SOFTENING THE HABITATS
SUSTAINABLE INNOVATION IN MINIMAL MASS STRUCTURES AND LIGHTWEIGHT ARCHITECTURES

Proceedings of the TensiNet Symposium 2019
Softening the habitats | 3-5 June 2019, Politecnico di Milano, Milan, Italy
Alessandra Zanelli, Carol Monticelli, Marijke Mollaert, Bernd Stimpfle (Eds.)

An algorithm to draw simulations of dynamic lightweight structural systems with schemas

Mesrop ANDRIASYAN*

*Textiles Hub, ABC department, Politecnico di Milano
Via Francesco Cucchi 7, Milan 20133, Italy,
mesrop.andriasyan@polimi.it

Abstract

Architects, designers and structural engineers throughout their education and practice work mostly with visually controlled 3D modelling applications that require manually controlled actions to get the final desired model. On the contrary, we can see the development of visual programming that is now intertwined with traditional software applications and uses data flow to generate geometry. Professionals in the AEC industry witness the capabilities of visual programming mainly by means of its output and the live demonstration of scripts that automate the big amount of manual work. Notwithstanding the fact that visual programming is very appealing as an asset, acquiring the necessary skillset is often a matter of struggle. An example of this is the dynamic simulations of structural systems (tensile, bending active etc.). There are means to perform them but they usually require knowledge of visual programming and several frameworks. This paper attempts to introduce an open-source algorithm cutting short the intricate process of learning new non-conventional software and directly giving the opportunity to design dynamic structural systems as a schema and getting the instant simulations therein.

Keywords: simulation, conceptual design, lightweight structures, form finding

Peer-review under responsibility of the TensiNet Association

1. Introduction

Form finding is a procedure to determine the shape of equilibrium of the desired prestress state and the boundary conditions (Stranghöne & Uhlemann, 2016). It is found that the behaviour of the investigated bending-active structures do not fall into clearly predictable categories, their load bearing is largely dependent on the variety of topologies and geometrical expressions that may be generated. Similarly to membrane structures, the geometry must be form-found, in this case simulating the elastic bending deformation (Lienhard, 2014). Ultra-lightweight structures often include form-active components that have large deformations and complex forms that are difficult to model with conventional 3D modelling processes. In recent years more and more attention is given to visual programming platforms such as Grasshopper for Rhino 3D to develop more complex geometrical solutions. A major step forward was the advent of Kangaroo solver that helps to create dynamic simulations of different structural behaviour. Notwithstanding the fact that kangaroo uses arbitrary values and does not provide accurate structural data it greatly helps in understanding how a certain type of structures might behave. Figure 1 represents the flow chart of the described Grasshopper-Kangaroo workflow.

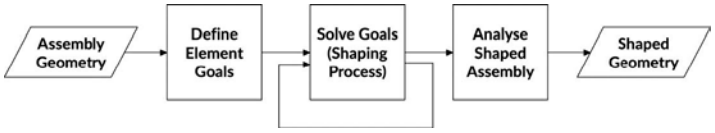

Figure 1 A design modelling pipeline using the Kangaroo2 library

(Deleuran, Pauly, Tamke, Tinning, & Thomsen, 2016)

2. Previous studies

The research presented in this paper is highly inspired by the results published in this paper (Deleuran, Pauly, Tamke, Tinning, & Thomsen, 2016). Several workflows are similar to what has been done. Some of the key differences are that the presented algorithm does not enforce the designer to operate on a given set of subdivision points to input geometry and is extended to incorporate other structural types including rigid elements (with fixed or hinged connections) and membranes. The purpose is the generalisation of the workflow to let the designer's work also on other structural types.

3. Methodology

The algorithm separates the processes of design and visual programming leaving the first part to the designer and automating the second one. What designers draw is considered a static schema of the structural system which is translated into the simulation model and drawn in a real-time manner in the same 3D environment. There are a few sets of rules that the designer has to follow. It is only required to draw the schema of the structural system with few indications of object types and some numerical values and these inputs will, later on, be carried by the algorithm. The algorithm classifies the input in to separate data streams, performs validation procedures and sequential steps of geometrical editing, creates the goal objects which represent the structural behaviour of the given objects, injects the designer's parameters and passes to the Kangaroo solver. The output is also classified into data streams for appropriate real-time visualisations and shape analysis value outputs.

