

Solidifying Wireframes

Vinod Srinivasan, Esan Mandal and Ergun Akleman
Visualization Laboratory
Department of Architecture
Texas A&M University
College Station, TX 77843-3137, USA
E-mail: vinod@viz.tamu.edu

Abstract

In this paper we present a method to convert a wireframe mesh into a 2-manifold mesh consisting of cylindrical pipes in place of the edges and joints in place of the vertices in the original mesh. Our method allows users to create unique artistic depictions of common objects and structures. The resulting mesh is also more effective at conveying the overall 3D structure and any internal elements of a model when compared to regular wireframe or boundary representations. The input wireframe mesh can be any collection of linear edges; they do not have to form a manifold surface or even be connected to each other. The result is always an orientable 2-manifold surface. Our algorithm replaces every edge in the wireframe mesh with a cylindrical *3D pipe*. The pipes are connected to each other using *3D joints* created at the vertices in the wireframe where the edges meet. Our method has been implemented as part of a polygonal mesh modeling system and has been used to create artistic models of popular architectural structures as well as to create conceptual sketches for virtual environments.

1. Introduction and Motivation

Wireframe meshes are commonly used to represent 3D objects in many modeling programs. A wireframe mesh consists of a set of vertices and a set of edges connecting those vertices. Although they represent 3D models, the individual components of the wireframe mesh (vertices and edges) are themselves thought of as having zero volume. Thus when shown on a 2D display, a wireframe mesh does not effectively convey the 3D information that it encapsulates [7]. The use of perspective projection reveals some of the depth information, but for complicated models with intricate geometry it is still not very effective. Boundary representations, such as polygonal meshes or NURBS surfaces, derived from the wireframe mesh can partly help with conveying 3D information. But in such models, internal structures are not visible, since they are occluded by exterior faces.

In comparison to using polygonal meshes or NURBS surfaces, modeling using wireframes is much simpler since it has fewer constraints. Furthermore, because no face information is stored, they are efficient in conveying a lot of information with very little resources. However, the lack of faces has a drawback in that it is difficult to tell which parts of the model are solid and which are empty [7].

Wireframe modeling is particularly well suited for modeling shapes such as frame structures which resemble wireframe meshes in their overall appearance. Shapes dominated by beams and columns are also well suited for wireframe modeling. Classical architecture is replete with examples of such shapes. In recent times, frame structures, such as space frames or plane frames, have gradually increased in popularity among

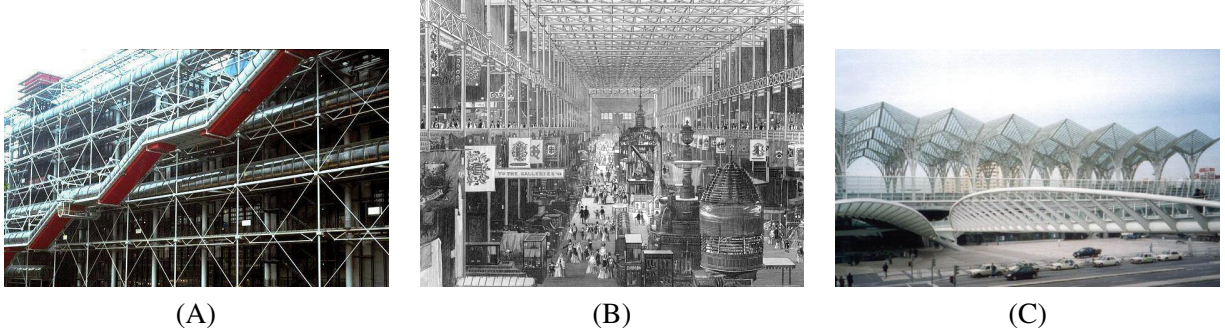


Figure 1: Examples of frame structures in architecture from 19th and 20th centuries. (A) Center Pompidou, Paris (1970s). Architects: Renzo Piano, Richard Rogers. (B) Crystal Palace, England (19th century). Designed by Joseph Paxton. (C) The Eastern Station, Lisbon (1990s). Architect: Santiago Calatrava.

designers. The Crystal Palace, designed in 1850 by Joseph Paxton is often cited as the first example of using pre-fabricated components [5]. Figure 1 shows examples of architectural designs from different periods, all of which are dominated by frame structures.

