Will Langford
December 20, 2016
Final Paper

Design and Optimization Tools for Discretely Assembled Mechanisms

Motivation

Interest in additive manufacturing has recently been spurred by the promise of multi-material printing and the ability to embed functionality and intelligence into objects, the ultimate goal of which is to "print" a complex robotic system in a single process. While there are a number of different materials and processes used in additive manufacturing, none have been able to produce devices with the range and performance of materials required to fabricate fully integrated robotic systems.

In contrast to additive manufacturing, we have previously presented a discrete assembly approach in which functional devices are assembled from a small set of standardized part-types. In past work we have shown the applicability of this approach for the assembly of electronics [1] and robotic spacecraft [2]. In this work, we specifically focus on the discrete assembly of planar mechanisms: detailing the development of a custom design tool for the design, simulation, and optimization of discretely assembled mechanisms.

From a structural optimization perspective, discretely assembled mechanisms offer a unique advantage over conventionally fabricated devices. In contrast with prior topology optimization work, which has largely tried to approximate continuum materials with discrete voxels or ground structure elements, this work performs a discrete-valued optimization of truly discrete materials; in essence, the physical parts are the finite elements themselves.

Assembly Architecture

In our assembly architecture, two-dimensional blocks are assembled into three-dimensional structures. The blocks interlock with neighboring ones, assembling into a regular rectangular lattice structure. Four part-types are sufficient to assemble a large diversity of mechanisms and linkages: a rigid part, a single-hinged part, a double-hinged part, and a missing part (lattice vacancy). Examples of this assembly architecture are illustrated in Figure 1.

The parts with embedded degrees of freedom are produced using a laminate bonding technique in which layers are pre-machined and then thermally bonded to create a part with embedded flexures. These flexures very closely approximate revolute joints and enable very large deformations with small forces.

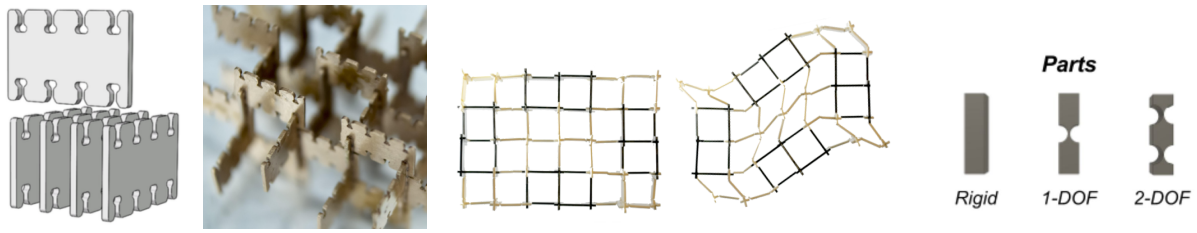

Figure 1. The proposed assembly architecture for discretely assembled planar mechanisms.

Background / Prior work / Literature Review

Mechanism Simulation

A number of different methods can be used for the simulation of mechanisms. Conventional mechanisms, which are typically made up of rigid links and revolute or prismatic degrees of freedom, are often simulated using rigid body models. While these are often analytical for mechanisms with only a handful of links and joints, numerical methods also exist. Automated analysis of arbitrary rigid body mechanisms is often complicated by the need to find closed kinematic loops to express the equations of motion. This approach is very accurate and efficient when the user has some knowledge of the mechanism being solved, but can problematic with mechanisms of changing and various topologies. Singularities, for example, are a classic problem in kinematic analysis in which the mechanism locks up and has multiple equally possible trajectories. [3]

Compliant mechanisms, on the other hand, are often modeled using finite element methods. For mechanisms undergoing large displacements, this necessitates nonlinear finite element solvers. [4]

A third approach, which is discussed by Jin et al., interpolates between these two approaches. Rather than modeling motion through the deflection of static elements, the authors describe a pseudo rigid body model in which virtual torsion springs are attached between each rigid bar. The problem of solving for the nonlinear kinematics of an arbitrary mechanism is then posed as an optimization which seeks to minimize the potential energy stored in the virtual springs. [5] Aviles et al. details a similar approach and does a more thorough treatment of the formulation with lagrange multipliers to efficiently solve the static equilibrium problem. [6]

Optimization

Prior work in the topological optimization of compliant mechanisms has largely focused on two approaches: continuum-based methods and ground structure-based methods. Sigmund shows that the formulation of the compliant mechanism problem using these frameworks is very similar to the problem of optimizing for light, stiff structures. However, rather than simply minimizing compliance, the objective is to minimize compliance while effectively translating an input force or displacement into desired output work. In this case, Sigmund outlines an objective function based on mechanical advantage between the input and output force subject to constraints including the volume of material and the minimum and maximum densities each pixel can represent. [7]

For the optimization of discrete-valued structures which aren't amenable to relaxing to a continuum problem, heuristic methods like genetic algorithms are often used. Jin et al., for example, demonstrate a genetic algorithm implementation to find suitable mechanisms given bounding constraints for their pseudo rigid body models (in this case, a displacement inverting mechanism). [5]

Methodology

Mechanism Simulation

Basic FEA formulations that make small angle approximations are not suitable for the very large displacements these mechanisms are capable of. On the other hand, rigid body linkage modeling often suffers from the necessity to identify closed kinematic chains, which can complicate the analysis of

mechanisms that are generated without human oversight. The analysis of the mechanism kinematics should be robust to under- and over-constraint conditions.

