and a more sophisticated algorithm. At the start of the project I sought after a 3D matching algorithm, in which fibre orientation optimisation could be included and in which the designer can have more control on the aggregation. But due to time constraints and complexity this was not found to be feasible. After two weeks the 3D approach was not able to provide viable results, requiring me to change the method form a 3D approach to a one-dimensional approach. In retrospect, I would have taken the same approach knowing if I had new the outcome at the start as the algorithm now is able to generate every type of portal frame in a few tries.

4. Did you encounter moral/ethical issues or dilemmas during the process? How did you deal with these?

The thesis is centred around building the algorithm, and has not touched to human aspect much. However as an engineer we always have to reflect ethically on the process and design as intentional manipulation of results can lead to failure of structures and even dead of people. A this is a theoretical assignment this is not the case, but still deign decisions were made to ensure a structurally safe and sound system. One of the design criteria was to create ductile system to prove an extra level of safety if one parts unexpectedly fails or if the algorithm has miscalculated. Results obtained from the algorithm and F.E.A. model are also not just assumed to be true but checked manually and altered until a they are found to be satisfactory. As a designer we can never fully just assume what the computer provides us with and we are required to always double check and put solutions in question.

Another aspect could be seen as moral decision making. The structural system is designed to be fully circular, however using glue to connected the pieces would results in a much better structural system and a much higher composite behaviour. Using glue however was not morally responsible as this does not aid the circular economy, and without real consideration cut out of the solution domain. Therefore extensive research was done on other types resulting is a system maybe not that structurally efficient but demountable. The moral issue here could be centred around the question, are all the measures worth it to avoid glue at any cost and adding fast amount of more material to get a working system. In this thesis it was chosen to focus on the ideology instead of the impracticalities, resulting in a prove of concept of design solution outside the know and traditional solution space, adding value previous academic works.

5. To what extent are the results applicable in practice?

The algorithm/ tool works, and is at a point were mechanical testing is required to validate the results. The results are clear and unexpected behaviour in the algorithm can be accounted for. Overall I believe that with more development and more knowledge on the structural system as a whole the tool could be introduced in practise, making reuse of a varying stock in a layered assembly possible for architects and engineers.

6. To what extent has the projected innovation been achieved?

A tool was developed which is able to select highly variable items from a stock and create an optimised layered assembly. The assembly is optimised in respect to the forces acting on it. The structural system is modular, circular and demountable. Meaning that the structure can be adjusted if loads or requirements change. The research aim and goal have been reached, through this new type of architecture and construction. I would say that the projected innovation is well achieved, and has resulted into something innovative, adding value to the themes of discrete design, timber and stock-constrained design.

7. Does the project contribute to sustainable development?

The structural system is made modular and circular, while glue should have been a much more efficient connection type the system can now be dismantled and due to the database/ material passport be reused infinitely. The algorithm allows to make compositions from a variable stock of reclaimed pieces. Giving previous, especially short pieces, viewed as unusable, a new perspective to be reused. The system can be made entirely out of reclaimed timber pieces, aiming to reduce virgin resources and deforestation. It aids to the development of optimisation as the optimised system in no longer constrained to its initial dimensions but can be reconfigured.

8. What is the socio-cultural and ethical impact and what is the relation between the project and the wider social context?

This project uses a new type of design, discrete design combined with structural optimisation. The resultant geometries are far outside of the traditional geometries and reflects a new type of architecture, a transformable efficient and sustainable type. This new type of architecture could not be within everyone's likings and understanding. Linear and straight beams and columns are often strived after more than the designed discrete system. The general societal concept of a structural system is afflicted and requires a change if the perspective of what aesthetically is pleasing before the system could be implement in real-time projects within society. Personally I love seeing true and honest structures that show the flow of forces and the structural behaviour, but it still might be too soon and a too large of an impact on the socio-cultural view on aesthetics to fully bring this on the market.

9. How does the project affects architecture / the built environment?

Similarly as stated above in question 8. The devised system shows a new type of transformable , circular and efficient architecture as falls well out of the traditional architecture. The system has the advantage that it is future proof, and can be adapted if requirements change. The system can be made entirely of reclaimed timber reducing virgin material usage and ultimately waste. This system allows architects implement the concept of design with what there already is.

10. How did the made choices affected the end results and in retrospect, could other choices have led to better results?

In the end I do doubt the chosen connection system as it adds a lot of material which I did not fully grasped until I was generating the case study. The method could have been improved to model, test and try more joints types to make an even better well weighted choice, on the other hand already a MCA was made to ensure a well weighted choice. This however, is a very theoretical approach and implementing decisions in practise can lead to different insights, which in this case have led to reconsiderations. I might have spent too much time developing the algorithm and too little on the report and in perfecting the structural system. Looking back I feel like I could have done processes concerning the algorithm more efficiently. But on the other hand, when I started this project I had virtually no experience in Python and had to self-teach myself everything. For what it is worth I am very proud of the result that I was able to put down and this whole thesis process has been a valuable lesson for a future research project.

11. What are the most important new insights in conducting research obtained from this master thesis?

I have learned so much during the whole graduation topic. Firstly, I learned to program with Python and also gained insight in how much time it takes to implement initially thought simple processes in a working program. Simple processed can take hours to program int a functional code. Moreover, I learned the importance of building as physical model again in this thesis. By making the model I found out that the initial design for dowel placement would have lead misalignment, based on these findings the dowel pattern was adjusted and circularity aspect enhanced. Lastly, I found that sometimes a problem, seemingly easy can become excessively difficult. To solve problems when being stuck it can help a great deal to dissect the problem in sub problems, by solving the sub problems the global problem can be reached more easily.

Acknowledgements

This thesis marks the end of my academic journey. The last two years at TU Delft were very exciting and instructive. The Building Technology program has helped me to figure out what I find most interesting and which career paths I want to explore. During these two years I have been surrounded by lots of friends, whom I want to thank for all the coffee breaks, insightful discussions and lots of fun. I like to thank everyone who has been part of this journey.

First and foremost, I like to thank my mentors Dr. Stijn Brancart and Ir. Hans Hoogenboom and express my sincere gratitude for your time to help me write this thesis. Each time we discussed the progress you both have taught me to think differently, outside the known boundaries and strive for an ideal instead of only applying practical solutions. You inspired me every session, igniting my enthusiasm further.

To my dearest friend Berjan, thank you for listening to my gibberish about programming issues, wood and optimisation while pretending to be interested, your cheerful and motivated words helped me overcome the biggest challenges. Loved sharing this process with you and I want to thank you for giving me space to work or stopping me from working from time to time to enjoy life.

To my parents, thank you for giving me the space to work and supporting me to get here where I am now. Without your support this would not have been possible.

Thank you dad for helping me with building the mock-up model, your insights through-out my entire academic journey have be most valuable

References

Basumalick, C. (2024). What is Dynamic Programming? Working, Algorithms, and Examples. https://www.spiceworks.com/tech/devops/articles/what-is-dynamic-programming/

Bertin, I., Saadé-Sbeih, M., Roy, R. L., Jaeger, J., & Feraille, A. (2022). Environmental Impacts of design for reuse practices in the building sector. Journal of Cleaner Production, 349, 131228. https://doi.org/10.1016/j.jclepro.2022.131228

Borgström, E., & Fröbel, J. (2019). The CLT Handbook: CLT structures – facts and planning. Swedish Wood. https://www.swedishwood.com/publications/list_of_swedish_woods_publications/the-clt-handbook/

Bruggen, R., & Zwaag, N. (2017). Knelpuntanalyses houtrecycling. TAUW BV. https://www.nedvang.nl/wp-content/uploads/2019/02/knelpuntenanalyse-houtrecycling1.pdf

Brütting, J., Senatore, G., & Fivet, C. (2021). Design and fabrication of a reusable kit of parts for diverse structures. Automation in Construction, 125, 103614. https://doi.org/10.1016/j.autcon.2021.103614

Brütting, J., Vandervaeren, C., Senatore, G., De Temmerman, N., & Fivet, C. (2020). Environmental impact Minimization of reticular structures made of reused and new elements through life cycle assessment and Mixed-Integer linear programming. Energy and Buildings, 215, 109827. https://doi.org/10.1016/j.enbuild.2020.109827

Buchanan, A. H., & Fairweather, R. H. (1993). Seismic design of glulam structures. Bulletin of the New Zealand national society for earthquake engineering, 26(4), 415-436.

Burger, J. A., & Zipper, C. E. (2009). Restoring the Value of Forests on Reclaimed Mined Land. Virginia Cooperative Extension. http://pubs.ext.vt.edu/460/460-138/460-138_pdf.pdf

Centraal Bureau voor de Statistiek. (2023). Winning, invoer en uitvoer van materialen naar soort; nationale rekeningen [Dataset]. https://opendata.cbs.nl/#/CBS/nl/dataset/83180NED/table?ts=1705422262070

Chen, D., Wang, G., & Chen, G. (2021). LEGO Architecture: Research on a Temporary building design Method for Post-disaster Emergency. Frontiers of Architectural Research, 10(4), 758–770. https://doi.org/10.1016/j.foar.2021.08.001

Cheret, P., Schwaner, K., & Seidel, A. (2013). Urbaner Holzbau : Handbuch und Planungshilfe ; Chancen und Potenziale für die Stadt.

Deleuze, G., & Guattari, F. (1987). A thousand plateaus: Capitalism and Schizophrenia. Minneapolis, MN: University of Minnesota Press. https://files.libcom.org/files/A%20Thousand%20Plateaus.pdf

European Parliment (2021). How the EU wants to achieve a circular economy by 2050. https://www.europarl.europa.eu/topics/en/article/20210128STO96607/how-the-eu-wants-to-achieve-a-circular-economy-by-2050

Eijk, J., van. (2021). Reusing waste wood for an exterior wall element. [Published master thesis: http://resolver.tudelft.nl/uuid:f4b7a7bb-abfc-46d2-b13e-7bf9c79cd2d0]. Delft university of technology.

Fang, D., Brown, N. C., De Wolf, C., & Mueller, C. (2023). Reducing embodied carbon in structural Systems: A review of early-stage design strategies. Journal of Building Engineering, 76, 107054. https://doi.org/10.1016/j.jobe.2023.107054

Giordano, L., Derikvand, M., & Fink, G. (2023). Bending properties and vibration characteristics of Dowel-Laminated timber panels made with short salvaged timber elements. Buildings, 13(1), 199. https://doi.org/10.3390/buildings13010199

Gordon, J. (2012). Structures or why things don't fall down. Springer Science & Business Media.

Gorgolewski, M. (2008). Designing with reused building components: Some challenges. Building Research and Information, 36(2), 175–188. https://doi.org/10.1080/09613210701559499

Hansen, J. L., Nielsen, M., Hansen, S. G., Kunic, A., & Naboni, R. (2023). A francture mechanical and anisotropic FEM model of the "RECONwood joint" and experimental verification. World Conference on Timber Engineering Oslo 2023. https://doi.org/10.52202/069179-0171

Hansen, S. G., Kunic, A., & Naboni, R. (2021). A reversible connection for robotic assembly of timber structures. Engineering Structures, 245, 112795. https://doi.org/10.1016/j.engstruct.2021.112795

HvA Urban Technology. (2024). Data Wood: Resthoud innemen met robots. https://www.hva.nl/kc-techniek/gedeelde-content/projecten/circular-transition/digital-production-research-group/data-wood.html

Heijne, N. (2023). Stock defined gridshells: about the computational optimization of gridshell structures from a finite stock. [Published master thesis: Published mater thesis: http://resolver.tudelft.nl/uuid:5bad81b9-3ab8-4edb-8743-cb237ea64d97]. Delft university of technology.

Holmberg, E., Torstenfelt, B., & Klarbring, A. (2013).

Stress constrained topology optimization. Structural and Multidisciplinary Optimization, 48(1), 33–47. https://doi.org/10.1007/s00158-012-0880-7

Huang, Y., Alkhayat, L., Wolf, C. G., & Mueller, C. (2021). Algorithmic circular design with reused structural elements: method and tool. International fib Symposium - Conceptual Design of Structures 2021. https://doi.org/10.35789/fib.proc.0055.2021.cdsymp.p056

Jensen, C., Sander, S., Kunic, A., & Naboni, R. (2023). Re-VoxLam Truss. Create-SDU. https://www.create-sdu.com/projects/revoxlam

Kochenderfer, M. J., & Wheeler, T. A. (2019). Algorithms for optimization. MIT Press.

Kumar, P. (2016). Synthesis of Large Deformable Contact-Aided Compliant Mechanisms using Hexagonal cells and Negative Circular Masks. Indian institute of technology Kanpur.