In spite of the prevalence of such designs, research into tools for creating such models is lacking. Van Kempen et al. in their work on computer aided drafting and three-dimensional modeling address the subject of pre-fabricated constructions, but not specific to the types of structures we are interested in [11]. Some of the authors of this paper have worked on methods to create models with large number of holes and handles and our present work develops on several of the ideas that were introduced in those papers [10, 3, 8, 9].

In this paper, we present a method to *solidify* a wireframe. Our method essentially converts a wireframe mesh into a 3D model in which every edge of the wireframe is replaced by a *3D pipe* of finite thickness and volume, in effect creating a *solid* wireframe. The pipes are connected together using *3D joints* at the vertices where the edges meet. The resulting model conveys the 3D information much more effectively than the plain wireframe model. Internal structures are also visible, since there are no occluding faces.

Our new method has been incorporated into our existing 3D mesh modeler [1, 9] and allows the user to create very complex models with minimal effort. Creation of models like those shown in Figure 1 is greatly simplified using our tool. Users can adjust the thickness of the pipe as well as its cross-section. We chose a semi-automatic approach to ease the tedium of modeling every individual column or beam, while at the same time giving the user a fair amount of control over the resulting shape and mesh structure.

One of the important features of our method is that the final mesh is guaranteed to be a valid 2-manifold surface without any cracks or gaps. This has important implications for several subdivision schemes which require that the input mesh be a valid 2-manifold [4, 6, 12]. The model created using our method is also well-suited for fabrication using a 3D printer or rapid-prototyping machine.

2. Creating the Solid Wireframe

As described above, our new method creates a solid wireframe from a given wireframe mesh. The input wireframe does not have to represent a closed surface and can be an arbitrary set of edges which may or may not be connected to each other. For simplicity, we will use the term “wireframe mesh” to refer to a regular wireframe mesh as well as an arbitrary set of edges.

The creation of the solid wireframe consists of two main steps. The first one involves creation of the 3D joints at each of the vertices in the input wireframe. The second step involves connecting the joints to create the 3D pipes corresponding to each edge in the input wireframe. The following paragraphs describe these two steps in more detail. The term “wireframe” will be used to refer to the input mesh, while the term “solid wireframe” will be used to refer to the output mesh.

2.1. 3D Joints In the first step of the process, 3D joints are created at every vertex of the input wireframe. The shape of the 3D joint at a vertex depends on the number of edges incident on that vertex, and the desired thickness and cross-sectional shape of the 3D pipes. There are three steps in the joint creation process as described below.

2.1.1. End-faces For every edge in the wireframe, we compute a *virtual 3D pipe* of user-specified thickness and cross-section, with the axis of the pipe aligned with the edge. At this stage we are only interested in obtaining the end-faces of each 3D pipe. We also establish a link between each vertex of the wireframe and the end-faces that will be used to create the 3D joint at that vertex.

The centroid of the end-faces lies on the original edge, with the normal to the face aligned with the edge. All the end-faces linked to a vertex are positioned at the same distance from the vertex. This distance is computed in such a way as to avoid intersection with the other end-faces linked to this vertex. To do this, we take end-faces linked to a vertex in pairs and find the minimum distance at which there is no intersection. We then take the maximum of these minimum distances and use that distance for all the end-faces linked to that vertex. If the thickness of the pipe is large, this approach can produce incorrect joints in situations where the mesh has very short edges near a sharp corner. We ignore this special case since the user can adjust the thickness of the pipes to avoid such situations.

2.1.2. Convex Hull For every vertex of the original mesh, we collect end-faces that correspond to the edges incident on the vertex. Using the vertices of these end-faces we then calculate a convex hull which defines the shape of the 3D joint. The distance of the end-faces from the vertex affects the shape of the convex hull. Since we avoid intersections between end-faces when computing the distances, the points making up each end-face are guaranteed to be a part of the convex hull.