Given the prior work, I implemented a few different approaches for comparison before ultimately focusing development on a nonlinear plane-frame finite element solver.

I first implemented an approach similar to that of Jin and Aviles, constructing a pseudo rigid body model composed of rigid beams connected at nodes via torsional springs; I then used a COBYLA (constrained optimization by linear approximation) routine to solve for the minimum potential energy configuration. While this approach is robust and is well suited for very large deformations, I found it to be too slow, requiring upwards of 40 seconds to solve for structures composed of only 20 nodes. This approach also does not offer insight into the structural performance of the mechanism, which is likely of importance to the designer.

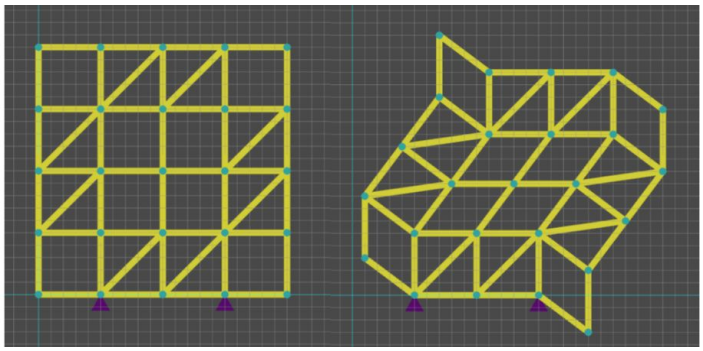

Figure 2. An example result using the pseudo rigid body formulation.

To gain insight into the structural performance of these mechanism assemblies, I developed a plane-frame finite element solver. This solver implements the classic displacement formulation in which nodal displacements are solved for by inverting a system stiffness matrix and multiplying it by a vector of external forces. The plane-frame model takes into account the axial forces, shear forces, and bending moments of the finite elements (and is depicted in Figure 4). This problem can be solved quite efficiently and works well for modeling mechanisms with small deformations. Critically, this method also provides insight into how forces are transmitted through the mechanism, which is useful for formulating the optimization.

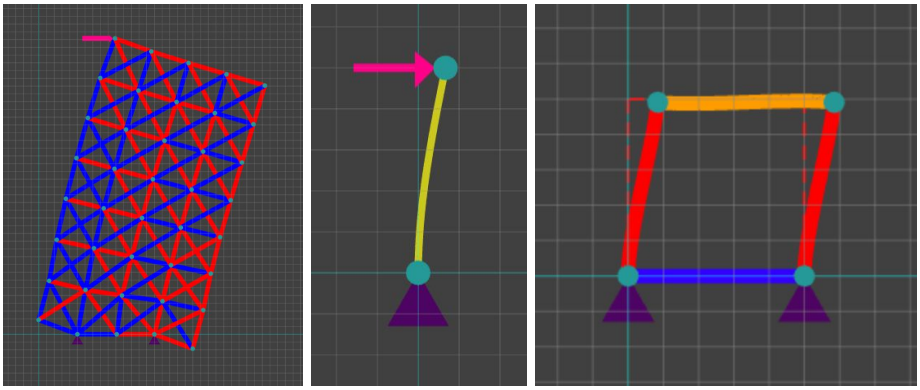

Figure 3. Example results from axial direct stiffness solver (left) and plane-frame finite element solver (middle and right)

By incrementally updating the stiffness matrix and repeating the linear plane-frame solve, this method can be extended to enable the simulation of large deformations. A number of methods exist for extending a linear finite element method for large-deformation nonlinear analysis; the most basic of these is the linear-incremental method which simply divides up an applied force into small linear steps and updates the stiffness matrix at each step (taking into account axial loads). [8] This method assumes that the axial force in the beams remains constant through the step, which is a fair assumption if the step size is small. The downside of this approach is that in can be computationally intensive, particularly if the system is composed of many elements, and the inversion of the stiffness matrix takes a non-insignificant amount of time. This approach also potentially suffers from accruing inaccuracies: since a linear approximation is made at each step, if the step size is too large, or the number of steps is large, the solver will diverge from the true solution.

An alternative approach is iterative at each step. Approaches like this, including Newton's method, update the system stiffness matrix throughout the step. While each step is more time intensive, large step sizes can be used. Because of the expense of recomputing the system stiffness matrix multiple times within each step, a modified approach is often used (called the modified Newton method), which uses the initial stiffness matrix throughout the step but iterates to converge on the same solution.