Kunic, A., Kramberger, A., & Naboni, R. (2021). Cyber-Physical Robotic Process for Re-Configurable Wood architecture - closing the circular loop in wood architecture. eCAADe proceedings. https://doi.org/10.52842/conf.ecaade.2021.2.181

Kunic, A., & Naboni, R. (2023a). ReconWood Slab. Computational design and structural optimization of reconfigurable timber slabs. Researchgate. Proceedings of the IASS Annual Symposium 2023 Integration of Design and Fabrication, Melbourne, Australië. https://www.researchgate.net/publication/372165873_ReconWood_Slab_Computational_design_and_structural_optimization_of_reconfigurable_timber_slabs

Kunic, A., & Naboni, R. (2023b). Collaborative design and construction of reconfigurable wood structures in a mixed reality environment. Blucher Design Proceedings. https://doi.org/10.5151/sigradi2022-sigradi2022_193

Kunic, A., Naboni, R., Kramberger, A., & Schlette, C. (2021). Design and assembly automation of the robotic reversible timber beam. Automation in Construction, 123, 103531. https://doi.org/10.1016/j.autcon.2020.103531

LETI. (2020). LETI emodied carbon primer: Supplementary guidance to the climate emergency design guide. The London Energy Transformation Initiative. https://www.leti.uk/ecp

Li, Y., & Chen, Y. (2010). Beam Structure Optimization for Additive Manufacturing based on Principal Stress Lines. Epstein Department of Industrial and Systems Engineering University of Southern California. https://doi.org/10.26153/tsw/15231

Lu, K. J., & Kota, S. (2005). Topology and dimensional synthesis of compliant mechanisms using discrete optimization. Journal of Mechanical Design, 128(5), 1080–1091. https://doi.org/10.1115/1.2216729

Mantje, M. (2023). Reuse of scrap wood in a building product. [Published master thesis: http://resolver.tudelft.nl/uuid:e70dc8f6-91a4-4d8a-9610-dc18618d1f19]. Delft university of technology.

McArthur, E., Zumwinkel, K., & Stuchtey, M. (2015). Growth within: A circular economy vision for a competitive Europe. Ellen McArthur foundation. https://www.ellenmacarthurfoundation.org/growth-within-a-circular-economy-vision-for-a-competitive-europe

Menges, A., Schwinn, T., & Krieg, O. D. (2016). Advancing Wood Architecture : A Computational approach. https://www.taylorfrancis.com/books/advancing-wood-architecture-achim-menges-tobias-schwinn-oliver-david-krieg/e/10.4324/9781315678825

Morris, F., Allen, S., & Hawkins, W. (2021). On the embodied carbon of structural timber versus steel, and the influence of LCA methodology. Building and Environment, 206, 108285. https://doi.org/10.1016/j.buildenv.2021.108285

Munck, E. D., de, Ravenhorst, G. J. P., & Jorssen, A. J. M. (2011). HTO Dictaat Houtconstructies. Centrum Hout Almere.

Naboni, R., & Kunic, A. (2019). A computational framework for the design and robotic manufacturing of complex wood structures. Blucher Design Proceedings. https://doi.org/10.5151/proceedings-ecaadesigradi2019_488

Parigi, D. (2021). Minimal-waste design of timber layouts from non-standard reclaimed elements: a combinatorial approach based on structural reciprocity. International Journal of Space Structures, 36(4), 270–280. https://doi.org/10.1177/09560599211064091

Paula, A., de. (2023). Discrete Automation: Robotic Construction Workflow for Reconfigurable Timber Housing. [Published master thesis: http://resolver.tudelft.nl/uuid:c3436d86-c7d7-48c2-833a-d2fad07fabe5]. Delft university of technology.

Pronk, A., Brancart, S., & Sanders, F. (2022). Reusing timber formwork in building construction: testing, redesign, and Socio-Economic Reflection. Urban planning, 7(2), 81–96. https://doi.org/10.17645/up.v7i2.5117

References

Retsin, G. (2019). In Part Whole: The Aesthetics of the Discrete: Vol. 89:120-127. https://doi.org/10.1002/ad.2488

Rossi, A., & Tessmann, O. (2018). From Voxels to Parts: Hierarchical Discrete Modeling for Design and Assembly. Advances in intelligent systems and computing, 1001–1012. https://doi.org/10.1007/978-3-319-95588-9_86

Sánchez, J. (2020). Architecture for the Commons: Participatory Systems in the Age of Platforms. Routledge. https://doi.org/10.4324/9780429432118

Sánchez, J. S. (2017). Combinatorial Commons: Social remixing in a sharing economy. Architectural Design, 87(4), 16–21. https://doi.org/10.1002/ad.2190

Scheibmair, F. (2012). Expedient moment connections for timber ortal frame buildings. [Published PhD thesis: https://researchspace.auckland.ac.nz/handle/2292/20009. University of Auckland.

Tam, K. M., & Mueller, C. (2015). Stress line generation for structurally performative architectural design. Other repository. https://dspace.mit.edu/handle/1721.1/125063

Terzidis, K. (2015). Permutation Design: buildings, texts, and contexts. https://www.researchgate.net/publication/372165873_ReconWood_Slab_Computational_design_and_structural_optimization_of_reconfigurable_timber_slabs

Tomczak, A., Haakonsen, S. M., & Łuczkowski, M. (2023). Matching algorithms to assist in designing with reclaimed building elements. Environmental Research: Infrastructure and Sustainability, 3(3), 035005. https://doi.org/10.1088/2634-4505/acf341

UN Environment Programme. (2019). Understanding circulairty. UNEP circularity platform. https://buildingcircularity.org/

United Nations. (2023). United nations: Global issues population. https://www.un.org/en/global-issues/population

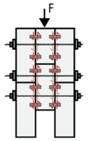
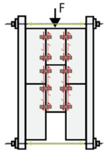
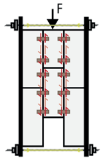
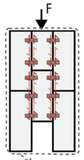
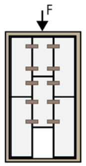
Warmuth, J., Brütting, J., & Fivet, C. (2021). Computational tool for stock-constrained design of structures. International Conference on Spatial Structures, Proceedings of the IASS Annual Symposium 2020/21, 1–9. https://infoscience.epfl.ch/record/287984

Xiao, K., Chen, C., Guo, Z., Wang, X., & Yan, C. (2020). Research on Voxel-based Aggregation Design and its Fabrication. Proceedings of the 25th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), 1, 13-22. https://doi.org/10.52842/conf.caadria.2020.1.013

Figures

Figures

Appendix A. Multi criteria analysis

Multi Criteria Analysis (Factors)		1 Architectural impact (Value)	0.05	2 Potential future reuse (Value)	0.30	3 Global stiffness (Value)	0.30	4 Ductility (Value)	0.10	5 Environmental impact (Value)	0.20	6 Fire resitance (Value)	0.05	Total 1.00
Dowels + bolts		Little visual modification.	4	Many holes on random location of the wood limit future reuse. Timber parts are affected a lot.	1	Shear strength form dowel + friction from bolts provide a high global stiffness.	4	The combiantion of timber dowels and steel bolts result in a high strength conenction. Strength from wood and ductility from steel	4	Three time more bolts needed than parts for the whole aggregate.	1	Steel bolts are protected inside the wood.	3	**2.45**
Dowels + timber links		Large visual modification robust look but stocky.	2	Timber parts are affected the least as possible due to external conenctions. Only staderdised dowel holes ares included.	4	Shear strength form dowel + friction from bolts provide a high global stiffness. Timber links cannot be tighend as much as steel, lower pressure is lower global stiffness.	3	Composite behaviour ensures that if a dowels fails the structure is kept togheter and acts as a safeguard. However, to much pressure can let internal pressure climb and cause a sliding brittle failure.	2	Few steel bolts or tension cables needed and timber as pressure distributor. Timber is more heavy and bulkyer than steel so not 4 points.	3	Timber has some natural resistancy and can be impregantead or treated.	2	**3.1**
Dowels + steel links		Large visual modification robust look very sleek.	3	Timber parts are affected the least as possible due to external conenctions. Only staderdised dowel holes ares included.	4	Shear strength form dowel + friction from bolts provide a high global stiffness. Steel links can acomadate high pressure and thus higher stiffenss.	4	Composite behaviour ensures that if a dowels fails the structure is kept togheter and acts as a safeguard. However, to much pressure can let internal pressure climb and cause a sliding brittle failure.	2	Few steel bolts or tension cables needed. Steel as pressure distributor, can be very thin sheeted, but adds to weight and steel usage.	1	Steel has some natural resitance and can be coated.	2	**3.05**
Dowels + binding strap		Large visual modification, can look a bit clutterd.	2	Timber can be crushed on the corners due to thin straps and large peak stresses.	3	Binding provides friction but also puts extra pressure on the dowels due to pressure for all sides. This results in higher internal forces and lower global stiffness.	2	Bindings encage the composition from all sides and keep the structure togheter even if dowels fail.	4	Very little environmental straining material used, steel straps from pallets or ratchet binders.	4	Straps do not have natural fire resistance and are hard to tread. Soil can be put over the structure but adds to the complexity and the overall weight.	1	**2.85**
Dowels + Multiplex cage		Large visual modification, optimised structure is not hidden.	1	Multiplex sheets are screwed on the structure, leave many holes and reducing future reuse potential.	1	Acts as a tublur stiff element reenforced with interal wood structure. Very stiff structure.	4	The multiplex cage keeps the structure stiff and will show cracks if failure is iminent.	4	Multiplex cage adds severly to the weight of the structure and the material consumption.	2	Multiplex sheets protect the inter construction from fire giving it a high resistance	4	**2.55**

Appendix B. Loading schemes

Loads on Portal	With bay size of 3.6 meters and height of 5 meters used in chapter 8												
	element	number of elements	Lengte m	dikte m	Breedte m	Volume m3	Soortelijk gewicht P_{mean} (kg/m2)	Soortelijk gewicht P_{mean} (kg/m3)	Gewicht kg	Weight on beam in kN	kN/m2	kN/m1 (local)	kN/m1 (global)
Span			7									7.0	7.0
Bay size			3.6										
height of column			6										
Dead load													
Selfweight	Main beam	1	7	0.32	0.2	0.45		420	188.16	1.85		0.3	
													0.00
	Roof joist	0	3.6	0.05	0.22	0.04		420	0.00	0.00	0.0000	0.0	0.00
	Roof structural slab (CLT)	1	3.6	0.12		0.00		420	0.00	0.00		1.8	1.78
	Insulation (EPS)	1	3.6	0.1				25				0.1	0.09
	PV_panels		3.6				50					*1.8*	1.77
Total S.W.												2.04	**3.63**

Live load Z direction:													
Snow											0.56		**2.016**
Maintenance only local (excentric)										1			
Live load facade													**1.00**

Live load Y direction:													
Wind load side pressure											0.66	**2.38**	**2.38**
Wind load side suction													**1.19**

load combinations

Combinations ULS		Fu.C.1	Fu.C.2	Fu.C.3	Fu.C.4
LC1	Selfweight	1.40	1.20	1.20	1.20
LC2	Snow		1.50		1.20
LC3	Wind			1.50	1.20

Combination SLS	EUROCODE 6.14	Ka.C.1	Ka.C.2	Ka.C.3	Ka.C.4
LC1	Selfweight	1.00	1.00	1.00	1.00
LC2	Maintenance	1.00			
LC3	Snow		1.00		
LC4	Live load			1.00	
LC5	Wind				1.00