2.1.3. Convex Hull Clean-up Figure 2A shows a close-up view of the convex hull for one vertex of a wireframe cube. As can be seen, the convex hull algorithm creates a triangulated mesh. In order to create 3D pipes of user specified cross-section and thickness, it will be necessary to clean up the triangulated mesh so that we have a face on each 3D joint which can serve as the end of a 3D pipe. By keeping track of which vertices came from which end-face we can remove edges which were not part of the end-face, in effect recreating the end-faces on the 3D joint. Note that we restrict this clean-up to edges between vertices from the same end-face. Figure 2B shows the convex hull after the end-faces have been recreated.

The convex hull can be cleaned-up further by removing edges which are adjacent to co-planar faces. That is, if two adjacent faces have the same face normal, the shared edge can be removed. This produces a cleaner 3D joint by removing edges which do not add any extra visual detail. Figure 2C shows the result of the additional clean-up.

2.2. 3D Pipes Once the 3D joints have been created, the creation of the 3D pipes is fairly straightforward. It involves insertion of a handle between matching faces of the joints. The connectivity information in the original mesh is used to determine the matching faces. The actual handle creation is done using the CREATEPIPE operator which is one of the operators provided by our mesh modeling system [1, 10, 9]. Figure 3 illustrates the main steps of the above algorithm using a wireframe cube as the initial shape.

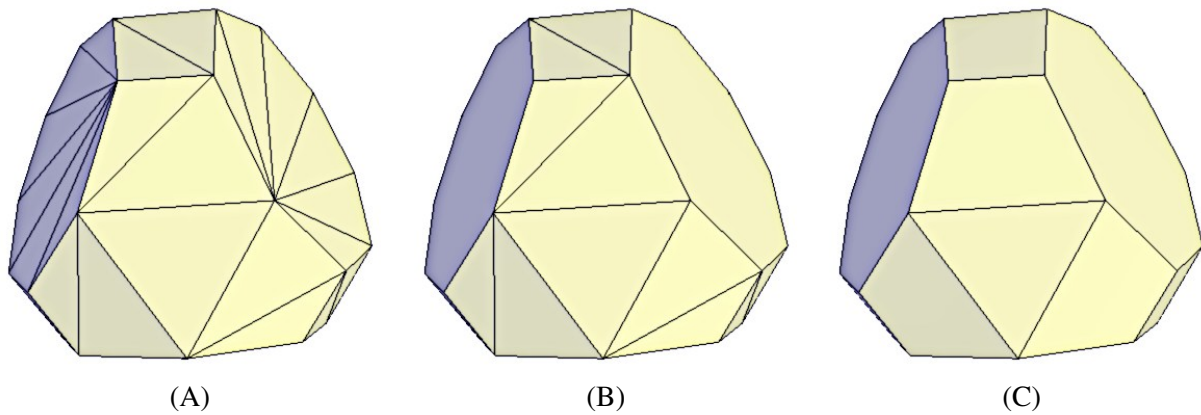


Figure 2: A closeup view of the 3D joint creation at one corner of a dodecahedral wireframe. (A) shows the joint after the initial convex hull has been created, (B) shows the joint after the end-faces have been recreated and (C) shows the joint after the additional clean-up.