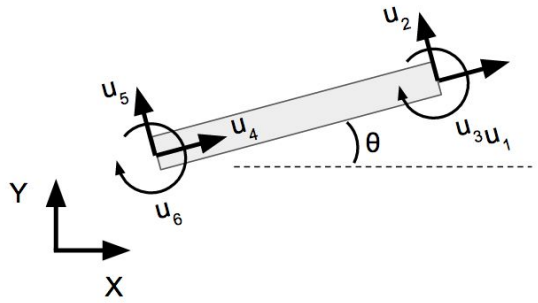

Figure 4. Planar frame element with six degrees of freedom.

Finally, if the designer cares about the dynamic performance of the structure or mechanism, the above linear-incremental method can be replaced with a more physical integration of the small linear steps: a structural-dynamic solution. In a nonlinear dynamic formulations, rather than solving for equilibrium directly, forces are converted into nodal accelerations which are integrated into displacements through a mass matrix. This method allows for incremental accumulation of nodal accelerations, avoiding the need to assemble a system stiffness matrix at each timestep. [9] This kind of method is particularly attractive for javascript and WebGL based applications since it enables the ability to parallelize the solve through the use tools such as WebWorkers and GL shaders.

I ultimately focused my work on implementing the basic linear incremental solver. While this method may not be as accurate as others, it converges quickly and the solutions are within an acceptable error tolerance for quantitative analysis of the mechanisms. Additionally, since the solution is physical at each timestep, the deflection of the mechanism can be animated while it is solved, giving the user critical feedback about how the solve is progressing.

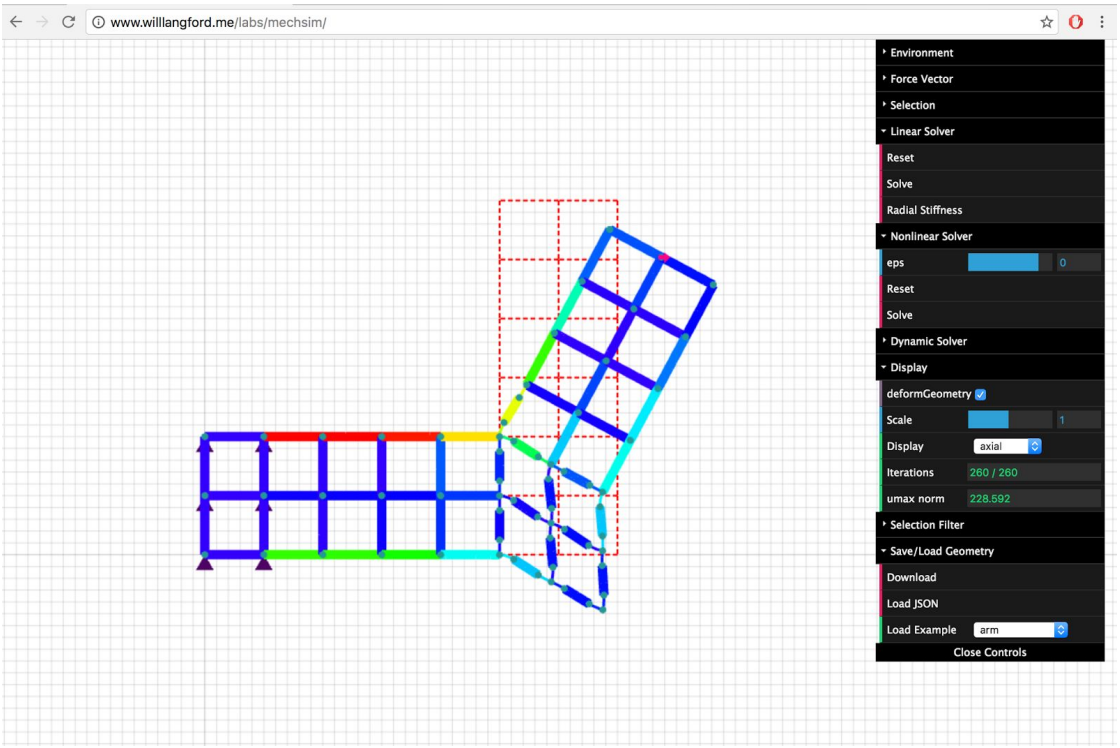

Figure 5. The design tool showing large deformation analysis of a mechanism.

Optimization

A major challenge of this work is in efficiently formulating the discrete optimization. Because the search space is discrete, there is no useful gradient that can be used to steer toward a better objective function. In topology optimization, this is usually dealt with by relaxing the problem and letting the design variables take on continuous values. However, since the parts used in my assembly framework have very rigidly defined degrees-of-freedom, it is unclear whether a relaxation will be suitable in this case.

Instead, I propose a heuristic optimization approach which uses beam-force information from the finite element solver to aid in identifying when one part-type should be substituted for another. The optimization would proceed interactively, enabling users to evaluate metrics such as on/off-axis stiffness (discussed in results) at various configurations of the mechanism.

As an example, consider the design of large displacement linear degree-of-freedom. The problem can be formulated as wanting to maximize the displacement of the loaded nodes (while maintaining a high-degree of stiffness off-axis). By sequentially loading the structure and then replacing elements with an update rule, the mechanism will converge to one that maximizes displacement.

