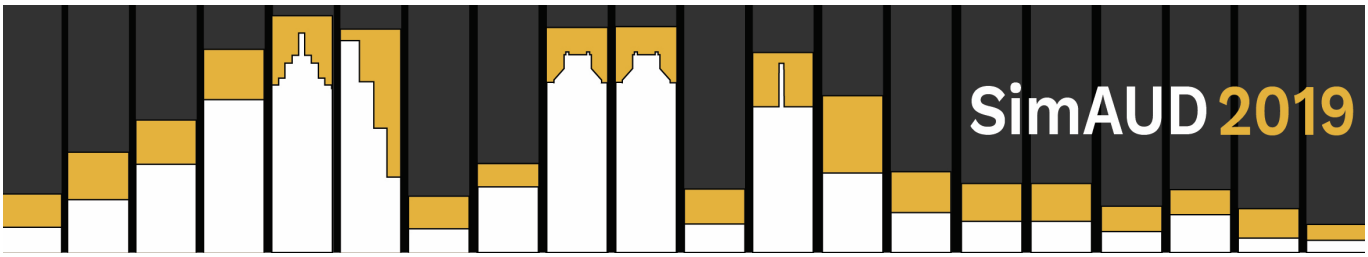

Towards Assembly Information Modeling (AIM)

Ayoub Lharchi, Mette Ramsgaard Thomsen and Martin Tamke

Centre for IT and Architecture (CITA), Copenhagen, Denmark, alha@kadk.dk

ABSTRACT

Nowadays digital tools support architects, engineers and constructors in many specific tasks in the construction industry. While these tools are covering almost all aspects of design and manufacturing, the planning and design for the assembly of buildings remain an unexplored area. This research aims to lay the foundations of a new framework for the design for assembly in architectural applications entitled Assembly Information Modeling. In practice, it is a central digital model containing the structure architectural design, construction details, three dimensional representations, assembly sequences, issue management and others. This framework forms the base for a multitude of novel applications for assembly design, planning and execution, such as assembly simulation and strategies communication, problem detections in the early design phases and interdisciplinary coordination. This paper describes the specifications of the digital assembly model and illustrate two use cases: collaborative assembly design using AEC cloud-based platforms and Augmented Assembly using Augmented reality devices.

Author Keywords

Design For Assembly; Digital Model; Assembly Modeling

1 INTRODUCTION

For an extended period, many architectural systems were strictly restricted in their general topology and geometrical differentiation because of both technical and economic factors [11]. These limitations increased the need for a toolbox that is adapted for complex geometries and led the architects and engineers to search and develop an entirely new set of methods and tools for design and manufacturing [3]. Today digital tools form the basis of nearly all design, construction, fabrication and management tools in all professions related to the building industry [13].

Overall, computational design tools and digital fabrication processes enabled a higher degree of differentiation between the elements in a single structure [11] and it became possible to design and manufacture large-scale freeform shapes. However, this new shape emergence posed significant challenges in terms of communication, fabrication, and assembly [9]. Although there are many attempts to standardize the information modeling and data sharing between the different interdisciplinary partners within one project [12], there are more issue that can rise and are not covered by traditional approaches such as Building Information Modeling (BIM).

This research aims to fill the existing gap in the design for assembly field, by suggesting a novel approach for handling assembly information in a construction context. By combining computer science techniques and design practices from other disciplines, this project defines in a first step a scheme by which professionals can describe, analyze and communicate assembly information.

This digital model forms the base for a multitude of novel approaches for assembly design, planning and execution (figure 1): now detailed digital assembly information can be shared and discussed between partners through cloud-based platforms, assembly sequences can be generated and optimized with the help of algorithms and new human-machine interfaces such as augmented reality can be used to assemble constructions. Finally, the model presents the base for future robotic assembly.

