Physical Finite Elements

Samuel Eli Calisch

B.A., Mathematics
Grinnell College, 2010

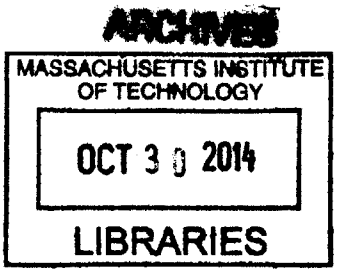
Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, in partial fulfillment of the requirements for the degree of

Master of Science

at the

Massachusetts Institute of Technology

September 2014



Signature redacted

Author .....
Sam Calisch
Program in Media Arts and Sciences
July 27, 2014

Signature redacted

Certified by ..
Prof. Neil Gershenfeld
Director, MIT Center for Bits and Atoms
Thesis Supervisor

Signature redacted

Accepted by .....
Prof. Pattie Maes
Interim Academic Head
Program in Media Arts and Sciences

Physical Finite Elements

by

Samuel Eli Calisch

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on August 8, 2014 in partial fulfillment of the
requirements for the degree of Master of Science.

Abstract

Engineering with digital materials, by discretely and reversibly assembling structure and function from a mass-produced construction kit of parts, is indeed an exciting vision. The ability to decouple conventionally linked material properties and reach new territory in parameter space has already been demonstrated [12] by the fabrication of ultralight samples with extreme specific stiffness. Further, these material properties can be spatially varied, opening new possibilities in engineering design. The discrete assembly process also frees us from constraints of monolithic manufacturing and the corresponding supply chains. Thus, this approach offers compelling promise for the design, manufacture, and deployment of large structures.

In this thesis, we argue that digital materials offer a further benefit in the power and accuracy of simulation possible, as compared to modeling materials with less order. At the most general level, this comes from mirroring the discrete nature of the structure in the mathematical model, creating a hierarchical representation of the assembly and treating each level independently. The results can reduce the cost to design and validate complex structures, in both the required computational resources, as well as the time and testing cycles of human engineers.

We outline several techniques for structural modeling of such digital material assemblies, focusing on workflow flexibility and engineering empowerment through custom design tools. We demonstrate two effective table-top part production techniques: one for producing many tightly-toleranced parts for validating simulations and one for producing high performance, directionally aligned composite parts in an out-of-autoclave process. We implement several structural tests in hardware and software, comparing results from modeling with empirical data. We show that at both the scale of individual parts, as well as of large assemblies, models synthesized from beam bending equations outperform more complicated and computationally intensive finite element simulations. Finally, we undertake an ambitious design study using these tools, using both simulation and physical testing to predict performance. The results suggest the feasibility of building skinned, lighter-than-air digital material structures, capable of withstanding atmospheric crush pressures and floating under the lift generated by the displaced air.

Thesis Supervisor: Neil Gershenfeld
Title: Director, MIT Center for Bits and Atoms

Physical Finite Elements

by

Samuel Eli Calisch

The following people served as readers for this thesis:

Signature redacted

Kenneth Cheung, PhD
Research Scientist, NASA Ames Research Center

5 Aug 2014
Date

Signature redacted

Geoffrey Irving, PhD
Research Scientist, Otherlab, Inc.

6 aug 2014
Date

Thanks

This thesis would not have been possible without the help and support of many people.

To Neil, thank you for providing the intellectually open atmosphere, the fleet of machines, the crew of strange and talented people, the weekly ~~battle royale~~ group meeting, and all the other things that make CBA possible.

To Will and Carlos, thanks for many, many therapy sessions and great collaborations.

To Nadya, Matt C, Matt K, James, Michael, Prashant, Noah, Lisa, Charles, Shuguang, Christian, and everyone else in CBA, thanks for being you.

To Spencer, thanks for your input and help with this work.

To Vadim Shapiro, thank you for your correspondence and insights.

To Joe, Ryan, John, Tom, and Jamie, thank you for morning time conversations and help making things happen.

To Sherry, thanks for taking me on adventures!

To the sponsors of CBA, thank you for the support to make this work possible.

To Kenny, thanks for inspiring me, and for your ability to drop everything for an hour-long conversation. Further, thanks for helping shape this work and for serving as a reader.

To Saul, thanks for your faith in me, for giving me an inspiring place to develop, and for providing the cover to learn how to do the things I pretended to know how to do.

To Tucker and Della, thanks for being great friends. In particular, thank you for roping me in to work on the double-donut chair.

To Geoffrey, thank you for writing code that serves as a model of elegance. Thank you for all your help over the years, including serving as a reader for this thesis.

To Keith and Martin, thanks for helping me develop as a programmer, and for some great collaborations.

To Dan, thanks for inspiring me in the quest to make beautiful things.

To Pete, thanks for showing me the power of stubbornness and a spreadsheet.

To Jonathan, Leila, Mike, Ryan, Forrest, Kevin, and many others at Otherlab, thanks for being excited, smart people who get things done.

To all denizens of Prentiss Palace, thank you for shared meals and late night conversations.

To Esther, thank you for your self-examined bravado.

To Nolan, Dad, and Mom, thank you for your ceaseless love and support, and for not being too mad about broken windows and such.

Contents

Chapter 1

Introduction

1.1 Digital Materials

The work of this thesis comes as a part of a research program in *digital materials*, where functional devices are reversibly assembled by placing discrete parts into discrete positions and orientations. In most instantiations, these positions and orientations are determined by kinematic connections between parts. As an example, Figure 1.1 shows a stack of identical parts, each with four joining locations. This joinery determines how the parts can be connected to create the lattice structure shown. The word *digital* is used here in analogy with modern communication and computation, where we perform operations with symbols, rather than with analog values of a global field (e.g. voltage or displacement). Like Lego toys, building with digital materials gives the ability to detect and correct errors, creating greater accuracy in the final assembly than was present in the assembly device (child, skilled fabricator, or robot).

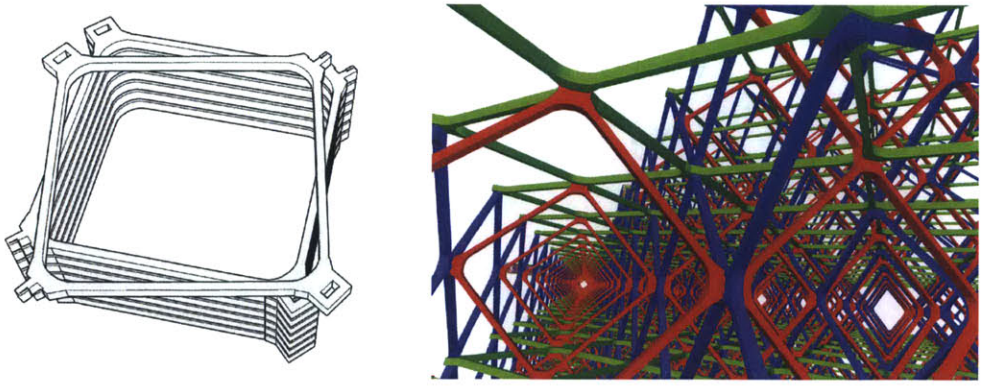


Figure 1.1: Discrete set of building blocks, assembled in discrete positions and orientations.

Due to their relative simplicity, digital material parts can be batch-produced at low cost and with few manufacturing constraints. This allows a great degree of flexibility in material selection, fabrication processes, and incorporation of functional elements. This also allows the final devices to exhibit a great degree of structural hierarchy, a well known strategy for creating high performance structures [26]. This claim has been demonstrated in the context of digital materials with the production of ultralight cellular solids of extremely high specific stiffness by discretely assembling oriented carbon-fiber-reinforced-polymer (CFRP) parts [11] [12].

Because assembly is discrete, functional properties can also be varied over the spatial extent of a part [38] to meet specific demands of an application. As an example, consider the design study shown in Figure 1.2, where we create a structure that is stiff in axial compression, but can deform along one bending axis. This behavior is determined by the design and placement of four distinct parts, shown in the top left image. Two of the parts form a cross section with zero Poisson ratio, allowing stretching without thinning. The other two parts create a stiff spine surrounded by a flexible part layer. By superimposing these behaviors in perpendicular part planes, we create a beam that can bend in excess of 60 degrees in one axis, but is stiff in the other bending axis, as well as in compression. The parts have also been designed with features for internal tendon routing for actuation, as shown in the top middle image. These tendons can be routed so motors can control the bending at several locations along the beam. For instance, using two such tendon pairs, we could produce the running gait shown in the lower image.

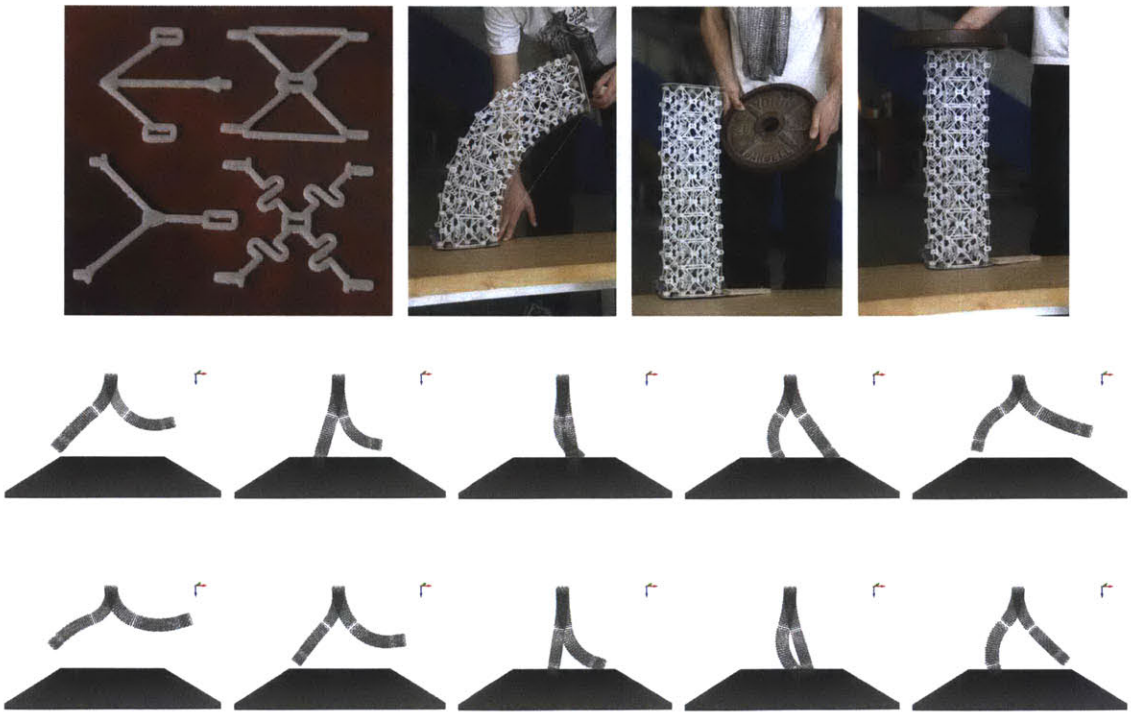


Figure 1.2: Assembling functionality: through assembly instructions four distinct part types determine a bending degree of freedom in one direction, and stiffness axially and in bending in the other direction.

While this example is structural in nature, digital material functionality can encompass much broader physical phenomena, with precedents in electromagnetics, fluid dynamics, and thermal processes. A good literature review of digital material approaches is given in [28].

The efficacy of this research hinges on the development of human-augmented and fully-automated assembly processes to scale these programmatic constructions to applications. Interconnect design is crucial to this development, as a small set of connection types can simplify the required action set of an assembly robot. As an example, consider the digital material "zippers" shown in Figure 1.3, inspired by [47]. The tapes shown have two joinery types (A) so that a two-step cam motion (actuated by human fingers) can zip the tapes together (B). Three such tapes

can form a stiff beam (C and D), more tapes can create internal structure (E), and curved profiles can code for three dimensional geometry (F). Crucially, the simple motions required to assemble the digital material allow for a repeatable process through human augmentation. One can imagine completely autonomous devices to place digital material parts. Such a robot could use the existing structure for locomotion and registration, placing new parts with simple moves. By parallelizing this automated assembly, the promises of digital materials could scale to great numbers of parts.

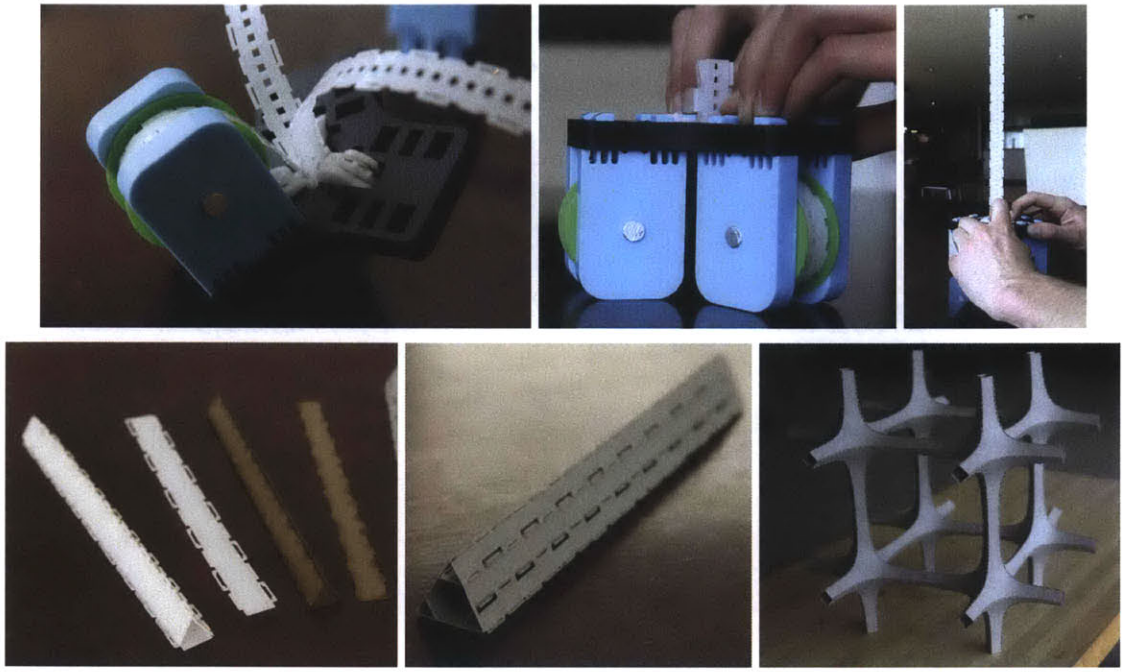


Figure 1.3: Tape-based digital material system with automated assembly, material flexibility, structural hierarchy, and capability for geometric complexity

Such discretely assembled structures also sidestep many of the size constraints of monolithic manufacturing (e.g. composite autoclave processes), allowing the rapid fabrication of structures much larger than the machines which made the parts. For instance, the expense of the massive machines necessary to make airplane-sized composites creates convoluted supply chains in the aerospace industry. We can imagine many of these facilities being replaced by factories manufacturing digital material parts using more tame, agile processes.

This line of reasoning can be pushed even further: If we significantly lower the performance and assembly costs of joined components, the scale of the structures we can build increases dramatically. Construction at such *geological* scales with discrete components has a precedent with Dolos and Kolos (or "Jackstones") [10], as well as honeycomb sea walls [5]. Both systems use a repeated, mass-manufactured element with interlocking geometry to fill vast volumes and mitigate the effects of coastal erosion. If digital material assembly processes were automated, one can imagine replacing the raw mass of these systems with the structural gains of over-constrained lattices. Such digital materials could also be designed to dissipate energy from incoming waves, a key feature of the Dolosse. With orders of magnitude less mass, such systems could be deployed quickly, in the face of an incoming natural disaster. Beyond such terrestrial applications, there is huge demand for large structures in space that can be assembled on-orbit. Digital material

systems for vast radio telescopes, solar collectors, and a variety of other infrastructure needs can easily be imagined.

To fully capitalize on these promises, however, we also need new kinds of design and modeling tools in the scope of digital materials. For instance, how might one optimize the bending behavior of the structure shown in Figure 1.2? How could we design a seawall construction system that dissipates the energy of an incoming wave? Given a volume to be filled, how can we work backward to determine the parts and connections that meet given specifications for the assembled structure? As with most inverse problems like this, the first step of its solution is the clarification and streamlining of the forward process. In this case, this means developing robust simulations that predict performance of discretely assembled structures under global constraints and inputs. It turns out that the discrete nature of digital materials is extremely helpful in this pursuit. As explained in the next section, the topic of this thesis is clarifying the extent to which global properties can be synthesized from the characteristics of such discretely assembled parts.

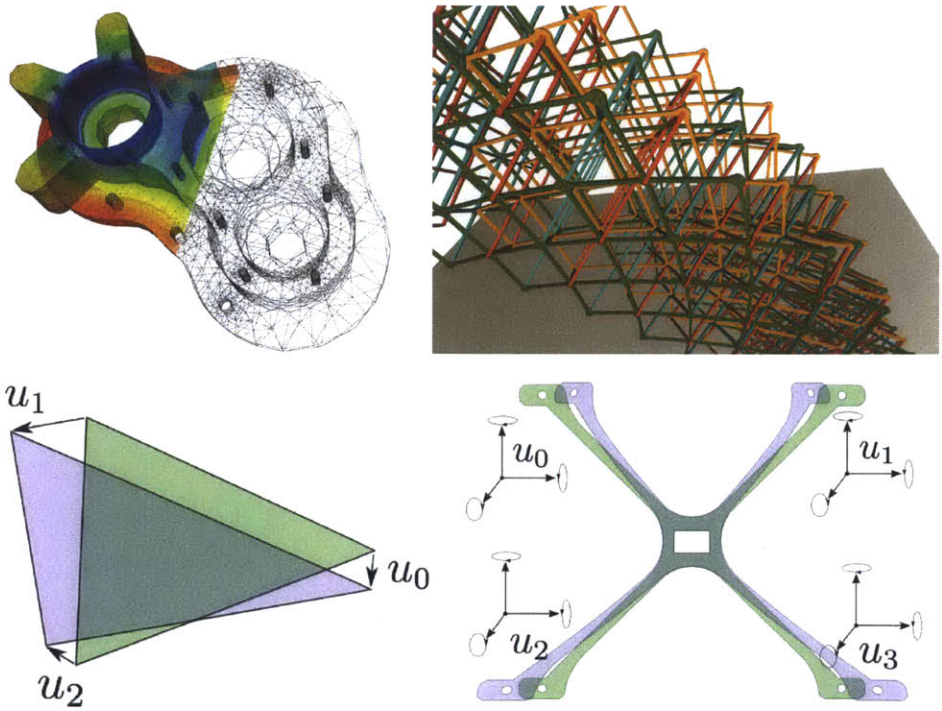


Figure 1.4: Defining deformations of a finite element in terms of nodal coordinates. Top left image from [45].

1.2 Modeling Digital Materials

At the most basic level, we seek to develop a modeling workflow that scalably exploits the fact that digital materials are composed of many copies of identical parts. That is, by spending work understanding the behavior of a part, can we abstract our findings and synthesize global behavior of the assembly? Effective mathematical models tend to use only the information relevant to the required answer, abstracting away the other unnecessary details [19]. To ask questions about the behavior of assemblies, we seek a description of the part based only on how it interacts with other

parts. Parts like those in Figure 1.2 use interlocking geometric features to enforce the assembly constraints, interacting only through these nodal features. Thus, we build a mathematical model for the forces and moments at nodal points only, disregarding the rest of the part.

In finite element analysis (FEA) [9], the geometry to be modeled is subdivided into many small, easy to analyze parts, like the tetrahedral mesh shown in Figure 1.4. We derive equations for the physics to be modeled on these elements, and solutions are specified by the displacements of the element nodes. The contribution of each element to the model is usually encapsulated in an *element stiffness matrix*, k_{el}. In the case of linear elasticity, this matrix relates the nodal displacements u_i with the nodal reaction forces r_i. The matrices can be aggregated into a global stiffness matrix K, and the vector U of all nodal displacements is the solution to a linear system $KU = R$, extremizing elastic strain energy.

Because we wish to simplify digital material parts to their nodal interactions, we can use the same process to model digital materials. In the case of conventional finite element analysis, we usually use the fact that the elements are small to make approximations to the constitutive equations and transform the partial differential equations into a linear system to solve. In the case of digital materials, the elements are not vanishingly small, so we must turn to other methods to derive the element stiffness matrices. These approaches are detailed in chapter 2 and chapter 3. To validate these approaches, in chapter 4 we produce many digital material parts. In chapter 5 we develop tools for designing digital materials based on high level descriptions, creating inputs for these simulations. In chapter 6, we build several assemblies of these parts and compare the results of structural testing to the predictions of these methods. Finally, in chapter 7, we undertake an ambitious design study to demonstrate the applications of a streamlined modeling workflow.

This approach to modeling offers several advantages over meshed finite element analysis of digital materials in engineering applications. First, and most simply, there is a huge reduction in the size of data structures and linear systems used to describe the problem. Correspondingly, the analysis is less costly, and can be iterated more times in the design process. Further, because the representation used to design and simulate is the same, these two stages of the engineering process can be more closely coupled, even occurring within the same interface. Finally, due the the reduction in complexity, physically meaningful design parameters can be pulled through the simulation loop more easily, offering better handles for structural optimization.

Chapter 2

Modeling part behavior

In chapter 1, we outlined a plan to create element stiffness matrices for digital material parts, abstracting away everything about the parts except the behavior of their nodes. In this section, we review the mathematical details of this process and describe several ways of calculating these matrices.

2.1 Finite element background

In its most general form, finite element analysis computes a scalar or vector field over a region, subject to a physical law and boundary conditions. In the case of structural analysis, we calculate a displacement field by minimizing strain energy over the domain. To do this, we break up the domain into small pieces, called elements, and specify a displacement field by its values at the set of nodes u_i defining the elements. The element can be thought of as a kind of spring connecting its nodes, and we can calculate the relationship between the nodal forces and nodal displacements. In this way, the physics is encapsulated in an element stiffness matrix k_{el}. Using this matrix, we compute potential energy in the case of static linear elasticity as

$$w_{el} = \frac{1}{2} u^\top k_{el} u \tag{2.1}$$

These element stiffness matrices are aggregated into a global stiffness matrix $K = \sum k_{el}$ and global potential energy is

$$W = \frac{1}{2} U^\top K U - R^\top U \tag{2.2}$$

where U is the global nodal displacement vector, and R, the *residual*, incorporates boundary conditions. For static analysis, the principle of virtual work implies that in stable equilibrium, strain energy is minimized across the structure. Thus, computing structural deformation amounts to solving the linear system $KU = R$.