An example update ruleset could include:

- Elements that are in (mostly) pure tension or compression above a certain threshold are removed.
- Elements that undergo positive or negative bending are replaced with a single-DoF part
- Elements that have antagonistic moments (nodal moments have the same sign) are replaced with a two-DoF part.

Figure 5 depicts this progression, using the design of a linear parallelogram linkage as an example case study. The first frame depicts the problem domain and boundary conditions. Once the loading is applied, the visualization of the shear and moment in the structure reveal that the interior parts contribute most to restraining motion in the desired direction. These parts are then replaced with 2-DoF parts and the process is repeated, replacing any parts that experience significant shear or moment.

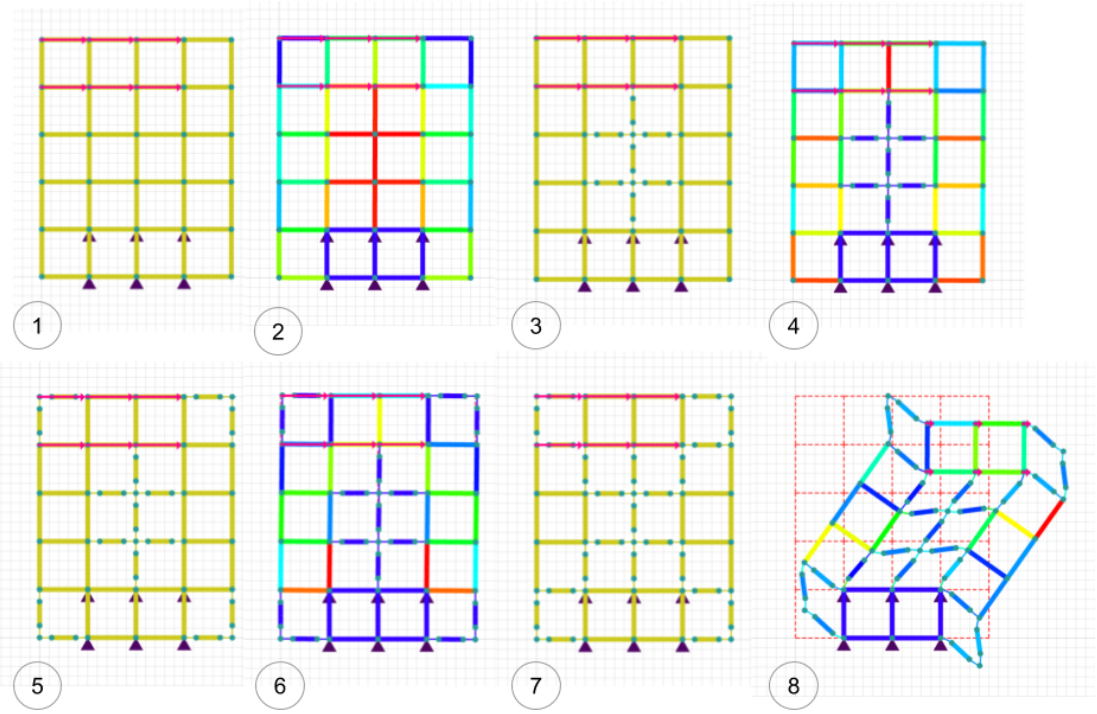

Figure 6. Heuristic interactive optimization process showing the sequential replacement of parts to allow displacement in the direction of the applied force.

As the process progresses, the mechanism displaces more and we see a sharpening of the radial stiffness of the mechanism about its operating point (as depicted in Figure 6). This indicates that the mechanism is becoming more compliant in the desired direction of motion relative to the off-axis direction.

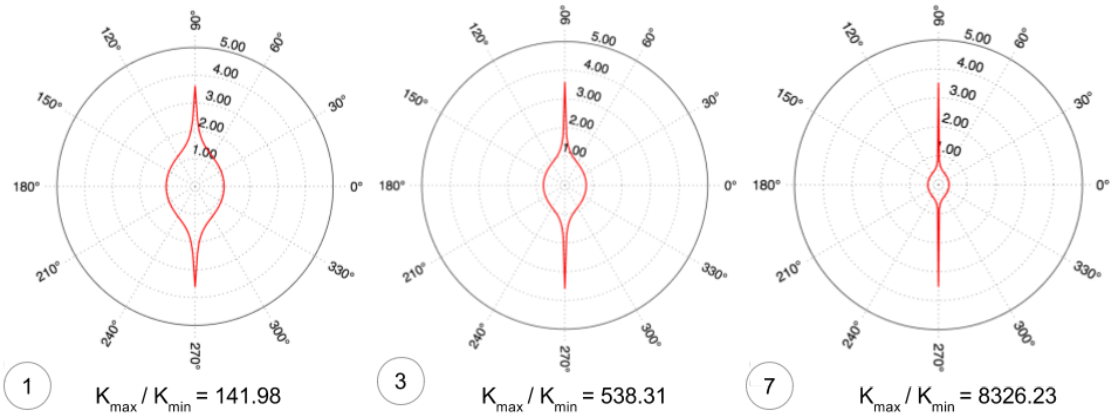

Figure 7. Radial stiffness plots calculated throughout the optimization process showing the sharpening of the on-axis vs off-axis stiffness ratio.