The algorithm is thought to minimize the necessary input, so the designer can focus on the creation process.

3.1. Used tools

Rhino 3D CAD software is the environment where the designer creates the model(schema) of the structural system. The visual programming plug-in Grasshopper is the place where the algorithm is assembled. The fundamental basis of most computational systems used for the design of surface structures is some form of equilibrium modelling (Forster & Mollaert, 2004). This equilibrium is achieved by Kangaroo package which is an interactive physics/constraint solver. Elefront is a plug-in package for both Rhino and Grasshopper together that allows the automated transfer of properties between one another and enables auto-referencing of geometry (Wortmann & Tunçer, 2017). For representation, a package called Human is used, since it creates a pipe-like thick representation of linear elements without affecting heavily the performance. Some parts of the algorithm are developed by Python and C#.

4. The algorithm workflow

When the designer runs the algorithm, it generates a set of layers that will be used by the designer to identify component types. No input from other layers is taken into account and even the elements present in working layers are filtered to match their proper use. For example, the linear elements in the membrane layer will not be taken into account.

The indication of the element type triggers the creation of appropriate properties for that element that the designer can use to input values. These include strengths (non-structural arbitrary quantity for Kangaroo solver), length goal factors, snaping planes for rigid objects etc.). The updated elements with imported values are referenced again.

Later the referenced elements pass through chains of geometrical modelling. This includes:

4.1. Interdivision of linear elements

Kangaroo 2 engine works leveraging the position of the points that comprise elements in regards to the active collection of goals. If we want an element to interact with another one, they must have at least one common vertex. In this regard, all the points that lie within the given distance (a number close to 0) from a linear element will act as a division point. This includes the intersection events between all linear elements, anchors, loads and boundary defining points of membranes Figure 2.

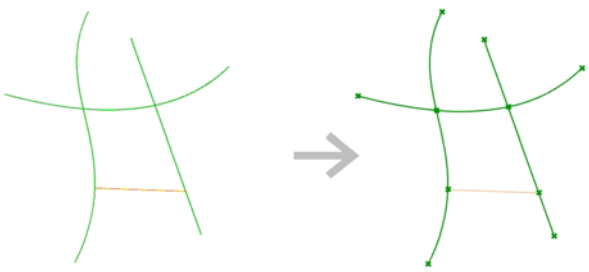

Figure 2Intersection event of different elements

4.2. Graph connected components

In order to let the designer draw connected rigid components, it was necessary after solving the intersection events of curves to group them by connectivity. Here a sub-algorithm is used to perform flat hard clustering based on curve-curve intersection events (Manning, Raghavan, &

Schutze, n.d.). It creates clusters of connected components in the graph. In Figure 3 the nodes represent curves. If the curves are intersecting (including end-to-end connections) then there's a connection line between those nodes. The algorithm clusters together all the connected curves and gives indexes of the curves from the initial unordered list as a data tree.

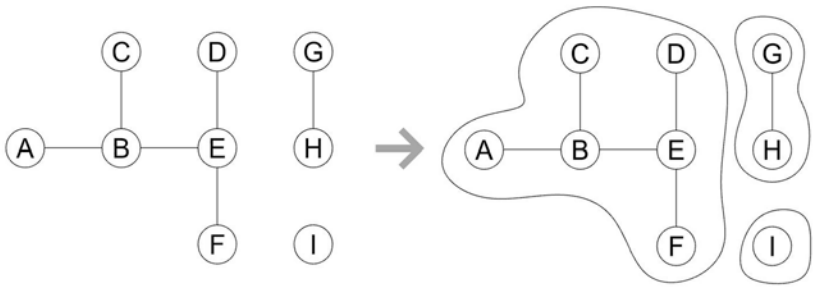

Figure 3Graph connected components clustering

4.3. Segmentation

Since Kangaroo 2 works with points, in order to simulate the behaviour of curved elements (rods, cables etc.) it is required to divide the designed curve elements into several linear segments Figure 4. The precision is one of the designer's inputs. It is usually based on the scale, the complexity of the project and the computing power of the used processors.