In the case of meshed finite element analysis, these element stiffness matrices are calculated using *shape functions* which span the space of nodal deformations in which we are interested. The simplest shape function is the linear hat function, a piecewise linear function defined to take the value one on a particular node, and have value zero on all other nodes. Taking linear combinations of such functions allows us to represent any piecewise linear deformation as the dot product of a

coefficient vector and the vector of shape functions. Such shape functions give us a handle on the otherwise infinite-dimensional space of geometric deformations, and allow us to calculate the stiffness matrices directly by integrating the governing equations, using a numerical integration scheme like Gaussian quadrature. By making the elements vanishingly small, we hope to capture any relevant strain field, no matter how complex.

As mentioned in chapter 1, if we take digital material parts as finite elements, they are not vanishingly small and hence the approximations used to calculate stiffness matrices in conventional finite element analysis are not readily applied. While this may seem like a big problem, there are three redeeming facts about our *physical finite elements*:

1. We build with many repeated copies of nearly identical parts instead of many uniquely shaped elements. We can afford to spend some work understanding the part, because we will reuse this work many times.

2. The parts tend to be simple, so we can analyze them with established engineering workflows.

3. As the name implies, we have physical access to the parts. In a pinch, we can simply measure them. This measurement can also serve as a calibration step on other methods of calculating part behavior.

To calculate the stiffness matrices, we first note that they are nothing more than relationships between nodal displacements/rotations, and the nodal reaction forces/torques. Like the constant in a spring equation, the stiffness matrix just determines the proportionality constant between a displacement and a force:

$$k_{el}u = r \tag{2.3}$$

So, if we want to probe the entries of k_{el}, we could simply enforce a unit displacement in the i^{th} entry of u, setting all other entries to zero. This picks out the i^{th} column of k_{el} and shows it is equal to the vector of reactions at the nodes of the element. This observation provides a method to calculate each column of the matrix, no matter how we determine these nodal reaction forces.

For instance, one strategy is to represent the part as an assembly of solid mechanical models for which the relationships between displacements and reactions can be determined analytically. For instance, many of the digital material parts in this thesis can be reasonably represented as a collection of beams connecting the nodes. In this case, we have stiffnesses corresponding to each beam which can be synthesized to create the part stiffness matrix. In section 2.2, we discuss the details of this process.

If we have no such solid mechanical description, we could also appeal to a conventional finite element simulation of the part. In Figure 2.1 we see such a simulation, where we enforce displacements in each of the six degrees of freedom at a node and measure the reaction forces and torques at all the nodes. By simulating in these configurations, we can calculate the stiffness coefficients for each node, and build up the full stiffness matrix for the part. chapter 6 describes comparing these tests to physical measurement. We performed the analysis in Figure 2.1 commercial finite element package ANSYS, but we also experimented with many other tools and methods, including Abaqus, a custom-written 2D and 3D linear elastic finite element simulation, and a pixel-based simulator implemented using PETSc [7].

In addition, in chapter 3 we describe progress towards a simulation engine that operates directly on a distance-based representation, without going through the meshing step required for

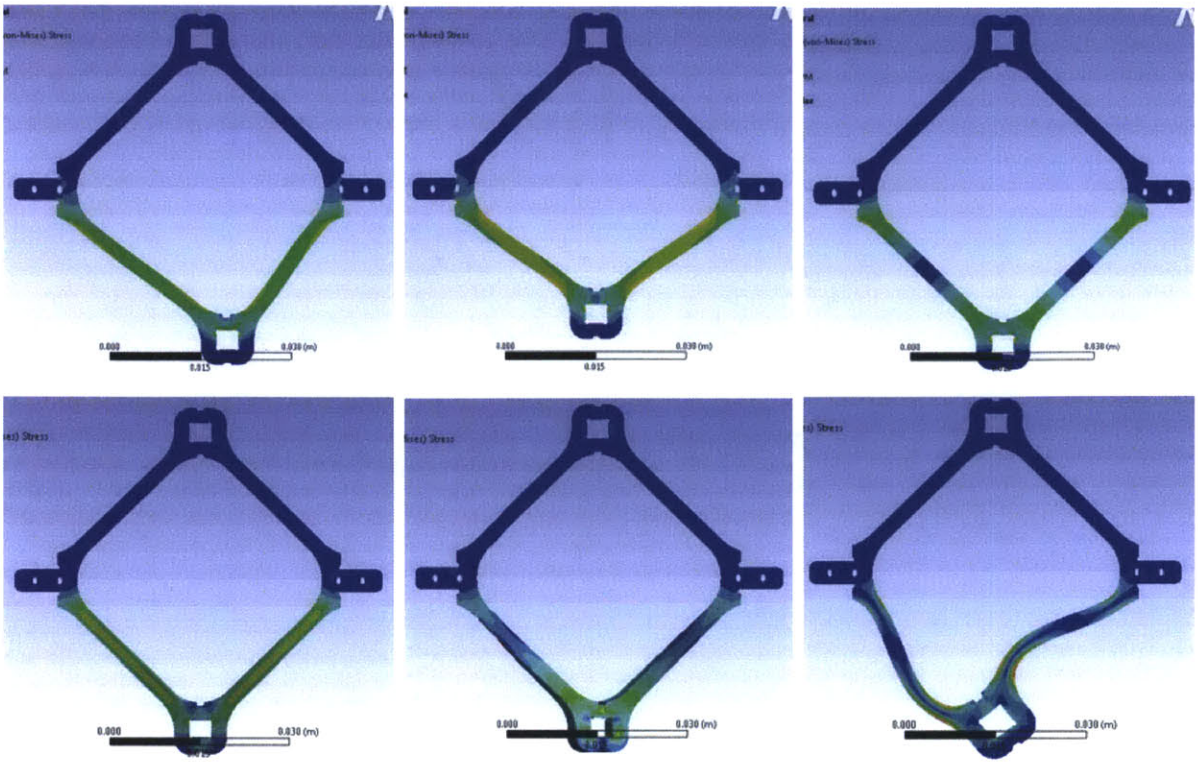


Figure 2.1: Generating a stiffness matrix through the force influence method: $\hat{x}, \hat{y}, \hat{z}, \phi_{\hat{x}}, \phi_{\hat{y}}, \phi_{\hat{z}}$.

conventional FEA. Much of the pain and innacuracy of computer-aided engineering comes from the many translation steps present in most simulation pipelines. Most of the designs in this thesis were created with a CAD tool [24] that uses an adaptively sampled distance field to represent geometry. Simulating directly in this representation could avoid some of the problems in these workflows.

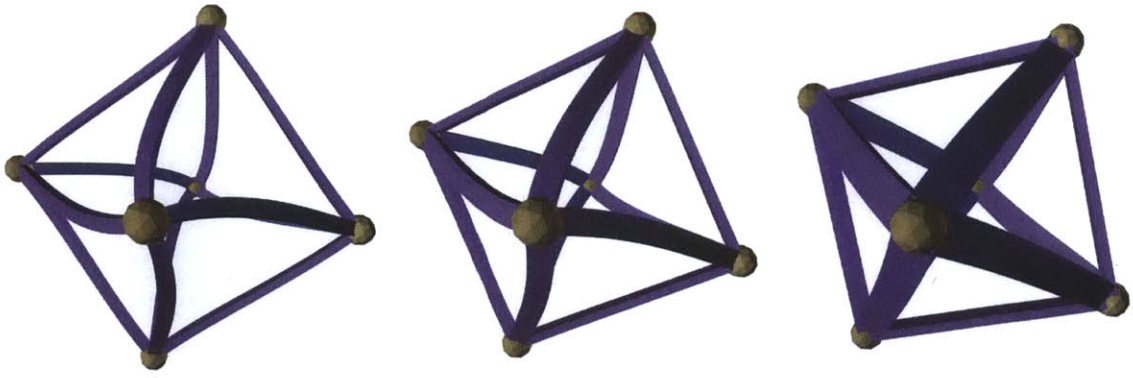


Figure 2.2: Varying piece parameters under global assembly, constraint, and load.

2.2 Beam-based model

If we can cleanly decompose a digital material part into well understood solid mechanical models like beams, our job of creating stiffness matrices is considerably easier. For instance, the part shown in Figure 2.1 can be represented by four beams, joined into a circuit, connected by relatively rigid regions. Using the same method described above to pick off columns of a stiffness matrix, we can calculate the stiffness matrix for a beam in three dimensions, taking into account forces related to bending, extension, and torsion. For this calculation, we assume the beam is oriented along the $\hat{x}$ axis, an assumption we will relax shortly. As the beam has six degrees of freedom at each of its ends (3 translations and 3 rotations), the stiffness matrix will be 12×12. Let x_i, y_i, z_i be the displacements of the i^{th} node, and ϕ_i^x, ϕ_i^y, ϕ_i^z be the i^{th} node rotations about the $\hat{x}$, $\hat{y}$, $\hat{z}$ axes. Using beam bending formulas, we calculate:

$$\begin{pmatrix}
\frac{AE}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-AE}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & \frac{-12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\
0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{-6EI_y}{L^2} & 0 & 0 & 0 & \frac{-12EI_y}{L^3} & 0 & \frac{-6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-GJ}{L} & 0 & 0 \\
0 & 0 & \frac{-6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & \frac{-6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\
\frac{-AE}{L} & 0 & 0 & 0 & 0 & 0 & \frac{AE}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{-12EIz}{L^3} & 0 & 0 & 0 & \frac{-6EI_z}{L^2} & 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{-6EI_z}{L^2} \\
0 & 0 & \frac{-12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 & 0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{-GJ}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 \\
0 & 0 & \frac{-6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} & 0 & \frac{-6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L}
\end{pmatrix}
\begin{pmatrix}
x_1 \\ y_1 \\ z_1 \\ \phi_1^x \\ \phi_1^y \\ \phi_1^z \\ x_2 \\ y_2 \\ z_2 \\ \phi_2^x \\ \phi_2^y \\ \phi_2^z
\end{pmatrix}
=
\begin{pmatrix}
f_1^x \\ f_1^y \\ f_1^z \\ \tau_1^x \\ \tau_1^y \\ \tau_1^z \\ f_2^x \\ f_2^y \\ f_2^z \\ \tau_2^x \\ \tau_2^y \\ \tau_2^z
\end{pmatrix} \tag{2.4}$$

where A is the cross sectional area, E is the elastic modulus, G is the shear modulus, J is the torsional constant, L is the beam length, and I_y and I_z are the second area moments of inertia about the $\hat{y}$ and $\hat{z}$ axes.

Now, if the beam is not oriented along the $\hat{x}$ axis, we can perform a coordinate transformation to align it. Thus to calculate the stiffness matrix of an arbitrarily-oriented beam, we can conjugate by the matrix of this transformation. Let T be a 3×3 rotation matrix mapping the beam axis to $\hat{x}$, and a reference vector in the beam cross section (e.g., a vector from the central axis to a face of a rectangular cross section) to $\hat{y}$. Denoting the matrix above as $k_e l^x$, we can form the beam stiffness matrix as

$$k_{el} = \begin{pmatrix} T^\top & 0 & 0 & 0 \\ 0 & T^\top & 0 & 0 \\ 0 & 0 & T^\top & 0 \\ 0 & 0 & 0 & T^\top \end{pmatrix} k_{el}^x \begin{pmatrix} T & 0 & 0 & 0 \\ 0 & T & 0 & 0 \\ 0 & 0 & T & 0 \\ 0 & 0 & 0 & T \end{pmatrix} \tag{2.5}$$

For a part with several such beams, we can add up the stiffness matrix contributions for each, summing entries for nodes shared between beams. In Figure 2.2, we show a simple test of the beam behavior, where we stitch together 8 beams into an octahedron in this way. We solve the assembled system for the linearized displacements and rotations at the nodes, and then plug these into analytic expressions for the bends and twists of the elements in order to visualize the beams.

2.3 Assembling part models

Once we have a stiffness matrix for a part, whether it comes from a collection of beams or from on of the other methods described above, we are ready to begin simulating assemblies of parts. This

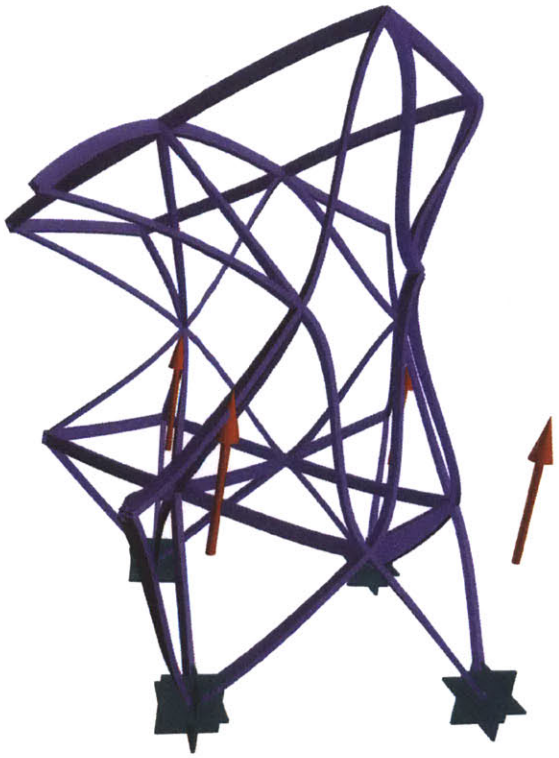


Figure 2.3: A fanciful simulation.

process is exactly analogous to the synthesis of multiple beams, as we simply add the contributions of each stiffness matrix, identifying nodes which are the same between the parts. For identical parts in different orientations, we again conjugate by a transformation matrix between coordinates. With this global stiffness matrix, K, we can set up a linear system for the problem:

$$KU = R \tag{2.6}$$

We add any traction boundary conditions to the residual R. We enforce essential boundary conditions by modifying the matrix and right hand side to have a trivial solution for the corresponding displacement, taking care to preserve the symmetry of the matrix [9][29].

A fancifully exaggerated loading of such an assembly is shown in Figure 2.3, illustrating this process. The stiffness matrix for each part is generated by modeling it as four connected beams. The parts are then used to generate a global stiffness matrix of the assembly. To impose boundary conditions, the nodes along the base are constrained, and a load is applied to the loads on the top face of the brick.

This assembly process need not use matrices from just a single part type. In Figure 2.4, we use one stiff part and one part with a flexural degree of freedom. By placing these part types on perpendicular planes, we can create a structure that exhibits anisotropy in flexural stiffness. The simulation shows the structure bending under a load in one direction, while remaining considerably stiffer in the perpendicular direction. In this way, we can now perform predictive simulation of structures like the digital material leg shown in Figure 1.2. While that structure was designed in a largely qualitative way to generate the running behavior, this simulation now gives

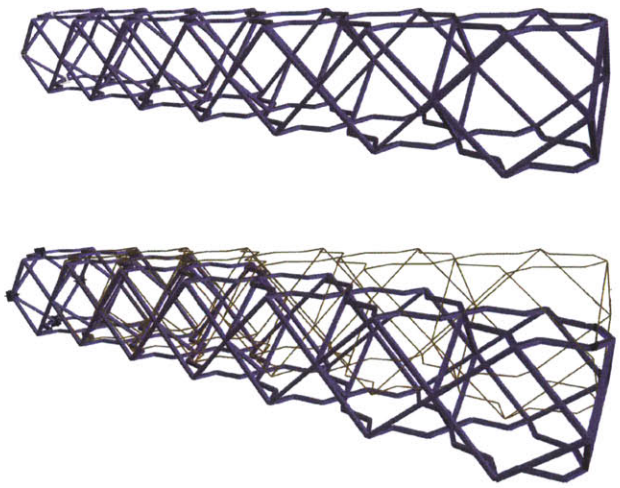


Figure 2.4: Placing two piece types in a beam can produce flexural anisotropy.

us the ability to quantitatively determine the characteristics of the part in terms of the bending stiffnesses we desire in the final assembly.

For a shape as simple as this, it is not hard to write routines to distribute and orient the parts and then transform and assemble their stiffness matrices. As these lattices are periodic, however, the process can be generalized to accommodate more complex lattice envelopes. In chapter 5 we discuss two workflows to generating lattices for engineering applications, after which the stiffness matrix assembly is automatic. These tools could be used to implement and optimize the full leg structure in Figure 1.2, drastically improving on the design tools available at the time of its construction.

First, however, in chapter 3 we detail progress towards another method of determining part stiffness matrices, by simulating direction in the distance field representation in which they were designed.

Chapter 3

Mesh-free simulation

This section describes work towards a mesh-free simulation framework based on a distance field geometry representation. Many of the digital material parts designed for this thesis were created in a CAD tool [24] using a functional representation to perform geometric operations. This functional description and the implementation are described in [25].

To generate stiffnesses for the parts in this thesis, we have either assumed a decomposition into more primitive elements like beams, or we have taken the geometry into a geometry pipeline involving contouring, simplification, translation to a boundary representation (e.g., STP format), and finally volumetric meshing. Such mesh-based workflows, though commonly used in engineering design, impose significant time costs on simulation iteration, often introduce human errors, require each step in the process to function with incomplete information, and reduce workflow flexibility.

Alternative formulations of geometric field modeling for engineering date back to the 1930s [23] and have received a recent revival due to work like [39], [40], [15], and [16]. In the pursuit of end-to-end workflows [37], we describe progress towards implementing such mesh-free modeling using the same distance-based description as used in the CAD tools mentioned. In what follows, we use the analysis of [16] quite closely, essentially applying it in the specific context of our design workflow.

3.1 Kantarovich's method: 1-D example

In the method proposed by Kantarovich [23], to model physical fields (e.g. elastic strain) over designed geometry subject to boundary conditions, we can assume a solution of the form $u = u^* + \sum_i \omega\chi_i$. Here ω is a smooth function measuring distance to the boundary where an essential condition is specified, u^* is a function defined on the entire domain interpolating the essential boundary condition values, and χ_i are shape functions (like the linear hat functions described in chapter 2) defined over the domain (but containing no information about the boundary).

To illustrate this representation, we reproduce the tensioned cable example of [16], the one-dimensional analog of the linear elasticity problem we wish to solve. A cable with tension λ runs from $x = a$ to $x = b$ and has an applied load of q over its span. The governing equation is $\lambda\frac{\partial^2 u}{\partial x^2} + q = 0$, where u gives the height of the cable. We have fixed boundary conditions $u(a) = u_1$ and $u(b) = u_2$. To find an approximate distance function ω from the boundary, we construct it

using the distances from each end of the interval, $\omega_1 = x - a$ and $\omega_2 = b - x$. We take function $\omega = \omega_1^2 + \omega_2^2 - \sqrt{\omega_1^2 + \omega_2^2}$ as an approximate distance function to the full boundary.

We assume the solution has a form $u = u_0 + u^* = \sum_{i=1}^{n} C_i \eta_i + u^*$, where u_0 satisfies homogeneous Dirichlet boundary conditions, and u^* is a function satisfying the boundary conditions. We use the distance functions to construct these functions: $u^* = \frac{\omega_1 u_2 + \omega_2 u_1}{\omega_1 + \omega_2}$ and $\eta_i = \omega \chi_i$, where χ_i is any shape function.

To allocate the shape functions, we simply fill an arbitrary interval $[x_i, x_f]$ containing $[a, b]$. We use n linear hat shape functions on this interval, resulting in a grid size of $h = (x_f - x_i)/n$. In this way, the nodes of the shape function have no topological relationship with the domain of integration. While this may not seem like a significant simplification in this one-dimensional example, in two and three dimensions this avoids a great many headaches in meshing.

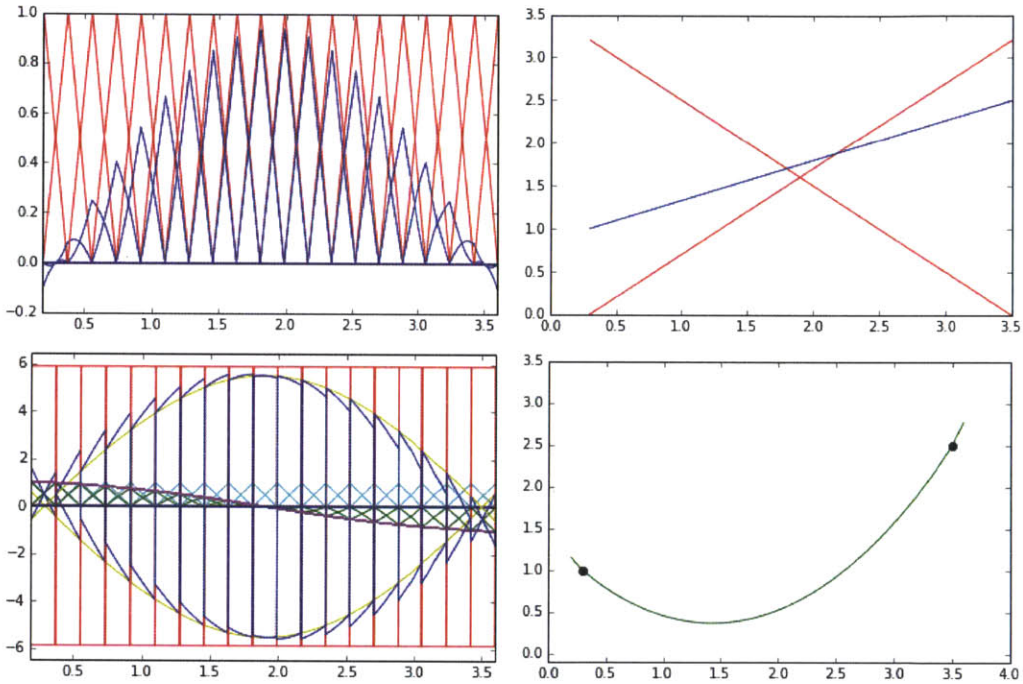


Figure 3.1: Following the one-dimensional example of [16], we can solve for the displacement of a tensioned cable under load.

