Cylindrical Mesh Morphologies

Study of Computational Meshes Based on Parameters of Force, Material, and Space for the Design of Tension-Active Structures

Moritz Fleischmann[1], Sean Ahlquist[2]
[1,2]Emergent Technologies and Design (EmTech), AA School of Architecture, London, UK,
[1,2]Institute for Computational Design (ICD), University of Stuttgart, Germany
www.arch-research.net
[1]moritz@arch-research.net, [2]sean@arch-research.net

Abstract: *In experimenting with digital processes for simulating the behavior of tension-active cable nets, a method was developed for creating informed geometries by utilizing computational meshes that carry properties of structure, space, and material. A spring-based particle system provided the dynamics to simulate the flow of tension force through the geometry. Particular functions were scripted to embed logics for fabrication and analysis of spatial parameters. This formulated a lightweight, reactive design tool for which multiple cable net morphologies could be quickly generated. This paper will describe the experiments in creating the method to generate such cable net morphologies, and discuss the potential application for this computational framework to apply to other architectural systems.*

Keywords: *Computation; particle system; spring; dynamic relaxation; Processing; fabrication.*

Introduction

Cable Nets, also defined as tension active structures, are compelling as architectural systems having an intimate connection between structural performance and the arrangement of material. The direct flow of structural forces through the material makes these systems efficient, attractive and unique from an aesthetic point of view. Cable nets, in comparison to membrane structures (also tension active structures), can be developed to realize a wider array of structurally stable forms. But, the modes for designing such structures are intensely technical, be it a through physical modeling or through digitally-based methods. As the engineering is critical in determining form, the normal design process must engage the structural logics of the system in more depth as compared to processes where structure can be rationalized from a given form. Our research has focused on generating a designer authored method digitally simulating the behaviors of tension active structures using simple geometric components

related to material and structural performance. The method is intended to be *design-oriented* where particular aspects of space and pattern are also parameterized behaviors. This envisions defining cable nets as heavily engineered *architectural* systems, where the method engages the forming of a digital mesh that is informed by design and structural characteristics. The research investigates scripting as a technique for data organization, sorting, and manipulation, inter-relating the various parameters of cable net architectural systems.

Traditional methods for solving tension active structures rely on both analog modeling techniques and the use of advanced engineering software. The complexity and laborious nature of both processes presents impedances to iterating through design variations. Design-oriented software packages have recently emerged, minimizing a need for intense specificity and technical detail in the setup of geometry (of which you will find in the engineering-oriented predecessors). While these software packages provide expediency, our research saw, in RhinoMembrane as one example, how their determination of force equilibrium does not always match to the dynamics of a cable net system. The packages also work without an open programming environment. Our research looks to locate the designer within a digital environment where specific parameters beyond pure structural constraints can be inserted. Such a closed system prevents the ability to embed parameters alternative to the structure-based variables.

Programming environments, such as Processing and Cinderella (both based in Java), are examples of accessible languages that are design-oriented and can deal efficiently with physics-based solvers. They are unique to architecture, in one sense, in that they work natively in 2d for graphical design. But experiments done by Axel Killian (MIT), Jeffrey Traer Bernstein (Princeton), and Chris Williams (University of Bath), have shown the environments to be robust and efficient for generating complex 3d structural systems. Utilizing the Processing environment, our research will focus on the use of a spring-based particle library to solve for force equilibrium in tension active geometries.

Data organization sits at the core of programmed (scripted), parametric design methods. Our research presents a framework for embedding data that goes beyond that of simple coordinate information and geometry association. The structuring of data is about both the coordination of geometric elements with particular constraints, and the allowance for efficient means of extracting design relevant data. In this work, the cylinder becomes a geometric foundation for defining spatial directionality, but also acts as a concept for defining the base data structure. Fabrication and pattern-recognition are key embedded components utilizing the logics of the data structure. As architects, the driver for this process is to develop physical architecture manifested through a digital method of form-finding, as opposed to hypothetical computational meshes that lack constraints towards material and structure.

This paper will describe the research in generating a computational form-finding method for designing cable net systems. We will cover the core technology for simulating structural performance, and methods for embedding particular characteristics of design and fabrication. The work will be concluded by examining the how this approach may be expanded to include other structural systems and design parameters, and discussing where this method sits within the entire process of producing architectural form.