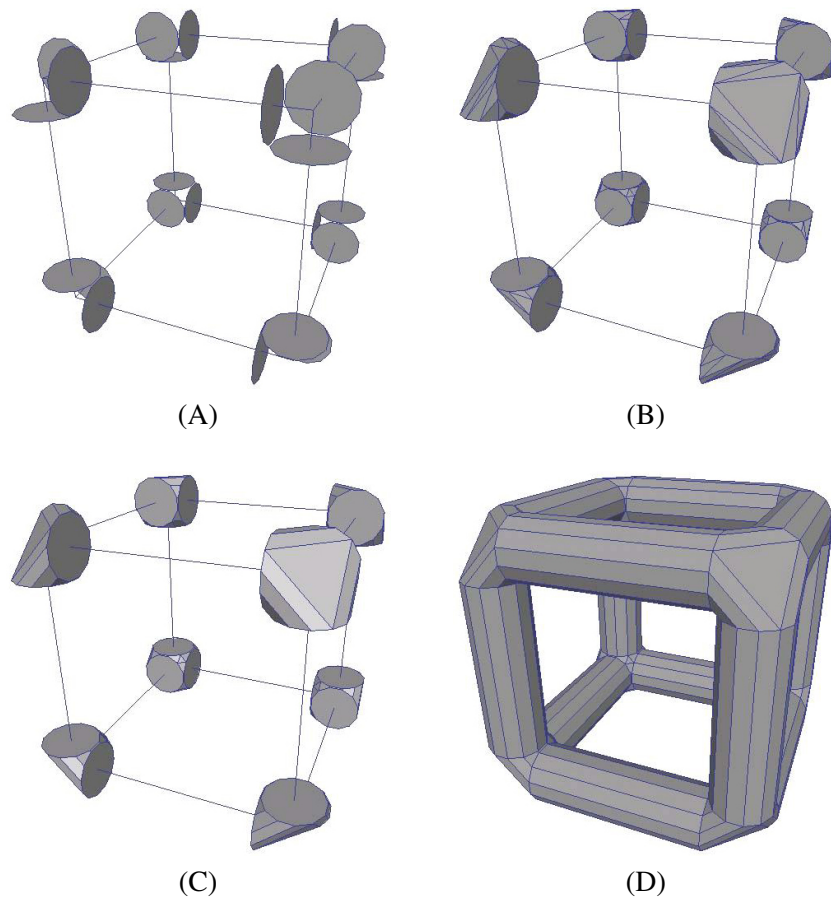


Figure 3: Creating a solid wireframe from a cube. (A) Creation of the end-faces. (B) Creation of convex hulls. (C) The 3D joints after clean-up of the convex hulls. (D) Final mesh obtained by inserting handles between the 3D joints.

3. Results

Our solid wireframe modeling method has been incorporated as an interactive tool into the three-dimensional mesh modeling system that we have developed [1, 2, 9]. Our system includes other features such as extrusions and re-meshing schemes, which when used in combination with the solid wireframe modeling tool, can create very interesting and complex shapes. The re-meshing schemes are particularly useful since they allows users to create several types of solid wireframe structures from the same initial mesh.

Figure 4 shows an example of applying the solid wireframe modeling tool to a dodecahedron with different values of thickness and cross-sectional shape. As can be seen, increasing the number of sides in the cross-section of the pipes, produces smoother looking pipes. Having a smaller number of sides produces angular pipes. By varying the thickness we can create thin wire-like structures or thick pillar-like wireframes.

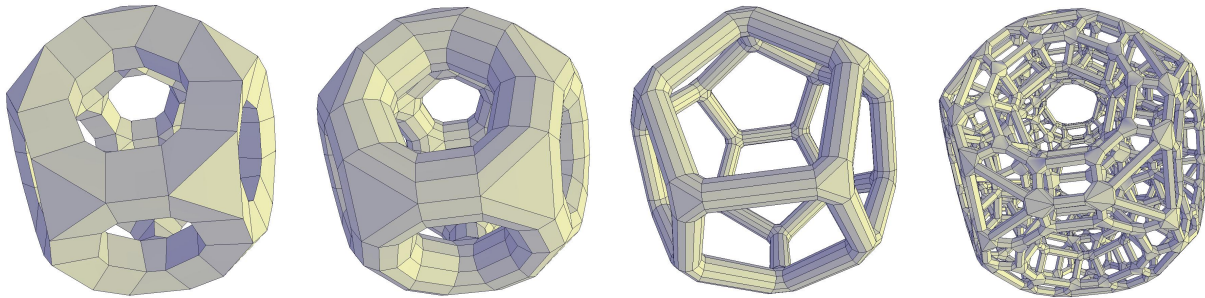


Figure 4: *The solid wireframe modeling tool applied to a dodecahedron with varying thickness and cross-section. The final model was created by applying the tool twice in succession.*

The solid wireframe modeling tool provides a simple, intuitive and easy-to-use interface for creating very complex structures in a few steps. It is well suited for use by architects and artists, and has been used to create conceptual models as well as stylized depictions of existing architectural models. Figure 5 shows an example of a model created using the solid wireframe modeling tool and its use in a conceptual sketch. Figure 6 shows a rendered model of a cathedral created using our new method.