Using the assumed form of the solution, we plug this into the governing equation.

$$\begin{aligned} \int_a^b \left(\lambda \left(\frac{\partial^2 u_0}{\partial x^2} + \frac{\partial^2 u^*}{\partial x^2} \right) + q(x) \right) \eta_j(x) dx &= 0, \quad j = 1, ..., n \\ -\sum_{i=1}^{n} C_i \int_a^b \frac{\partial \eta_j}{\partial x} \lambda \frac{\partial \eta_i}{\partial x} dx &= -\int_a^b q(x) \eta_j(x) dx + \int_a^b \frac{\partial \eta_j}{\partial x} \lambda \frac{\partial u^*}{\partial x} dx \quad j = 1, ..., n \end{aligned}$$

That is, we have a linear system $Ax = b$ where $A_{ij} = -\int_a^b \frac{\partial \eta_j}{\partial x} \lambda \frac{\partial \eta_i}{\partial x} dx$, $b_i = \int_a^b \frac{\partial \eta_j}{\partial x} \lambda \frac{\partial u^*}{\partial x} dx - \int_a^b q(x) \eta_j(x) dx$, and $x_i = C_i$. Differentiating, we have $\frac{\partial \eta_i}{\partial x} = \frac{\partial \omega}{\partial x} \chi_i + \omega \frac{\partial \chi_i}{\partial x}$. Using ω as above, we have $\frac{\partial \omega}{\partial x} = \frac{\omega_1 - \omega_2}{\sqrt{\omega_1^2 + \omega_2^2}}$. Similarly, $\frac{\partial u^*}{\partial x} = \frac{u_2 - u_1}{\omega_1 + \omega_2} = \frac{u_2 - u_1}{b - a}$. Using these definitions, in Figure 3.1 we

plot the unscaled and scaled shape functions in the top left image. In the top right, we show the distances from each boundary condition, as well as the boundary condition interpolation u^*. At the bottom left, we build the scaled shape function derivatives, and at the bottom right, we plot the computed solution $u^* + \sum_i C_i \eta_i$, taking care to exclude any shape functions completely outside the integration domain from the solver.

3.2 Applying to geometry described with ASDF

The geometry engine in [24] facilitates the design used for the digital material parts in this thesis. It uses a *functional description* of the geometry, where a mathematical formula evaluates to `True` for all points contained in the volume of a part, and `False` everywhere else. This representation provides an elegant method for implementing Booleans and many other operations, but for rendering, toolpathing, and export, the functional representations must be efficiently evaluated.

For this, [25] uses an *adaptively sampled distance field* (ASDF) data structure to limit the number of distance evaluations necessary to produce arbitrarily-detailed reproductions of the functionally-described geometry. Briefly, the functional description is evaluated on an octree-like spatial data structure. Starting with a bounding region, we subdivide and evaluate cell corners. If a cell is fully contained on one side of the boundary (FILLED or EMPTY states), or if it is smaller than a given threshold (LEAF states), we stop subdividing. LEAF cells are combined if this operation does not exceed an interpolation error threshold. In Figure 3.2 at the left, we show a two-dimensional slice of a digital material part represented with an ASDF data structure. The larger cells do not intersect the part boundary, and so don't need to be divided further.

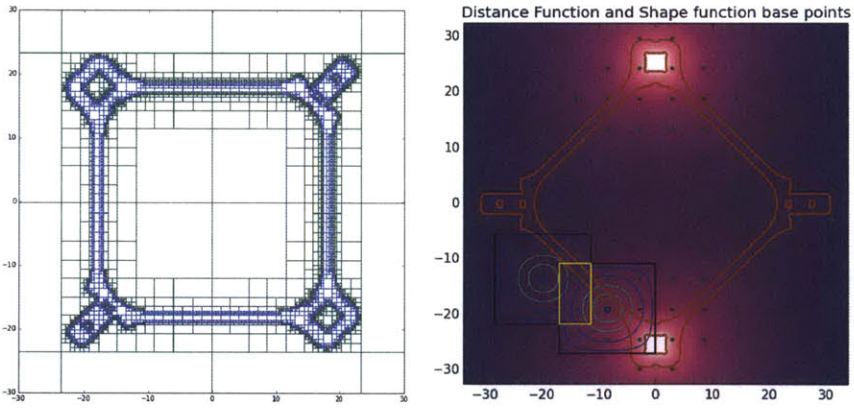


Figure 3.2: A) A digital material part represented as ASDF, B) Two shape functions with intersecting supports, and the distance field to a nodal boundary.

We explore using the functional description, as well as the ASDF data structure to create distance functions, so we can scale our shape functions as in the one-dimensional example. At the right in Figure 3.2, for example, we see the distance field describing essential boundary conditions on two opposing nodes. We also plot the contours of two overlapping B-spline shape functions to be scaled. These shape functions (more precisely, their derivatives) are used in the calculation of solid mechanics integrals. Unlike the one-dimensional example, however, in three dimensions we

must take more care to keep the problem tractable. In what follows, we outline progress towards this goal.

As a problem statement, we assume we have geometry described by an ASDF data structure, as well as a set of geometry descriptions for essential boundary conditions.

3.3 Shape Functions

As in the 1-D field modeling example, the first step is selection and allocation of shape functions over the domain. We use B-splines, a flexible, multivariate class of shape functions that include and generalize linear hat functions. They are easy to implement, have a solid theoretical basis, and can be combined to represent a large set of functions. A good overview of their properties and uses is given by [22].

Next, we must allocate the B-splines over the domain represented by the ASDF. In Figure 3.3, we draw an algorithm for this process in two dimensions. We begin with a uniform grid of size h and eliminate the B-splines whose support does not intersect a FILLED or LEAF cell. The support for each remaining shape function is divided along each spatial axis, spawning eight new shape functions of grid size $\frac{h}{2}$. This process is repeated to create an array of shape functions that is adaptive to the modeling geometry.

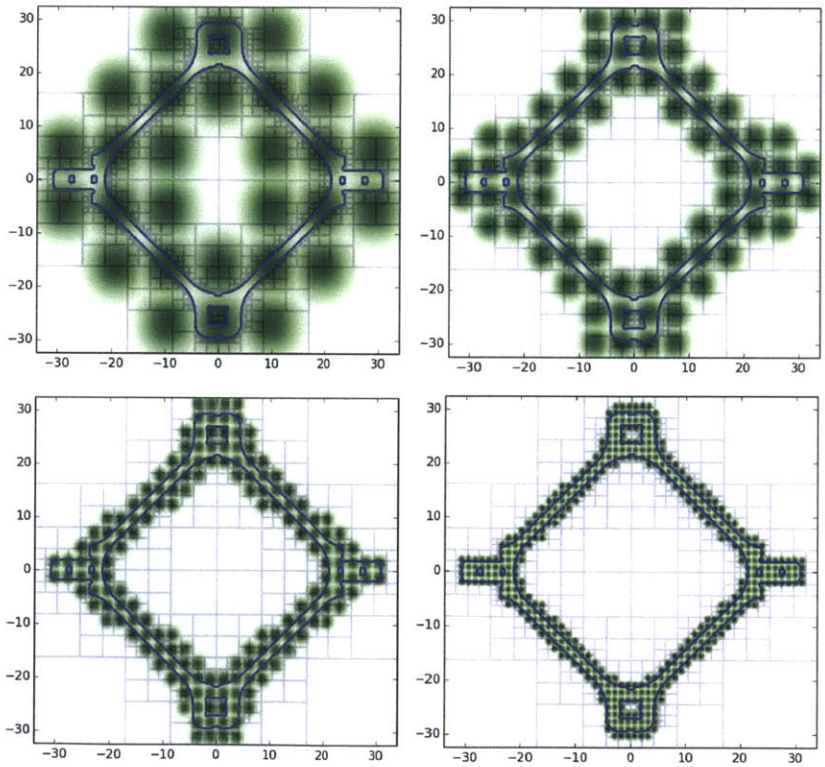


Figure 3.3: Shape function allocation and refinement

The ASDF hierarchy offers interesting potential for shape function arrays with varying grid sizes, but care must be taken to satisfy the interpolation conditions required of shape functions used for field modeling. There has been considerable research on this topic, most notably Grinspun's

conforming, hierarchical, adaptive refinement methods (CHARMS) [21]. For the scope of this work, however, we only used uniformly sized shape functions.

3.4 Solid mechanics integrals

With the shape functions allocated, we need to calculate their contributions to the stiffness matrix and to the right hand side of the linear system. Assuming we have a number of essential boundary conditions to enforce, we have a distance function ω_i for each. For the sake of example, suppose as in Figure 3.2 that we have two essential boundary conditions to enforce, given by distance functions ω_1 and ω_2. The boundary conditions are given by displacement vectors $\mathbf{U_1}$ and $\mathbf{U_2}$ for the bodies.

As in the one-dimensional case, we assume a decomposition of the solution into homogeneous and inhomogeneous parts:

$$\mathbf{u} = \sum_i \eta_\mathbf{i} \mathbf{C_i} + \mathbf{u}^*, \quad \eta_\mathbf{i} = \omega \begin{pmatrix} \chi_{i,x} & 0 & 0 \\ 0 & \chi_{i,y} & 0 \\ 0 & 0 & \chi_{i,z} \end{pmatrix}, \quad \mathbf{C_i} = \begin{pmatrix} C_{i,x} \\ C_{i,y} \\ C_{i,z} \end{pmatrix}$$

where the distance function ω is zero on all essential boundary conditions and the function $\mathbf{u}^*$ interpolates the essential boundary condition displacements over the entire domain. We construct ω using R-conjugation [39]:

$$\omega = \omega_1 + \omega_2 - \sqrt{\omega_1^2 + \omega_2^2}$$

We construct $\mathbf{u}^*$ as

$$\mathbf{u}^* = \frac{\omega_1 \mathbf{U_2} + \omega_2 \mathbf{U_1}}{\omega_1 + \omega_2}$$

which, by inspection, interpolates the boundary condition values between the zero sets of ω_1 and ω_2.

We plug these expressions into the weak form of the solid mechanics equations:

$$-\sum_{i=1}^{n} \int_\Omega \mathbf{B}[\eta_\mathbf{i}]^\top \mathbf{D}\mathbf{B}[\eta_\mathbf{j}] d\Omega = \int_\Omega \mathbf{B}[\eta_\mathbf{j}]^\top \mathbf{D}\mathbf{B}[\mathbf{u}^*] d\Omega - \int_\Omega \eta_\mathbf{j} \mathbf{F} d\Omega \tag{3.1}$$

where $\mathbf{F}$ is a body force, $\mathbf{B}$ is the strain-displacement matrix, and $\mathbf{D}$ is the stress-strain matrix:

$$\mathbf{B} = \begin{pmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{pmatrix}, \quad \mathbf{D} = \begin{pmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 2\mu \end{pmatrix}$$

for Lamé parameters λ and μ. We can also include a traction term on the right hand side of this system, but for present purposes we omit it.

Thus, to calculate the linear system coefficients, we need to differentiate the expressions associated with the distance functions. Applying differentiation rules, we have:

$$\frac{\partial}{\partial x}\mathbf{u}^* = \frac{\omega_1 \frac{\partial \omega_2}{\partial x} - \omega_2 \frac{\partial \omega_1}{\partial x}}{(\omega_1 + \omega_2)^2} (\mathbf{U_1} - \mathbf{U_2})$$

and

$$\frac{\partial}{\partial x}\omega = \frac{\partial \omega_1}{\partial x} + \frac{\partial \omega_2}{\partial x} - \frac{\omega_1 \frac{\partial \omega_1}{\partial x} + \omega_2 \frac{\partial \omega_2}{\partial x}}{\sqrt{\omega_1^2 + \omega_2^2}}$$

and similarly for the other coordinate directions. To implement these formulas, we use Python's `sympy` module [42] for symbolic manipulation and the contained `ccode` command to turn the large expressions into code.

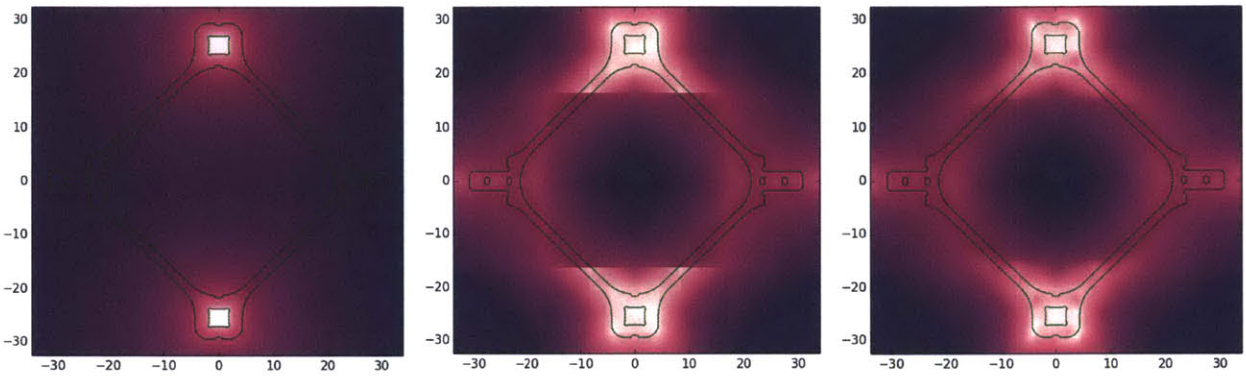


Figure 3.4: Functional description of boundary conditions, ASDF Sampling, MPU sampling

3.5 Distance functions

To implement the functions ω_i, we can use a functional description if the boundary is simple enough. The left most image in Figure 3.4 shows a distance function computed in this way, using a combination of R-functions [39]. While we can construct such functions for more complex boundaries [40], these descriptions can become expensive to evaluate many times. Thus, we pursue methods to use the ASDF for distance evaluations. The middle image in Figure 3.4 shows the raw distance information contained in the ASDF, based on trilinear interpolation of the distance values stored at cell nodes.

To guarantee convergence of our solution, however, we require more smoothness in the distance function. Thus, the naive use of the ASDF distance values will not suffice. To triage this situation, we bring in a technique from surface reconstruction: Multi-level Partition of Unity (MPU) approximation [32]. This method is used to stitch together many local approximations into a smooth function without requiring a global solving step (as with most radial basis function approaches).

In our case, the trilinear interpolation Q_i on each ASDF cell constitutes a local approximation to the distance function. To stitch them together, we introduce a compactly-supported weight function w_i for each cell, so that the support of w_i properly contains the cell. The global distance function can then be approximated as

$$f(x) \approx \sum_i \frac{w_i(x) Q(i)}{\sum_j w_j(x)} \tag{3.2}$$

To implement this, we use a trivariate B-spline for each w_i (and hence each ASDF cell), with support equal to a dilation of the cell. The dilation factor controls the amount of smoothing

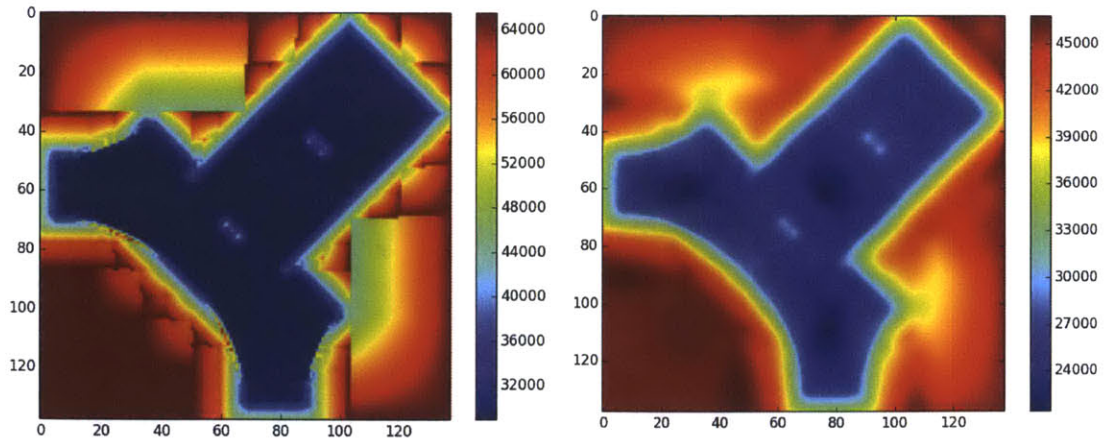


Figure 3.5: Comparing the distance function contained in the ASDF and after MPU sampling.

present in the function approximation. With zero dilation, the method returns the same value as a simple ASDF evaluation, but as we increase the dilation, the smoothness (and computational cost) of the distance function increases.

To evaluate the MPU distance value at a point, we start by traversing the ASDF tree, accumulating the values of $\sum_i w_i(x)Q(i)$ and $\sum_j w_j(x)$. If the point isn't contained in the dilation of current cell, we leave them unchanged. If the cell is LEAF, FILLED, or EMPTY, we calculate $w_i(x)$ and $Q(i)$ and add them to our accumulated values.

This implementation allows us to prune large sections of the ASDF tree that are irrelevant to the point in question. It is also adaptive to the refinements of the ASDF, smoothing with finer spatial resolution near the part boundary. In the right image in Figure 3.4, we compare the functional, ASDF, and MPU distance functions. In Figure 3.5 we show a detail of the digital material part, plotting the pure ASDF distance field and the MPU-smoothed field. Many of the artifacts have disappeared, and the distance field appears smooth, even in this region of geometric complexity.

3.6 Numerical Integration

With all the functions of the integrands in place, we must perform the integration in Equation 3.1 over the geometrically complex domain. Numerical integration is usually carried out by allocating function evaluation points according to Gaussian quadrature rules and taking weighted sums of their values. There are three-dimensional quadrature rules [41], but for smooth functions, repeated one-dimensional integration is often used to compute integrals in multiple dimensions [35]. The problem of integration over our complex domain, then, is a problem of allocating rays over which to allocate one-dimensional Gaussian quadrature points.

Fortunately, the ASDF data structure provides an effective way to do this. The domain is divided according to the cells of the ASDF. EMPTY cells make no contribution to the integral, while FILLED cells can be integrated as a simple prism. Figure 3.6 shows a slice of a digital material part, with integration points allocated in the FILLED cells according to repeated Gaussian quadrature.

LEAF cells straddling the boundary present a greater challenge. In [27], the importance of proper coordinate system choice for computing integrals over such regions is stressed. As an

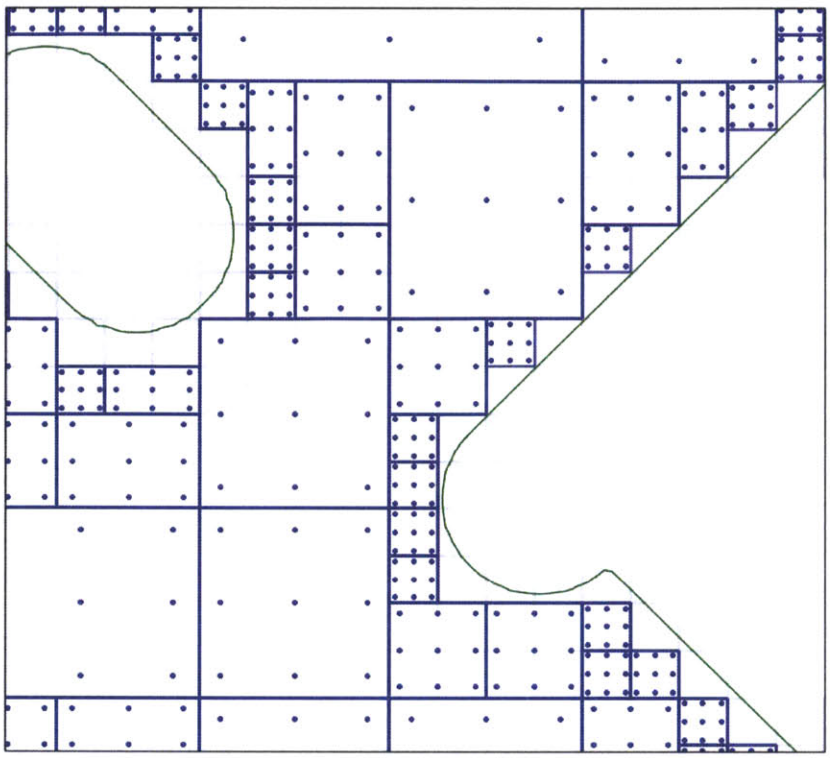


Figure 3.6: Detail of a part, showing integration rules in two dimensions on filled cells only.

illustrative example, the authors consider computing the integral of a constant over a quarter circle using cartesian and polar coordinate systems:

$$\int_0^1 \int_0^{\sqrt{1-x^2}} 1 dy dx = \frac{\pi}{4} = \int_0^{\pi/2} \int_0^1 r dr d\phi \tag{3.3}$$

Symbolically, these integrals are identical, but numerically, the polar form integral converges must faster due to better integration point placement.

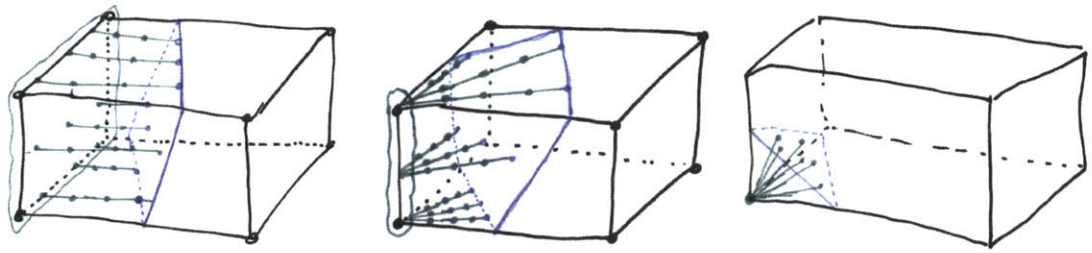


Figure 3.7: Integration rules for leaf cells.

With this in mind, the authors suggest placing integration points along a boundary using rules based on the marching cubes algorithm (drawn in Figure 3.7). If a face of LEAF cell is contained in the part, the integration points are allocated by cartesian rules over this face, and along rays

from each point to the boundary intersection. If only an edge is contained, points are allocated along the edge, and rays to the boundary from each are laid out in cylindrical coordinates. If no edge is contained, we pick a contained vertex and lay out rays by spherical coordinates from this vertex to the boundary.