Solvers for tension active structures

There are many solvers available for the digital simulation of pre-stressed cable net structures and membranes. The most commonly encountered ones when looking into structural analysis software are the Dynamic relaxation, the Force-density method, and Finite-Element Method. In general, the first 2 solvers are used for the simulation of cable net structures even though in their long history the Dynamic

relaxation method has been implemented to simulate shells, cable nets and membranes as well as plates (Lewis, 2003). It is first used as a solver for the analysis of cable nets by Day and Bunce in 1970 (Day and Bunce 1970) while FEM-based solvers are more universal and often used for the simulation of membrane behavior.

The dynamic relaxation method proved to be an efficient computational method for simulating tension-active systems based on springs. In 2004, Prof. Ochsendorf and Simon Greenwold, through a workshop at MIT (ocw.mit.edu/OcwWeb/Architecture/4-491Fall-2004/LectureNotes/: Mar 2008), conducted digital form-finding experiments based on Gaudi's analog form-finding experiments with catenaries. One product of this workshop was an algorithm built for the Processing language based on the physics of springs. Axel Killian, as a part of his PhD research at MIT, utilized this algorithm to digitally form-find linear catenaries. He extended the application to solve for tensile surfaces by using a network of springs, eventually building this into a software called CADenary.

Springs are a part of a computational physics model where their force, generally, is connected to the distance at which the two ends of the spring sit apart from each other. The two ends of a spring are defined by particles. A particle is a node element that carries information for mass. Springs are one type of force typically sitting within a particle system library that contains other elements which can undergo other forces, such as gravity or magnetism, in the computational simulation. Hooke's Law for springs specifies the amount of force through the degree of difference between a spring's actual length and its rest length. A spring-based computational system serves as an efficient solver for tension-active systems. The work by Axel Killian utilized the tension aspect of a spring, but developed forms in which it is imagined that the forces would be inverted into a compression-only structure (simply put, a computational version of the Gaudi hanging chain models).

The particular system by Greenwold being placed within the programming environment of Processing (a Java based language) allows it to be lightweight, flexible, and openly accessible. This spring based particle system library is the core of our digital design process, providing the algorithms for spring force determination and dynamic relaxation to solve for force equilibrium in a network of springs.