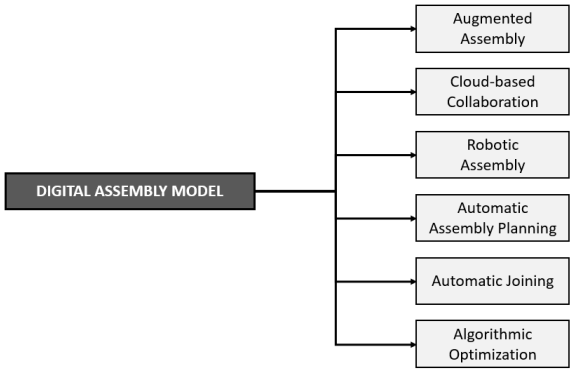

Figure 1. Potential Applications of Assembly Information Modeling

2 DESIGN FOR ASSEMBLY IN ARCHITECTURE

SimAUD 2019 April 07-09 Atlanta, Georgia

A proficient assembly planning should be part of any successful design. This can reduce assembly time, allow flexibility and improve the quality and reliability of the final product [15]. Design for Assembly (DfA) is a well-established practice in other disciplines since the 80s [2], especially in industrial and mechanical design. It is often combined with Design for Manufacturing (DfM) techniques in order to optimize the manufacturing and assembly within industrial applications. DfA aims to have a full understanding of the assembly process and to extract principles that influence the design iterations. In general, it involves two crucial steps: minimization of the number of the separated parts that constitute the global structure (figure 2), and the improvement of the "Assemblability" of the remaining parts [2]. Research has been conducted on assembly planning, little focus was however set on the integration of questions of assembly in the design phase [14]. Furthermore, the existing approaches to DfA are all rooted in industrial fabrication processes for e.g. machines and consumer products, which makes it difficult to have a direct transfer of these to an architectural context due to material and scale considerations.

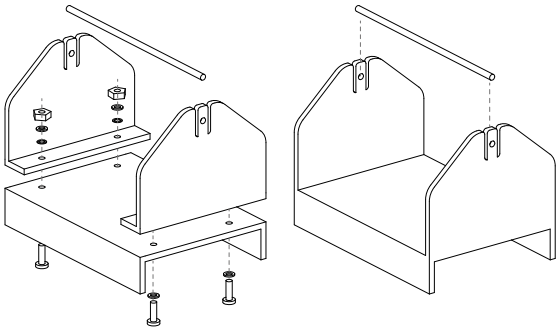

Figure 2. Pieces number reduction for optimized Assembly [2]

3 ASSEMBLY INFORMATION MODELING

An Assembly Information Model (AIM) as described here is a digital framework directed for application in building processes, that aims to include all the necessary data to describe precisely an assembly sequence and at the same time to bridge the different authoring tools used by the stakeholders. The model specifications are freely available online and are covering most of the common needs in a typical architectural assembly planning process, but can be extended as well if needed. In order to assure high interoperability between different software package, the implementation presented in this paper is intentionally software agnostic and is using exclusively open source libraries to allow an eventual use in commercial software.

The AIM implementation we propose is composed of the following elements:

- A library called AIM.Core defining the basic functions and managing the input (geometry and meta-data) and output (files, documentation, etc.).
- A set of plug-ins for different host CAD systems. Each plug-in uses the geometric capabilities of the host software and the AIM.Core library to generate an Assembly model (.adm).

The AIM.Core library was written in C# [5], which is a powerful object-oriented language and using the .NET Core Framework, which in itself is a further approach to ensure a cross-platform compatibility. For demonstration and testing purposes an AIM.GH - plug-in for the Grasshopper [10] environment was developed. Additional plug-ins to generate, read and manipulate the assembly model can be written easily by persons with a certain knowledge of the host application API, using any .NET programming language [8].