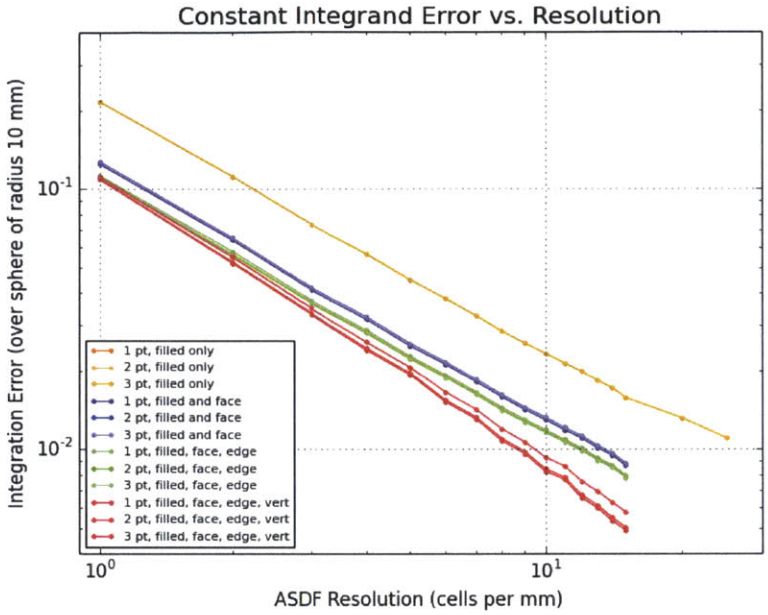


Figure 3.8: Testing numerical integration routines over the sphere

We implemented these integration routines for the ASDF data structure. As a test, we consider integrating a unit integrand over the sphere. The result should converge to the sphere volume, so we can evaluate the error. Figure 3.8 plots the error for this test, showing how each additional leaf cell case increases integration accuracy for a given resolution.

3.7 Results and Future Work

With these integration routines in place, we can construct the linear system in Equation 3.1 and solve the elasticity problem. In the scope of this thesis, we had hoped to use this as another method for constructing the digital material stiffness matrices. In reality, however, the results produced by the implemented mesh-free simulation engine were not ready for such use. Extracting well behaved linear systems was a challenge, and would likely benefit from tuning and testing. Further, to extract the stiffness measurements needed for use in characterizing digital material parts, additional surface integration routines were needed to calculate traction forces. In the shortness of time, this mesh-free simulation engine was set aside as a direction for future work.

Chapter 4

Part Production

With the models of part behavior and assembly modeling covered, we next turn to physical fabrication of digital material parts. To validating the proposed hierarchical approaches for modeling, we need an ample supply of parts to test and assemble. First, we demonstrate a workflow for quickly producing many parts, with tolerances sufficient for reversible, load-transferring node connections. To simplify modeling part behavior, we require that these parts should be made from material that behaves as a quasi-isotropic continuum down to length scales around 1 mm.

Second, we demonstrate an out-of-autoclave, net-shape molding process for producing high-performance, directionally-aligned composite parts, like those made by [12]. Using resin transfer molding (RTM), we can automatically wind dry fiber into mold, adding strength and stiffness where it is needed, finally pulling and curing resin to create a part that requires no post-cure machining.

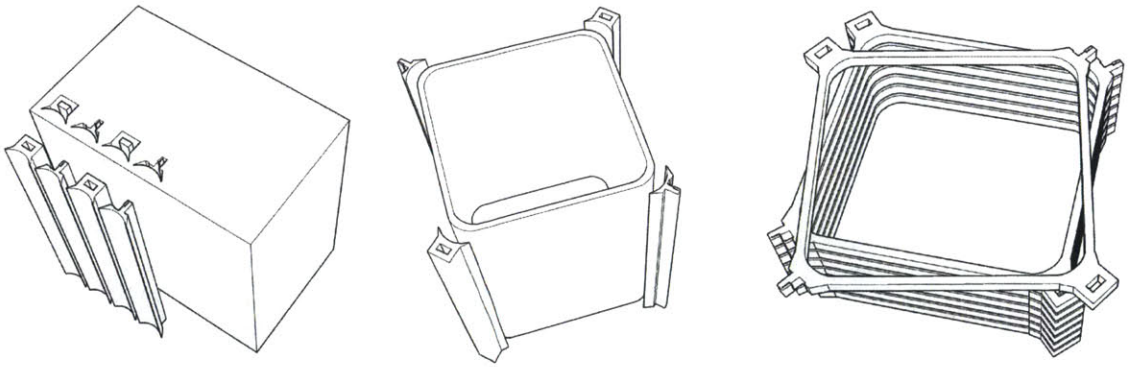


Figure 4.1: First proposed method for standardizing part interface

4.1 First steps

We fabricated the first potential part design for rapid production using a wire electric discharge machining center. Attempting to multiplex part production, a stack of standardized interfaces were cut from a large aluminum block (see Figure 4.1, left). These interfaces were then glued to square tube stock and wafer-cut to create individual parts. This process allows a wide variety

of stock to be used as the tendon material, while maintaining a standard, robust aluminum part interface.

As shown in Figure 4.2, using .010" EDM wire, we cut connections from a 4" aluminum block. These pieces were joined to a fiberglass square tube using Loctite Hysol. The parts were cut from this assembly using a precision wafering saw.

Figure 4.2: First experiments with standardizing part interface, using EDM machining.

Unfortunately, this process had several drawbacks. First, the wafering operation was too time-intensive for a rapid prototyping process. Second, after cutting the parts, the glued patches between fiberglass and aluminum had too little area to stand up to the loads required of the these parts. A barbed design could overcome this last drawback, but in light of the time-intensity, we abandoned this fabrication process.

4.2 Milled Phenolic

Next, we turned to milling the parts on a desktop CNC router. To increase throughput, we fabricated a custom vacuum fixture table, allowing higher cutting forces (and correspondingly higher feeds) while keeping the required tolerances on loose parts. We considered two materials for parts: a cotton-phenolic composite (.065" thickness) and medium quality baltic birch plywood (3-ply at .0625" thickness). The phenolic was heavier (1.32 vs 0.52 g/cm^3) and more expensive, but was slightly stiffer and more uniform at small scales.

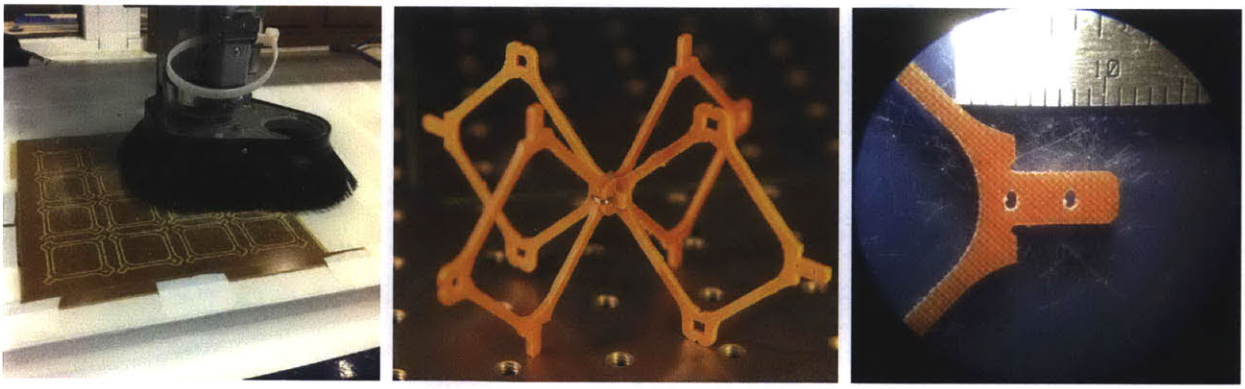


Figure 4.3: Milling with vacuum fixture, a pinned joint, and a microscope photograph of a node.

To decide between these materials, we milled coupons of each to determine modulus and isotropicity. Using the Instron 4411 with a 500N load cell, we applied loads up to $400N$ at a rate

of .25 mm/min. In tension, the phenolic had slightly higher modulus and was more isotropic than the birch. The anisotropy of the birch is even more evident in bending (as opposed to tension), since it has only three plies. These results suggested the phenolic was a better choice, but proceeded to make some parts from both. These parts showed that the lattice from phenolic parts was much easier to assemble without the risk of breaking parts as the features of the parts were on the same magnitude of size as the grain of the birch. In light of this reasoning, we selected the phenolic as the material for our structural testing.

4.3 Pins

To facilitate faster iteration times for testing assemblies, we also fabricated forming tools for making custom pins to join the digital material parts together. Figure 4.4 shows a simple press for bending .030" steel wire sections. We cut these sections in large numbers on a shear, and bent them in groups of thirty at once using this press.

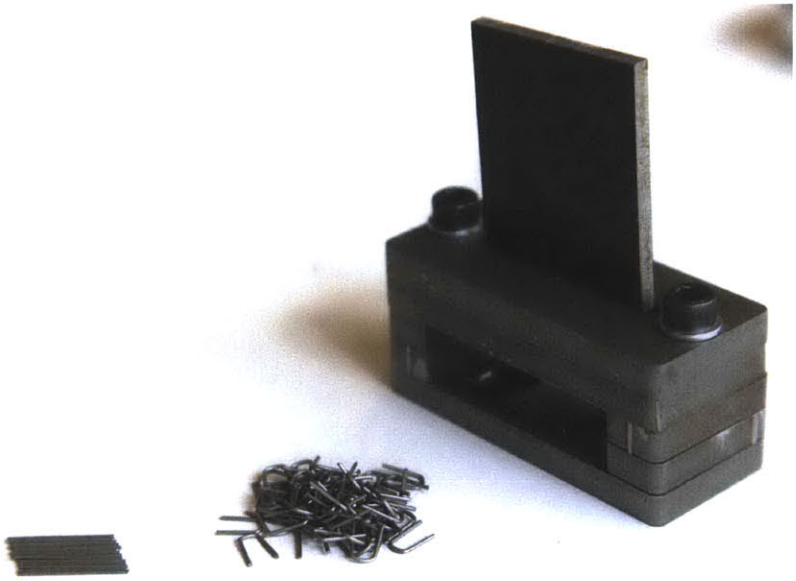


Figure 4.4: Pin press

4.4 Ganged Resin Transfer Mold

In addition to the milled phenolic parts for high throughput testing, we also developed workflows for higher-performance, directionally aligned carbon-fiber-reinforced polymer (CFRP) parts. This parallel production technique was based on that of [11], but used plastic layers to build up a resin transfer mold (RTM) for many identical parts, instead of machining them from a monolithic part after curing. These interlocked mold layers provide the necessary compaction and vacuum integrity for a quality out-of-autoclave (OOA) composite part, while also providing a scaffold for dry fiber winding. This produces composite parts with controlled fiber fill fractions, few voids, and directional fiber alignment. In this way, we can achieve the directional stiffness seen in

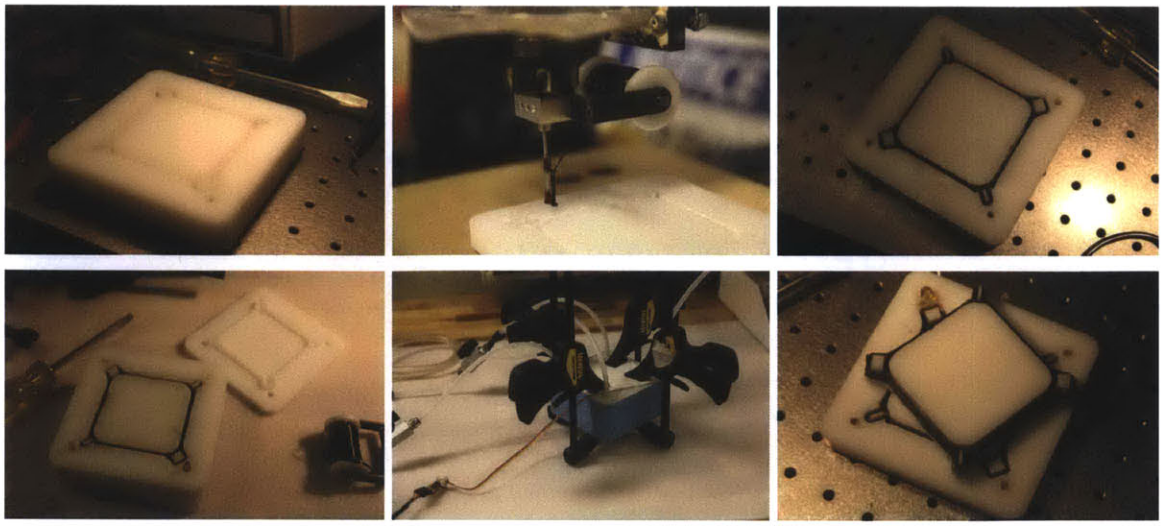


Figure 4.5: A) Mold layer, ready for winding. B) Automated winding attachment in a 3 axis CNC machine. C) Mold layer, dry wound. D) With male compression layer ready to insert. E) RTM with one resin inlet and 3 vacuum outlets, E) Demolding layers of composite parts.

conventional filament winding (e.g. for pressure vessels) with fewer constraints on geometry. The chief differences between this method and that of [11] are that the parts are produced in their net-shape form, eliminating the need for any post-cure machining, and that room-temperature-cure resins can be used.

The process is outlined in Figure 4.5. First, layers of a mold are machined from an engineering plastic. We used high-density polyethylene (HDPE), which machines well and provides the surface finish necessary for the molded surfaces. The first layer has a channel in the shape of our desired composite part, but considerably deeper than the desired part thickness, to facilitate easy winding and sufficient compaction.

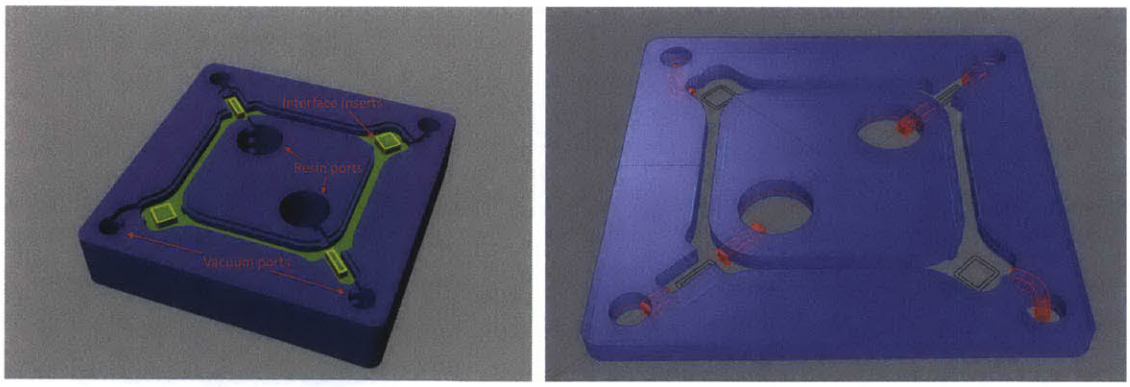


Figure 4.6: A) Alternative instantiation of resin paths. B) Showing manifolds and ports for resin transfer.

Dry fiber is wound by hand or with a three-axis computer-controlled machine into this channel. Figure 4.5 shows a custom attachment for a Shopbot Desktop holding a spool of 12k carbon fiber tow. The tow is guided into the channel through silicon carbide loops mounted on a precision rod. The programmed path can be designed to loop around features requiring more reinforcement,

filling in fiber where it is needed in the part.

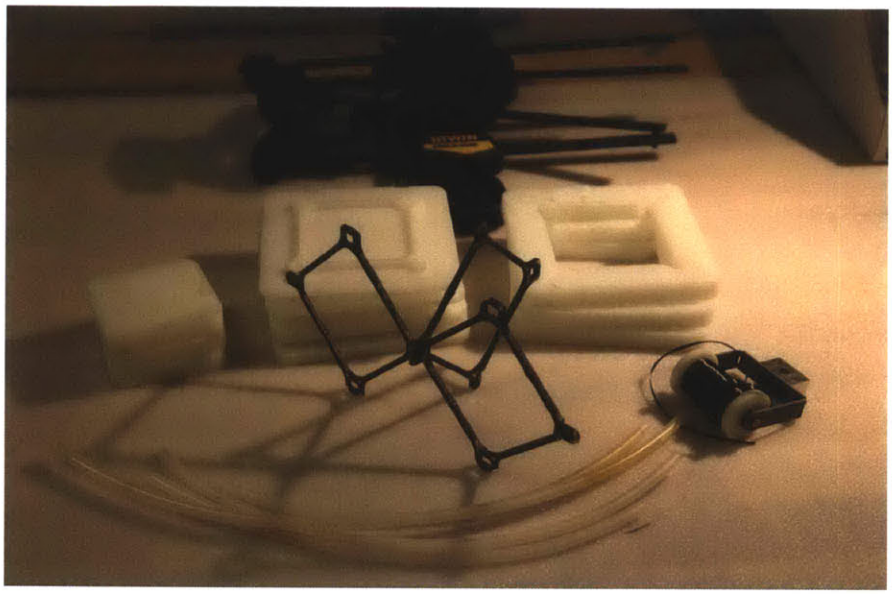


Figure 4.7: All parts of the process (back to front): clamps, three types of mold pieces, three identical sample composite parts from the mold assembled, automated winding attachment with 100 feet of 12k carbon fiber tow to the spool, and tubing used for resin transfer.

When an appropriate fiber fill fraction has been reached, the fiber is compacted using the next mold layer. The next layer has a feature in the shape of the part, projecting from the bottom surface. The height of this feature is less than the channel depth by precisely the desired thickness of the part. This layer also seals the fiber, providing vacuum integrity for the resin transfer process.

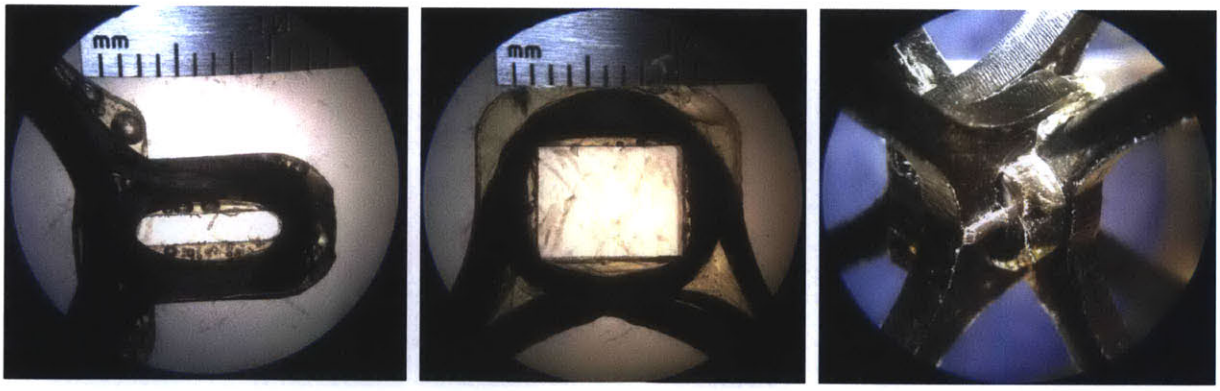


Figure 4.8: Microscope photographs of filament-would part nodes and a pinned joint.

Besides providing features to wind and compact the fiber, each mold layer also contains four holes and small channels connecting the mold cavity to these holes (shown in Figure 4.6). With many layers, these features make up the manifolds for the RTM process, connecting each part cavity to the vacuum and resin reservoirs. The top mold layer accepts conventional plumbing fixtures and the bottom layer is blocked off. When a sufficient number of layers have been added, resin is pulled through all the cavities with a vacuum pump in an RTM process. In the mold in

Figure 4.7, we used a vacuum pressure of 7-10 psi to pull in West Systems 105 epoxy system, a readily available, high-strength, room-temperature-cure resin.

This parallel production technique allows a large number of parts to be produced in a single cure, the longest and most costly step in most composites production. As resin flows much faster through vacant channels than those filled with dry fiber, a great number of parallel molds can be ganged together without affecting time required for resin infusion. This technique also provides fine-tuned control over the infusion, as we can add as many vacuum manifolds as desired and use external valves to control their relative pulls.

A critical feature of this technique is that each mold piece contacts the composite part along a surface that parts along a single vector. This guarantees that there are no internal corners on any mold piece which would cause problems during demolding. Because of this, complex geometry can be created by molds that can be reused for many runs with little degradation.

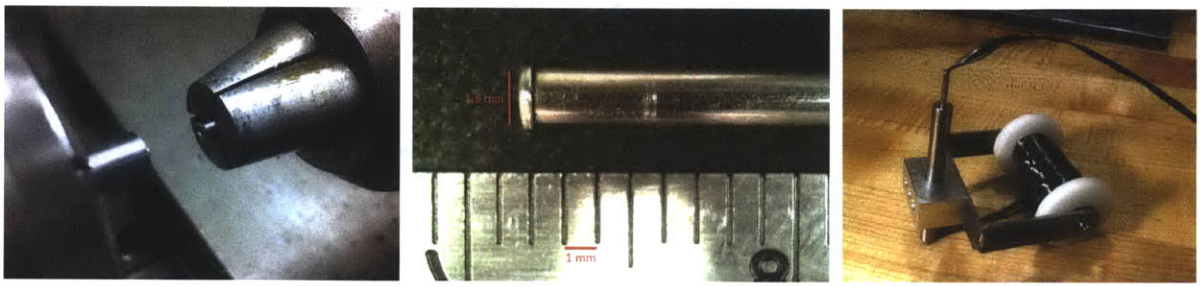


Figure 4.9: Forming the stainess steel tube end, a scale comparison, and the tube in the winder

Figure 4.8 shows microscope images of the joints of these composite digital material parts, where loops of the composite fiber are linked. Due to the fiber alignment, these parts exhibit significantly better stiffness-to-weight performance than the milled, quasi-isotropic phenolic parts.