Tabel NB.5 — Extreme stuwdruk in kN/m² als functie van de hoogte

Hoogte	Gebied I			Gebied II			Gebied III	
m	Kust	Onbebouwd	Bebouwd	Kust	Onbebouwd	Bebouwd	Onbebouwd	Bebouwd
1	0,93	0,71	0,69	0,78	0,60	0,58	0,49	0,48
2	1,11	0,71	0,69	0,93	0,60	0,58	0,49	0,48
3	1,22	0,71	0,69	1,02	0,60	0,58	0,49	0,48
4	1,30	0,71	0,69	1,09	0,60	0,58	0,49	0,48
5	1,37	0,78	0,69	1,14	0,66	0,58	0,54	0,48
6	1,42	0,84	0,69	1,19	0,71	0,58	0,58	0,48
7	1,47	0,89	0,69	1,23	0,75	0,58	0,62	0,48
8	1,51	0,94	0,73	1,26	0,79	0,62	0,65	0,51
9	1,55	0,98	0,77	1,29	0,82	0,65	0,68	0,53
10	1,58	1,02	0,81	1,32	0,85	0,68	0,70	0,56
15	1,71	1,16	0,96	1,43	0,98	0,80	0,80	0,66
20	1,80	1,27	1,07	1,51	1,07	0,90	0,88	0,74

load combinations ULS

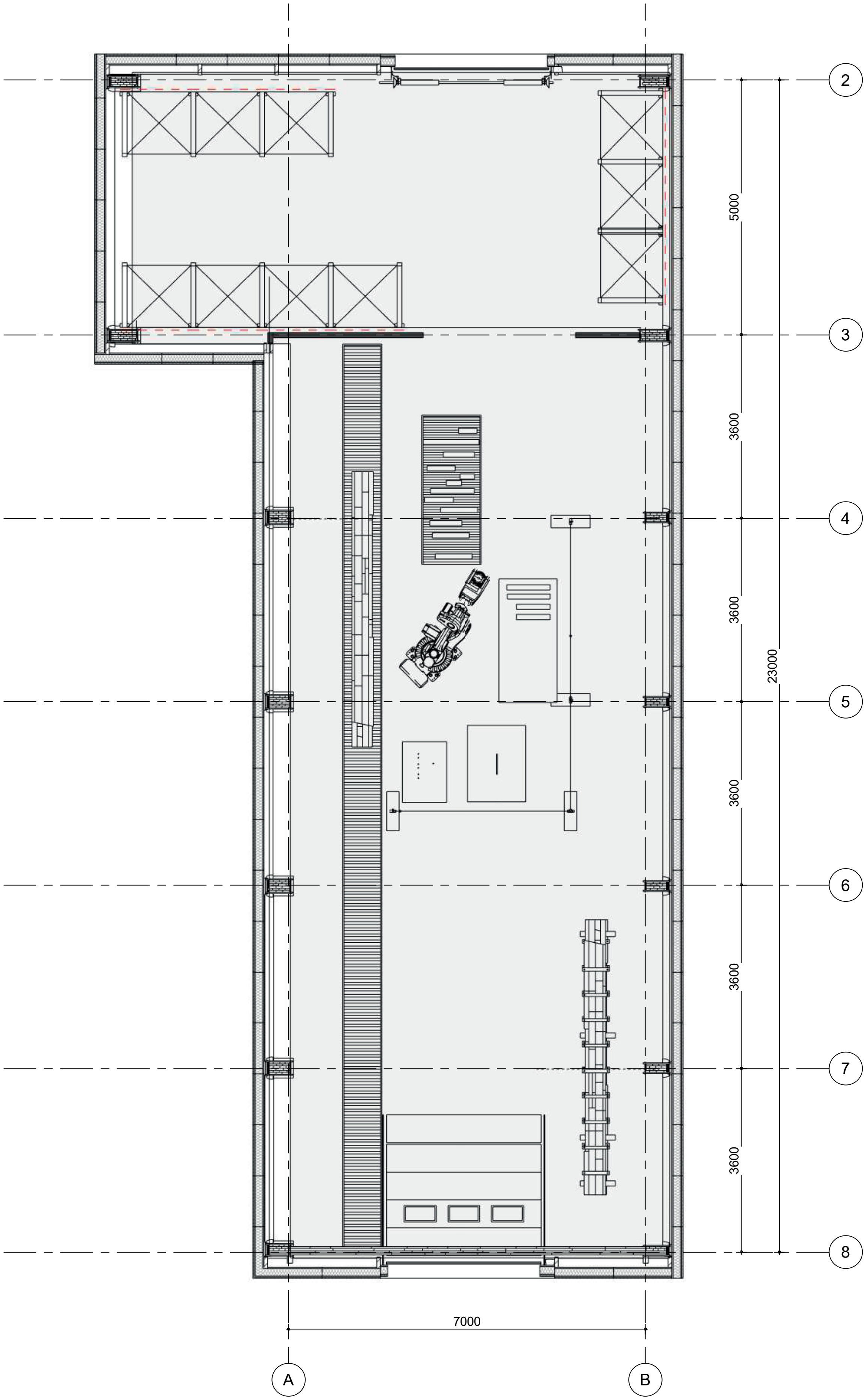
Key and decisive load combination	
Fu.C.3	1.20* LC1 + 1.50* LC3
Fu.C.5	1.20* LC1 + 1.20* LC2 + 1.20* LC3

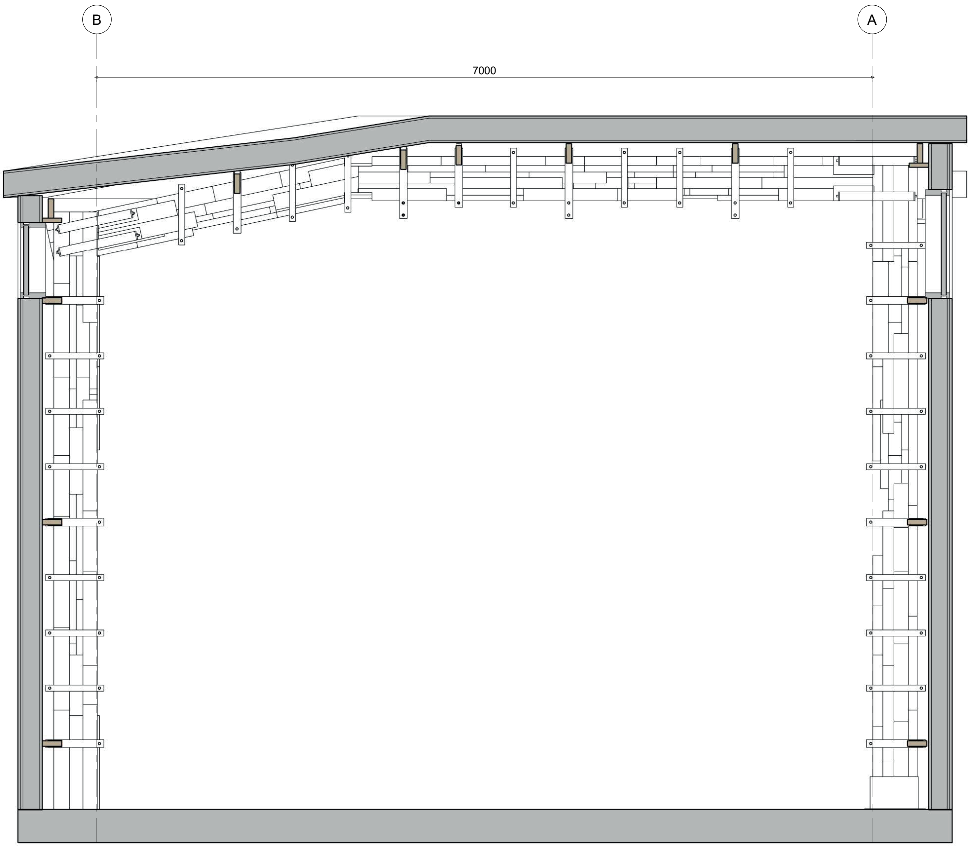
Loads on Portal	With bay size of 5 meters and height of 6 meters used in chapter 9												
	element	number of elements	Lengte m	dikte m	Breedte m	Volume m3	Soortelijk gewicht P_{mean} (kg/m2)	Soortelijk gewicht P_{mean} (kg/m3)	Gewicht kg	Weight on beam in kN	kN/m2	kN/m1 (local)	kN/m1 (global)
Span			10									10.0	10.0
Bay size			5										
height of column			6										
Dead load													
Selfweight	Main beam	1	10	0.32	0.2	0.64		420	268.8	2.64		0.3	
													0.00
	Roof joist	15	5	0.05	0.22	0.06		420	346.50	3.40	3.0902	0.3	0.34
	Roof structural slab (CLT)	1	5	0.12		0.00		420	0.00	0.00		2.5	2.47
	Insulation (EPS)	1	5	0.1				25				0.1	0.12
	PV_panels		5				50					*2.5*	2.45
Total S.W.												3.08	**5.39**

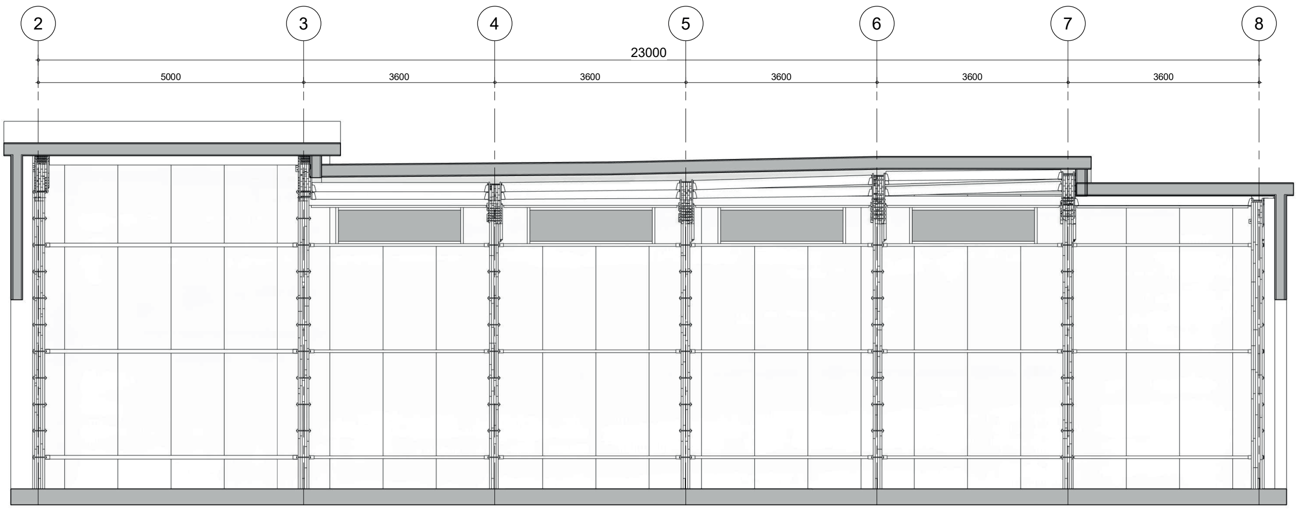
Live load Z direction:													
Snow											0.56		**2.8**
Maintenance only local (excentric)										1			
Live load facade													**1.00**

Live load Y direction:													
Wind load side pressure											0.71	**3.55**	**3.55**
Wind load side suction													**1.78**

Appendix C. Technical drawings of case study




B
A
7000


2
3
4
5
6
7
8
23000
5000
3600
3600
3600
3600
3600

Appendix D. Python code

Creating fake database and send data to SQL

```python
import scriptcontext
pg2 = scriptcontext.sticky['psycopg2']

def setup_database_connection():
    conn = pg2.connect(dbname='Wooddealer_database', user='postgres', password=              ', host='localhost')
    cur = conn.cursor()
    return conn, cur

def close_database_connection(conn, cur):
    cur.close()
    conn.close()

def clear_data(conn, cur):
    try:
        cur.execute("TRUNCATE salvaged_wood RESTART IDENTITY")
        conn.commit()
        print('Data removed')
    except pg2.Error as e:
        print("Error clearing data: {}".format(e))

def apply_data(conn, cur, waste_wood_data):
    try:
        for piece_of_wood in waste_wood_data:
            # Ensure each piece_of_wood is a tuple with five elements
            if isinstance(piece_of_wood, tuple) and len(piece_of_wood) == 6:
                # Extract values from the tuple
                id_wood_piece, length, width, depth, strength_grade, painted = piece_of_wood

                # Convert the dimensions to integers
                id_wood_piece, length, width, depth = map(int, (id_wood_piece, length, width, depth))

                # Insert data into the table, including the new column
                cur.execute("INSERT INTO salvaged_wood (id_wood_piece, length_wood_mm, width_wood_mm, depth_wood_mm, strength_grade, painted) \
                             VALUES (%s, %s, %s, %s, %s, %s)", (id_wood_piece, length, width, depth, str(strength_grade), painted))
            else:
                print("Invalid data structure.")

        conn.commit()
        print('Data import success')
        return 'Data import success'
    except Exception as e:
        print("Error during data import:", e)
        conn.rollback()

conn, cur = setup_database_connection()

if Apply:
    clear_data(conn, cur)
    apply_data(conn, cur, waste_wood_data)

close_database_connection(conn, cur)
```