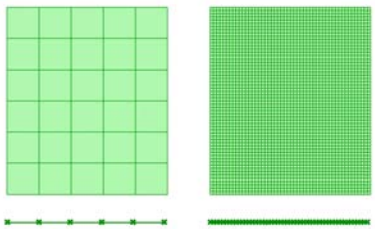

Figure 4 Low and high segmentation controlled by segment size

4.4. Membrane adjacencies

One of the challenges in simulating the membranes with Kangaroo is the solution of adjacencies of different membrane meshes with non-matching edge vertices. In these cases, since there is no vertex wise superposition, the meshes act as separate units. As an inherent part of this algorithm, a special part is written to address this issue. It finds the connected edges between membranes and creates local goals of collinearity for each point with the nearest two points on the other membrane. This ensures that the seams are correctly simulated. The designer has the choice to weld or not different membranes Figure 5.

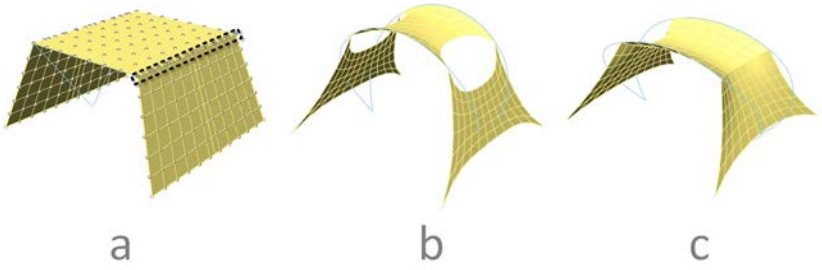

Figure 5 a) drawn model b) separate membranes c) joint membranes

After the presented geometrical operations, each group of geometry is composing a goal of Kangaroo 2 engine. The necessary parameters are passed through the designer's input. The structural and representation goals are combined into an ordered data tree and are inserted into Kangaroo 2 solver. Each time the designers changes the schema, the whole script is recomputed and the simulation model is updated to include the altered schema elements.

4.5. Geometry analysis

The algorithm post-processes and analyses the simulated geometry. Curve-like components that were segmented for simulation are re-interpolated into curves. These curves are used to generate curvature values and provide dynamic output as a curvature graph on top of the model with corresponding numerical values of curvature Figure 6. It also tracks the mean curvature values of the membranes. Although in this case there's a back and forth translation to a surface object which may reduce the accuracy Figure 7.

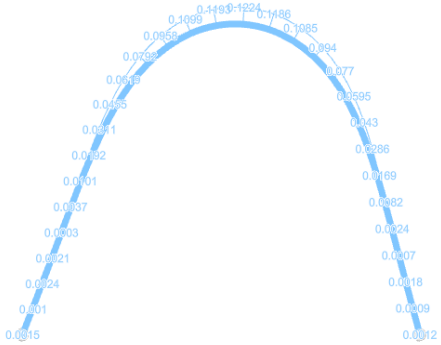

Figure 6 Rod curvature graph

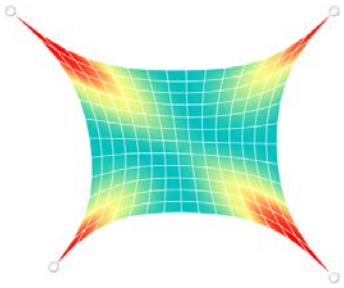

Figure 7 Colored representation of mesh curvatures

Figure 8 Depicts the flow chart of the entire algorithm.