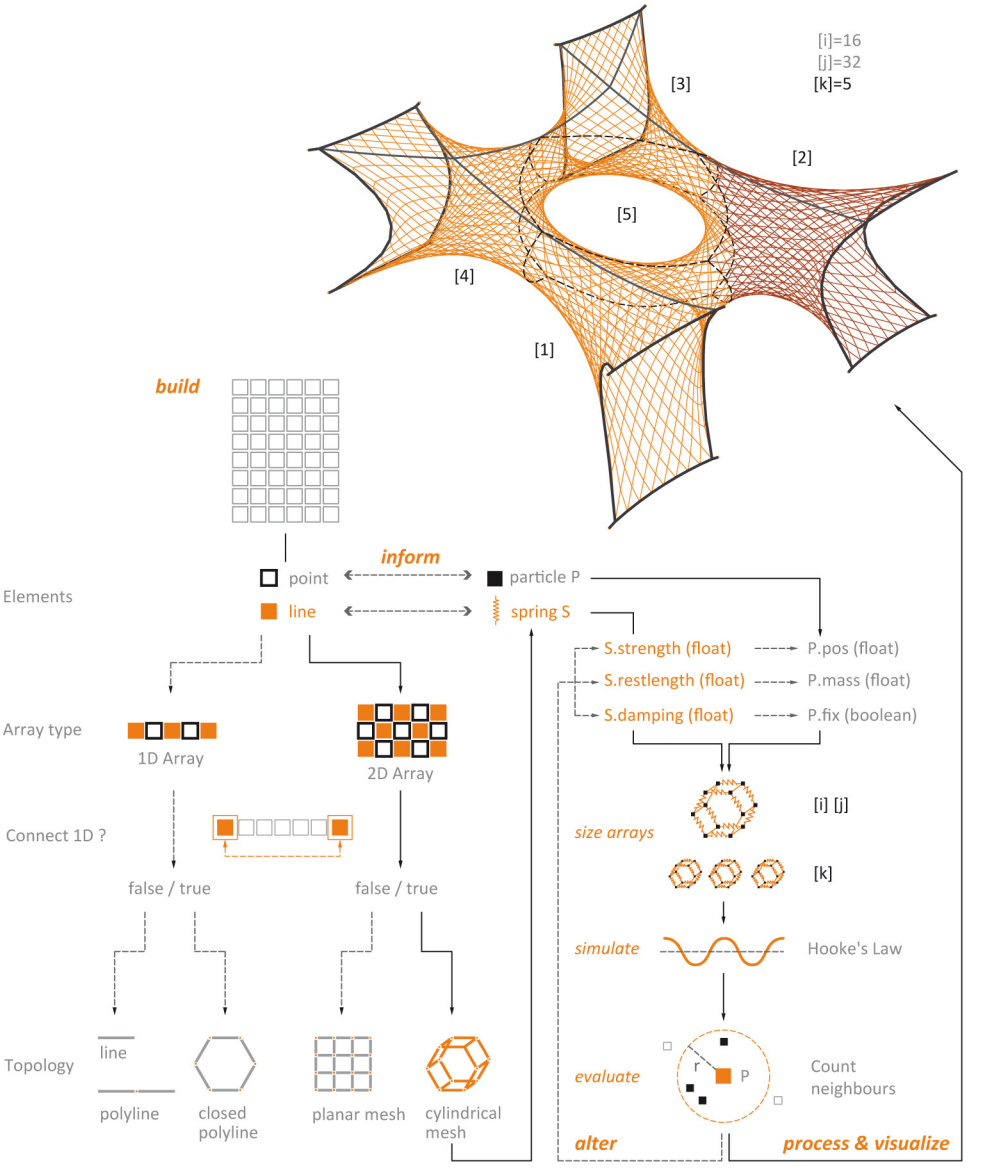
Design pressures for shaping form

Cylindrical topology

Like Killian's CADenary software, our approach was to utilize a network of springs to computationally form-find a tension-active surface. This means the primary engine for developing form is the spring force along with the locations of fixed particles (definition of the boundary condition). This pressure sits mainly within the category of structure. Where we looked to advance our work was by trying to induce pressures based on *design-oriented* decisions. This is a process by which elements do not fall into place simply based on the flow of spring force, but through the negotiation of multiple form-influencing parameters. In Killian's paper on particle systems and form-finding (Killian and Oschendorf, 2005), he mentions the choice of mesh pattern (connectivity between springs) as a decision which influences the degree of optimization in the arrangement of springs, in terms of force flow. If the pattern is based on a design choice, then the arrangement of geometry is not driven purely towards an optimized solution about structural efficiency.

In our experiments, the choice of an overall organizational strategy was a significant design pressure placed upon the system. The cylinder is of particular interest for the arrangement of a cable net structure, in comparison to other base geometries such as the plane or sphere. The cylinder can describe a boundary and also imply a direction. Frei Otto specifically refers to this type as a *spatial* cable net (Otto, 1962). It is unique in terms of tension-active forms, where they are typically derivatives of planar assemblies (expressed as saddle-shaped or conical forms). As a

Figure 1
Computational mesh elements



parameter for the form-finding of cable nets, the cylinder form can be realized through the negotiation of its spatial qualities and the imposition of tension-force in the arrangement of structure.

The network of particles and springs is generated through a simple but highly structured computational process. The foundation of the network is a 2-dimensional map of particles, where each particle is an instance within a double-array. The particles are connected with springs forming a rigorous dia-grid pattern. The network, continuous along 1-dimension, is what defines the cylindrical topology. Its highly structured nature, with the double-array of particles and rigorous connection method of springs, allows for ease and extensibility in generating topologies of different numerical scales. It is also critical for methods of manipulating parameters, and reading and extracting data.

Springs can exhibit either tension or compression force, having inherent translation to particular materials. Filtering through a more critical analysis of the magnitudes of force for individual springs, material assignments can be more specifically determined. The determination of force only occurs when all springs interact with each other. This introduces the difference between mesh topology and geometry. Topology is the association of springs, in this case, in a manner akin to a cylinder, and in a diagrid-like pattern. Geometry is the arrangement of springs once they reach a force equilibrium (this process is achieved through the method of dynamic relaxation). Understanding and controlling the topology, at the level of the individual particles and springs, allows for an advanced degree of control in generating geometries, and extensibility in the range of geometries that can be produced. This also exposes the intimate connection between the manipulation of mesh and the resulting form. While the topology is not being changed in our experiments, the individual elements within the topology are being adjusted to affect force distribution and thus affect the overall form. The highly ordered computational system is setup to allow such local manipulation to occur. The rigorous numerical nature of the logic also allows for patterns of variation across the entire network to be investigated.