The full model specifications and the implementations source code are available online at: **`https://www.github.com/ALharchi/AIM`**

The digital model is composed of the following classes:

3.1 Elements

An element refers to any physical element in the structure. It can be either a part of the structure (beam, column) or a fastener (bolt, screw, etc.). The model stores the defined position, orientation and the geometry of all the elements. For each one, there is a corresponding file containing the geometry. Depending on the modeling approach and the software used, a NURBS or MESH description is used. If the used software support both NURBS and Mesh geometries, dual representation can be enabled, and both geometries representation will be stored to maximize the interoperability. The NURBS is stored in a .STEP file, while the MESH is in a .STL file.

For similar elements, only one file (or two if dual representation is enabled) is used to reduce the model size and memory usage.

3.2 Transformations

Transformations are the sequence of geometrical operations that are necessary to get an element from the entry position to the final position. The entry position can be any safe position without collision or obstacles (on the ground for example). The final position corresponds to the correct spatial location and orientation within the structure. Two main types of transformations are defined using simple mathematical concepts:

- Translation: Linear movement in space, defined by a three-dimensional vector.
- Rotation: Circular movement defined by a rotation plane and a rotation angle (expressed in radian).

Using these two transformations, complex spatial movements can be described. When loaded, the appropriate AIM plug-in will convert it to match with the software native spatial library.

3.3 Components

Elements are grouped together in components. Typically, one component includes several parts and the necessary fasteners. Single-element components are also possible. Transformations can also be assigned to components.

3.4 User Manipulation Zone (UMZ)

The user manipulation zone is the necessary area to operate during an assembly step. It is represented by a sphere defined by radius from the fastener element.

3.5 Joints

Joints define the relation between two or more elements (figure 3). The order or the Assembly Sequence (AS) is specified. Each joint is expressed in the necessary number of transformations.

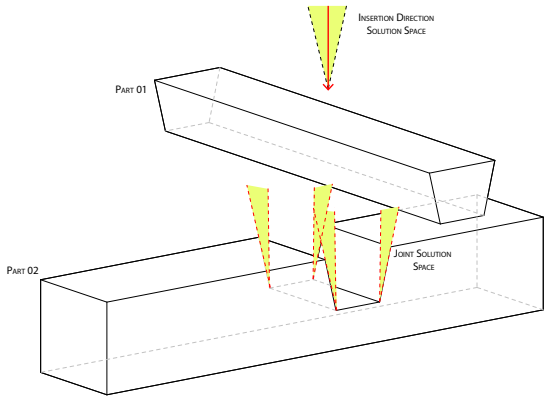

Figure 3. Joints in space with vectors

3.6 Issues

Issues are problems concerning the assembly process that needs to be communicated. It can be either manually defined by the user during the model creation or automatically raised later by the computation engine (as described in 4).

Issues are divided into categories according to the severity and impact on the assembly process:

- Critical: The assembly is impossible.
- Moderate: The assembly is partially possible.
- Suggestion: The assembly is possible, but can be improved.

Every issue is linked to the user that created it, which can be used later for an issue management platform (see 6.1).

3.7 Documentation

Technical drawings of the different elements can be automatically pre-generated and included in the digital model. They are saved in PDF and PNG format (figure 4).

4 MODEL COMPUTATION

One of the main benefits of the assembly modeling approach is to have all the information in one single model, which facilitates any desired analysis or computation. The model computation helps to detect issues in the assembly process in the early stages. Although some of these problems can be detected using 4D simulation of the construction process, the adoption of such techniques remains very slow [4]. They are

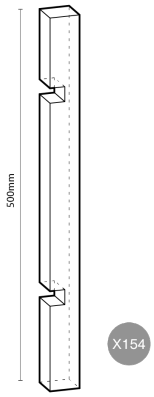

Figure 4. Example of documentation included in the model

usually related to the geometry of the elements [6], for example, if the parts cannot be assembled because of incompatible joints, collision with other elements or due to a wrong assembly sequence order. The problem can also be related to logistics: the assembly is possible in theory but very difficult or impossible to realize in practice. This is often the case if there are some unpredicted constraints on site, such as a smaller crane (only one insertion direction; from above) or existing construction that would prevent the spatial movement of large elements.