Results

Solver Accuracy Verification

In order to validate the accuracy of my solver, I performed an experiment with a known geometry with which I could compare my results with other commercial FEA solutions as well as analytical results. I apply 100N to the end of a 100mm long beam element. The beam element has a base of 20mm and a thickness of 2mm, with a Young's modulus equal to that of of a standard bronze alloy (97 GPa).

I chose Autodesk Fusion 360's NASTRAN nonlinear solver as a benchmark for it's legacy and stability. [10] I also compare the my solver with analytical results derived from elliptic integrals. [4] All three results are within 5% of each other, with my solution indicating a deflection just slightly less than the nonlinear FEA solver, but slightly more than the elliptic integral solution. The difference in these results can be explained by tendency of iterated linear approximations to exaggerate deformations relative to the continuously integrated solution.

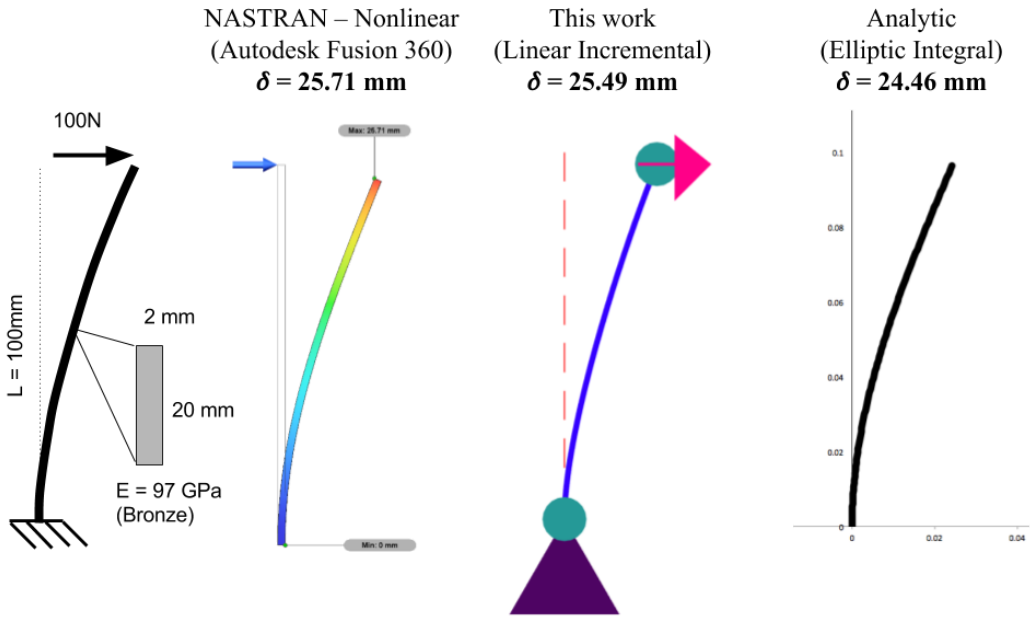

Figure 8. The solution for a loaded single beam element is compared with commercial FEA and analytical results.

Nonlinear Finite Element Mechanism Solver

I have implemented a nonlinear finite element solver to calculate the deformed configuration of the discretely assembled mechanisms. The solver uses the linear-incremental formulation with user-controllable error-tolerance. The solver is fast enough to solve mechanisms with ~50 elements with very large displacements in ~20 seconds.

Naively implemented, this kind of solver is incredibly slow. My first implementation ran far too slowly to be useful as an interactive design tool, but through successive refinement of the core solving loop, I was able to achieve a solving speed that enables fast (enough) analysis of large numbers of discretely assembled mechanism-parts over large deflections.

The two main computational expenses in solving the large-displacement problem are in assembling the system stiffness matrix and solving the system. The system stiffness matrix is inherently sparse and symmetric. This is especially certain in my case because of the topology of the discretely assembled mechanisms: there are no beam elements that span outside of their local neighborhood. This trait can be exploited to more efficiently assemble and solve the system by using sparse matrix data representations. These representations don't store zero values and instead represent the matrix by separate vectors containing the values and their respective indices. Figure 9 depicts a typical system stiffness matrix in which each black square represents a non-zero value. The system has 210 degrees of freedom with 1244 nonzero elements out of 44100.