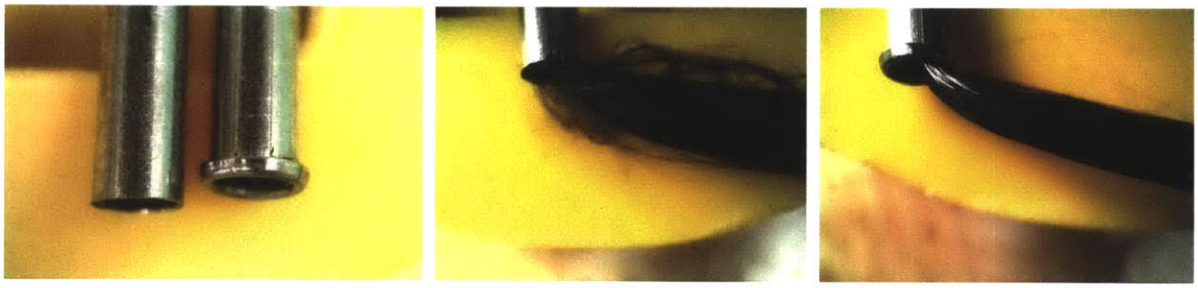


Figure 4.10: Without micro-tube-end forming, the winder slices and tangles the tow. With the lip, the tow can pass smoothly.

To scale this production method, it was necessary to optimize the automated fiber winding process. The winding attachment described above could not place fiber into channels as slim as desired for our digital material parts. Because of this, we made several iterations on the tip of the attachment, finally settling on using a section of 1/16" diameter stainless steel tubing with .005" wall (shown in Figure 4.9). The cross section of this tip was on the order of the cross section of the tow to be wound, much smaller than the original tip with the silicon carbide guides.

With such thin-walled tubing, however, the edges would damage the tow as it was pulled through the tube (see Figure 4.10. To fix this, we fabricated a set of three progressive micro-tube-

end-forming tools. Using these forming dies from the motion system of a lathe, we were able to reliably turn out and fold back the rib of the tubing, significantly softening the edge, and allowing the tow to pass unscathed.

The part production techniques detailed above provide means of producing both parts for testing assemblies, as well as high performance parts for performance critical applications. In the next chapter, we describe custom design tools for digitally specifying digital material designs with high level descriptions.

Chapter 5

Design tools

Specifying lattices subject to geometric and performance constraints is a peculiar task not well addressed by general CAD tools. Thus, to efficiently generate digital models of lattices filling complex geometry, we built a set of specialized, custom tools. As described in chapter 2, if we can use such tools to describe the lattice based on high level constraints and describe a part as a stiffness matrix, the simulation pipeline becomes reasonably automated.

5.1 Describing lattices

The first ingredient in a digital material design tool is an appropriate way to easily specify the pattern of a lattice. Borrowing notions from crystallography, a lattice is simply a cell, usually a simple cubic prism, that tiles space under the three unit translations. To define a `Lattice` object, then, we simply specify how parts are placed into this cell. In Figure 5.1, we show cells with parts placed for a lattice of vertex connected octahedra (*cuboct*) and one of bi-truncated cubes (*Kelvin*). These arrangements of parts can be tiled with spatial transformations to fill space with the lattices, as shown in Figure 5.2.

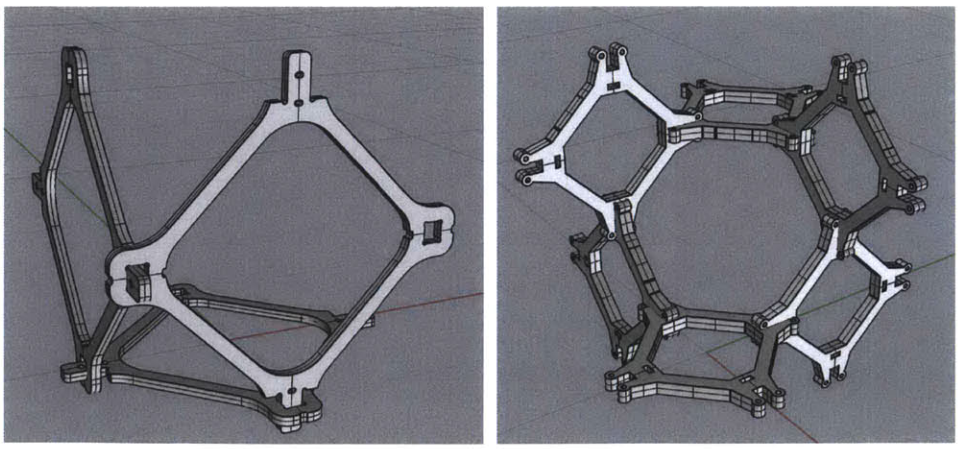


Figure 5.1: A) Cubuct lattice cell with square parts placed. B) Kelvin cell with parts placed.

We make two concessions from this very simple lattice definition. The first, illustrated by both lattices, is the need for edge cases. If we were to build a structure from these cells, we would likely

want to place special edge cells to close off the structure at its boundaries. Based on this desire, we add to our `Lattice` object a cell definition for boundaries along one, two, and three spatial directions. These correspond to the faces, edges, and vertices of the lattice envelope. This addition also allows us to specify that particular part types should be placed at these boundaries, a useful option for carrying surface loads and skinning the structure.

The need for the second concession is shown by the Cuboct lattice, where we alternate the orientation of part overlap for consecutive cells in each spatial direction. To accommodate this, we add to the cell definition an awareness of cell index number in each spatial direction, so we can use this information in part layout (for instance, by offsetting by the part thickness times the cell index modulo two).

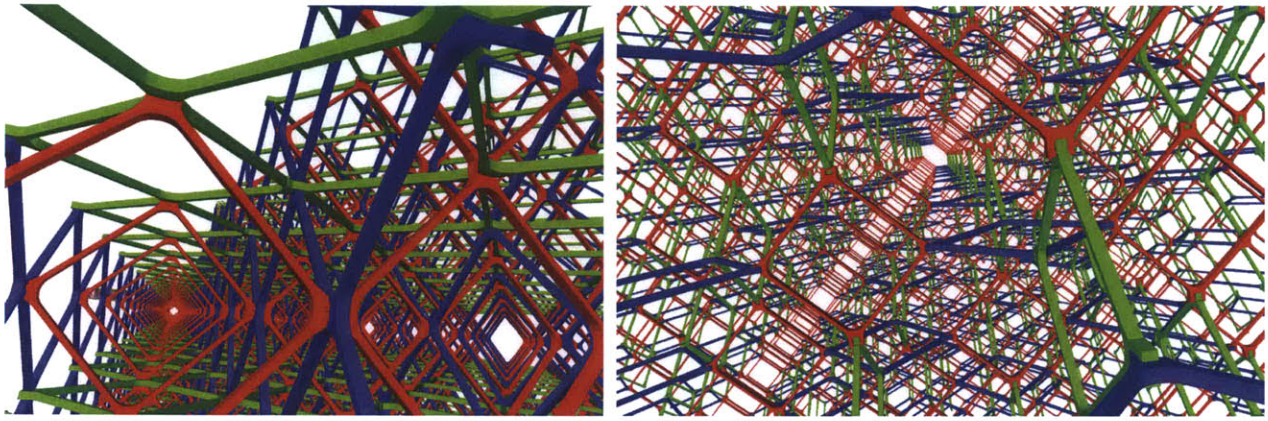


Figure 5.2: Two lattice types: Cuboct (left) and Kelvin (right)

Using the stiffness matrices inherited from the parts, transformed by their orientations within their cell, we can assemble the global stiffness matrix cell-by-cell, consolidating nodes lying on cell boundaries. Once the lattice is specified in terms of cells, this assembly process is efficient and easy to implement. The left image in Figure 5.3 shows an assembly of the square parts into a 3x3x3 cuboct brick (one of tests used in chapter 6), the middle shows loads and constraints, and the right shows the deformed shape.

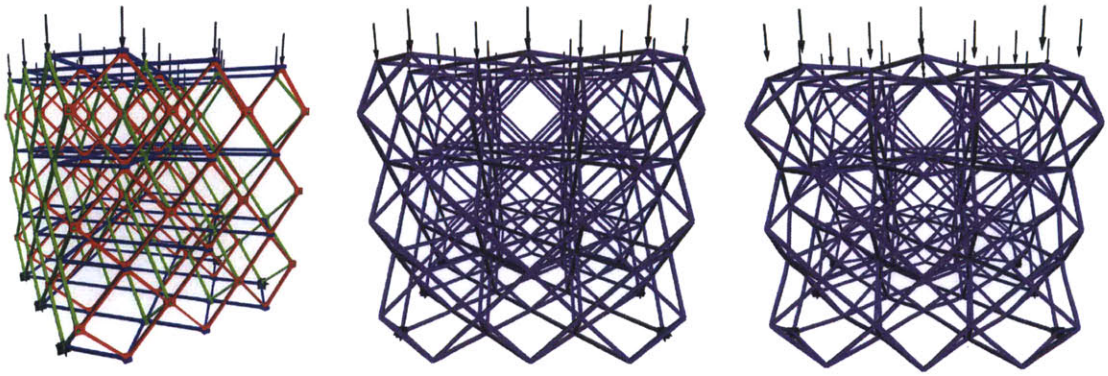


Figure 5.3: A) Parts placed in 3x3x3 test volume, B) Loads and constraints, C) The deformed shape.

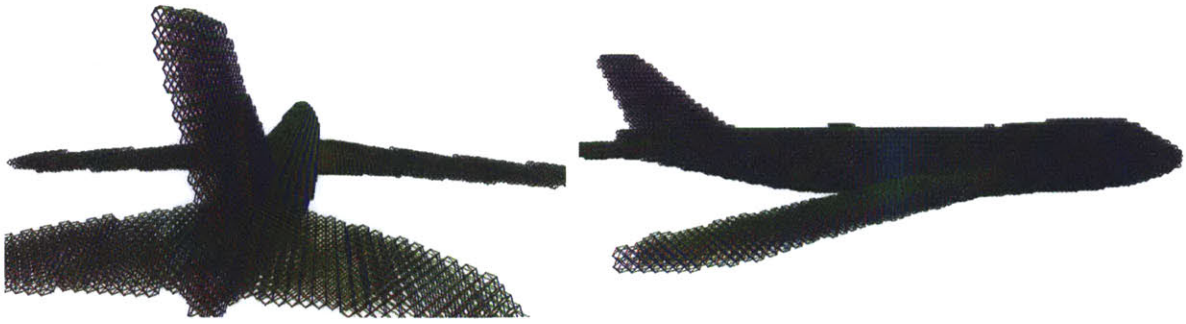


Figure 5.4: Filling a jumbo jet with a lattice.

5.2 Filling

We can leverage this cell-based description to create lattices filling an envelope of our own design. For example, in Figure 5.4, we show a lattice propagated over the shape of a jumbo jet. Starting from an STL file (a triangle mesh), we evaluated the distance function to the surface to determine which cells were contained. To do this evaluation, we used the open-source geometry library Geode [33]. Once we have the contained cells, we can test for the boundary cells and apply the boundary rules supplied by the lattice definition to close the edges, skin the fuselage, or apply greater load-carrying capacity to the wing surfaces.

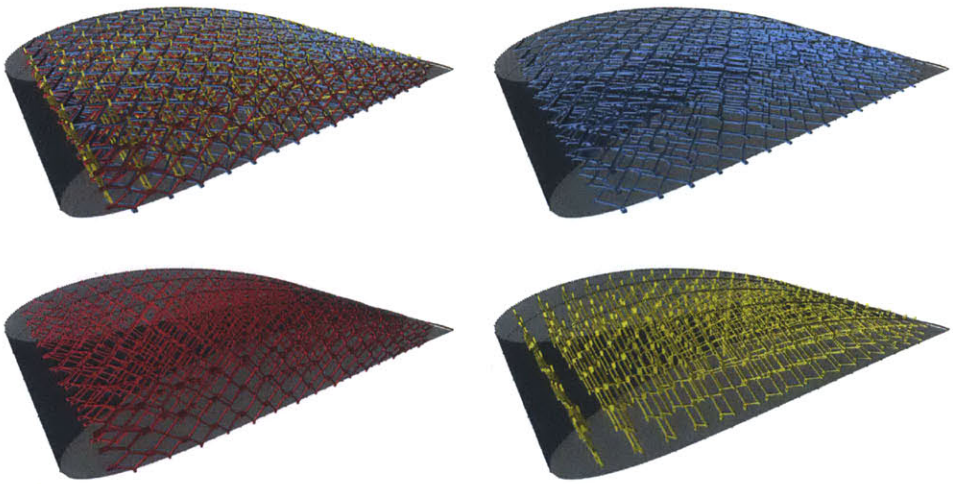


Figure 5.5: Warping parts to fill a NACA profile

5.3 Warping

We can also use the cell description to work backwards from envelope geometry to calculate part shapes that conform to surfaces. As an example, Figure 5.5 shows how an arbitrary NACA profile [30] can be used to generate parts that assemble to exactly match the boundary. This is

different from the filling operation above, because we avoid stair-stepping phenomena and can more precisely create the curved surfaces required by applications like building wings.

To do this, we use a parameterization of the top and bottom surfaces provided by the NACA code. These can be combined to parameterize the volume to be filled. We take the cells to be equally spaced prisms in the parameter space, and apply the mapping to the geometry of the parts.

The trade-off for creating such complex surfaces is a huge increase in the number of unique parts required. An interesting direction for future work is write an optimization routine that would allow a designer to explore the trade-off curve between the number of unique pieces and the fidelity of the surface reproduction.

Chapter 6

Testing

With the parts produced and the assembly design tools working, we now compare the results of structural testing with the behavior predicted by the simulations. This testing will take place on two scales: that of individual parts and that of large assemblies. The goals of testing parts are to down-select the models of part behavior and to examine the effect of the joints between parts. The goals of the assembly testing were to provide load conditions representative of typical engineering applications, and to compare the measured results with simulation results. For this, we chose two static tests to span the types of loading typically seen in applications. The first test was a simple compression test, capable of determining an effective Young's modulus for the homogenous material model of the lattice. The second test was a three-point bend, a test used to characterize bending stiffness.

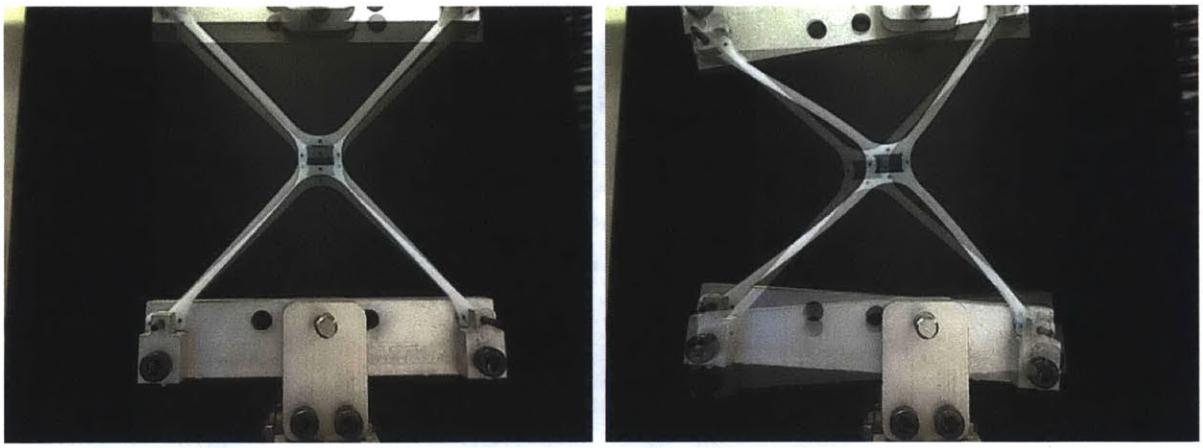


Figure 6.1: First prototype of instron fixture to verify piece level model.

6.1 Infrastructure

6.1.1 Fixturing

For the part tests, we used the Instron 4411 material characterization machine with 500N load cell. The first prototype testing fixture for parts was designed to measure stiffness in multiple degrees

of freedom of the part. Figure 6.1 shows this fixture, with two loading conditions composited. The left configuration is designed to impose a pure stretch of the part while allowing rotation at the nodes about the transverse axis. Careful inspection of the image markers shows that this pure stretch does transmit a torque about this constrained axis. The configuration on the right applies differential force to left and right nodes, imposing a bend on the piece.

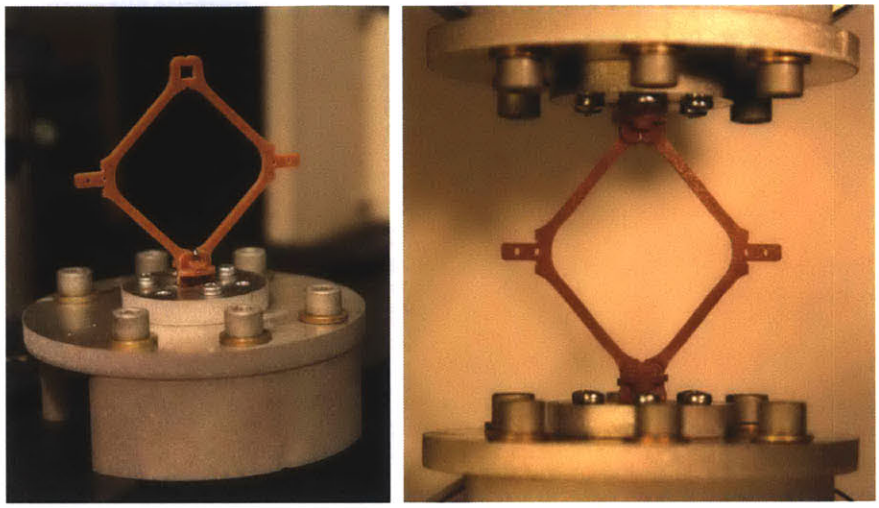


Figure 6.2: Measuring part stiffnesses, tensile test.

We determined this fixture created more confounding variables than were warranted by its generality. The method we settled on instead uses the same method to fixture the part as is used to join the parts together in the digital material assembly. To do this, we waterjet-cut aluminum pads with phenolic joint pieces glued in. These phenolic joints attach to the structure like another digital material piece, and are pinned in place (see Figure 6.2). The pads required many M3 holes to be drilled and tapped, so to facilitate this we used a Tapmatic tapping head and a Bridgeport mill with conversational programming. To test the parts, these aluminum pads are bolted to a standard fixture in the load string, which in turn is held by wedge-action specimen holding jaws.

For the assembly testing, we used an Instron 5985 with a 250 kN load cell. To fixture the test samples to the instron, we used the same aluminum pads as above to interface with the lattice structure. The aluminum pads are bolted to large end plates, which in turn are fixtured to the Instron. To make the aluminum plates, we used the OMAX waterjet in combination with a Shopbot Desktop CNC mill for pocketed features.

6.1.2 Material Properties

All of the structural simulations used in this thesis require material data as input. We determined this empirically to avoid discrepancies between our materials and published values. The density and elastic modulus of the phenolic composite were determined empirically (Figure 6.3) to be roughly 1320 kg/m^3 and 3.2 GPa, respectively.

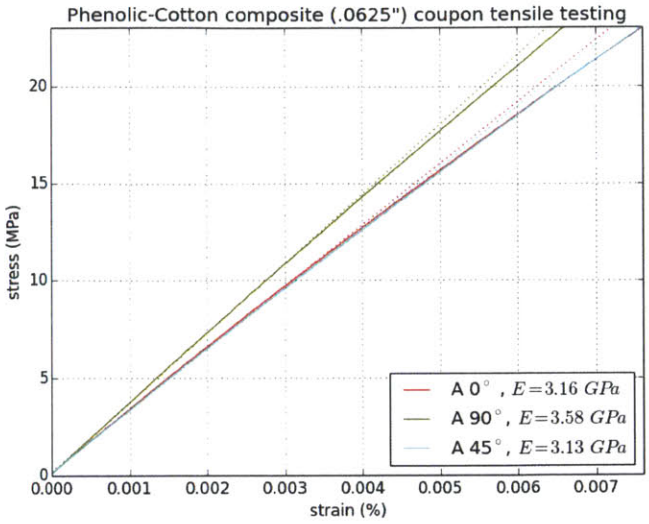
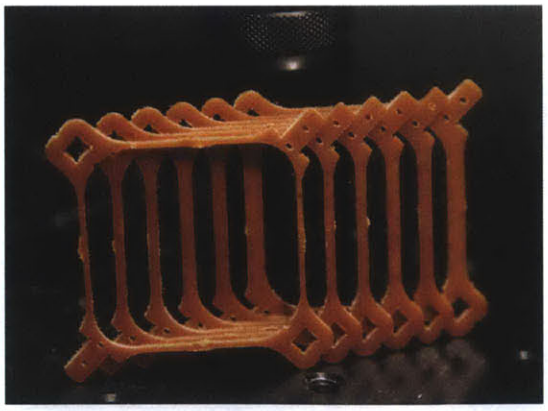


Figure 6.3: Determining elastic modulus and isotropicity of phenolic coupon

6.2 Testing Parts

6.2.1 Slender array

With testing infrastructure in place, we set up several part tests. First, to generate a dataset for comparing models of behavior, we machined an array of digital material parts, sweeping over the slenderness ratio, s, of the tendons. This ratio is defined as the tendon length ℓ over the tendon width w. For this test, we used a stock with a constant thickness t of .065", a constant tendon length $\ell = (\sqrt{2})$" (corresponding to 2" pitch), and let w range so s takes values from 10 to 25. The image at left in Figure 6.4 shows an array of six parts with varying slenderness.



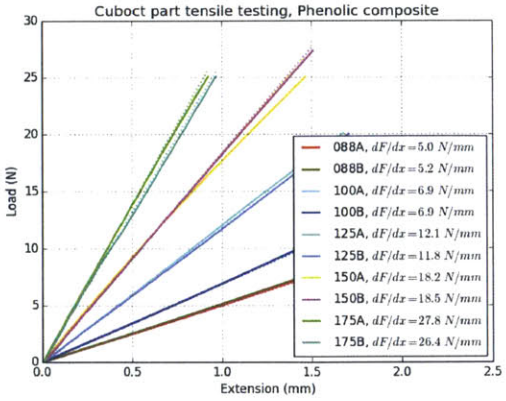


Figure 6.4: Slenderness array testing.

Each part was subjected to a tension test on two opposing nodes, while the other nodes remained free, as shown in Figure 6.2. We ramped the extension at .25 mm/min to a maximum of $25N$. The stiffness results are shown in Figure 6.4 at right.