Importing data form SQL database

```
import scriptcontext
pg2 = scriptcontext.sticky['psycopg2']

def setup_database_connection():
    conn = pg2.connect(dbname='Wooddealer_database', user='postgres', password=              , host='localhost')
    cur = conn.cursor()
    return conn, cur

def close_database_connection(conn, cur):
    cur.close()
    conn.close()

def import_data(conn, cur, id_wood, length, width, depth, strength_grade, painted, length_range, width_range, import_limit):
    try:
        cur.execute('SELECT * FROM salvaged_wood WHERE length_wood_mm BETWEEN %s AND %s AND width_wood_mm BETWEEN %s AND %s AND d
        data_set = cur.fetchall()
        print('Data imported. Number of rows: {}'.format(len(data_set)))
        for data in data_set:
            id_wood.append(data[0])
            length.append(data[1])
            width.append(data[2])
            depth.append(data[3])
            strength_grade.append(data[4])
            painted.append(data[10])
    except pg2.Error as e:
        print("Error during data import: {}".format(e))
    except Exception as e:
        print("An unexpected error occurred: {}".format(e))

# Create lists to store data
id_wood = []
length = []
width = []
depth = []
strength_grade = []
painted = []

# Establish a database connection
conn, cur = setup_database_connection()

try:
    if Apply[0]:
        import_data(conn, cur, id_wood, length, width, depth, strength_grade, painted, length_range, width_range, import_limit)
except Exception as e:
    # Print any errors that occur during the database operations
    print("An error occurred: {}".format(e))
finally:
    # Close the database connection
    close_database_connection(conn, cur)
```

Filtering and loading a random seeding sample of stock into the program

```python
import random
from collections import Counter

random.seed(sample_value)

purlin_sections = [t for t in purlin_sections if t[3] in purlin_heights]
purlin_sections = [t for t in purlin_sections if t[2] in purlin_widths]
framework_sections = [t for t in framework_sections if t[3] in framework_heights]
framework_sections = [t for t in framework_sections if t[2] in framework_widths]

consider_strength_purlin = percentage_strength_purlin[-1]
consider_strength_framework = percentage_strength_framework[-1]
consider_visualaspect = percentage_painted[-1]

amount_of_purlins = int(total_parts_considerd / 100 * percentage_purlins)
amount_of_framework = total_parts_considerd - amount_of_purlins

structural_classes = ['C14', 'C18', 'C24', 'C30', 'C35']
strength_requirements_purlin = list(zip(structural_classes, percentage_strength_purlin))
strength_requirements_framework = list(zip(structural_classes, percentage_strength_framework))
print(strength_requirements_framework)

# Step 1: Calculate total percentage for purlins and framework
total_percentage_purlin = sum(percentage for _, percentage in strength_requirements_purlin)
total_percentage_framework = sum(percentage for _, percentage in strength_requirements_framework)

# Check if the sum of percentages is equal to 100
if total_percentage_purlin != 100:
    raise ValueError("Error: The sum of strength percentages for purlins must be equal to 100.")
if total_percentage_framework != 100:
    raise ValueError("Error: The sum of strength percentages for framework must be equal to 100.")

# Step 2: Determine number of tuples to extract for each strength grade for purlins
num_to_extract_purlin = {grade: int((amount_of_purlins / 100) * percentage) for grade, percentage in strength_requirements_purlin}

# Step 3: Determine number of tuples to extract for each strength grade for framework
num_to_extract_framework = {grade: int((amount_of_framework / 100) * percentage) for grade, percentage in strength_requirements_framework}

print(num_to_extract_framework)

# Step 4: Check if enough pieces are available for purlins
for grade, count in num_to_extract_purlin.items():
    if sum(1 for tup in purlin_sections if tup[4] == grade) < count:
        print("Not enough pieces available for purlin strength grade " + grade + ".")

# Step 5: Check if enough pieces are available for framework
for grade, count in num_to_extract_framework.items():
    if sum(1 for tup in framework_sections if tup[4] == grade) < count:
        print("Not enough pieces available for framework strength grade " + grade + ".")


# Step 6: Extract tuples randomly based on percentages for purlins
extracted_pieces_purlin = []
if consider_strength_purlin == 1:
    for grade, count in num_to_extract_purlin.items():
        tuples_of_grade = [tup for tup in purlin_sections if tup[4] == grade]
        extracted_tuples = random.sample(tuples_of_grade, count)
        extracted_pieces_purlin.extend(extracted_tuples)
else:
    extracted_pieces_purlin = random.sample(purlin_sections, amount_of_purlins)

# Step 7: Extract tuples randomly based on percentages for framework
extracted_pieces_framework = []
if consider_strength_framework == 1:
    for grade, count in num_to_extract_framework.items():
        tuples_of_grade = [tup for tup in framework_sections if tup[4] == grade]
        extracted_tuples = random.sample(tuples_of_grade, count)
        extracted_pieces_framework.extend(extracted_tuples)
else:
    extracted_pieces_framework = random.sample(framework_sections, amount_of_framework)

# Step 8: Filter extracted framework pieces to include max 20% with length < 0.1
max_len_0_1_count = int(0.2 * len(extracted_pieces_framework))
len_0_1_tuples = [tup for tup in extracted_pieces_framework if tup[1] < 0.1]
other_tuples = [tup for tup in extracted_pieces_framework if tup[1] >= 0.1]

# Limit to 20% with length < 0.1
if len(len_0_1_tuples) > max_len_0_1_count:
    len_0_1_tuples = random.sample(len_0_1_tuples, max_len_0_1_count)

# Combine tuples ensuring the 20% limit
filtered_framework_pieces = len_0_1_tuples + other_tuples
if len(filtered_framework_pieces) > amount_of_framework:
    filtered_framework_pieces = random.sample(filtered_framework_pieces, amount_of_framework)

selected_salvaged_timber = extracted_pieces_purlin + filtered_framework_pieces

# Sorting the Waste_wood tuple based on the Piece_width
selected_salvaged_timber = sorted(selected_salvaged_timber, key=lambda x: (x[2], x[1]))

piece_strengths = [item[4] for item in selected_salvaged_timber]
occurrences_of_strength = Counter(piece_strengths)

# Creating a new list to store piece strengths and their occurrences
piece_strength_occurrences = [(piece_strength, count) for piece_strength, count in occurrences_of_strength.items()]
piece_strength_occurrences = sorted(piece_strength_occurrences)

strength_section_stats = []
for strength, count in piece_strength_occurrences:
    strength_section_stats.append(("Piece Height[mm] :", strength, ", Occurrences:", count))
    print("Piece Height:", strength, ", Occurrences:", count)
```

Initialize dictionaries

```python
import ghpythonlib.treehelpers as gt

# Dictionary to hold sublists based on index[2]
sublists = {}
volume = []
available_widths = []
available_height = []
total_lengths = []  # List to store total lengths for each sublist
height_occurrences = []  # List to store occurrences of heights for each sublist
strength_occurrences = []

for wood in waste_wood:
    # Check if the value at index[2] is already a key in the dictionary
    if wood[2] in sublists:
        # If yes, append the current wood tuple to the existing sublist
        sublists[wood[2]].append(wood)
    else:
        # If no, create a new key with the value at index[2] and initialize a list with the current wood tuple
        sublists[wood[2]] = [wood]

selected_salvaged_wood = sublists.values()

# Calculate total volume and total length for each sublist
for key, sublist in sublists.items():
    total_volume = sum((length * width * height) for id , length, width, height, strength_grade, visual_state in sublist if height >= 0.1)
    volume.append(total_volume)
    available_widths.append(key)
    occurrences_of_height = {}  # Initialize dictionary to store counts of height occurrences for this sublist
    total_length_by_height = {}  # Initialize dictionary to store total length for each height in this sublist
    occurrences_of_strength = {}  # Initialize dictionary to store counts of strength occurrences for this sublist
    
    for id , length, width , height, strength_grade, visual_state in sublist:
        if height in occurrences_of_height:
            occurrences_of_height[height] += 1
            if height in total_length_by_height:
                total_length_by_height[height] += length
            else:
                total_length_by_height[height] = length
        else:
            occurrences_of_height[height] = 1
            total_length_by_height[height] = length

        if strength_grade in occurrences_of_strength:
            occurrences_of_strength[strength_grade] += 1
        else:
            occurrences_of_strength[strength_grade] = 1
    
    height_occurrences.append(occurrences_of_height)
    total_lengths.append(total_length_by_height)
    strength_occurrences.append(occurrences_of_strength)


present_heights = []
for height_dict in height_occurrences:
    present_heights.append(list(height_dict.keys()))

heightgraph = [[1000 * i for i in j] for j in present_heights]

height_count = []
for height_dict in height_occurrences:
    counts = list(height_dict.values())
    height_count.append(counts)

available_inventory_length = []
for length_dict in total_lengths:
    lengths = list(length_dict.values())
    updated_lengths = [length * 0.92 for length in lengths]
    available_inventory_length.append(updated_lengths)

available_heights = gt.list_to_tree(heightgraph)
height_counts = gt.list_to_tree(height_count)
total_lengths_combined = gt.list_to_tree(available_inventory_length)
print available_inventory_length
```

Y-combinatorial problem

```python
import itertools
import math

# Define your initial inventory as a list of tuples (available_width, available_volume)
inventory = list(zip(available_width, available_volume))

purlin_sections = [width for width in available_width if width > 0.045]
framework_sections = [width for width in available_width if width < 0.045]

# Define function to calculate max usage for a single box
def calculate_max_usage_for_box(dimensions, inventory):
    box_width = dimensions[1]
    box_length = dimensions[0]
    box_height = dimensions[2]
    max_usage = []
    limit_purlin_constraint = []
    
    # Define purlin sections and framework sections
    purlin_sections = [width for width in available_width if width > 0.045]
    framework_sections = [width for width in available_width if width < 0.045]

    # Define box types based on box_width
    type_1 = box_width <= 0.26
    type_2 = 0.26 < box_width <= 0.3
    type_3 = 0.3 < box_width <= 0.37
    type_4 = 0.37 < box_width <= 0.45
    type_5 = 0.45 < box_width <= 0.5
    type_6 = box_width > 0.5
    
    # Determine the valid type based on box_width
    if type_1:
        max_purlin_usage = 0
        valid_type = 'type_1'
    elif type_2:
        max_purlin_usage = 1
        valid_type = 'type_2'
    elif type_3:
        max_purlin_usage = 2
        valid_type = 'type_3'
    elif type_4:
        max_purlin_usage = 3
        valid_type = 'type_4'
    elif type_5 or type_6:
        max_purlin_usage = 4
        valid_type = 'type_5' if type_5 else 'type_6'
    
    limit_purlin_constraint.append(max_purlin_usage)
    
    # Calculate max usage considering constraints
    for width, volume in inventory:
        if width in purlin_sections:
            usage = math.floor((volume) / (width * box_length * box_height))
            max_usage.append(min(usage, max_purlin_usage))
        else:
            max_usage.append(math.floor((volume) / (width * box_length * box_height)))
            
    return max_usage, limit_purlin_constraint

# Define function to find optimal layers for a single box
def find_optimal_layers_for_box(dimensions, available_width, inventory):
    original_width = dimensions[1]
    min_diff = float('inf')
    optimal_layers = []
    total_combos = 0
    
    max_usage, limit_purlin_constraint = calculate_max_usage_for_box(dimensions, inventory)
    
    # Create a new list of widths based on the maximum usage of each width
    new_widths = []
    for width, usage in zip(available_width, max_usage):
        new_widths.extend([width] * int(usage))

    # Calculate the maximum and minimum combination length
    max_width = max(available_width)
    min_width = min(available_width)
    max_combination_length = int(math.ceil((dimensions[1] - (limit_purlin_constraint[0] * max_width))/ min_width + limit_purlin_constraint[0]))
    min_combination_length = int(math.floor((dimensions[1] - (limit_purlin_constraint[0] * max_width))/ min_width + limit_purlin_constraint[0]))

    # Generate all possible combinations of widths from the new list of widths
    for num_layers in range(min_combination_length, max_combination_length + 1):
        for combo in itertools.combinations(new_widths, num_layers):
            # Count the number of purlin sections in the combination
            purlin_count = sum(1 for width in combo if width in purlin_sections)
            
            # Check if the number of purlin sections exceeds the limit
            if purlin_count > limit_purlin_constraint[0]:
                continue

            total_combos += 1
            total_width = sum(combo)
            
            # Check if the total width is greater than or equal to the original width
            if total_width < original_width:
                continue
                
            # Calculate the difference between the total width and the original width
            diff = abs(total_width - original_width)

            # Update the optimal layers if the combination is closer to the original width
            if diff < min_diff:
                min_diff = diff
                optimal_layers = list(combo)
                chosen_sum = total_width

    # Sorting list smaller sections, outside, larger section in centre.
    # Sort the optimal layers in descending order
    optimal_layers.sort(reverse=True)
    
```

```
    # Divide the sorted list into two halves
    list_half_1 = []
    list_half_2 = []
    for i, val in enumerate(optimal_layers):
        if i % 2 == 0:
            list_half_1.append(val)
        else:
            list_half_2.append(val)
    
    list_half_1.reverse()
    sorted_optimal_layers = list_half_1 + list_half_2

    # If no valid combination is found, raise an error message
    if not optimal_layers:
        raise ValueError("No valid combination found, adjust inventory")

    return sorted_optimal_layers

# Define function to calculate volume per layer for a single box
def calculate_volume_per_layer_for_box(dimensions, layer_distribution):
    box_length = dimensions[0]
    box_height = dimensions[2]
    volume_per_layer = []
    for width in layer_distribution:
        volume = width * box_length * box_height
        volume_per_layer.append(volume)
    return volume_per_layer

layer_width_composition = []

# Loop through each box in box_dimensions
for dimensions in beam_dimensions:
    if not dimensions:  # Check if dimensions list is empty
        continue
    try:
        layer_distribution = find_optimal_layers_for_box(dimensions, available_width, inventory)
        layer_width_composition. append(layer_distribution)
        print("Optimal Layers:", layer_distribution)

        # Calculate the total width for the current box
        total_width = sum(layer_distribution)
        print("Total width for box with dimensions {}: {}".format(dimensions, total_width))

        # Calculate max usage for the current box
        max_usage, _ = calculate_max_usage_for_box(dimensions, inventory)
        print("Max usage for box with dimensions {}: {}".format(dimensions, max_usage))

        # Call the function to calculate volume per layer for the current box
        volume_per_layer = calculate_volume_per_layer_for_box(dimensions, layer_distribution)

        # Update the available_volume in inventory after creating the current box
        for width_used, volume_used in zip(layer_distribution, volume_per_layer):
            for i, (width_inventory, volume_inventory) in enumerate(inventory):
                if width_used == width_inventory:
                    inventory[i] = (width_inventory, volume_inventory - volume_used)
                    break
        
_layer_width_composition = layer_width_composition
print _layer_width_composition

y_coords = []

# calcualte y_coordiantes
for i in layer_width_composition:
    y_coords_layer = [0]  # Initialize a new sublist for cumulative sums
    start_value = 0  # Initialize a variable to store the cumulative sum
    for j in i:
        start_value += j  # Update the cumulative sum
        y_coords_layer.append(start_value)  # Append the cumulative sum to the sublist
    y_coords.append(y_coords_layer[:-1])  # Append the cumulative sublist to the cumulative list

global_y_coordinates = y_coords

y_coordinates_column_elements = global_y_coordinates[-2:]
y_coordinates_beam_elements = global_y_coordinates[:-2]

y_coords_beam_a = global_y_coordinates[0]
if len(global_y_coordinates) >= 2:
    y_coords_beam_b = global_y_coordinates[1]
else:
    y_coords_beam_b = []

# Calculate sum of each sublist
global_beam_widths = []
for sublist in layer_width_composition:
    sublist_sum = sum(sublist)
    global_beam_widths.append(sublist_sum)
```