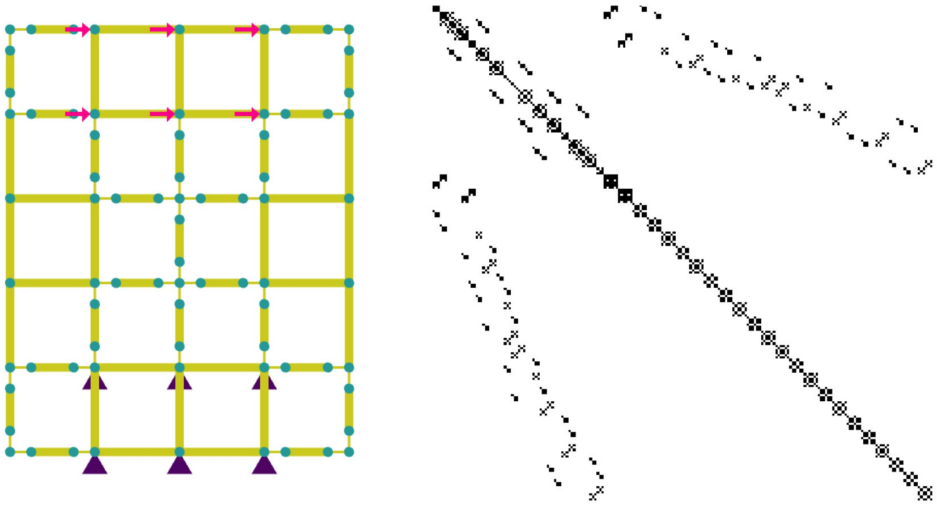

Figure 9. System stiffness matrix for parallelogram linkage example.

This sparse representation greatly increases the speed of both the assembly and the solving of the system. A further speed increase can be achieved by solving the system of equations in a thoughtful way. Inverting a large matrix is a classically computationally expensive task. Instead, this system can be solved by performing an LU-decomposition whereby the matrix is first triangularized into upper and lower triangular matrices and then solved by forward and back substitution.

Even with these enhancements, there is still a fundamental bottleneck with this formulation of the problem in that the whole system stiffness matrix needs to be updated at each time step. In my current implementation, of the time spent looping through linear solves, approximately 52% of the time is spent updating and assembling the matrix while only 38% of the time is spent actually solving the system (the remaining 10% accounts for updating geometry to be rendered).

Figure 10 shows the scaling of the solve speed as a function of the number of degrees of freedom in the system. Systems with more than 1000 degrees of freedom (~300 nodes) slow down significantly and are difficult to model with my current implementation.

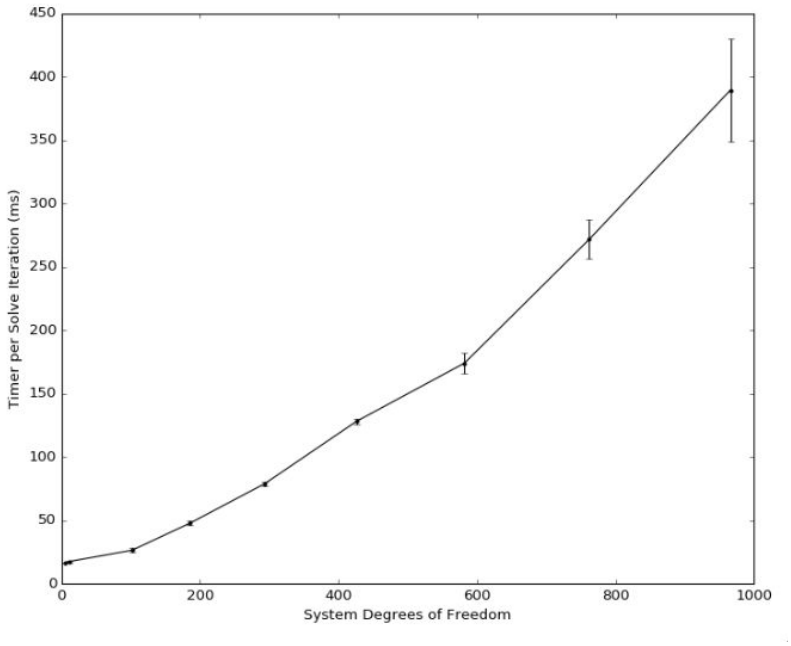

Figure 10. System stiffness matrix for parallelogram linkage example.

The allowed tolerance of the solver also contributes to the speed at which a nonlinear solution can be found. Here, 10^{ε} represents the maximum allowed deflection per incremental step. By default, I set ε=0, such that the maximum displacement per step is 1mm (a 1% displacement relative to beam length). Figure 11 shows the total accrued position error with respect to displacement of a shearing mechanism with various values of ε. With ε=0, the total position error is necessarily less than 1%, which I deemed sufficient for the purposes of my simulation.