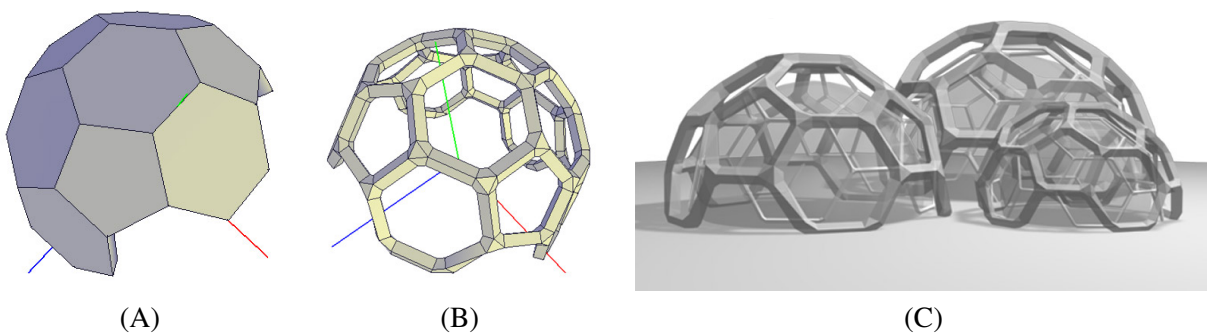


Figure 5: *Example of a frame structure created using the solid wireframe modeling tool. (A) shows the initial mesh and (B) shows the end result. (C) is a rendered image from a conceptual sketch (created by Ozan Ozener).*

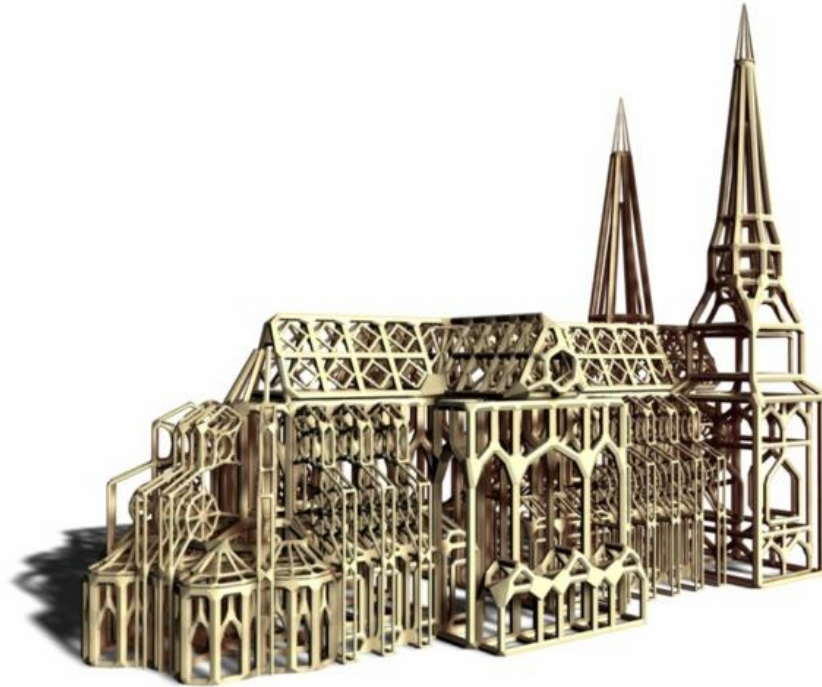


Figure 6: *Model of a cathedral created using the column modeling tool.*

4. Conclusions

In this paper we have presented a new method for converting a regular wireframe mesh into a mesh in which the edges of the wireframe are replaced by cylindrical pipes of finite thickness and volume, a process we refer to as “solidifying” the wireframe. Our method allows users to create *solid* wireframes. The resulting model conveys 3D information about the object, including interior structures, much better than a traditional wireframe rendering or a surface rendering.