Fabrication logics

Rigid numerical ordering is common practice in computational methods, for maintaining a data structure and for providing elements with unique identifiers. In these experiments, each particle is created with a unique double-array [i][j] value, and springs are connected between in a methodical manner forming a dia-grid like pattern. The distinctive computational effort here is the re-use of this ordering method and unique ID structure to dictate a physical assembly method. As opposed to applying another ordering system, the particle array values in digital space defines the pattern for stepping through the nodes in physical space.

The logic for mapping the particle topology is universal. As mentioned before, the internal topology for the network is not manipulated. This may have a limitation in form-making, but for providing information for fabrication, where the geometric outcome can vary greatly, the method for sorting through the nodes/particles for fabrication is the same for each result. Because sorting is based on a function that recursively tracks through the particles of a series of springs, the bottom-up system is scalable and geometrically un-constrained.

In the effort of building a cable net from the computational model, the considerations of force, both distributed and within each element, are necessary. A spring, by default, has infinite stretch. This is advantageous for computational form-finding, in that a solution will almost always be found. In prescribing a spring, and its actual length, to a material length, the elasticity of the specific material has to be considered in the translation process. In our experiments, a spring's length is measured and the value reduced by a particular percentage to reflect the elasticity in the material being used. This way of viewing "stretch" in the physical model is different from the way it is viewed in the computational

model. This is simply because the computational model uses springs all with the same strength and rest length parameters. The physical model is built of a series of uniquely sized elements. In the physical model, each element ends up being stressed to the same degree when the entire structure is assembled. In the computational model, where each spring has the same rest length, the force distribution (the amount of pull on each string) is varied.

We looked to find a closer comparison in the computational model to the physical model where each spring's rest length is changed to match its actual length (minus a certain percentage to reflect the same translation method as described above). In this method, the computational model would exhibits a series of unique springs does show a force diagram akin to that the physical model.

Densifying behaviors

The described process simulates the general behaviour of tension-active structures in the form of cable nets and created the necessary output for the assembly of a form (in this case the AA Exhibition Installation). The choice of the cylinder as a base topology is a fixed parameter in these experiments. What we looked to do computationally was visualize the dynamics of a cylinder, or multiple inter-connected cylinders. The computational analysis is done through a rudimentary tool to identify denser ("closed") and less dense ("open") regions within the structure in order to highlight areas, that would form a visual threshold indicating a surrounding space.

As the cylindrical surface was dissolved into a network of linear elements (cables) and distorted significantly, the geometry did not directly indicate an underlying cylindrical topology anymore. By looking at the simulation from different angles, it became obvious, that in a mesh the distinction between "front" and "back" was harder to define visually. But the overlapping mesh regions were of particular interest for the further development as the threshold became an emergent property of the structure, highly dependent on the spectator's viewing position, the overlapping layers and the general mesh density.

Looking to insert visual experience as a recognized characteristic, the analysis function reads node (particle) proximities and displays the ranges in which the nodes have accumulated or dispersed given a certain perspective. In order to calculate the proximity it was not satisfactory to simply measure the actual distance between one particle and another. In a projected plane, a node that is perceived as a "neighbor" to another node might be quite far away, in X,Y,Z space. We developed a method that evaluated only two-dimensional distances by projecting each particle (P) onto a plane (result P') with regards to a specific point in space (Spectator).

The analysis is embedded as an "extension" of the base particle class itself in Processing. This maintains the lightweight nature of the process where a global function for analysis all particle positions was not necessary. Each particle simply contains a reader for how many neighbours the particle (P) has within a particular radius (r) around its Projection (P'). The trait is set at the local level, but is activated when a particle is part of a larger network.

This tool was able to measure the interesting observed effects of multi-layered meshes, POV-dependencies and physical mesh-densities, without slowing the digital form-finding process down significantly as it was programmed in a OOP-fashion (extension of a class).

Expanding the topology

Initially, our research aimed to digitally form-find a cylindrical cable net structure, defining certain computational parameters to test degrees of variation in the cylindrical morphologies that could be generated. This method was established and tested through a constructed installation for the AA Projects Review exhibition in the spring 2009. Looking at how a cable net, through its varied pattern densities, can both define and disintegrate a surface, we chose to pursue a method for multiplying the number of cable

nets within an architectural system, and begin to ascertain the emerging characteristics of overall form and pattern. We chose to define two parameters through which the computational network could be expanded: changing the number of spring elements within a cylinder, and connecting multiple cylinders.