Z-combianatorial problem

```python
import copy
import itertools
import math

## Iterate through sorted_wood
## Create a dictionary using a dictionary comprehension
dict_height_sections = dict_height_sections = {(float_val): list_val for float_val, list_val in zip(width_keys, available_heights)}
dict_available_inventory_length = {(float_val): list_val for float_val, list_val in zip(width_keys, available_inventory_length)}

max_heights = [max(i) for i in available_heights]
max_height = max(max_heights, key=abs)
min_heights = [min(i) for i in available_heights]
min_height = (min(min_heights, key=abs)) + 0.03

def find_optimal_layers(beam_dimensions, width_distribution, dict_height_sections, dict_available_inventory_length):
    optimal_height_distributions_all = []
    z_coords_all = []  # List to store z coordinates for each beam
    z_incrementations_per_layer_all = []
    
    remaining_inventory = {}  # Initialize remaining inventory dictionary for each beam
    
    for beam_index, dimensions in enumerate(beam_dimensions):
        box_length = dimensions[0]
        original_box_height = dimensions[2]  # Store the original box height

        optimal_height_distributions = []
        z_coords = []  # List to store z coordinates for each layer in the beam
        z_incrementations_per_layer = []
        
        # Flag to track whether min_combination_length has been reduced
        reduced_combination_length = False
        # Flag to track whether z_incrementation has been performed
        z_incrementation_made = False

        # Calculate the maximum and minimum combination length for the current beam
        max_combination_length = int(math.ceil(original_box_height / min_height))
        min_combination_length = int(math.ceil(original_box_height / max_height))
        
        layer_distribution = width_distribution[beam_index]  # Get layer distribution for the current beam

        for layer_num, layer_width in enumerate(layer_distribution):
            available_heights = dict_height_sections[layer_width]  # available heights for the current layer
            inventory_lengths = dict_available_inventory_length[layer_width]  # available inventory lengths for the current layer

            original_width = layer_width
            min_diff_layer = float('inf')
            optimal_layers_layer = []

            if layer_width not in remaining_inventory:
                remaining_inventory[layer_width] = dict(zip(available_heights, inventory_lengths))

            # Set current box height for the current layer
            box_height = original_box_height

            best_combo = None
            best_z_coords = None
            best_diff = float('inf')

            total_combinations = 0
            best_remaining_space = float('inf')
            found_valid_combination = False
            num_valid_combinations = 0  # Counter for valid combinations considered

            z_increment = 0
            while True:
                print("Layer:", layer_num)
                print("Minimum Combination Length:", min_combination_length)
                print("Maximum Combination Length:", max_combination_length)
                print("Z Increment:", z_increment)
                #print("Box Height:", box_height)

                # Generate combinations for each length within the specified range
                combinations = []
                for num_layers in range(min_combination_length, max_combination_length + 1):
                    combinations.extend(list(itertools.product(available_heights, repeat=num_layers)))
                
                
                for combo in combinations:
                    total_combinations += 1
                    total_height = sum(combo)  # total height of the layers
                
                    # Check if there is sufficient inventory for the selected combination
                    valid_inventory = True
                    for height in set(combo):
                        total_avaialbe_inventory_for_height = combo.count(height) * box_length
                        if remaining_inventory[layer_width][height] < total_avaialbe_inventory_for_height:
                            valid_inventory = False
                            break

                    if total_height >= box_height + 0.04:
                        continue
            
                    if valid_inventory:
                        # Check the constraint: heights smaller than 0.1 can only be at the start or end
                        if any(height < 0.1 for height in combo):
                            if any(height < 0.1 and i not in [0, len(combo) - 1] for i, height in enumerate(combo)):
                                continue
                                
```

Z-combianatorial problem

```python
                        z_coords_layer = [sum([z_increment] + list(combo[:i + 1])) for i in range(len(combo))]

                        # Check for clashes
                        if not any(check_clashes(z_coords[:layer_num] + [z_coords_layer], z_incrementations_per_layer)):
                            # Calculate remaining space
                            remaining_space = abs(box_height - sum(combo))
                            num_valid_combinations += 1  # Increment counter for valid combinations

                            # Check if the current combination is better than the current best combination
                            if (remaining_space < best_remaining_space or
                                    (remaining_space == best_remaining_space and len(combo) > len(best_combo)) or
                                    (remaining_space == best_remaining_space and len(combo) == len(best_combo) and len(set(combo)) > len(set(best_combo)))):
                                    #(remaining_space == best_remaining_space and len(combo) == len(best_combo) and len(set(combo)) > len(set(best_combo))) or
                                    #(remaining_space == best_remaining_space and len(combo) > len(best_combo))):
                                best_remaining_space = remaining_space

                if found_valid_combination: #and best_remaining_space < 0.09
                    break
                else:
                    # No valid combination found, adjust parameters
                    if z_increment < 0.03:
                        #if any(height < 0.1 for height in optimal_height_distributions[-1]):
                        if optimal_height_distributions and optimal_height_distributions[-1][0] < 0.1:
                            z_increment += 0.02  # If previous layer has height < 0.08, use z_increment of 0.02
                        else:
                            z_increment += 0.03
                        box_height -= z_increment  # Decrease box_height
                        z_incrementation_made = True  # Set flag for z_incrementation
                        # Reduce min_combination_length only once per beam and only if z_incrementation is made
                        if not reduced_combination_length and z_incrementation_made:
                            min_combination_length -= 1
                            reduced_combination_length = True
                    elif z_increment <= 0.1:
                        z_increment += 0.02
                        box_height -= z_increment  # Decrease box_height
                        z_incrementation_made = True  # Set flag for z_incrementation
                    else:
                        raise ValueError("No valid combination found even after adjusting starting z_coordinate.")

            if found_valid_combination:
                for height in best_combo:
                    remaining_inventory[layer_width][height] -= box_length
                optimal_height_distributions.append(list(best_combo))
                z_coords.append(best_z_coords)
                z_incrementations_per_layer.append(z_increment)

                # Print z_coords_layer for the chosen combination
                print("Chosen combination for layer {} in beam {}: {}".format(layer_num, beam_index, best_combo))
                print("Number of Valid Combinations Considered:", num_valid_combinations)

            else:
                # Handle the case where no valid combination is found
                print("No valid combination found for Beam {} and layer {}. Skipping this layer.".format(beam_index, layer_num))
                continue

            # Print remaining inventory for the current layer
            print("Remaining inventory after layer {} in beam {}: {}".format(layer_num, beam_index, remaining_inventory[layer_width]))

            # Calculate and print remaining height after filling the current layer
            remaining_height = box_height - sum(best_combo)
            print("Remaining height after layer {} in beam {}: {}".format(layer_num, beam_index, remaining_height))

        # Append the optimal distributions and z_coords for the current beam
        optimal_height_distributions_all.append(optimal_height_distributions)
        z_coords_all.append(z_coords)
        z_incrementations_per_layer_all.append(z_incrementations_per_layer)

    return optimal_height_distributions_all, z_coords_all, z_incrementations_per_layer_all
```

```python
def check_clashes(z_coords, z_incrementations_per_layer):
    clashes = []
    for i in range(1, len(z_coords)):
        prev_layer_z = z_coords[i - 1]
        current_layer_z = z_coords[i]
        z_increment_prev_layer = z_incrementations_per_layer[i - 1]

        # Check clashes for the first item in the current combo with both z_coordinates and z_incrementation of the previous layer
        prev_z = prev_layer_z[0]
        prev_z_incremented = z_increment_prev_layer
        for curr_z in current_layer_z:
            if abs(curr_z - prev_z) <= 0.049 or abs(curr_z - prev_z_incremented) <= 0.049:
                clashes.append((i - 1, i))
                break

        # Check clashes for the rest of the items in the current combo with the z_coords of the previous layer
        for prev_z in prev_layer_z[:-1]:  # Exclude the last z-coordinate of the previous layer
            for curr_z in current_layer_z:
                if abs(curr_z - prev_z) <= 0.049:
                    clashes.append((i - 1, i))
                    break

    return clashes
```

```python
try:
    height_distribution, z_coordinates, z_incrementations_per_layer_all = find_optimal_layers(beam_dimensions, width_distribution, dict_height_sections, dict_available_inventory_length)
    print("Optimal height distributions:", height_distribution)
    print("Z coordinates:", z_coordinates)
except ValueError as e:
    print(e)

z_coordinates = [[i[:-1] for i in j] for j in z_coordinates]
# Transform z_coords to include z-coordinates for each layer in each beam
z_coords = []
for beam_index, beam_z_coords in enumerate(z_coordinates):
    beam_z_coords_transformed = []
    if beam_index < len(z_incrementations_per_layer_all):
        for layer_index, layer_z_coords in enumerate(beam_z_coords):
            if layer_index < len(z_incrementations_per_layer_all[beam_index]):
                layer_start_point = z_incrementations_per_layer_all[beam_index][layer_index]
                layer_z_coords_transformed = [layer_start_point] + list(layer_z_coords)
                beam_z_coords_transformed.append(layer_z_coords_transformed)
            else:
                print("Warning: Layer index {} exceeds the length of z_incrementations_per_layer_all[{}]".format(layer_index, beam_index))
    else:
        print("Warning: Beam index {} exceeds the length of z_incrementations_per_layer_all".format(beam_index))
    z_coords.append(beam_z_coords_transformed)


mid_z_points = []

for beam_z_coords, beam_heights in zip(z_coords, height_distribution):
    coord_mid_points = []
    # Iterate over each layer in the beam
    for layer_z_coords, layer_heights in zip(beam_z_coords, beam_heights):
        # Adjust the z-coordinate to represent the midpoint of the layer
        z_mid = [z + height / 2 for z, height in zip(layer_z_coords, layer_heights)]
        coord_mid_points.append(z_mid)
    mid_z_points.append(coord_mid_points)

global_z_coords = z_coords
z_coords_beam_elements = z_coords[:-2]
z_coords_column_elements = z_coords[-2:]

height_distribution_column_elements = height_distribution[-2:]
height_distribution_beam_elements = height_distribution[:-2]
```