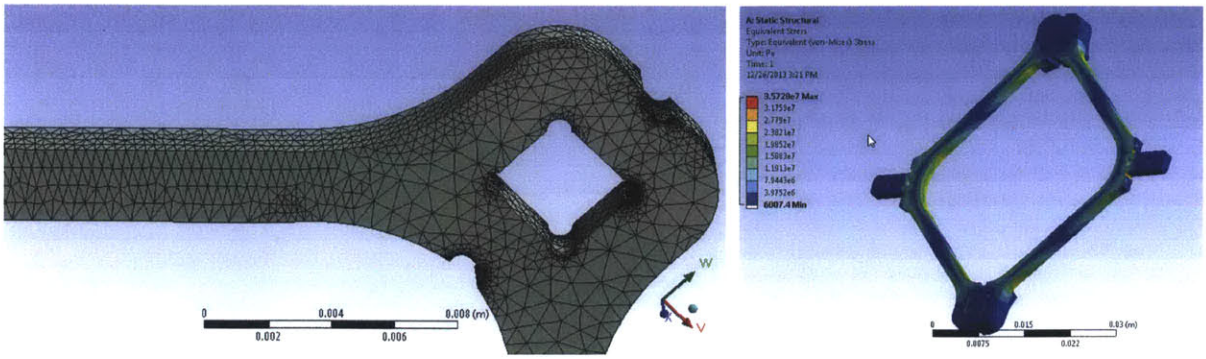


Figure 6.5: Simulation mesh and stress.

To generate simulated predictions for this test, we examined three separate approaches. First, we used ANSYS, a commercial finite element simulation package, to implement the test. For fast design iteration, we represented the part design in a volumetric form (using [24]), but ANSYS requires a surface representation. To do translate between the two, we generated part outlines using edge detection, and cleaned the result using a pass of the Ramer-Douglas-Peucker algorithm. Then we create an extrusion of this outline using a CAD program (in this case, Rhinoceros), and export a STP file to be readable by ANSYS. Once in the software, we can create a mesh of the part (shown in Figure 6.5), apply constraints and loads to faces, and extract displacements of the nodes based on the solutions returned. Packages like ANSYS Workbench attempt to shield users from intricacies of finite element analysis, automatically selecting the "best" element type for a given simulation. For these tests, we used a ten-node tetrahedral element because it accommodates quadratic displacements and does a good job capturing irregular geometry [6].

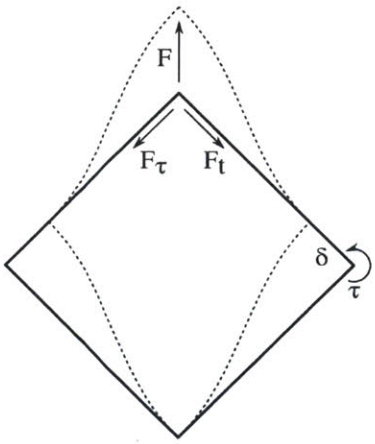


Figure 6.6: Deriving the two-dimensional model

The second approach is a simple, pen-and-paper two-dimensional model of the part using beam deflections. Modeling the part as a rhombus with rigid joints, with one node fixed and the opposite node pulled in tension by a force F, as shown in Figure 6.6. We let δ be the displacement of one of the free, unloaded nodes, measured in the direction towards the loaded node. Assuming δ is small, a symmetry argument shows that the node is displaced in this direction without rotating, and that the loaded node does not rotate. Thus, we can simplify the problem to a one-dimensional

beam problem with one clamped and one end with constrained angle. We write

$$F_t = \frac{12\delta EI_z}{\ell^3} \tag{6.1}$$

where F_t is the force along the tendon joining this node to the loaded node. Now, to satisfy our assumptions of no rotations, there must be induced moments about our nodes, given by

$$\tau = \frac{6EI\delta}{\ell^2} \tag{6.2}$$

each. To remain in equilibrium, this moment must be balanced by a force $F_\tau = \frac{12EI\delta}{\ell^3}$ at the loaded node, perpendicular to the tendon. Taking the components of these forces in the direction of our global force F, and adding the contributions of both sides of the rhombus, we have

$$F = 2(F_t^y + F_\tau^y) = \frac{24\sqrt{2}\delta EI_z}{\ell^3} \tag{6.3}$$

If Δ is the displacement of the loaded node, we have $\Delta = \sqrt{2}\delta$, and so

$$k = \frac{F}{\Delta} = \frac{24EI_z}{\ell^3} \tag{6.4}$$

As we can readily calculate I_z from the part parameters w and t, this formula gives us a prediction of the stiffness measured by our Instron test across the slenderness value array.

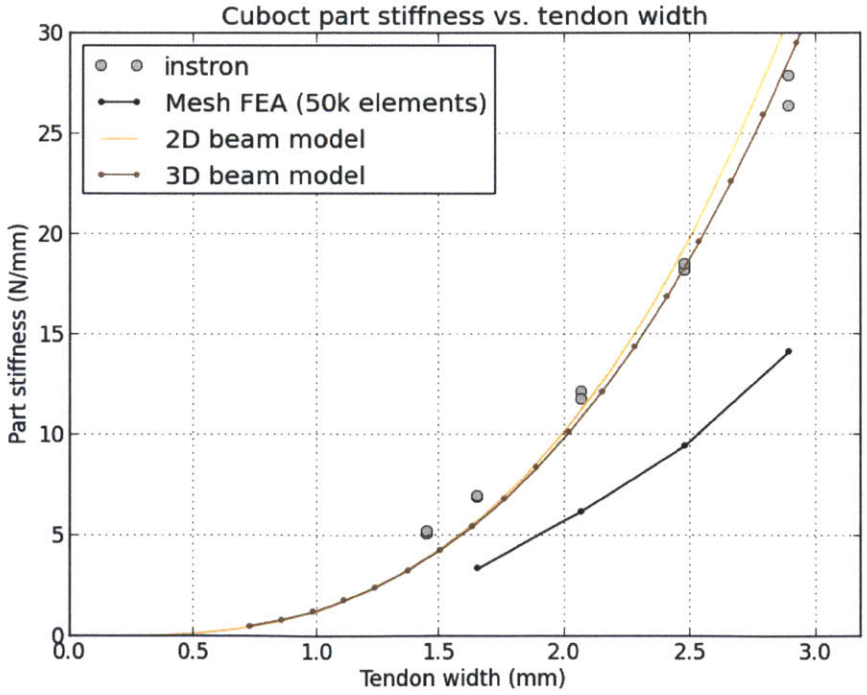


Figure 6.7: Part testing results, all methods.

Finally, as the last approach, we implement the full three-dimensional stiffness matrices of the beams and solve the corresponding system to minimize strain energy. For this test, we used the

Frame3dd library [18]. This library implements geometric stiffness effects with a Newton-Raphson iterative method. Geometric stiffness captures the effect of axial loading on deformation in the cross-sectional plane of the beam elements [17]. We chose this library over our own linear elastic solver in order to see how good the simple two-dimensional model performs when compared with an independently verified simulation which includes many more solid mechanical phenomena.

We plot the results in Figure 6.7. Very evidently, both beam element methods perform considerably better than the more computationally-intensive meshed FEA. To give the meshed FEA a fair trial, we experimented with several possible sources of errors. First we performed several refinements in the simulation mesh and recomputed the solution. Extrapolating the trend of such a solution sequence is a commonly accepted method to eliminate the effects of the mesh approximation. While the more finely meshed solutions were slightly closer to the measured results, the apparently convergent values still showed significant discrepancies.

We also experimented with ways of enforcing the boundary conditions in the meshed simulation, another common cause of simulation errors. Despite building elaborate geometry that more closely mimicked the physical test, transmitting loads and constraints to the part through computed contact patches, the results did not significantly change.

The two beam models agree remarkably well, especially in the narrow tendon limit. When the tendon becomes very wide, as at the right side of the graph, there is an out-of-plane deformation not captured by the two-dimensional model. This explains the divergence of the curves and the better agreement of the three-dimensional model with test results for wide tendons.

There is some question for both beam models about how to determine ℓ from the part geometry. Clearly, the beam length is not simply the distance between node centers, as the joinery and fillets shorten the effective beam length. For the purposes of this test, we took the length to be measured between points on fillets where the width was 25% greater than the thinnest tendon width. In practice, this step can serve as a model calibration, where we tune our beam simulations using measurements of a part before simulating large assemblies of parts.

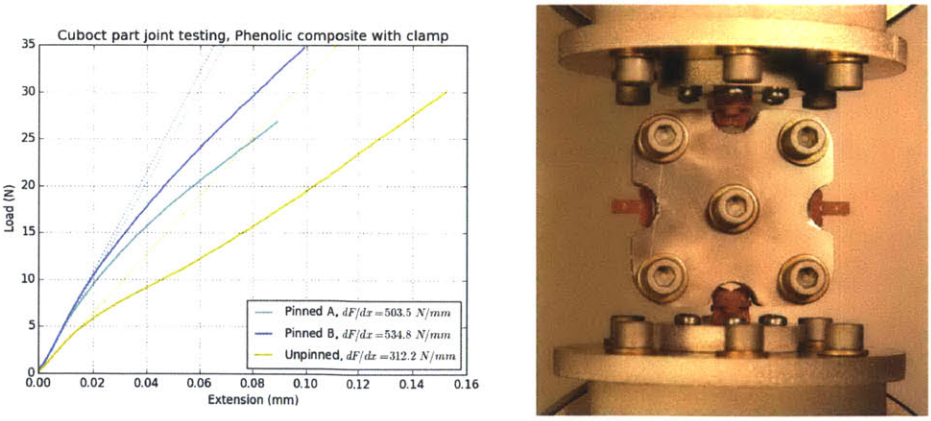


Figure 6.8: Backing out the joint effects.

6.2.2 Joints

In these simulations, we model elastically-joined nodes as rigid connections. To validate this assumption, we next investigated the discrepancies in behavior between the two. To this end,

we subjected a part from each of the slenderness values above to a clamped test, where the deformation of the part is constrained by an additional aluminum fixture, shown in Figure 6.8 at right. Two parts with pinned joints and a third with unpinned joints were tested. This test measured the effective stiffnesses of the joints for small deformations (roughly 500 N/mm pinned, and 300 N/mm unpinned).

These results show that the pin element is crucial to maintaining a robust joint, nearly doubling the stiffness in extension. Second, when appropriately designed and pinned, these phenolic joints between parts can exhibit stiffness within a factor of three of that of a monolithic piece. Given that the joint regions of digital material parts are generally over-engineered, this suggests that modeling the joints as rigid is a valid assumption for most relevant loading conditions.

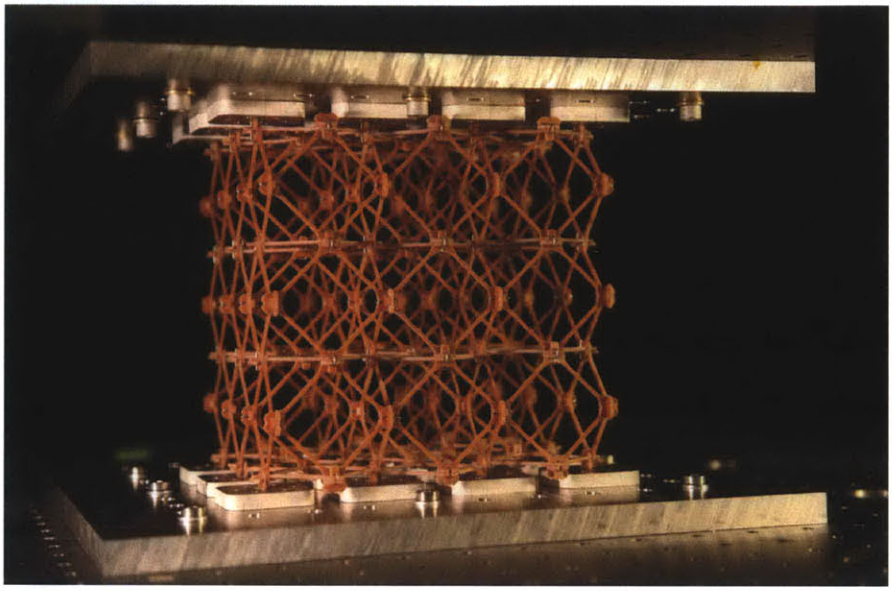


Figure 6.9: 3x3x3 phenolic brick in Instron, ready for compression testing

6.3 Assembly testing: 3x3x3 Brick compression

For the compression testing, we assembled 2x2x2 and 3x3x3 bricks of phenolic parts. For both this sample and the one described in section 6.4, hitting tolerances and streamlining the part fabrication and assembly process were both essential to finishing the tests in a timely manner. For this, the milling fixtures and pin press described in section 6.1 were both critical.

With the 3x3x3 brick fixtured in the instron (Figure 6.9), we ramped the load until a break event occurred at one of the interior tendons at roughly 800 N. Based on the dimensions of the brick, we use the load and displacement data to estimate an elastic modulus of 2.532 MPa.

To implement this test in the simulation tools, we needed to describe the geometry and materials. These parts were milled with square tendon cross section of .065" and a lattice pitch of 2". We specify the cuboct lattice cell unit with these dimensions, and propagate it across the 3x3x3 volume. As described in chapter 5, we apply the edge cell definitions in order to close off the lattice. The nodes on the bottom face are constrained, and a ramped force is applied to the nodes on the top face. After solving the system, the displacements of the loaded nodes are collected and

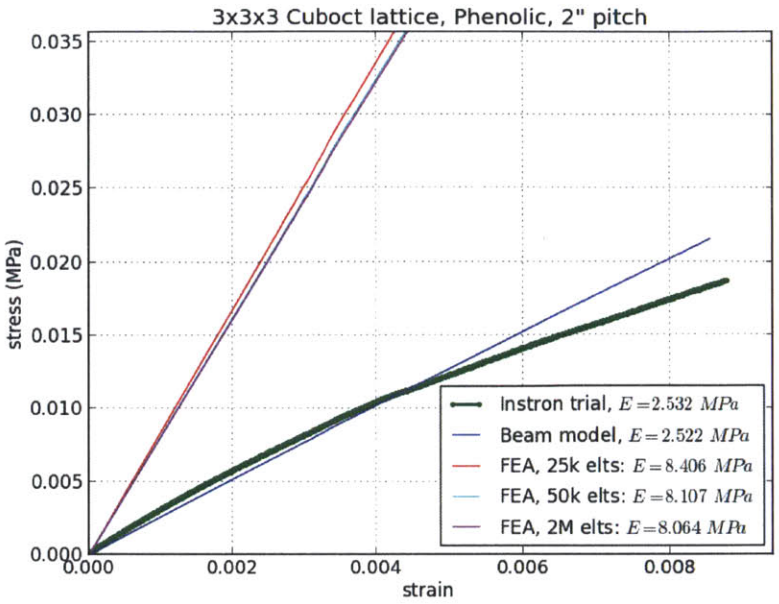


Figure 6.10: Compression test results results

averaged. Figure 6.10 plots the stress and strain for the empirical test, as well as the simulated tests.

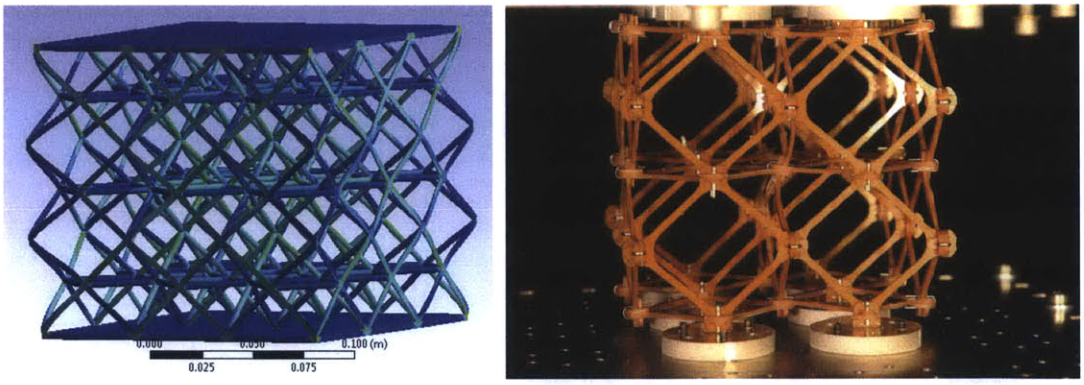


Figure 6.11: a) Brick simulation in ANSYS, b) Preliminary 2x2x2 test, showing coordinated buckling.

As with the part testing, we also implemented this compression test as a meshed FEA simulation, using ANSYS Workbench. Elements are again 10-node tetrahedron type, according to the program's automatic settings. For this simulation, we added the top and bottom fixture plates, shown in Figure 6.11. Several small displacements (1mm) were specified on the top plate, and the reaction forces were measured. The results are shown in Figure 6.10 for several mesh resolutions. The simulated elastic modulus stabilizes quickly, showing very little change between meshes with $50k$ and $2M$ elements.

Again, the beam model is considerably more effective than meshed FEA, despite requiring considerably fewer calculations. The empirically measured modulus and that of the beam model show agreement to better than 1%. Neither test, however, is sophisticated enough to capture

the reversible but nonlinear stress-strain curve measured in the testing. This is a geometric phenomenon, rather than a material phenomenon (e.g. plasticity), resulting from coordinated buckling of nodes (as shown at the right in Figure 6.11). To fully capture this, more sophisticated large displacement methods would be necessary.

We should note that these tests can be used to work out the power law scaling relationship often used in cellular solids literature (c.f. [20]). Let $E_{lattice}$ and $\rho_{lattice}$ be the elastic modulus (as determined above) and density of the lattice, respectively. The power law scaling posits that $\frac{E_{lattice}}{E_{phenolic}} \propto \left(\frac{\rho_{lattice}}{\rho_{phenolic}}\right)^s$. Using the results above, we have

$$s = \frac{\log E_{lattice} - \log E_{phenolic}}{\log \rho_{lattice} - \log \rho_{phenolic}} = 1.755 \tag{6.5}$$

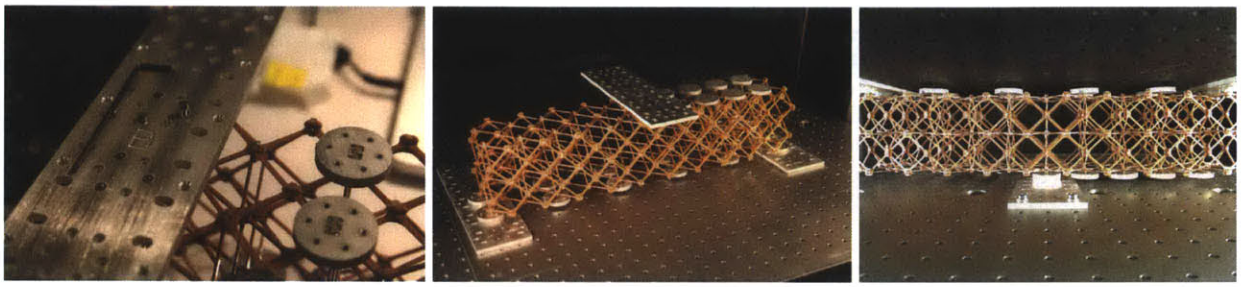


Figure 6.12: Testing fixture for three point bend.

6.4 Assembly testing: Three point bending

For the three-point bend test, we constructed a 10x2x2 structure and mounted the aluminum pads to the top and bottom faces. The milled fixture plates allowed us to select between which pads we induce a bend. This allowed multiple measurements across different span lengths from a single sample. In Figure 6.12, we show the process of mounting the sample in the Instron for a ten cell span. Again, we ramped the applied load, but stopped short of irreversible changes so that we could perform multiple tests without affecting the results.

Implementing this test in simulation was identical to the compression tests except for the boundary conditions. For this, we constrained the bottom, middle nodes (again, see Figure 6.12) and applied loads to the top, outer nodes. Again, after solving we average the displacements of the loaded nodes to determine the bending stiffness.

The empirical and simulated test results are shown in Figure 6.13. The agreement looks good, with error on the same order as the variance in the measurements.

As a point of comparison, consider applying a homogenous material model, using a beam bending equation with the overall geometry of the sample and the elastic modulus derived in the compression testing. For large spans, this works as well as our method, but for shorter spans it significantly overestimates the stiffness (by as much as 50% for the six cell span).

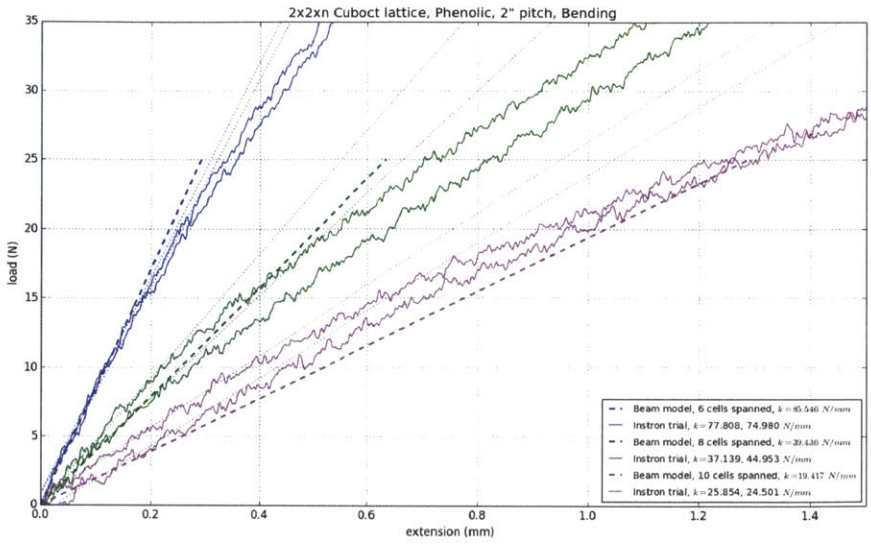


Figure 6.13: Three-point bend test results

6.5 Conclusions

These three structural tests show that using beam-based models to describe digital material parts and synthesize behavior of assemblies is an effective way to predict first order stiffness in a range of meaningful load conditions. Further, this relatively simple and inexpensive method outperformed meshed FEA with commercial tools where they were compared. These results are very encouraging, and so next we undertake an ambitious design study to leverage these advantages.

Chapter 7

Application: Vacuum Balloons

To demonstrate the utility of building and designing structures using the methods described, we designed an ambitious ultralight structure.