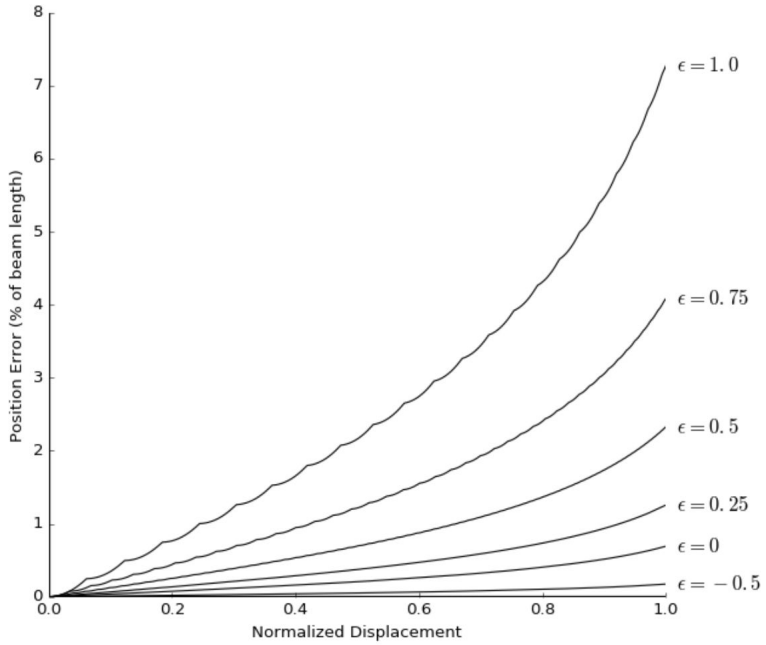

Figure 11. Nonlinear solver accuracy with varying convergence tolerances.

Interactive Design and Simulation

With the goal of creating a tool that enables exploration of the discrete mechanism design space, I have been developing the mechanism modeling tool using Javascript and WebGL in the browser. While Javascript is not traditionally thought of as a language that is especially well suited for numerical computing, its speed, ease of development, portability, and interactivity make it a good choice for this design tool.

I have implemented a number of features to allow users to interact with discretely assembled mechanisms and explore the design space. With draggable window selection tools, the user is able to add and remove geometry to define the problem bounds. Additionally, the user is able to fix or un-fix nodes, apply forces, and change part-types using this selection tool. Informational tooltips displaying the axial and shear forces as well as the moments in each beam element allow the user to visualize the load paths in the mechanism. Together, all of these tools enable a range of mechanisms to be modeled. A few examples are pictured in Figure 5.

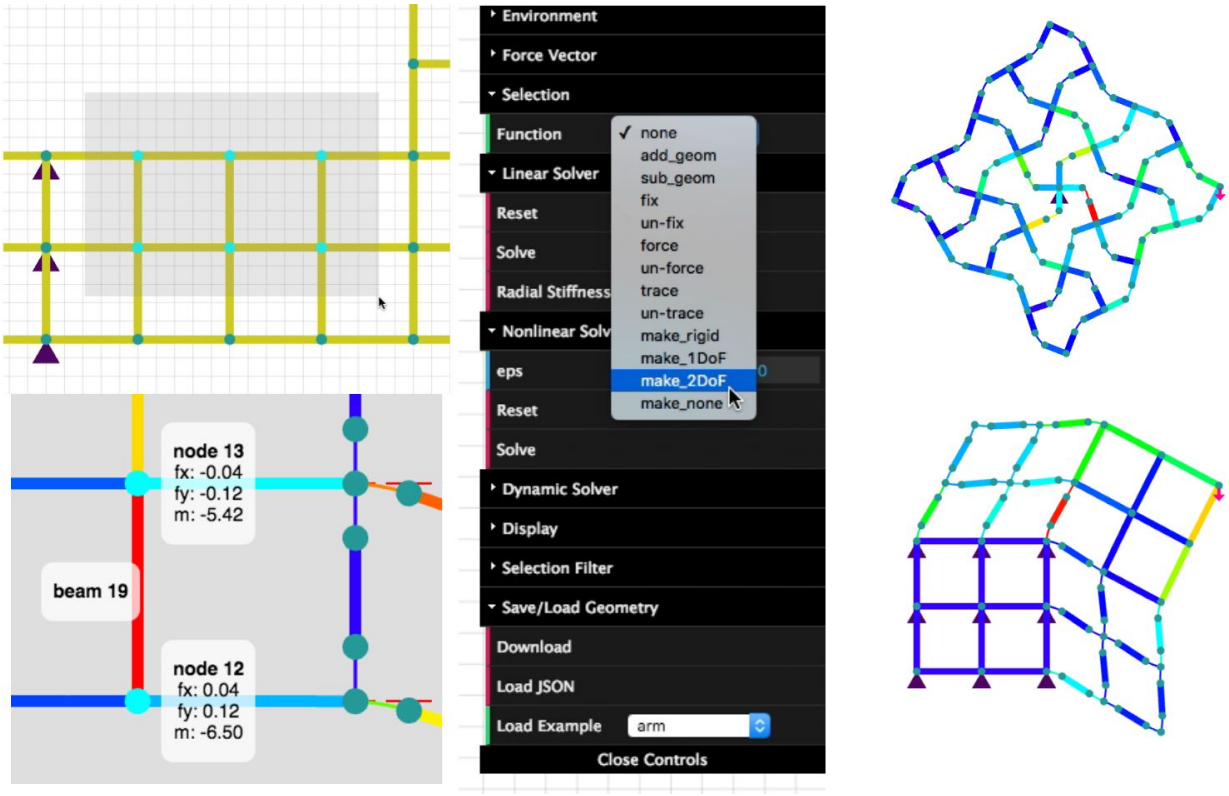

Figure 12. Interactive features enable the creation and simulation of a variety of discretely assembled mechanisms.