Since the AIM.Core does not include any geometrical kernel, the model computation is executed on the software side (using the adequate AIM plug-in).

4.1 Mathematical Conflict Detection

Using the information embedded within the joint definition, issues in the assembly sequence can be detected. Before using computationally expensive Boolean operations to detect collisions, the joints design space are evaluated and conflicting solutions are flagged.

4.2 Physical Collision Detection

Using the transformations, every element in the model is interpolated from the initial entry position until the final position within the structure. The computational Engine is detecting any collision between the current part, and all the previously placed parts during the assembly operation (figure 5). If there is none, the part is marked as safe; otherwise, an issue is raised and recorded in the model as well. A custom collision detection algorithm had to be written based on the existing Boolean operations available in the HCS. The issue with the existing collision detection methods is that they flag two touching solids as intersecting, while in an assembly context, it is accepted or necessary within certain tolerances (such as sliding an element in between two already placed elements).

4.3 Fabrication Constraints

The user can provide additional information about the available logistics. This is intended to be used in coordination and assembly sequence evaluation. All the existing constraints or construction on site are stored within the model. An abstract

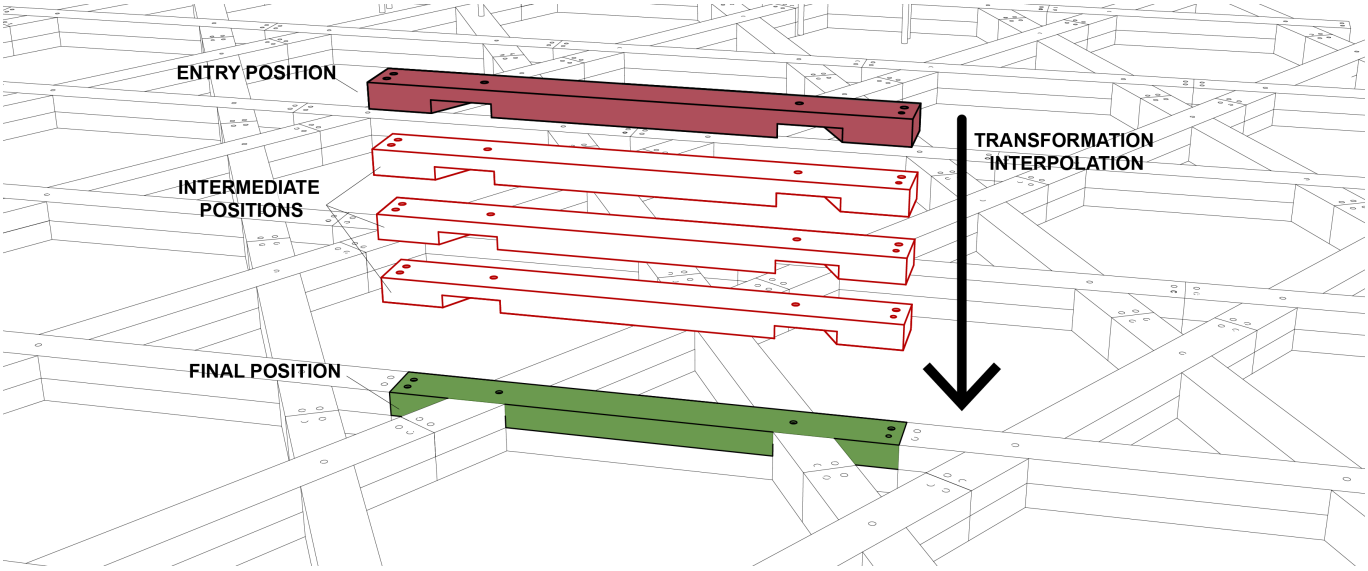

Figure 5. Transformation interpolation from entry to final position.

representation (reduced mesh) is used to reduce the model size.