7.1 Concept

For this design study, we set out to build a volume from which the contained air could be evacuated while the structure resists the resulting force from atmospheric pressure. If this structure is made light enough, the mass of the displaced air (at a density of 1.225 kg/m^3) will be enough to generate lift.

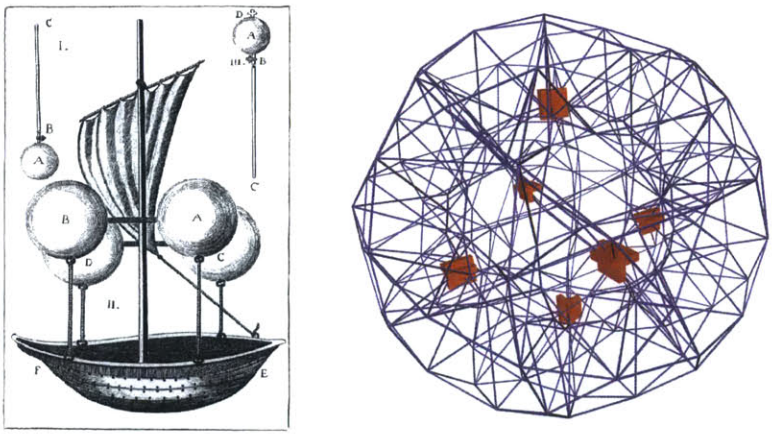


Figure 7.1: A) Francesco Lana de Terzi's flying boat concept c.1670 [46], B) Discretely assembled sphere design

This *vacuum balloon* concept has been around for quite some time, appearing recently in several articles and patents [31][4] but dating back to at least 1670 with Italian Jesuit Francesco Lana de Terzi [2], whose design is shown in Figure 7.1 at left. This airship was based around large spheres (roughly 7 m in diameter) made of thin copper foil. These spheres were reasoned to withstand atmospheric pressure based on a symmetry breaking argument, but this discounted inevitable imperfections in the manufacturing of such spheres. The vacuum airship was never

built, and arguments were constructed for its impossibility based on the assumption of using solid, homogenous materials for the spherical shells.

If instead of solid materials we use sparse lattices, the scaling looks much more plausible. In this section, we propose a design based on digital material assembly and use the hierarchical simulation tools developed in this thesis to investigate performance.

Figure 7.2: Constructing the spherical lattice by truncation of polyhedra.

The lattice underlying the design can be constructed starting from the Platonic icosahedron, shown in Figure 7.2. We first truncate it to get the familiar "soccer ball" polyhedron. Then, we truncate again, effectively replacing each vertex by a triangle between the midpoints of the incident edges. This polyhedron is composed of triangles, pentagons, and hexagons, where each vertex has degree four. Taking just just the edge network of this polyhedron, we create two concentric copies as the inner and outer boundary of our spherical shell. To connect these boundaries we introduce new edges as follows: First, give each existing edge a direction so that all triangles have a counterclockwise orientation with respect to the outward normal. Next, draw a new edge between the first vertex of an edge on the outer surface and the second vertex of the corresponding edge on the inner surface.

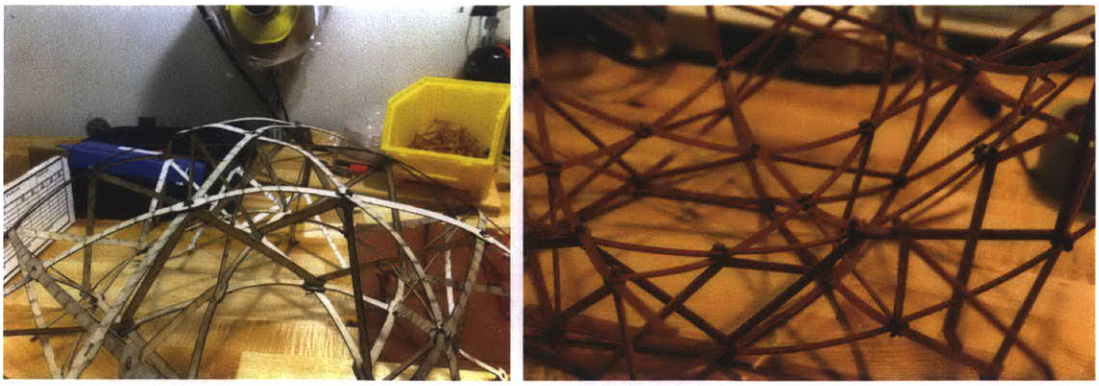


Figure 7.3: Sphere section prototypes from plywood and phenolic.

In the resulting framework, there are 540 members, 180 joints, and each vertex has degree six. Thus, the structure is not isostatic, but it meets the necessary conditions for kinematic determinacy [14]. Each triangular wedge is similar to Kenneth Snelson's "octet truss" [13], [43], a small tensegrity structure which is *super-stable*, that is, one made stiffer by prestress.

This framework can be realized from planar components in several ways. Two prototypes in laser-cut plywood and milled phenolic composite are shown in Figure 7.3. These prototypes are particularly convenient for fabrication, as the pieces are one-dimensional, and so can be produced efficiently. The pieces join in a manner similar to the other lattice structures in this thesis, with slots, tabs, and pins.

Figure 7.4: First vacuum balloon part prototyping

To obtain the necessary stiffness in the actual balloon design, we used the molding techniques described in section 4.4. We split the carbon fiber winding into two phases, inserting a lightweight core in between. This effectively makes custom-shaped, fiber-aligned sandwich panels (see Figure 7.4), significantly reducing the mass necessary to achieve a given bending stiffness. For the core material, we selected end-grain balsa for its relatively low cost, availability in a large selection of thicknesses, and ease of laser cutting. Balsa is well known for its low density, and the graph in Figure 7.5 shows the difference in compressive modulus between end-grain and side-grain. In the image at the right, we measure the layer thicknesses of a part fabricated in this way. Using a 2.5 mm balsa layer, we attempted to add .5 mm carbon fiber face sheets. The measurements show that this process has roughly 50 μm precision in layer thickness.

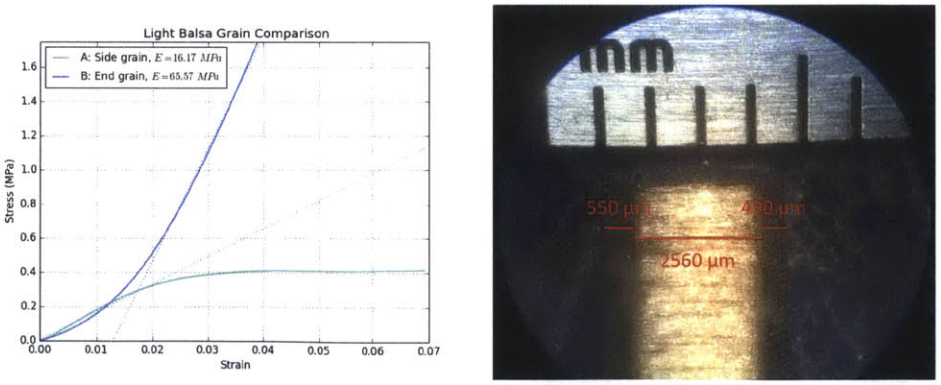


Figure 7.5: A) Stress strain plot for grain directions of balsa., B) Cross section of molded sandwich part.

7.2 Skin Design

Besides the lattice structure, we also need a method to skin the lattice and maintain vacuum integrity under load. As an ambitious first attempt, we explored making reversibly assembled skin elements using a custom-built extrusion machine. Using polypropylene pellets, this machine used an auger to melt and push the molten polymer through custom electric-spark machined die with a "ziploc" profile. These extrusions would be heat sealed onto laser-cut panels, allowing the panels to be reversibly joined to form an air-tight seal. We prototyped an extruder using a waterjet-cut aluminum housing, NEMA 23 stepper motor, nichrome wire heating element, and 300° C thermistor temperature control, shown in Figure 7.6.

Figure 7.6: A) Ziploc extrusion machine, B) Microscope photograph of interlocking ziploc elements (roughly 6mm across), C) Graphical user interface for the extruder.

Ultimately, more development was needed to consistently produce the ziploc elements, but the approach showed promise. For the scope of this thesis, we instead switched to irreversibly joined panels. The sphere design can be panelized into large triangles and smaller pentagons. With appropriate allowance, these panels can be joined with spray adhesive, applied through a stencil. The shear stress applied to this bond will be bounded by

$$\tau \leq \frac{3P_{atm}l^2}{8wd} \tag{7.1}$$

where $P_{atm} = 101\ kPa$, l is the sphere cell length, w is the width of the bond, and d is the dip of the skin furthest from a supporting frame element. To derive this equation, we assign to each structural member a kite-shaped patch of the skin surface, shown in Figure 7.7. Plugging in nominal values to this equation, the shear is very conservatively bounded by 300 Pa, well below the rated shear strengths of 300 psi of 3M products [3].

7.3 Evaluation

To evaluate the performance of this design and the fabrication process, we developed models for the mass of the parts, the bending stiffness of the parts, and the stability of the spherical shell.

7.3.1 Mass

To calculate the part masses for a sandwich construction, we use the thicknesses (th_{cfrp} and th_{balsa}) and densities (ρ_{cfrp} and ρ_{balsa}) of the constituent layer materials, along with the tendon

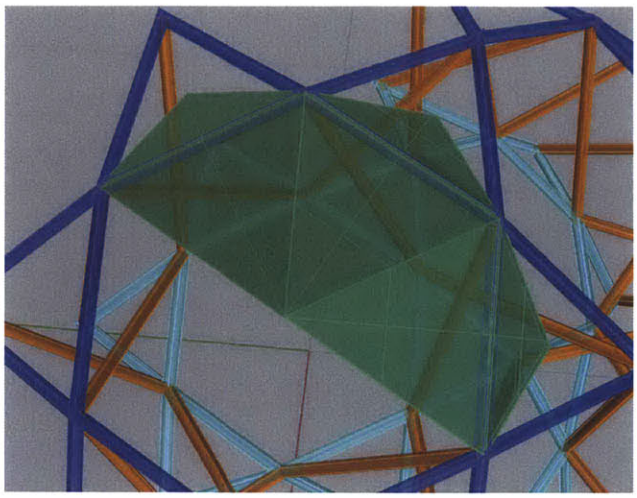


Figure 7.7: Each strut can be assigned to a kite-shaped patch for calculations.

lengths. For a spherical shell of outer radius r_1 and inner radius r_2, the cell length of the outer and inner tangential parts are $l_1 = r_1\frac{\pi}{9}$ and $l_2 = r_2\frac{\pi}{9}$. Using the law of cosines, the length of the radial type parts is $l_3 = \sqrt{r_1^2 + r_2^2 - 2r_1r_2\cos\frac{\pi}{9}}$. Then the part masses are given by

$$m_i = wl_i(2th_{cfrp}\rho_{cfrp} + \rho_{balsa}th_{balsa}) \tag{7.2}$$

To calculate the mass of the full sphere lattice, we count parts by type. Starting from the truncated icosahedron, we have 60 vertices. There is a one-to-one correspondence between these vertices and the triangular faces after the next truncation. Every edge touches one and only one triangle, so the number of edges after truncation is 180. Each tangent-type part spans two cell lengths, so there are 90 instances of each. Each radial-type part corresponds to a unique polyhedron edge, so there are 180 instances. Therefore the overall mass is

$$m_{lattice} = 180w(l_1 + l_2 + l_3)(2th_{cfrp}\rho_{cfrp} + \rho_{balsa}th_{balsa})$$

where w is the tendon width as calculated above.

For completeness, we also include calculations for thin-walled pultruded parts with square cross section of dimension a and wall thickness t. The lattice mass is then

$$m_{lattice} = 180(l_1 + l_2 + l_B)4\rho_{cfrp}(ath_{cfrp} - th_{cfrp}^2)$$

Besides the lattice structure (the mass of which is on the order of a few kilograms), there are other minor contributors to overall mass. First, the parts are joined with pins, whose mass must be accounted. Using $\frac{1}{16}$" CFRP rod, diced into pins, the overall pin mass is $5.4g$. Second, the skin, as it is in tension, is not a significant mass contributor. Measuring roughly, Dura-Lar film (a candidate skin material) weighs in at 1.67 mg/cm^2. Estimating a sphere with radius 1 m, has surface area $4\pi \approx 12.5m^2$, which works out to 17 g. All of these masses are relatively small, indicating that the mass of the lattice will dominate.

7.3.2 Bending Stiffness

To reliably predict bending stiffness, we first must produce a composite layer with predictable fiber fill fraction and adhesion to the core layer. Using the shear properties of core, we can then predict the bending stiffness of the part.

Fill fraction

To make sure to wind an appropriate amount of fiber into these molds for the desired carbon thickness and fiber fill fraction, we do some calculations. In the first test, we attempt .5 mm thickness on top and bottom. For the test part, the tendon width is 5 mm. We use a fiber fill fraction of 50%, (in the range of common values reported in composites research [1]) with 12k carbon fiber tow (having measured compacted cross sectional area of .7 mm^2 (see Figure 7.8). Comparing areas, this means we should cover the entire face area with approximately two wraps of tow.

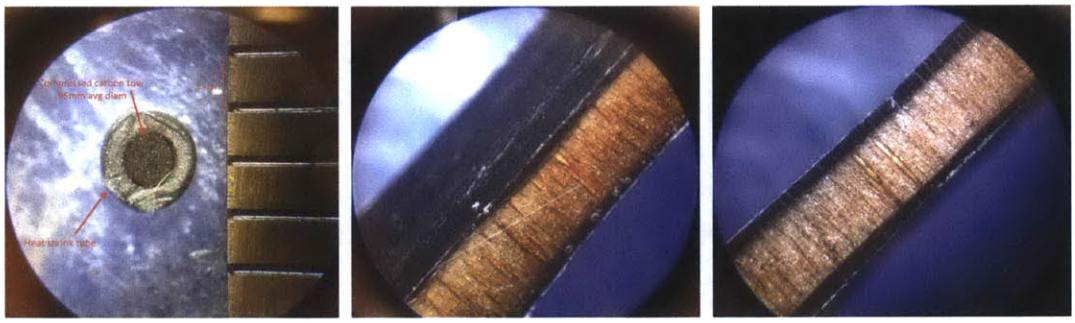


Figure 7.8: A) Measuring tow packing density for calculating windings. B) and C) Balsa-CFRP sandwich section part.

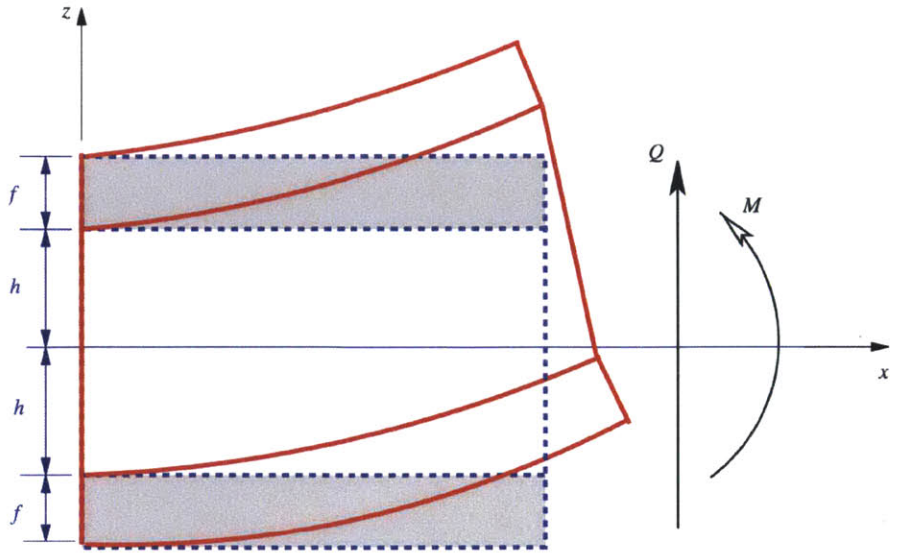


Figure 7.9: Shear mode of sandwich panel. [8]

Core shear

If we neglect shear, increasing the dimension of the core material would appear to generate a stiffer beam without limit. In reality, there is a shear mode of the core that lowers the stiffness of the sandwich beam (see Figure 7.9) as the core thickness increases. Following the analysis of [34], the deflection δ of a sandwich beam in a three point loading is given by

$$\delta/F = \frac{L^3}{48(EI)_{eq}} + \frac{L}{4(AG)_{eq}} \tag{7.3}$$

For a sandwich panel with core thickness c, face sheet thickness t, core elastic modulus E_c, face elastic modulus E_f, core shear modulus G_c, width w, we have

$$(EI)_{eq} = \frac{E_f w t^3}{6} + \frac{E_f w t d^2}{2} + \frac{E_c w c^3}{12} \tag{7.4}$$

$$(AG)_{eq} = G_c w d \tag{7.5}$$

where $d = t + c$. In practice, the second term of $(EI)_{eq}$ dominates the calculation, and we can ignore the contributions of other terms.

Measuring the shear modulus G_c accurately can be difficult due to fixturing issues. Figure 7.10 shows an experimental setup for shear modulus testing. A sheet of end grain balsa is glued to waterjet-cut aluminum, and a shearing load is applied. Based on the dimensions of the sample, the forces and displacements can be converted to shear stresses and strains, and the shear modulus can be estimated. For such a thin sample ($2.5mm$), the force required to make the measurement is enough to separate the glued interface over significant portions of the interface area. Therefore, this test should be considered only a lower bound on G_c.

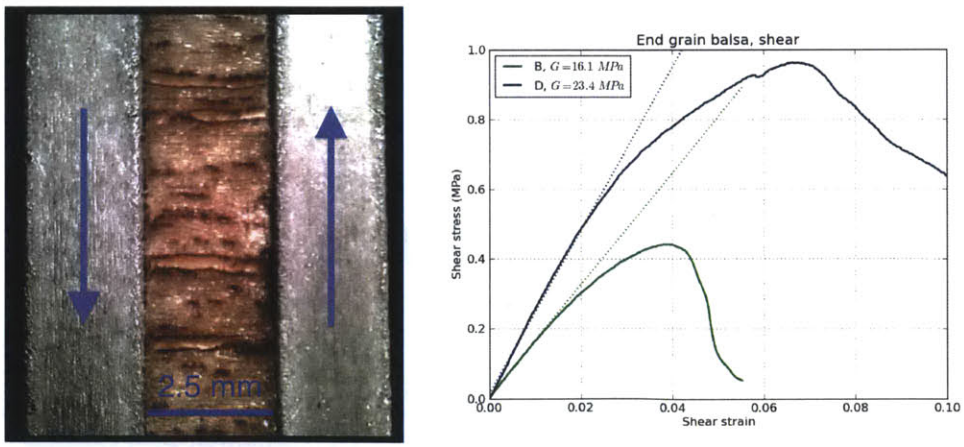


Figure 7.10: Experimental estimation of G_c for end-grain balsa.

To get a more accurate measurement of G_c, we performed a three point bend test on the prototype CFRP-balsa sandwich parts. The results are shown at left in Figure 7.11. The bending stiffness equations with the shear correction above determine the stiffness predicted for this test. Plotting the results (at left), we can back out an estimate of G_C of roughly 60 MPa.

Once we have an estimate of the bending stiffness in the direction of the sandwich stack (EI_y), we can calculate what value for beam width will give uniform bending stiffness. To guarantee that

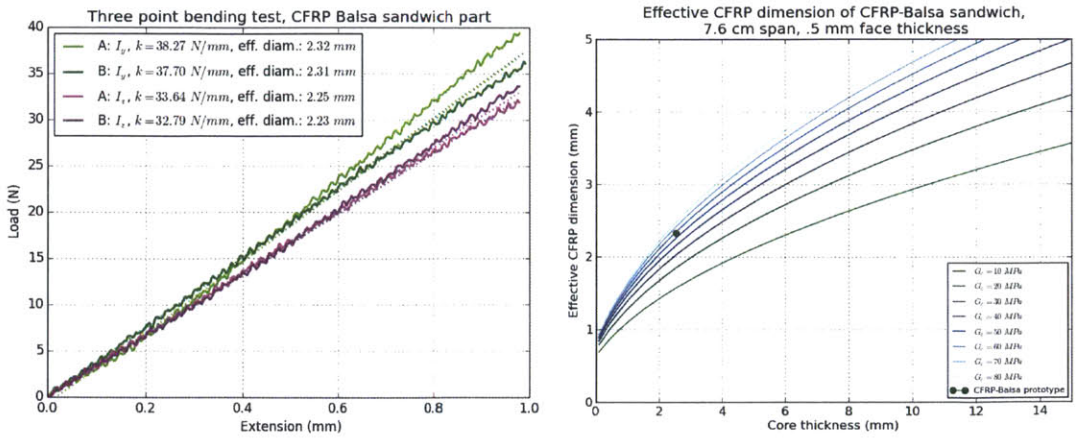


Figure 7.11: A) Three point bend test, B) Using the test to determine shear modulus G_c.

$I_y = I_z$, then we take $w = (\frac{6I_y}{th_{cfrp}})^{\frac{1}{3}}$. This value can then be used to calculate the mass savings over using a uniform beam (not a sandwich panel) to achieve the same bending stiffness. For the relevant parameter values, this mass savings factor is significant (around 2).

7.3.3 Stability Calculations

Based on the analysis above, we can calculate maximum pressure values to keep material stress at safe values. Even if we stay below these limits, however, we are still in danger of failure due to instability modes of the structure [44]. For this reason, we focus our simulation efforts on predicting instabilities in the lattice. Further analysis would find parameter choices predicting that all failure modes occur at the same critical pressure. In [36], a linear programming approach is devised to do just this, but the stiffness approximations are too coarse and conservative for our applications.

We now undertake to use the simulation workflows developed in this thesis to determine critical pressures for our spherical shell lattice. Before we even use the simulations, however, we can do some scaling analysis. Consider a spherical shell with a constant ratio between internal and external radii. Further, suppose it is composed of a lattice structure of constant tendon cross section. As we vary r (the external radius), the mass of the lattice material scales linearly. The volume of the shell varies cubically. Therefore, the relative density ρ_{rel} varies as $\frac{1}{r^2}$. Using the stability criterion for a thin spherical shell: $P_{crit} = \frac{2Et^2}{3(1-\nu^2)r^2}$. That is, we have $P_{crit} \propto E$. Now from cellular solids literature [20], we have $E \propto \rho_{rel}^s$, where the value of s depends on whether the lattice the stretch- or bend-dominated (In chapter 6, we calculated the value of this parameter for the 3x3x3 cuboct brick sample). This means $P_{crit} \propto \frac{1}{r^{2s}}$. This scaling law will serve as a sanity check on the simulation results.