Our algorithm works on any set of input edges – they need not be connected to each other, nor do they have to represent a manifold surface. The fact that models created using our new tool can be fabricated (using 3D printers or rapid prototyping machines) is particularly pertinent for architects who want to create conceptual sketches and produce real 3D models from them.

The tool has been used to create real and virtual environments, artistic depictions of known objects and architectural conceptual models. It can also be used to easily and quickly create complicated artistic shapes that would be difficult to generate using traditional modeling methods. Figures 7 and 8 show some more examples of conceptual design projects which make use of the solid wireframe modeling tool

Our new method makes available a powerful tool for end users. However there is still room for improvement. For example, it would be useful if users could create spherical joints. This would add the ability to create models of molecules and other chemical structures, which when fabricated could be used as educational aids. Another useful addition would be the option to have symmetric but non-circular cross-sections, such as elliptical cross-sections, which are found in several engineering structures in the real world.

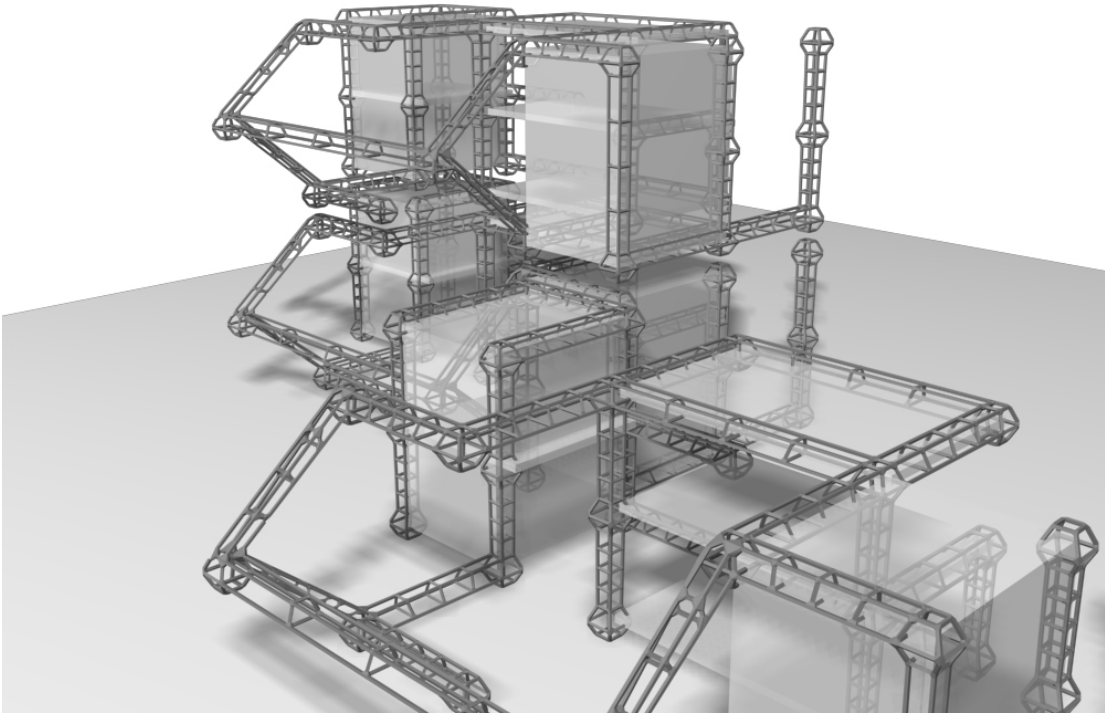


Figure 7: Conceptual work done using the column modeling tool. Design and modeling by Ozan Ozener.