5 FILE FORMAT

The Assembly Digital Model (ADM) is intended to be used for various applications (see section 6). It is crucial to embed all the information within one comprehensive to allow the desired interoperability. Furthermore, one self-contained file facilitates the exchange with cloud-based platforms, robotic interfaces and other devices. According to the specifications, the information is divided into three categories:

- Geometry data: This includes the elements' geometry. It is a collection of STEP and STL files.
- Graphical data: Technical drawings (PNG or PDF).
- Meta-data: Further data, which provides additional information about the geometry (transformations, issues, joints etc.), relationships to overall project, related models, processes etc.. The metadata is plain text.

The ADM is open-source and the implementation uses freely available libraries that are running on the three majors operating systems (Windows, Linux and MacOS).

5.1 ADM Container

The ADM container we propose is new file format that uses the extension .adm. It is a compressed ZIP file that encapsulates the geometry and graphical files and one single database for the meta-data (see section 5.2). The files are organized in hierarchy using folders (figure 6).

Every element is stored in separated file to allow a quick loading for single elements and to facilitate the replacement/adjustments.

5.2 Meta-Data storage

It was necessary to define a way to store the metadata that is flexible enough to allow an integration of interlinking data

```
filename.adm
    data.db
    geometry/
        part_01.stp
        part_02.stp
        fastener_01.stp
        ...
    docs/
        part_01.pdf
        part_02.pdf
        ...
```

Figure 6. Hierarchy in the .adm file

with the possibility of complex queries. The most commonly used format for this purpose are relational databases and text storage format such as JSON (JavaScript Object Notation) and XML (Extensible Markup Language). While XML and JSON offers the advantages of being human-readable, in this case most of the processing is automated. Therefore the advantage was given to organization querability.

For this purpose, we chose to use a relational database because it is allowing the definition of relations between elements, that once in the system, can be queried using a specific language SQL (Structured Query Language). Common operations queries are already provided with the model and the user can easily write his own query if needed. Among the different relational database systems, we chose to use SQLite because of its advantages as it is a serverless database system so it does not need a server and the generated file can be manipulated directly within the different application (table 1).

6 APPLICATIONS

As we describe a general approach to formalise assembly in a digital model, many promising areas of applications can be imagined. For example the ADM can be a unified base

Name	Advantages	Disadvantages
SQLite	File Based Embedded Applications Serverless Little resource	No User System Single Query
MySQL	Extensible Multi-user support Scalable	Requires separated hosting Heavy
PostgreSQL		

Table 1. Comparison of relational database systems [7]

for communication between different stakeholders, engineers or architects can receive instant feedback on their assembly strategies through further computation or analysis of the ADM, as the ADM can provide detailed data for robotic assembly.

In this research, we evaluated our approach with two areas of applications, which require an effective communication of assembly strategies and operate at different design stages.

6.1 Collaborative Design for Assembly

The collaborative design for assembly is a platform for the communication, viewing and discussion of assembly data. A web-based platform uses the ADM and makes it available to the different stakeholders with a powerful built-in 3D viewer. The requirement was to create an integrated platform that would run on most modern web browsers without the need of any additional plug-ins. This would include mobile devices such as smartphones and tablets for eventual on-site usage. The presented platform utilize the Autodesk Forge cloud services [1]. We used the Forge Viewer API and Model Derivative API. The application backend was written in ASP.NET using the .NET Core framework; meanwhile, the front-end was mainly in HTML, CSS and JavaScript.

One user can upload an ADM file that is processed by the Autodesk Forge cloud, translated into SVF format. Afterwards, the assembly information is injected into the 3D model. The model is then made available for all the users of the platform for editing and viewing (figures 7 and 8).