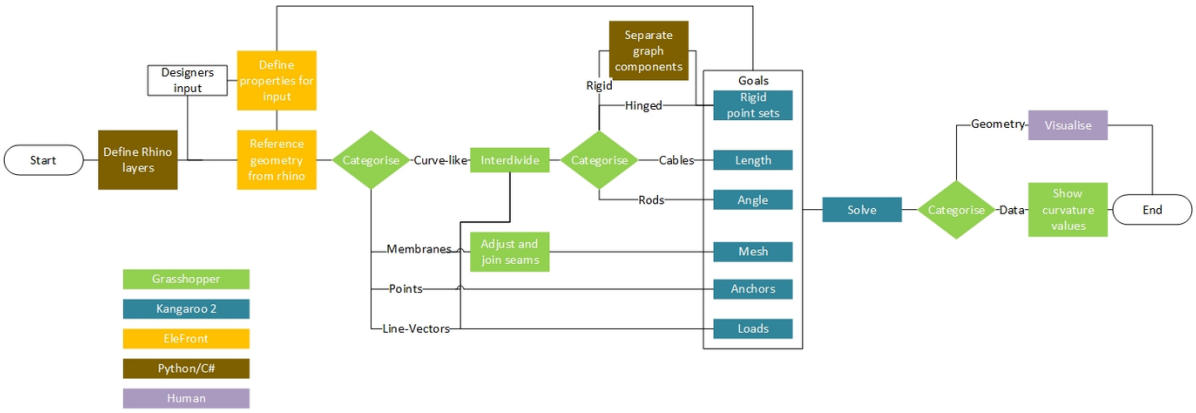

Figure 8 Macro process flow chart of the algorithm

5. Use cases

In order to validate the algorithm, many different design solutions were tested to see if it correctly carries out the simulations. The tool was distributed to the students of the lightweight structure design class and their feedbacks were taken into account to continuously expand the capabilities of the algorithm. This includes the possibility to snap rigid objects to planes, join membrane seams, insert individual strength values and many more. Some of the use cases are described below.

5.1. Gridshell

Conventionally it is possible to simulate a gridshell with the Kangaroo 2 engine. However, it takes some effort and knowledge of both Grasshopper and Kangaroo. With the help of the algorithm, the designer only has to draw a flat network of rods, lines that represent the initial and moved positions of the anchors and a load vector line to lift the structure. This workflow is more intuitive. Figure 9 shows the interaction between rods, moveable anchors and a load.

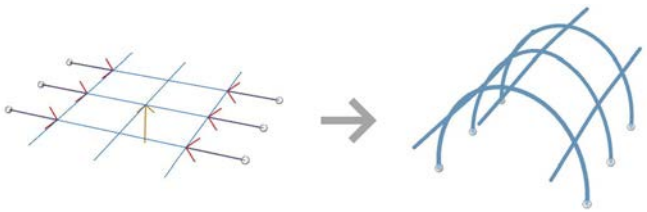

Figure 9 Gridshell schema and the ready model

5.2. Hybrid structure

Hybrid structures are very tedious to model since the joint elements of various structural types close require precise modelling and in case of simulation with Kangaroo, they also require a matching segmentation and anchor positioning. The algorithm streamlines all these processes and the schema that was drawn by the designer is effectively transformed into a correctly interconnected hybrid structure. Figure 10 shows the joint simulation of anchors, membranes and a rod.

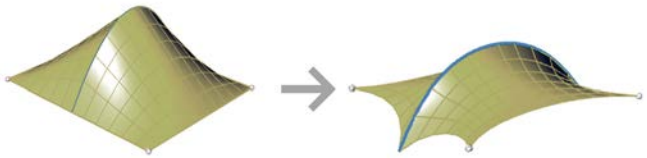

Figure 10 Hybrid structure

6. Conclusion

The aim of the research was to investigate the possibilities of streamlining the workflows of manual creation of Grasshopper+Kangaroo algorithms that are required to perform simulations of various structural systems. As a result, a robust algorithm was developed that works intuitively for the designer to translate the drawn schema into a simulation model and get instant geometry analysis data. However, the algorithm has not been tested yet on large scale projects and predictably will be heavily slowed due to a large number of operations for each component.

7. Further work

The given algorithm helps to easily simulate dynamic structural systems and to understand at a conceptual level how do things work. However, it does not provide accurate structural data. Later on, a FEM package may be incorporated to carry on with the structural calculations and validation. User feedback will be continuously collected to improve the performance and add requested functionality.

References

Deleuran, A. H., Pauly, M., Tamke, M., Tinning, I. F., & Thomsen, M. R. (2016). Exploratory Topology Modelling of Form-active Hybrid Structures. *Procedia Engineering*, *155*, 71–80.

Forster, B., & Mollaert, M. (2004). *European design guide for tensile surface structures*. *Tensinet*.

Lienhard, J. (2014). *Bending-Active Structures* (PhD thesis). Universität Stuttgart.

Manning, C. D., Raghavan, P., & Schutze, H. Flat clustering. In *Introduction to Information Retrieval* (pp. 321–345). Cambridge University Press.

Stranghöne, N., & Uhlemann, J. (2016). *Prospect for European Guidance for the Structural Design of Tensile Membrane Structures*. JRC.

Wortmann, T., & Tunçer, B. (2017). Differentiating parametric design: Digital workflows in contemporary architecture and construction. *Design Studies*, *52*, 173–197.