Dynamic programming arrangement

```python
import copy
import itertools
import math
from itertools import chain

dict_inventory = {(float_val): list_val for float_val, list_val in zip(width_keys, sorted_wood)}
length_margin = 0.8
single_beam = len(beam_dimensions)

def dynamic_arrangement(dict_inventory, beam_dimensions, width_distribution, height_distribution, z_coords):
    inventory_copy = copy.deepcopy(dict_inventory)
    
    arrangements_all_beams = []
    strength_info_all_beams = []
    start_points_all_beams = []
    global_used_blocks_all_beams = []

    for beam_index, (dimensions, width_distribution_beam, height_distribution_beam, z_coords_beam) in enumerate(zip(beam_dimensions, width_distribution, height_distribution, z_coords)):
        box_length = dimensions[0]
        box_height = dimensions[2]
        element_type = dimensions[3]
        layer_keys = width_distribution_beam
        height_distribution_per_layer = height_distribution_beam
        arrangements_per_layer = []
        global_used_blocks = set()
        strength_info = []
        start_points = []

        lengths_of_crossections_per_layer = [[box_length + box_height] * len(count) for count in height_distribution_per_layer]

        num_layers_start_odd = math.ceil(len(lengths_of_crossections_per_layer) / 2)
        num_layers_start_odd = int(num_layers_start_odd)
        start_point_odd = (box_length - (box_length + box_height))
        end_point_even = box_length

        for i, length_list in enumerate(lengths_of_crossections_per_layer):
            if single_beam == 3 and element_type == 'beam':
                # Update length for layers smaller than num_layers_start_odd
                if i < num_layers_start_odd:
                    lengths_of_crossections_per_layer[i] = [box_length + box_height*2] * len(length_list)
                else:
                    # Update length for layers equal to or larger than num_layers_start_odd
                    lengths_of_crossections_per_layer[i] = [end_point_even] * len(length_list)
            else:
                # Update length for layers equal to or larger than num_layers_start_odd
                if i >= num_layers_start_odd:
                    lengths_of_crossections_per_layer[i] = [end_point_even] * len(length_list)
                    
        for i, length_list in enumerate(lengths_of_crossections_per_layer):
            if element_type == 'column':
                if i == num_layers_start_odd -1:
                    lengths_of_crossections_per_layer[i] = [end_point_even + box_height - 0.2] * len(length_list)
        print lengths_of_crossections_per_layer
        dp_table_mask = []
        for layer_index, (layer_key, layer_lengths, width_distribution_beam, layer_z_coords) in enumerate(zip(layer_keys, lengths_of_crossections_per_layer, height_distribution_per_layer, z_coords_beam)):
            rows = [[] for _ in range(len(width_distribution_beam))]
            composition = [[] for _ in range(len(width_distribution_beam))]
        used_blocks = {i: set() for i in range(len(width_distribution_beam))}
        block_coordinates = []
        row_x_coords = [0] * len(layer_lengths)

        if element_type == 'column':
            if layer_index < num_layers_start_odd :
                for i in range(len(layer_lengths)):
                    row_x_coords[i] = start_point_odd
        else:
            if layer_index < num_layers_start_odd:
                for i in range(len(layer_lengths)):
                    row_x_coords[i] = start_point_odd

        start_points.append(row_x_coords[:])

        low_strength_count = 0
        high_strength_count = 0

        for i, (row_count_value, row_z_coord) in enumerate(zip(width_distribution_beam, layer_z_coords)):
            if element_type == 'beam':
                if abs(box_height / 3 * 2) < row_z_coord:
                    region = "HSR"
                    high_strength_count += 1
                else:
                    region = "LSR"
                    low_strength_count += 1
            elif element_type == 'column':
                if abs(box_height / 4 * 3) < row_z_coord or abs(box_height / 4 * 1) > row_z_coord:
                    region = "HSR"
                    high_strength_count += 1
                else:
                    region = "LSR"
                    low_strength_count += 1

            # Filter available blocks based on region and beam's used blocks
            available_blocks = [block for block in inventory_copy[layer_key] if block[0] not in global_used_blocks and block[3] == row_count_value]
            #sorted_blocks = available_blocks
            if element_type == 'beam':
                if region == "HSR":
                    sorted_blocks = sorted(available_blocks, key=lambda x: (
                        x[4],
                        x[1] if x[4] in ["C24", "C30", "C35"] else x[1]
                    ), reverse=True)
                else:
                    sorted_blocks = sorted(available_blocks, key=lambda x: (
                        x[4],
                        -x[1] if x[4] in ["C18", "C24", "C30", "C35"] else -x[1]
                    ), reverse=False)
            else:
                if region == "HSR":
                    sorted_blocks = sorted(available_blocks, key=lambda x: (
                        x[4],
                        x[1] if x[4] in ["C24", "C30", "C35"] else x[1]
                    ), reverse=True)

            else:
                sorted_blocks = sorted(available_blocks, key=lambda x: (
                    x[4],
                    -x[1] if x[4] in ["C18", "C24", "C30", "C35"] else -x[1]
                ), reverse=False)
```

Dynamic programming arrangement

```python
                # Initialize DP table and populate cells
                dp_table = [[0] * (int((layer_lengths[i] + length_margin) * 100) + 1) for _ in range(len(sorted_blocks) + 1)]
                for j in range(1, len(sorted_blocks) + 1):
                    for k in range(1, int((layer_lengths[i] + length_margin) * 100) + 1):
                        block_index, block_length, block_width, block_height, block_strength, block_visual_state = sorted_blocks[j - 1]
                        if block_length * 100 <= k:
                            if dp_table[j - 1][k - int(block_length * 100)] + block_length > dp_table[j - 1][k]:
                                dp_table[j][k] = dp_table[j - 1][k - int(block_length * 100)] + block_length
                            else:
                                dp_table[j][k] = dp_table[j - 1][k]
                        else:
                            dp_table[j][k] = dp_table[j - 1][k]

                arrangement_valid = dp_table[len(sorted_blocks)][k] >= layer_lengths[i]
                # Iterate through each value in the bottom row of the DP table
                dp_mask = []
                for j in range(len(dp_table[len(sorted_blocks)])):
                    # Check if the value is greater than or equal to the corresponding layer length
                    if dp_table[len(sorted_blocks)][j] >= layer_lengths[0]:
                        dp_mask.append(True)
                    else:
                        dp_mask.append(False)
                dp_table_mask.append(dp_mask)

                target_length = dp_mask.index(True) if True in dp_mask else None
                if target_length is None:
                    raise ValueError("Insufficient length available to create row. Please check the inventory or consider increasing the length_margin.")

                # initialize start cell in dp (cell that is closest but larger or equal to target length)
                k = target_length

                # start backtracking process to get the blocks of the optimal combination
                blocks_used = []
                for j in range(len(sorted_blocks), 0, -1):
                    if dp_table[j][k] != dp_table[j - 1][k]:
                        block_index, block_length, block_width, block_height, block_strength, block_visual_state = sorted_blocks[j - 1]
                        blocks_used.append((block_index, block_length, block_strength))
                        k -= int(block_length * 100)

                for block_index, block_length, block_strength in blocks_used:
                    used_blocks[i].add(block_index)
                    global_used_blocks.add(block_index)
                    block_coordinates.append((block_index, row_x_coords[i], row_z_coord))
                    rows[i].append((block_index, block_length, row_x_coords[i], row_z_coord))
                    composition[i].append((block_index, block_strength, region, block_length))
                    row_x_coords[i] += block_length

            excess_length = [row_length - sum(length for block_id, length, x_coord, z_coord in row) for row_length, row in zip(layer_lengths, rows)]
            arrangements_per_layer.append((rows, excess_length, used_blocks, block_coordinates, row_x_coords))
            strength_info.append(composition)
        arrangements_all_beams.append(arrangements_per_layer)
        strength_info_all_beams.append(strength_info)
        start_points_all_beams.append(start_points)
        global_used_blocks_all_beams.append(global_used_blocks)

        # Update inventory_copy for the next beam
        for layer_key, blocks in inventory_copy.items():
            blocks_to_remove = []
            for block in blocks:
                if block[0] in global_used_blocks:
                    blocks_to_remove.append(block)
            for block in blocks_to_remove:
                blocks.remove(block)

    return arrangements_all_beams, strength_info_all_beams, start_points_all_beams, global_used_blocks_all_beams, dp_table_mask

arrangements, strength_info, start_points, global_used_blocks, dp_table_mask = dynamic_arrangement(dict_inventory, beam_dimensions, width_distribution, height_distribution, z_coords)

start_points_per_layer = start_points
used_pieces_all_beams = []
excess_length_all_beams = []
pieces_per_layer_all_beams = []
composition_all_beams = []

for beam_index, (arrangements_beam, strength_info_beam) in enumerate(zip(arrangements, strength_info)):
    used_pieces_beam = []
    excess_length_beam = []
    pieces_per_layer_beam = []

    for layer_index, arrangement in enumerate(arrangements_beam):
        layers, excess_length, used_blocks, block_coordinates, layer_x_coords = arrangement
        print("Arrangements in layer", layer_index + 1, "of beam", beam_index + 1, ":")
        block_count = sum(len(used_blocks[i]) for i in used_blocks)
        pieces_per_layer_beam.append(block_count)
        for i, row in enumerate(layers):
            if row:
                for block in row:
                    used_pieces_beam.append(block[0])
            else:
                print("No blocks placed in this layer")
            print("Excess_length in row", i + 1, ":", excess_length[i])
            excess_length_beam.append(excess_length[i])
        print("Used blocks in layer:")
        for i, used_blocks_in_layer in used_blocks.items():
            print("Row", i + 1, ":", used_blocks_in_layer)

    used_pieces_all_beams.append(used_pieces_beam)
    excess_length_all_beams.append(excess_length_beam)
    pieces_per_layer_all_beams.append(pieces_per_layer_beam)
    composition_all_beams.append(strength_info_beam)

x_coords_all_beams = []

for arrangements_beam in arrangements:
    x_coords_beam = []
    for arrangement in arrangements_beam:
        layers, _, _, block_coordinates, layer_x_coords = arrangement
        layer_x_coordinates = []
        for row in layers:
            row_x_coordinates = [block[2] for block in row]
            layer_x_coordinates.append(row_x_coordinates)
        x_coords_beam.append(layer_x_coordinates)
    x_coords_all_beams.append(x_coords_beam)

global_used_pieces = list(chain.from_iterable(used_pieces_all_beams))

pieces_per_layer_column_elements = pieces_per_layer_all_beams[-2:]
pieces_per_layer_beam_elements = pieces_per_layer_all_beams[:-2]
global_pieces_per_layer = pieces_per_layer_all_beams
```

Cutting parts with excess length

```
length_margin = 0.3
double_cut = 0.5

double_cut_pieces = []

for sublist in excess_length_all_beams:
    double_cut_sublist = []
    for value in sublist:
        double_cut_sublist.append(abs(value) >= double_cut)
    double_cut_pieces.append(double_cut_sublist)
    

start_beams = []
# Iterate over each sublist in composition_all_beams
for sublist_outer, double_cut_sublist in zip(composition_all_beams, double_cut_pieces):
    bool_index = 0  # Reset bool_index for each sublist
    last_tuples_sublist = []
    for sub_sublist, double_cut_value in zip(sublist_outer, double_cut_sublist):
        for sub_sub_sublist in sub_sublist:
            if not double_cut_value:
                last_tuples_sublist.append(sub_sub_sublist[0])
            else:
                last_tuples_sublist.extend(sub_sub_sublist[:2])
            bool_index += 1
    start_beams.append(last_tuples_sublist)

excess_length_double_cut = []

for sublist, cuts in zip(excess_length_all_beams, double_cut_pieces):
    new_sublist = []
    for value, cut in zip(sublist, cuts):
        if cut:  # If cut is True, replace the original value with two new values
            new_sublist.extend([-length_margin, value -- length_margin])
        else:
            new_sublist.append(value)
    excess_length_double_cut.append(new_sublist)

cut_for_disposal = []
length_reduction_for_disposal = []
length_reduction_for_reuse = []

for sublist in excess_length_double_cut:
    cut_for_disposal_sublist = []
    for value in sublist:
        cut_for_disposal_sublist.append(abs(value) <= length_margin - 0.001)
        if abs(value) <= length_margin - 0.001:
            length_reduction_for_disposal.append(value)
        else:
            length_reduction_for_reuse.append(value)
    cut_for_disposal.append(cut_for_disposal_sublist)

cut_for_disposal_beams = []
cut_for_reuse_beams = []

# Iterate over each sublist in cut_for_disposal and composition_all_beams simultaneously
for cut_for_disposal_sublist, beams_sublist in zip(cut_for_disposal, start_beams):
    # Iterate over each boolean value and corresponding tuple
    for dispose, beam in zip(cut_for_disposal_sublist, beams_sublist):
        # If dispose is True, append the corresponding beam tuple to cut_for_disposal_beams
        if dispose:
            cut_for_disposal_beams.append(beam)
        else:
             cut_for_reuse_beams.append(beam)
             

# Create new tuples based on index [0] and values in the cut lists
pieces_for_reuse = [(cut_for_reuse_beams[i][0], length_reduction_for_reuse[i]) for i in range(min(len(cut_for_reuse_beams), len(length_reduction_for_reuse)))]
pieces_for_disposal = [(cut_for_disposal_beams[i][0], length_reduction_for_disposal[i]) for i in range(min(len(cut_for_disposal_beams), len(length_reduction_for_disposal)))]

cutoff_waste = [tup for tup in pieces_for_disposal if -0.001 > tup[1]]
uniform_distribution = [tup for tup in pieces_for_disposal if -0.001 < tup[1]]
cutoff_reuse = []

# Create a dictionary for faster lookup
pieces_dict = dict(pieces_for_reuse)
```