Due to the discontinuity in the amount of lift available through vacuum evacuation (i.e., we cannot create a pressure differential greater than atmospheric pressure), it is difficult to apply this type of scaling argument to determine the fraction of the critical pressure necessary for lift. Instead, we will calculate this quantity through simulation directly.

Critical Pressure Simulation

To calculate the point of instability of a structure, we use modal analysis. As a simple, illustrative example, consider a tetrahedron with three nodes fixed and a compressive load applied to the fourth node. We can compute its lowest frequency mode and the corresponding frequency (shown in Figure 7.12 at left). As we increase the load applied to the fourth node, this fundamental frequency starts to drop. Theoretically, in the zero frequency limit, the mode shape is the deformation associated with structural instability (independent of inertial properties). The load at which the fundamental frequency crosses zero is the computed buckling load of the structure, after which the stiffness matrix is no longer positive definite. In the middle image of Figure 7.12, we plot of the lowest frequency versus load applied. We can see for this example the buckling load appears to be around 18 kN. To perform these calculations, we used the Frame3dd library [18].

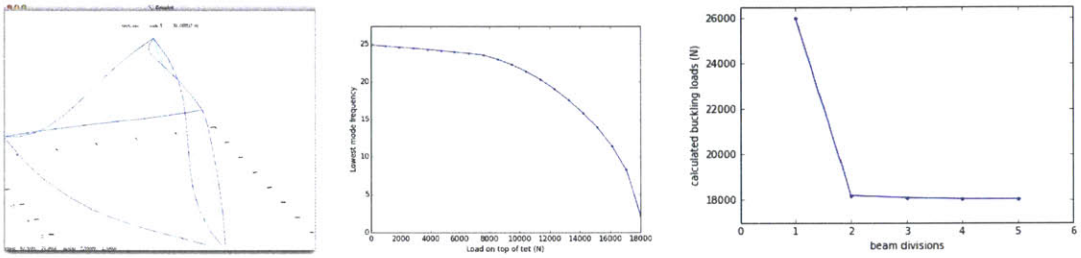


Figure 7.12: Stability example with tetrahedron

As an implementation detail, this buckling analysis is more accurate if we subdivide the beams. We illustrate this in the right graph in Figure 7.12, where we plot the calculated buckling loads versus number of beam divisions. We see that the estimate without subdivision significantly overestimates the limit value, but that we converge quickly after adding a division.

We can apply the same analysis to the sphere, but we first need to translate the pressure difference into a load on the structure. To do this, we again partition the boundary of the sphere into kite-shaped patches for each tendon of length l (shown in Figure 7.7). These kites have area $\frac{4}{3}$ times that of a equilateral triangle of side l. This area equals $\frac{l^2}{\sqrt{3}}$. Thus, the force from a pressure P per unit length of the tendon supporting the load is $\frac{Pl}{\sqrt{3}}$. To implement this in simulation, we apply uniform loads along each of the outer members with this magnitude.

This modal analysis technique is convenient because we don't need to know a priori which mode shapes will cause an instability. To perform the analysis, we must eliminate the six rigid body degrees of freedom. If we simply apply six constraints to a single node, however, modal analysis will find shapes that do not represent a physically meaningful deformation. To produce relevant mode shapes, we enforce that the extreme nodes on each coordinate axis stay on those coordinate axes while remaining free to rotate. This is equivalent to enforcing radial deformations on a subset of the nodes, a common technique used to derive analytical stability formulas for less complicated structures [44].

Using the lattice design tools, we set up the beam elements in the shape of the spherical shell. In this case, we again specify the lattice by a unit cell, but instead of using translations to propagate the cell in space, we use elements of the icosahedral symmetry group. To perform the modal analysis, we use the Frame3dd library [18]. For this buckling analysis, the geometric

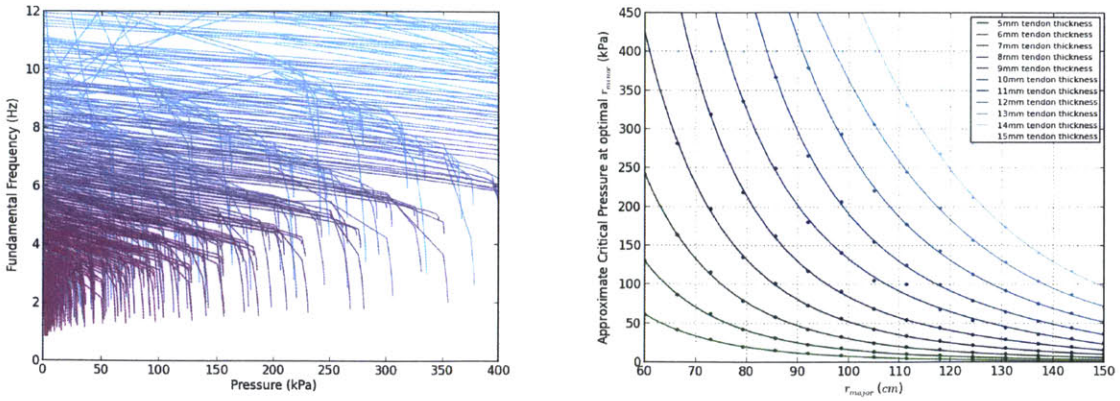


Figure 7.13: Left, lowest mode frequency vs. pressure difference. Right, aggregated critical pressures for an array of balloon parameters.

stiffness calculation performed by this library is a critical difference from a simple linear elasticity calculation.

In Figure 7.13, we perform parameter sweeps on the radii and tendon thicknesses of the spherical shell. For each parameter combination, we estimate the critical pressure which drives the fundamental frequency to zero. To do this, we perform a binary search on the pressure value, starting from a known stable pressure (near zero), bracketing the zero crossing, and taking diminishing steps based on the results of the simulation. When the step size is below a small acceptable tolerance, we exit.

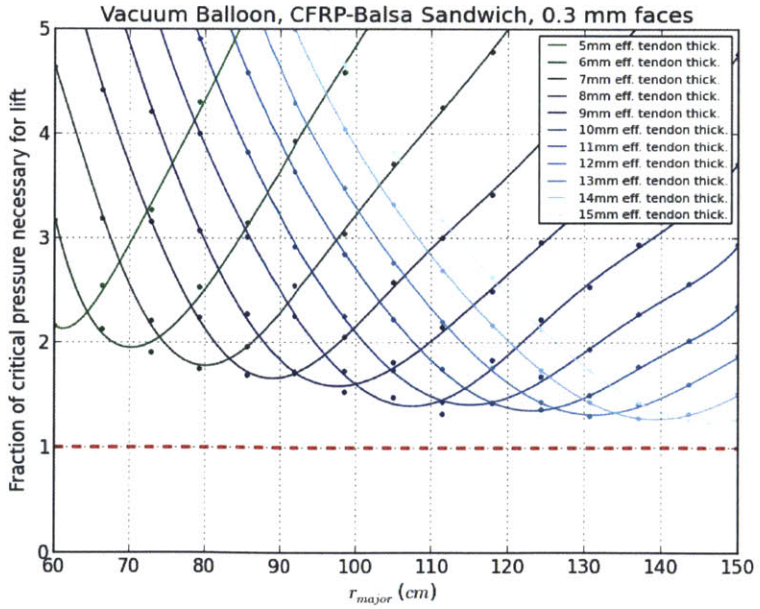


Figure 7.14: Plotting the fraction of the calculated critical pressure needed for lift, using the CFRP-balsa sandwich parts.

The left graph plots the fundamental frequency versus pressure curves for many parameter combinations, representing major radius of the sphere, the minor radius, and the equivalent cross sectional dimension of the frame elements. We can use these equivalent cross sectional dimensions to parameterize the sandwich construction described in section 7.3.2.

Given the number of simulations required for this computation, we implemented it as an array job for a 176-core computing cluster. The job scheduler dispatches each parameter combination to the cluster nodes, and each node reports back with the corresponding fundamental frequency curve.

Interestingly, we observed the maximum critical pressure value when the minor radius was a set fraction of the major radius, approximately .57. This suggests it may be possible to derive a formula for this quantity, but this was out of the scope of the present work. As we sweep over the major radius and pick the minor radius optimally, we can observe how the critical pressure scales for each tendon dimension. The right graph in Figure 7.13 shows this, illustrating as predicted that a larger sphere is unstable at a lower pressure difference. For each series, this graph fits a function $a + bx^{-1} + cx^{-2} + dx^{-3}$ to the simulation results. This function fits the data well, but with fewer terms, the fit is significantly worse. This indicates (with our scaling law $P_{crit} \propto \frac{1}{r^{2s}}$), that s is (very roughly) near 1.5, a value that is nicely within expectations (e.g. the 3x3x3 brick produced a value of 1.755).

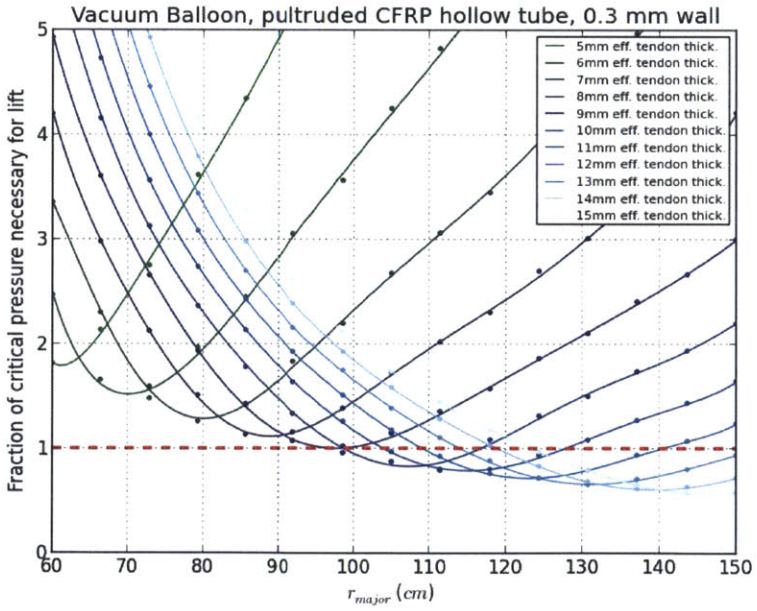


Figure 7.15: Plotting the fraction of the calculated critical pressure needed for lift, using hollow pultruded CFRP parts.

Once we can estimate the buckling pressure of a sphere lattice, we need to compare the generated lift to the mass required for the structure. Calculating the generated lift is a simple matter of multiplying the density of air by volume of the sphere. Estimating the mass requires us to use the equation encapsulated by Figure 7.11, backing out the density of the sandwich structure from the effective cross section used in the simulation. The critical line at unity in the graph

separates designs that are predicted to float without imploding from those that are predicted to fail.

The same analysis can be adapted to simulate composite parts with a hollow, pultruded cross section (although the part design would be different). We show the same critical pressure fraction graph for this geometry in Figure 7.15. We see the hollow parts outperform the sandwich parts on this metric. This is because the sandwich parts grow excessively wide to maintain uniform stiffness, whereas the hollow parts are inherently uniform in both cross section directions.

These graphs suggest that while the task of building a vacuum balloon is indeed a difficult one, it is certainly not outside the realm of the possible! These results are meant more to illustrate the application of the simulation and design tools than as a recommendation for the best vacuum balloon design. Likely, a higher performance structure could be achieved by separating the functions of tension and compression, adapting the design parameters to create a tensegrity structure.

Chapter 8

Conclusions and Future Work

In this thesis, we have demonstrated the ability to assemble linear elastic descriptions of digital materials in a way closely analogous to the physical assembly process. This hierarchical method offered a significant reduction in problem size and running time versus a meshed approach for the same modeling. Indeed, the conventional finite element analyses we ran for comparison not only took considerably more resources to solve, but also involved time-consuming geometry conversion pipelines. In practice, these effects limit the overall time required for a simulation-validation cycle, a serious hindrance for the design of a complex part. Further, the solutions from the hierarchical method showed better correlation with empirical measurements of structural performance for all tests performed. The most likely cause for this is simply the inherent model complexity of meshed FEA, as compared to the relative simplicity of the hierarchical representation of digital materials. In practice, as described in [16], the aforementioned geometry conversion pipelines are ripe with potential for errors, both human and systematic in nature. The user must keep track of many more files and often perform manual simplification and de-featuring operations. The disconnected, multi-stage processes of reinterpretation and healing can also alter geometry, often erasing critical features. Thus, using the hierarchical structure inherent in the geometry to simplify the simulation model and avoid converting between representations can produce more accurate, reliable results.

We also demonstrated production techniques for digital material part production, including a table-top process for fabricating high-performance, directionally-aligned composite parts. We generalized this technique to accommodate sandwich core composites, and produced parts with significantly greater specific bending stiffness per mass than even the directionally aligned parts. Both techniques are exciting directions for future work in expanding the possible types of geometry and streamlining the automated winding process.

We incorporated the constraints of this fabrication technique into a design for a skinned, lighter-than-air balloon structure. We then used the hierarchical simulation tools to explore the parameter space of this design, subject to the strict constraint of withstanding atmospheric crush pressure. Using modal analysis, we calculated critical pressures for structural stability and showed optimal relationships between design parameters. Based on this analysis, we found portions of parameter space where flotation based on the lift from the displaced air appears feasible. Because of the combinatorial explosion required for this type of analysis, such an extensive design study would have been prohibitively expensive for a simulation framework that did not exploit the substructure of the digital material design.

Not only are these results very exciting for the prospect of engineering with discrete assembled materials, but they also leave a tremendous amount to be done. The same principles of analysis can be extended beyond static linear elasticity to include dynamics, and nonlinear responses. Further, the modeling of any field (thermal, electromagnetic, etc.) that gives rise to integral whose domain is confined to the part only (as opposed to including the space around them) could be made similarly hierarchical. For any of these cases, this work has shown that if we intend to build discretely assemble structures, there is great advantage in exploiting this structure in modeling.

In these simulations, the behavior of structural elements usually had a single characterization, iterated over the structure many times. For real-world engineering use of these techniques, we could imagine a more useful design tool that simultaneously creates geometry for a variety of structural elements, keeps track of behavior models, and generates simulation inputs based on stitching these models together. Such an integrated design and modeling environment would be a valuable tool for agile engineering.

The design study undertaken for this work also opens exciting directions for future work. These preliminary results indicate that with more iterations of the simulate-prototype-test cycle, physically realizing a vacuum balloon may be possible. The study only considered a single lattice design with a single part fabrication process, so there is considerable room for exploration and improvement using the demonstrated workflows.

Chapter 9

Bibliography

[1] F. H. Abdalla, M. H. Megat, M. S. Sapuan, and B. B. Sahari. Determination of volume fraction values of filament wound glass and carbon fiber reinforced composites. *ARPN Journal of Engineering and Applied Sciences*, 3(4):7–11, 2008.

[2] Adam and Charles Black, editors. *Encyclopaedia Britannica, Ninth Edition, Volume 1.* Edinburgh, 1875. Aeronautics.

[3] 3M Industrial Adhesives and Tapes. 3m sprayable adhesives. http://multimedia.3m.com/mws/mediawebserver?mwsId=66666UF6EVsSyXTtn8TVNXT_EVtQEVs6EVs6EVs6E666666--.

[4] A. Akhmeteli and A. Gavrilin. Layered shell vacuum balloons, February 23 2006. US Patent App. 11/127,613.

[5] R.T.L. Allen. *Concrete in Coastal Structures.* Telford, 1998.

[6] Inc ANSYS. Ansys mechanical apdl element reference, 2012. Release 14.5.

[7] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry F. Smith, and Hong Zhang. PETSc Web page. http://www.mcs.anl.gov/petsc, 2014.

[8] Bbanerje. Sandwichbendinit. Own work. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons http://commons.wikimedia.org/wiki/File:SandwichBendInit.svg#mediaviewer/File:SandwichBendInit.svg, 2010. [Online; accessed 18-July-2014].

[9] Allan F. Bower. *Applied Mechanics of Solids.* CRC Press, 2009.

[10] P.V Chandramohan, V. Sundar, S.A. Sannasiraj, and A. Arunjith. Development of ?kolos? armor block and its hydrodynamic performance. In *Proc.8th Intl. Conf. on Coastal and Port Engineering in Developing Countries*, PIANC-COPEDEC VIII, pages 1344–1354, February, 2012.

[11] Kenneth C. Cheung. *Digital Cellular Solids: Reconfigurable Composite Materials.* PhD thesis, Massachusetts Institute of Technology, 2012.

[12] Kenneth C. Cheung and Neil Gershenfeld. Reversibly assembled cellular composite materials. *Science*, 341(6151):1219–1221, 2013.

[13] R. Connelly and Allen Back. Mathematics and tensegrity. *American Scientist*, 86(2):142–151, 1998.

[14] R. Connelly, P.W. Fowler, S.D. Guest, B. Schulze, and W.J. Whiteley. When is a symmetric pin-jointed framework isostatic? *International Journal of Solids and Structures*, 46(3?4):762 – 773, 2009.

[15] Michael Freytag, Vadim Shapiro, and Igor Tsukanov. Field modeling with sampled distances. *Computer-Aided Design*, 38(2):87 – 100, 2006.

[16] Michael Freytag, Vadim Shapiro, and Igor Tsukanov. Finite element analysis in situ. *Finite Elements in Analysis and Design*, 47(9):957 – 972, 2011.

[17] Henri Gavin. Cee 421l. matrix structural analysis. Duke University class notes, http://people.duke.edu/~hpgavin/cee421/.

[18] Henri P. Gavin. Frame3dd: Static and dynamic structural analysis of 2d and 3d frames. http://frame3dd.sourceforge.net/. [Online; accessed 6-July-2014], GPL License.

[19] Neil A. Gershenfeld. *The Nature of Mathematical Modeling.* Cambridge University Press, 1998.

[20] L.J. Gibson and M.F. Ashby. *Cellular Solids: Structure and Properties.* Cambridge Solid State Science Series. Cambridge University Press, 1999.

[21] Eitan Grinspun, Petr Krysl, and Peter Schröder. Charms: A simple framework for adaptive simulation. In *ACM Transactions on Graphics*, pages 281–290. ACM Press, 2002.

[22] Klaus Hollig. *Finite Element Methods with B-Splines.* Society for Industrial and Applied Mathematics, 2003.

[23] Leonid Vital'evich Kantorovich and Vladimir Ivanovich Krylov. *Approximate methods of higher analysis.* Groningen : Noordhoff, 1958. First published in 1936 under title: Metody priblizhennogo resheniia uravnenii v chastnykh proizvodnykh (transliterated).

[24] Matthew Keeter. Kokopelli: Script-based cad/cam in python. https://github.com/mkeeter.

[25] Matthew J. Keeter. Hierarchical volumetric object representations for digital fabrication workflows. Master's thesis, Massachusetts Institute of Technology, 2013.

[26] Roderic Lakes. Materials with structural hierarchy. *Nature*, 361(6412):511–515, 02 1993.

[27] Brian Luft, Vadim Shapiro, and Igor Tsukanov. Geometrically adaptive numerical integration. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*, SPM '08, pages 147–157, New York, NY, USA, 2008. ACM.

[28] Robert MacCurdy, Anthony McNicoll, and Hod Lipson. Bitblox: Printable digital materials for electromechanical machines. *The International Journal of Robotics Research*, 2014.

[29] CSIRO Mathematical and Information Sciences. Boundary conditions, fastflo tutorial guide, 1998.

[30] Jack Moran. *An Introduction to Theoretical and Computational Aerodynamics.* Courier Dover Publications, Mineola, NY, USA, 1984.

[31] David Noel. Lighter than air craft using vacuum. *Correspondence, Speculations in Science and Technology*, 6:262–266, 1 1983.

[32] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, July 2003.

[33] Otherlab. Geode: A computational geometry library for c++ and python. https://github.com/otherlab/geode.

[34] Achilles Petras. *Design of Sandwich Structures.* PhD thesis, Robinson College, Cambridge, 1998.

[35] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing.* Cambridge University Press, New York, NY, USA, 1992.

[36] D. Rayneau-Kirkhope, R. S. Farr, and Y. Mao. Fractal-like dependence in the designs of efficient pressure-bearing structures. *EPL (Europhysics Letters)*, 93(3):34002, 2011.

[37] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Trans. Comput. Syst.*, 2(4):277–288, November 1984.

[38] Tobias A. Schaedler, Alan J. Jacobsen, and Wiliam B. Carter. Toward lighter, stiffer materials. *Science*, 341(6151):1181–1182, 2013.

[39] Vadim Shapiro. Semi-analytic geometry with r-functions. *Acta Numerica*, 16:239–303, 5 2007.

[40] Vadim Shapiro and Igor Tsukanov. Implicit functions with guaranteed differential properties. In *Proceedings of the Fifth ACM Symposium on Solid Modeling and Applications*, SMA '99, pages 258–269, New York, NY, USA, 1999. ACM.

[41] A.H. Stroud. *Approximate calculation of multiple integrals.* Prentice-Hall, 1971.

[42] SymPy Development Team. *SymPy: Python library for symbolic mathematics*, 2014.

[43] M.F. Thorpe and P.M. Duxbury. *Rigidity Theory and Applications.* Fundamental Materials Research. Springer, 1999.

[44] Stephen P. Timoshenko and James M. Gere. *Theory of Elastic Stability, 2nd Edition.* Dover Civil and Mechanical Engineering. Dover Publications, 2009.

[45] Wikipedia. Elmer-pump-heatequation. Own work. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons http://commons.wikimedia.org/wiki/File:Elmer-pump-heatequation.png#file, 2010. [Online; accessed 23-July-2014].

[46] Wikipedia. Vacuum airship — wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Vacuum_airship&oldid=607943247, 2014. [Online; accessed 6-July-2014].

[47] George Woodruff, Paul Muench, and Gary Witus. Zipper mast for enhanced communications and surveillance. In *Proc. SPIE*, volume 8045, pages 804512–804512–6, 2011.