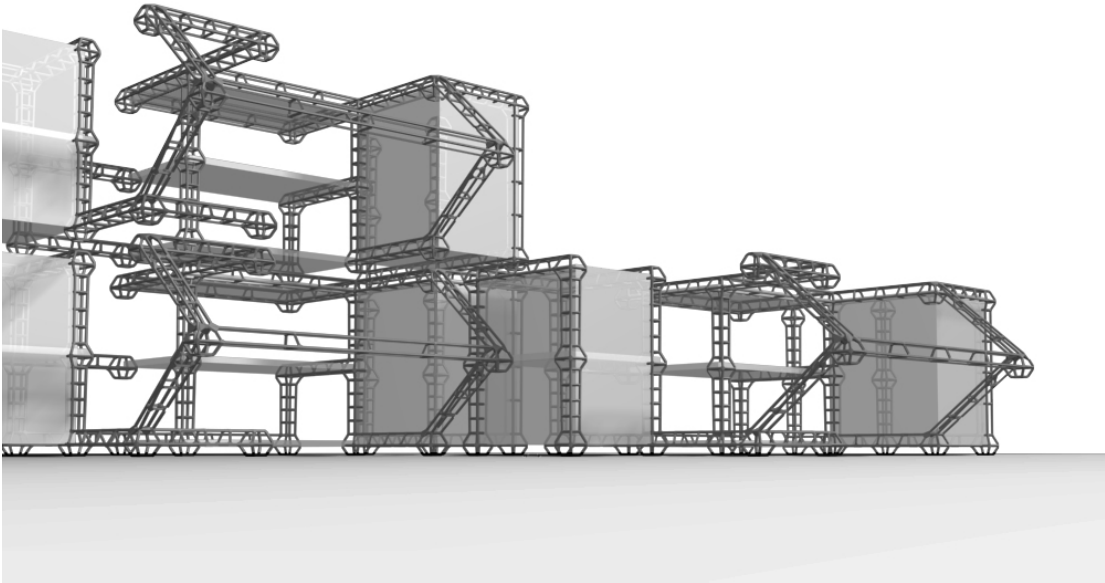


Figure 8: Conceptual work done using the column modeling tool. Design and modeling by Ozan Ozener.

References

- [1] AKLEMAN, E., CHEN, J., AND SRINIVASAN, V. An interactive shape modeling system for robust design of functional 3d shapes. In *Proceedings of ACADIA 2001* (Oct. 2001), pp. 248–257.
- [2] AKLEMAN, E., CHEN, J., AND SRINIVASAN, V. A prototype system for robust, interactive and user-friendly modeling of orientable 2-manifold meshes. In *Proceedings of Shape Modeling International 2002* (Banff, Canada, May 2002).
- [3] AKLEMAN, E., SRINIVASAN, V., AND CHEN, J. Interactive rind modeling. In *Proceedings of Shape Modeling International 2003* (Seoul, Korea, May 2003).
- [4] CATMULL, E., AND CLARK, J. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design 10* (1978), 350–355.
- [5] CATURANO, U., AND SANSEVERINO DI MARCELLINARA, C. Solid modelling by low-cost hardware and software: the crystal palace’s node. In *Proceedings of eCAADe Conference* (Eindhoven (The Netherlands), Nov. 1993).
- [6] DOO, D., AND SABIN, M. Behavior of recursive subdivision surfaces near extraordinary points. *Computer Aided Design 10* (1978), 356–360.
- [7] MANDAL, E. Wire and Column Modeling. Master’s thesis, Texas A&M University, 2004.
- [8] MANDAL, E., SRINIVASAN, V., AND AKLEMAN, E. Wire modeling. SIGGRAPH 2003, Sketches and Applications, July 2003.
- [9] SRINIVASAN, V. *Modeling High-Genus Surfaces*. Ph.D. dissertation, Texas A&M University, 2004.
- [10] SRINIVASAN, V., AKLEMAN, E., AND CHEN, J. Interactive construction of multi-segment curved handles. *Pacific Graphics 2002* (Oct. 2002).
- [11] VAN KEMPEN, A., KOK, H., AND WAGTER, H. Design modeling. *Automation in Construction 1* (May 1992), 7–13.
- [12] ZORIN, D., AND SCHRÖDER, P. Subdivision for modeling and animation. ACM SIGGRAPH 2000 Course Notes, 2000.