Cutting parts with excess length

```python
# Iterate over each sublist in list_of_sublists
for sublist in inventory:
    # Iterate over each tuple in the sublist
    for tup in sublist:
        # Check if the ID of the tuple matches any ID in pieces_for_reuse
        if tup[0] in pieces_dict:
            # Get the length value from pieces_for_reuse
            length = abs(pieces_dict[tup[0]])
            # Update the tuple with the new length
            new_tup = (int("77" + str(tup[0])), length, tup[2], tup[3],tup[4] )
            # Add the updated tuple to cutoff_reuse list
            cutoff_reuse.append(new_tup)
updated_composition = []

# Create dictionaries for faster lookup
pieces_dict_reuse = dict(pieces_for_reuse)
pieces_dict_disposal = dict(cutoff_waste)

# Iterate over each sublist in composition_all_beams
for sublist_outer in composition_all_beams:
    updated_sublist_outer = []
    # Iterate over each sub_sublist in the sublist
    for sub_sublist in sublist_outer:
        updated_sub_sublist = []
        # Iterate over each sub_sub_sublist in the sub_sublist
        for sub_sub_sublist in sub_sublist:
            updated_sub_sub_sublist = []
            # Iterate over each tuple in the sub_sub_sublist
            for tup in sub_sub_sublist:
                beam_id = tup[0]  # Get the ID of the tuple
                # Check if the tuple ID matches any ID in pieces_for_reuse
                if beam_id in pieces_dict_reuse:
                    # Update the length value
                    updated_sub_sub_sublist.append((beam_id, tup[1], tup[2], tup[3] + pieces_dict_reuse[beam_id]))
                # Check if the tuple ID matches any ID in pieces_for_disposal
                elif beam_id in pieces_dict_disposal:
                    # Update the length value
                    updated_sub_sub_sublist.append((beam_id, tup[1], tup[2], tup[3] + pieces_dict_disposal[beam_id]))
                else:
                    # If no match is found, keep the original tuple
                    updated_sub_sub_sublist.append(tup)
            updated_sub_sublist.append(updated_sub_sub_sublist)
        updated_sublist_outer.append(updated_sub_sublist)
    updated_composition.append(updated_sublist_outer)

updated_inventory = []
```

```python
# Iterate over each sublist in inventory
for sublist in inventory:
    updated_sublist = []
    # Iterate over each tuple in the sublist
    for tup in sublist:
        beam_id = tup[0]  # Get the ID of the tuple
        # Check if the tuple ID matches any ID in pieces_for_reuse
        if beam_id in pieces_dict_reuse:
            # Update the length value
            updated_sublist.append((beam_id, tup[1] + pieces_dict_reuse[beam_id], tup[2], tup[3], tup[4]))
        # Check if the tuple ID matches any ID in pieces_for_disposal
        elif beam_id in pieces_dict_disposal:
            # Update the length value
            updated_sublist.append((beam_id, tup[1] + pieces_dict_disposal[beam_id], tup[2], tup[3], tup[4]))
        else:
            # If no match is found, keep the original tuple
            updated_sublist.append(tup)
    updated_inventory.append(updated_sublist)
```

```
from copy import deepcopy

# Deepcopy composition for manipulation
composition_copy = deepcopy(composition)

# Define a dictionary to map string values to numeric values
value_mapping = {'C14': 14, 'C18': 18, 'C24': 24, 'C30': 30, 'C35': 35}

#Function to get the numeric value of the second element in the tuple
def get_numeric_value(item):
    return value_mapping.get(item[1])

# Sorting each sub-sublist based on the second element of each tuple
for beams_index, beams in enumerate(composition_copy):
    for sublist_index, sublist in enumerate(beams):
        for sub_sublist_index, sub_sublist in enumerate(sublist):

            # Sort based on the third element of the first tuple
            if sub_sublist[0][2] == 'LSR':
                sub_sublist.sort(key=get_numeric_value, reverse=False)  # Sort in descending order

                # Apply the custom sorting logic
                sorted_sub_sublist = []
                half_1 = []
                half_2 = []
                for i, val in enumerate(sub_sublist):
                    if i % 2 == 0:
                        half_1.append(val)
                    else:
                        half_2.append(val)

                half_1.reverse()
                sorted_sub_sublist = half_1 + half_2
                sub_sublist[:] = sorted_sub_sublist

            elif sub_sublist[0][2] == 'HSR':
                sub_sublist.sort(key=get_numeric_value, reverse=True)  # Sort in ascending order

                # Apply the custom sorting logic
                sorted_sub_sublist = []
                half_1 = []
                half_2 = []
                for i, val in enumerate(sub_sublist):
                    if i % 2 == 0:
                        half_1.append(val)
                    else:
                        half_2.append(val)

                half_1.reverse()
                sorted_sub_sublist = half_1 + half_2
                sub_sublist[:] = sorted_sub_sublist

# Calculate new_x_coords for each sub-sublist
new_x_coords_list = []
for beams_index, beams in enumerate(composition_copy):
    new_x_coords_beam = []  # New list to hold x coordinates for each beam
    for layer_index, layer in enumerate(beams):
        new_x_coords_layer = []  # New list to hold x coordinates for each layer
        for row_index, sub_sublist in enumerate(layer):
            new_x_coords_row = []
            start_point = start_points_per_layer[beams_index][layer_index][row_index]  # Adjust start point based on beam, layer, and row
            new_x_coords_row.append(start_point)  # Insert start point at the beginning
            cum_sum = start_point  # Initialize cumulative sum
            for tuple_item in sub_sublist:
                cum_sum += tuple_item[3]  # Cumulative sum of block-lengths
                new_x_coords_row.append(cum_sum)

            # Remove the last value
            new_x_coords_row.pop()

            new_x_coords_layer.append(new_x_coords_row)
        new_x_coords_beam.append(new_x_coords_layer)
    new_x_coords_list.append(new_x_coords_beam)

global_x_coordinates = new_x_coords_list

x_coordinates_column_elements = global_x_coordinates[-2:]
x_coordinates_beam_elements = global_x_coordinates[:-2]

#x_coords_beam_a = global_x_coordinates[0]
#if len(global_x_coordinates) >= 2:
#    x_coords_beam_b = global_x_coordinates[1]
#else:
#    x_coords_beam_b = []

blocklist = composition_copy
block_identifiers = []

for beams_index, beams in enumerate(blocklist):
    new_blocklist = []
    for sublist in beams:
        new_sublist = []
        for sub_sublist in sublist:
            new_sub_sublist = []
            for item in sub_sublist:
                new_sub_sublist.append(item[0])
            new_sublist.append(new_sub_sublist)
        new_blocklist.append(new_sublist)
    block_identifiers.append(new_blocklist)
```

Reconfiguration of pieces based on optimisation criteria

```
# get new order of blocks in a list of lists
global_order = []
for beams_index, beams in enumerate(composition_copy):
    beam_order = []
    for sublist in beams:
        for sub_sublist in sublist:
            for item in sub_sublist:
                beam_order.append(item[0])
    global_order.append(beam_order)

order_of_blocks_column_elements = global_order[-2:]
order_of_blocks_beam_elements = global_order[:-2]
```

Clash check for pieces overlapping in x direction, unfinished but works

```
clash_info = []

for beams_index, beams in enumerate(new_x_coords_list):
    new_clash_info = []
    for layer_index, layer in enumerate(beams):
        merged_layer = []
        for row_index, row in enumerate(layer):
            merged_row = []
            block_identifiers_row = block_identifiers[beams_index][layer_index][row_index]  # Get block identifiers for the current row
            for x_coord in row:
                block_index = row.index(x_coord)  # Get the corresponding index in the row
                block_id = block_identifiers_row[block_index]  # Get the block identifier
                merged_row.append((x_coord, block_id))  # Append tuple (x_coord, block_id)
            merged_layer.append(merged_row)
        new_clash_info.append(merged_layer)
    clash_info.append(new_clash_info)

total_clashes_per_layer = [0] * (len(clash_info[0]) - 1)  # Assuming all beams have the same number of layers

# Iterate through consecutive layers
for beams_index, beams in enumerate(clash_info):
    for i in range(len(beams) - 1):
        current_layer = beams[i]
        next_layer = beams[i + 1]
        
        # Check if row numbers differ by at most 1
        if abs(len(current_layer) - len(next_layer)) <= 1:
            # Iterate through all combinations of coordinates between consecutive layers
            for current_row_index, current_row in enumerate(current_layer):
                for next_row_index, next_row in enumerate(next_layer):
                    # Exclude the first tuple in each row
                    for (current_x, current_block), (next_x, next_block) in zip(current_row[1:], next_row[1:]):
                        current_block_id = block_identifiers[beams_index][i][current_row_index][current_row.index((current_x, current_block))]
                        next_block_id = block_identifiers[beams_index][i+1][next_row_index][next_row.index((next_x, next_block))]
                        # Check if the row numbers differ by at most 1
                        if abs(current_row_index - next_row_index) <= 1:
                            # Check if there's a clash between consecutive layers
                            if abs(current_x - next_x) <= 0.02:
                                total_clashes_per_layer[i] += 1
                                print("Clash, same startpoint found between blocks in beam", beams_index, "layer", i, "row", current_row_index + 1, "X", current_x, "Bloc

# Print total clashes per layer
for layer_index, total_clashes in enumerate(total_clashes_per_layer):
    print("Total clashes in Layer", layer_index, "count:", total_clashes)
```

Sending pieces to the new database

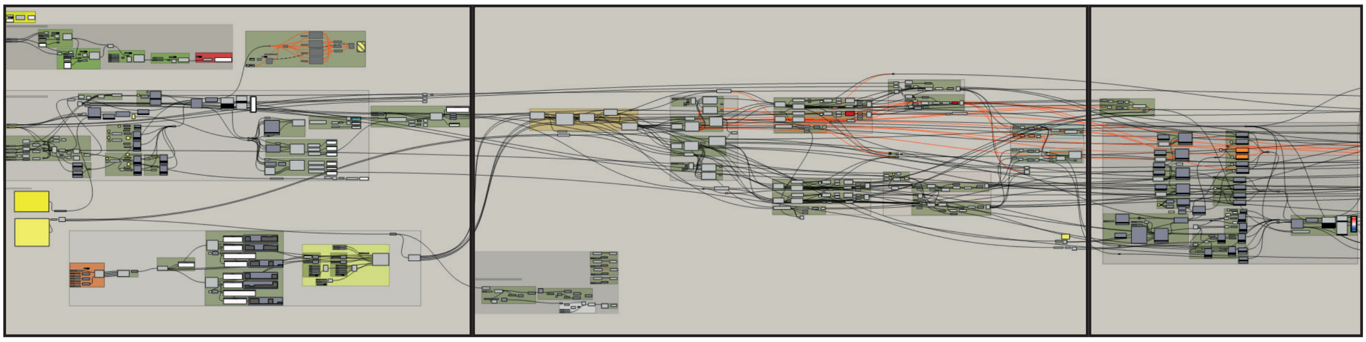
```python
import scriptcontext
pg2 = scriptcontext.sticky['psycopg2']

def retrieve_used_pieces(data, id_list):
    flattened_list = []
    for sublist in data:
        for tup in sublist:
            if tup and tup[0] in id_list:
                flattened_list.append(tup)
    return flattened_list

used_pieces = retrieve_used_pieces(inventory, remaining_pieces)

def extend_inventory(tuple_list, original_seller, status, project_name, composition_usage, composition_id):
    extended_tuples = []
    for tup in tuple_list:
        extended_tuple = tup + (original_seller, status, project_name, composition_usage, composition_id)
        extended_tuples.append(extended_tuple)
    return extended_tuples
    
# Extend each tuple in the list with the specified strings
extended_inventory = extend_inventory(used_pieces, status, original_seller, project_name , composition_usage, composition_id)

def setup_database_connection():
    conn = pg2.connect(dbname='WoodResurge_database', user='postgres', password=              , host='localhost')
    cur = conn.cursor()
    return conn, cur

def close_database_connection(conn, cur):
    cur.close()
    conn.close()

def apply_data(conn, cur, used_pieces):
    try:
        for pieces in used_pieces:
            if isinstance(pieces, tuple) and len(pieces) == 11:
                # Extract values from the tuple
                id_wood_piece, length, width, depth, strength_grade, painted, original_seller, status, project_name, composition_usage, composition_id = pieces
                # Convert the dimensions to integers
                id_wood_piece, length, width, depth = map(int, (id_wood_piece, length * 1000, width * 1000, depth * 1000))
                # Insert data into the table, including the new column
                cur.execute("INSERT INTO reclaimed_modulair_timber (id_wood_piece, length_mm, width_mm, depth_mm, strength_grade, painted, status, original_seller, project,
                            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)", (id_wood_piece, length, width, depth, str(strength_grade), painted,  str(status), str(ori
            else:
                print("Invalid data structure.")

        conn.commit()
        print('Data import success')
        return 'Data import success'
    except Exception as e:
        print("Error during data import:", e)
        conn.rollback()
```