To achieve the expansion of the overall network, the cylinder is defined as an object within a more clearly structured hierarchical system. Object-oriented programming allows for the definition of a unique element and the statement of the exact variables and parameters that can be accessed. In our case, the cylinder is the object, where particle count and spring count are examples of variables that can be manipulated, and particle ID is an example of a parameter that can be accessed.

Generating multiple instances of the object demands a strategy for connecting each instance. This is critical when applied to a system of tension-active objects, where the flow of force means that each object influences the other in determining the final form. The method in which objects connect is through defining various point and edge conditions along the open boundaries of the cylindrical object. The rule is not constrained by connecting the entire open boundary of one object to the entire open boundary of another. Association occurs by connecting particles from one component to another, or strings of particles between two components.

This computational structure, setup as a fixed hierarchy, allows for the production of the geometry where performance-based hierarchies can be transformed. Ridge cables emerge as continuous lines of springs carrying more force, and having more influence on the shaping of the overall form. Branching was found as one of the emergent behaviors. This was not a directly scripted behavior; rather a particular arrangement of associations between particles and strings-of-particles amongst multiple components produced the branched volumetric mesh. While the computational model can be broken down into a series of cylindrical objects, the geometry is represented a one hierarchy of a single continuous network. The geometry (position of elements in space) is generated through the equalization of springs force across the entire network.

Analysis and conclusion

Meshes are the foundation of form generation for designing architecture through digital means. As a network of vertices and edges (or points and lines, as shown in Figure 1), a mesh acts as an organizational system for digital geometry. Even in working with NURBS, another organizational system for position of a series of vertices in space, a mesh underlies this network when architecture (as it typically does) has to be converted in a series of planar geometries. The organizational systems are primarily scale-less, allowing them to visualize any material assembly of architecturally prescribed elements. In our experiments, we viewed geometry as being formed through the primary parameters of shape organization (cylindrical topology) and distribution of spring force (Hooke's Law and Dynamic Relaxation). The "mesh" was an accumulation of all of these parameters and constraints, expressed in a network of springs and particles. What makes this applicable to methods for designing other architectural/structural systems is to view a computational mesh as the primary parameter, constructed of vertices, edges, and rules for association between elements (see Figure 1). The constraints of structure, material, and space are characteristics embedded within the individual elements that constitute the mesh. The constraints inform the position of vertices and edges through their own negotiation. The insertion of material-based parameters introduces a specification of scale to the previously scale-less computational mesh.

When additional criteria for the generation of form can be embedded into this building block of digital form, the mesh itself becomes more purposeful and directed to the production of specific, relevant, and more effective architectural shapes. The challenge, as we see it, is to inform the initial stages of the design generation process with values

of structure, function, and performance; these being active tools in constraining and providing feedback during form generation. The information supplied in the mesh can flow forward in the design process for further specification, validation, and analysis. The "rough" values and approximate shape can be further specified and evaluated in successive processes of computational analysis; for instance, material details and componentry in GenerativeComponents, structural analysis in Ansys, and environmental calculations in Ecotect. The purpose of informed meshes is not to have absolute accuracy available during the initial formation of architectural shapes, but to work within a computationally lightweight method that supports the production of information beyond position and association of elements in space.

Acknowledgements

This research was initially developed as a part of the Emergent Technologies and Design Programme at the AA in London, under the direction of Mike Weinstock, Michael Hensel, and Achim Menges. The work is currently being expanded and taken into new directions as Phd Candidates at the Institute for Computational Design at the University of Stuttgart with the Institute Director, Achim Menges.

References

Killian, A. and Ochsendorf, J.: 2005, Particle-Spring Systems for Structural Form-Finding, Journal for the International Association for Shell and Spatial Structures: IASS, Vol. 46 n. 147.

Pottman, H.,Asperl, A. Holfer, M. and Killian, A.: 2007, Architectural Geometry, Bentley Institute Press, Exton.

Reas, C. and Fry, B.: 2007, Processing: A Programming Handbook for Visual Designers and Artists, MIT Press, Cambridge.

Moncrieff, E.: 2005, Systems for Lightweight Structure Design, in Textile Composites and Inflatable Structures, Springer, Netherlands, pp. 17-28.

Frei, O.: 1962, Zugbeanspruchte Konstruktionen, Ullstein, Frankfurt.

Lewis, W. J.:2003, Tension Structures. Form and Behaviour. Thomas Telford, London.

Day, A. S. and Bunce, J. H.: 1970, Analysis of Cable Networks by Dynamic Relaxation, in Civil Engineering and Publics Review, pp. 383-386.