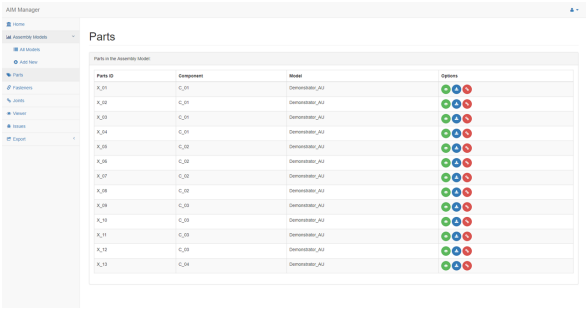

Figure 7. Individual elements management

6.2 Augmented Assembly

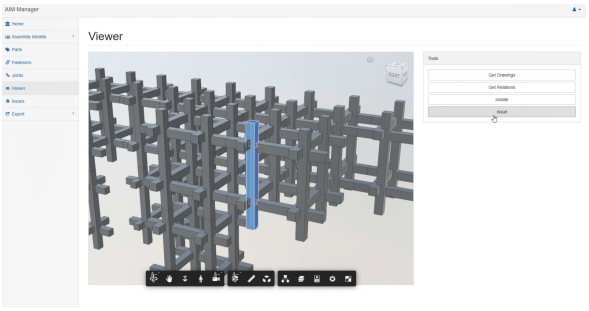

Figure 8. 3D Viewer

Using the ADM model, it is possible to extract step by step information, which guides a user through an assembly process. Augmented Reality (AR) devices are particularly adapted for this usage as they allow overlaying a digital input onto the user view and thus align the relevant information to the objects of interest [16]. Our tests show (figure 9) that empowered an AR device, an inexperienced user can assemble complex structures without any previous knowledge or training.

The Augmented Assembly interface was implemented for the head mounted device Microsoft Hololens. Using the autonomous self-tracking features, it was possible to track the user in the space and display assembly information that overlap with the environment (figure 10). The software running on the Hololens was developed using the Unity 3D Game Engine in C#. The assembly sequence was displayed to the user in sequential order. The assembly was animated in a way that the user can distinguish clearly the orientation of the element as well as the necessary transformations to put it in the correct place.

The user interface uses two input systems of the Hololens:

- Gesture-Based: To select and manipulate the different elements. It also allows extracting information from the digital assembly model and displaying them directly in the augmented view.

- Vocal commands: This provides a hands-free interface to control the assembly animation (showing the next or previous piece, pausing the animation etc.).

7 CONCLUSION AND FUTURE WORK

We present Assembly Information Modeling (AIM) as a framework to describe, analyze and communicate assembly strategies in the AEC sector. While the current specification of the digital model is covering many typical assembly planning needs (Collision detection, documentation generation, 4D simulation etc.), many technical improvements are necessary to cover more specific cases such as generic joints and interfacing with existing manufacturing techniques. This includes defining the joints class more abstractly and providing additional queries to extract and export data from the model. Furthermore, the implementation of a geometric kernel that would be embedded in the AIM.Core library is envisioned, to unify the geometric operations and to skip the translation phase to the native format within the host CAD software. In