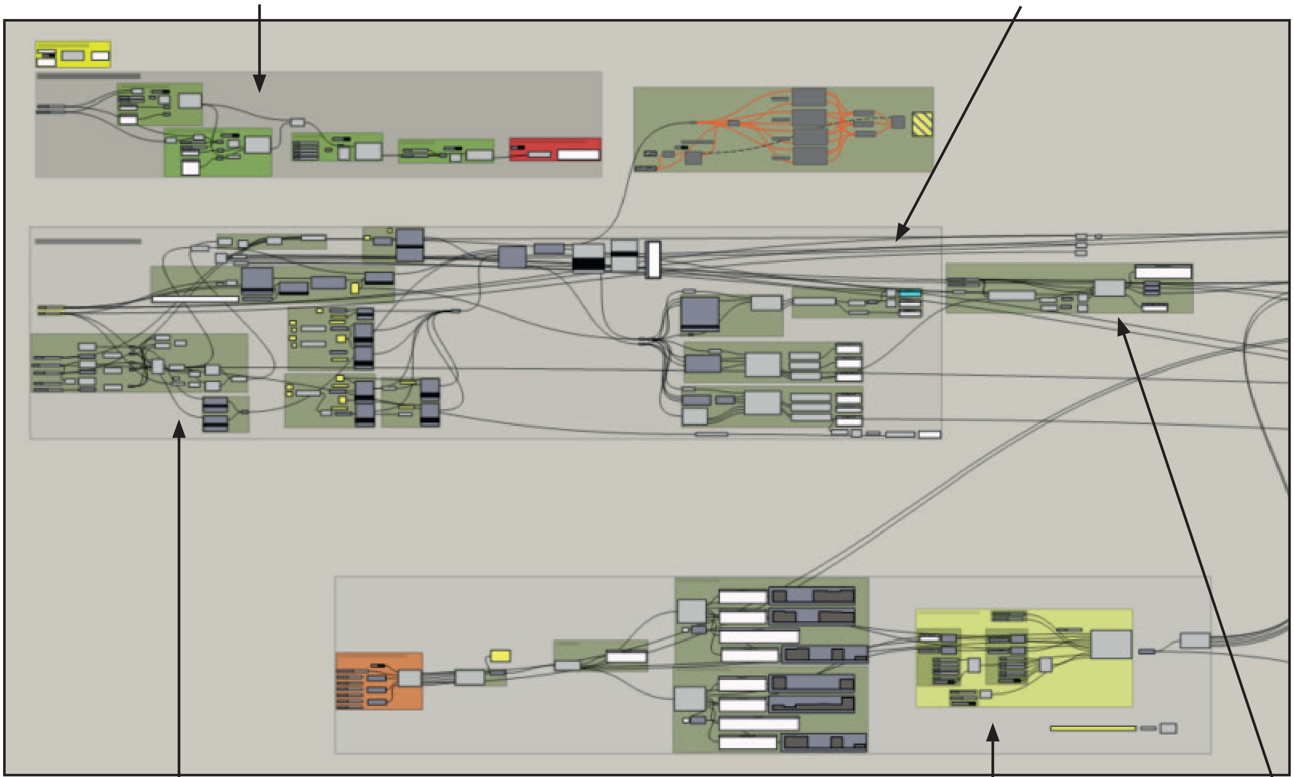
Removing pieces form the old database

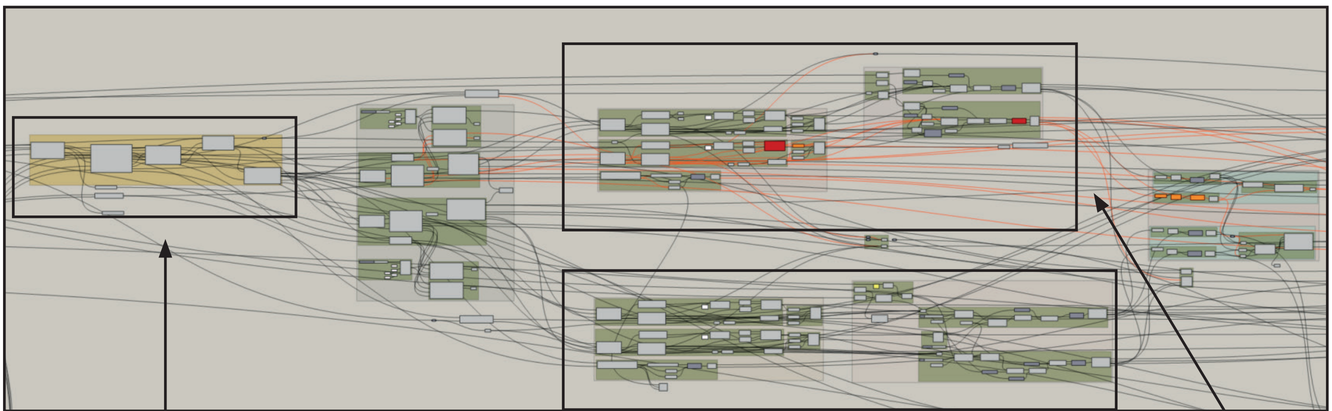
```python
import scriptcontext
pg2 = scriptcontext.sticky['psycopg2']


def delete_id(id_number):
    #parses the used pieces from the wood_dealer_database
    cur.execute("DELETE FROM salvaged_wood WHERE id_wood_piece = (%s)", (id_number,));
    conn.commit()
    print(str(id_number) + 'removed')

if Apply:
    conn = pg2.connect(dbname='Wooddealer_database', user='postgres', password=              , host='localhost')
    cur = conn.cursor()
    for id_number in remaining_pieces:
        delete_id(id_number)
```

Appendix E. Grasshopper canvas

Overview of initializing, matching pre-processing and finite element modelling

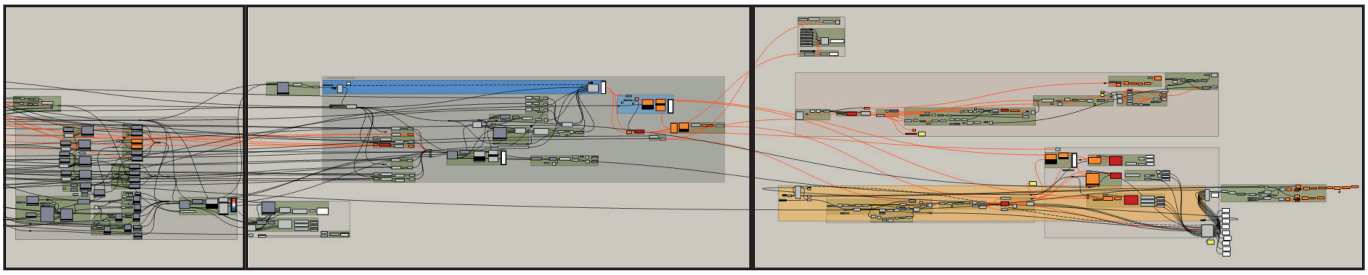






Grasshopper canvas

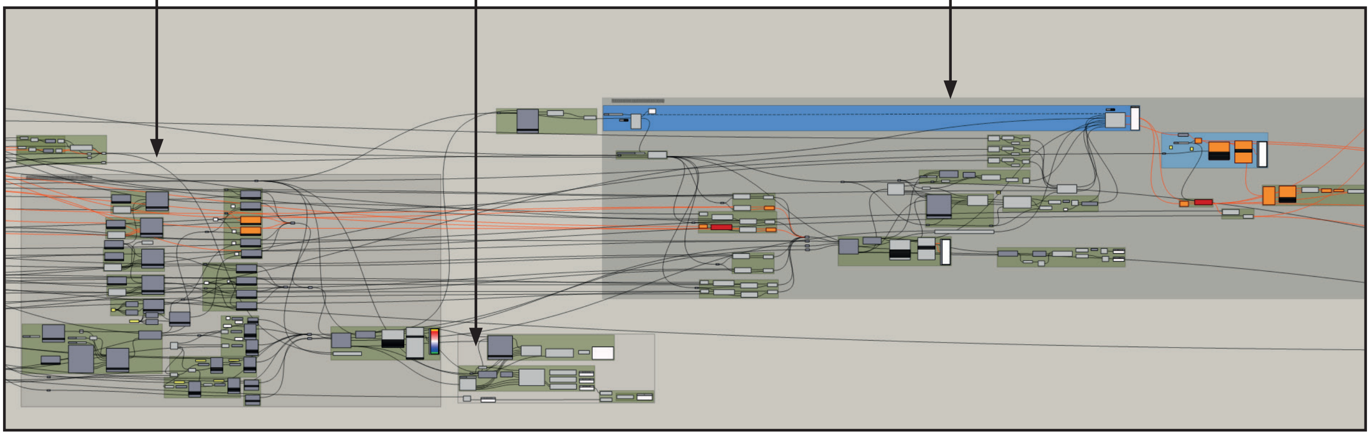
Overview of finite element modelling, optimisation loop and post-processing



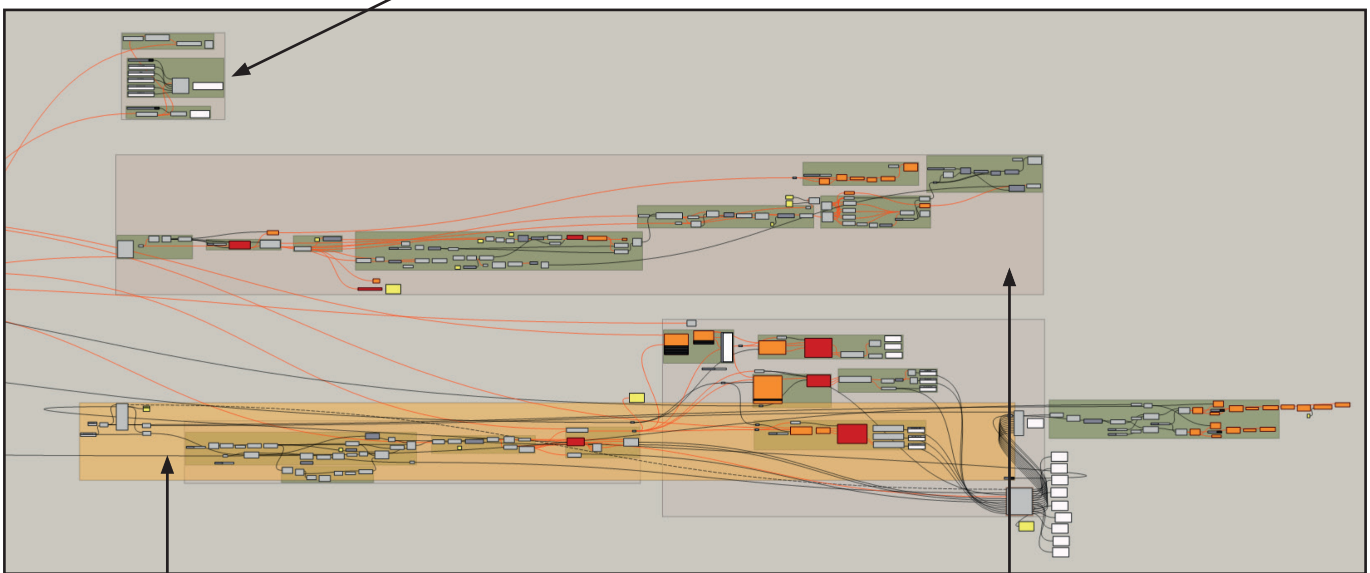
FEM, assembling model with karamba3D

Extracting resultant deformation and utilisation

Optimisation loop, selecting piece with lowest utilisation and removing it with plug-in Anemone



Sending pieces to material database and removing them from original database



Making slices through structure and obtaining detailed structural information

Creating 2D drawings with dowel holes, (unfinished project).