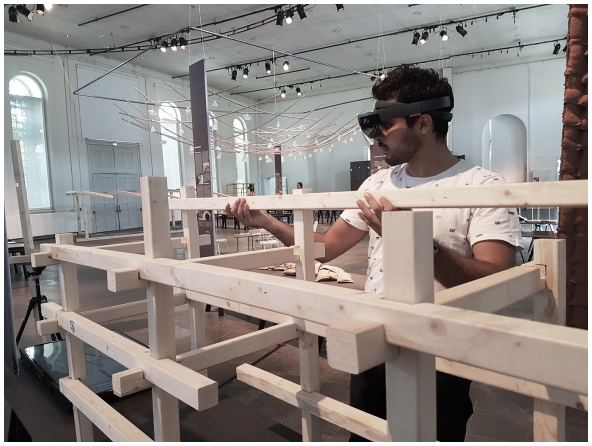

Figure 9. A user performing augmented assembly

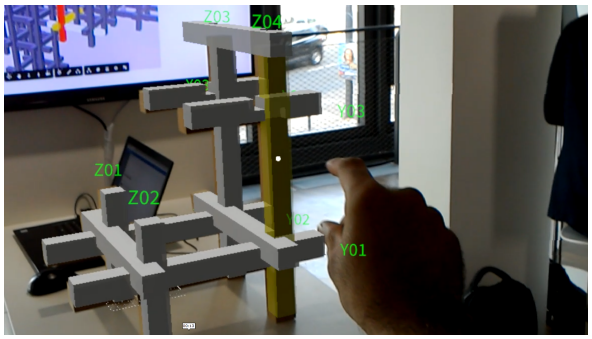

Figure 10. Augmented assembly view

addition, this will speed up the necessary computation time for collision detection.

Two areas of exploration seem especially pressing in respect to the general move of AEC sector:

- Mechanization of the construction site, through robotic assembly where the ADM is uploaded directly to a robotic arm for an automatic assembly.
- Integration of simulation tools in early design planning, where AIM can be used for an integrated path planning, using a system capable of generating collision free robotic paths for the construction. Ultimately, the model can serve as a base for machine learning algorithm to assist the designer for assembly choices.

8 ACKNOWLEDGMENTS

This project was undertaken as part of the Innochain Early Training Network. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie Grant Agreement No. 642877. We would also like to express our gratitude to the industrial partners of this research; Blumer Lehmann and Design-To-Production, as well to Autodesk, especially the Forge development team.

REFERENCES

1. Autodesk Forge: Cloud-based AEC developer tools . `https://forge.autodesk.com`.
2. Boothroyd, G. Design for assembly—the key to design for manufacture. *The International Journal of Advanced Manufacturing Technology 2*, 3 (1987), 3–11.
3. Boothroyd, G. *Assembly automation and product design*. CRC Press, 2005.
4. Boton, C., Kubicki, S., and Halin, G. The challenge of level of development in 4d/bim simulation across aec project lifecyle. a case study. *Procedia Engineering 123* (2015), 59–67.
5. C# Programming Language. `https://docs.microsoft.com/en-us/dotnet/csharp/`.
6. Czmoch, I., and Pekala, A. Traditional design versus bim based design. *Procedia Engineering 91* (2014), 210–215.
7. Relational Databases Comparison. `https://www.digitalocean.com/community/tutorials/sqlite-vs-\mysqlvs-postgresql-a-comparison-of-relational-\database-management-systems`.
8. .NET Framework, a free, cross-platform, open source platform for building apps. `https://www.microsoft.com/net`.
9. Eigensatz, M., Kilian, M., Schiftner, A., Mitra, N. J., Pottmann, H., and Pauly, M. Paneling architectural freeform surfaces. *ACM transactions on graphics (TOG) 29*, 4 (2010), 45.
10. Algorithmic Modeling for Rhinoceros 3D. `https://www.grasshopper3d.com`.
11. Krieg, O. D., Dierichs, K., Reichert, S., Schwinn, T., and Menges, A. Performative architectural morphology: Robotically manufactured biomimetic finger-joined plate structures.
12. Tamke, M. Aware design models. In *Proceedings of the Symposium on Simulation for Architecture & Urban Design*, Society for Computer Simulation International (2015), 213–220.
13. Tamke, M., and Thomsen, M. R. Digital wood craft. *Joining Languages, Cultures and Visions: CAADFutures* (2009), 673–686.
14. Usai, S., and Stehling, H. La seine musicale. In *Humanizing Digital Reality*. Springer, 2018, 201–209.
15. Wilson, R. H. On geometric assembly planning. Tech. rep., STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, 1992.
16. Zollmann, S., Hoppe, C., Kluckner, S., Poglitsch, C., Bischof, H., and Reitmayr, G. Augmented reality for construction site monitoring and documentation. *Proceedings of the IEEE 102*, 2 (2014), 137–154.