Radial Stiffness Plot

Flexural mechanisms often need to be designed with low stiffness in one axis (to enable motion) while still resisting forces in all other axes. Furthermore, it's often the case that a mechanism which is resistant to off-axis loads in its initial configuration is much less so once it is has been displaced by a large amount. To enable designers to interactively play with mechanisms and keep this constraint in mind,

I implemented a radial stiffness plot. A unit force is swept in a full circle about a node of interest and the resulting deflection of the node is measured and used to generate a measure of compliance in each direction. The figure below shows an example plot overlaid on the geometry. The plot clearly reveals that the stiffest direction is in line with the fixed supports and that the mechanism is most compliant in the direction orthogonal to that.

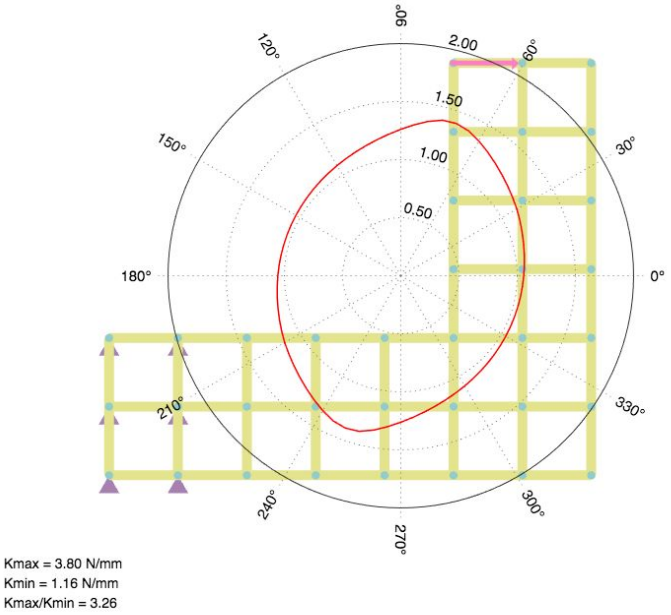

Figure 13. A radial stiffness plot overlaid on the physical geometry shows the direction of most and least compliance.

The radial plot tool can also be used to track how the direction of most compliance changes throughout the stroke of a mechanism. Figure 14 demonstrates an example of this in which a shearing mechanism is displaced, resulting in a radial stiffness plot that rotates (pointing the stiffest direction toward the fixed supports) and becomes less sharp (indicating that ratio of on-axis to off-axis stiffness has been reduced).

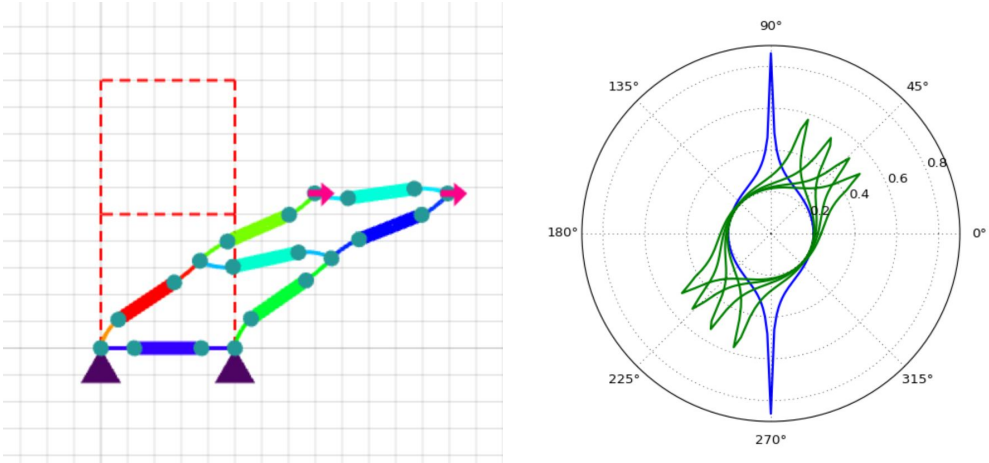

Figure 14. Radial stiffness of the mechanism (left) plotted at five discrete steps throughout its trajectory

Case Study – Optimizing the Design of a Large Displacement Linear Stage

With the parallelogram linkage explored earlier as a starting point (depicted in Figure 6), I would like to synthesize a mechanism that can constrain linear motion over a large range. This means that it has to have a high ratio of off-axis stiffness to on-axis stiffness and must produce straight line motion over a large displacement (~1 beam length). While the parallelogram linkage achieves the former, it fails with the latter: displaying a large off-axis parasitic displacement as well as a rotated radial direction of maximal compliance.

Given these objectives, I used the interactive features of this design tool to model a number of different candidates and used the rapid simulation and feedback to narrow the search. After a number of iterations I converged upon a solution which improves upon the design of the parallelogram linkage. The result is a mechanism that does not exhibit the same sharp stiffness directionality in its initial state, but maintains the directional stiffness throughout its stroke. When loaded with an equal force on-axis and off-axis, it displaces five-times more in the desired direction.

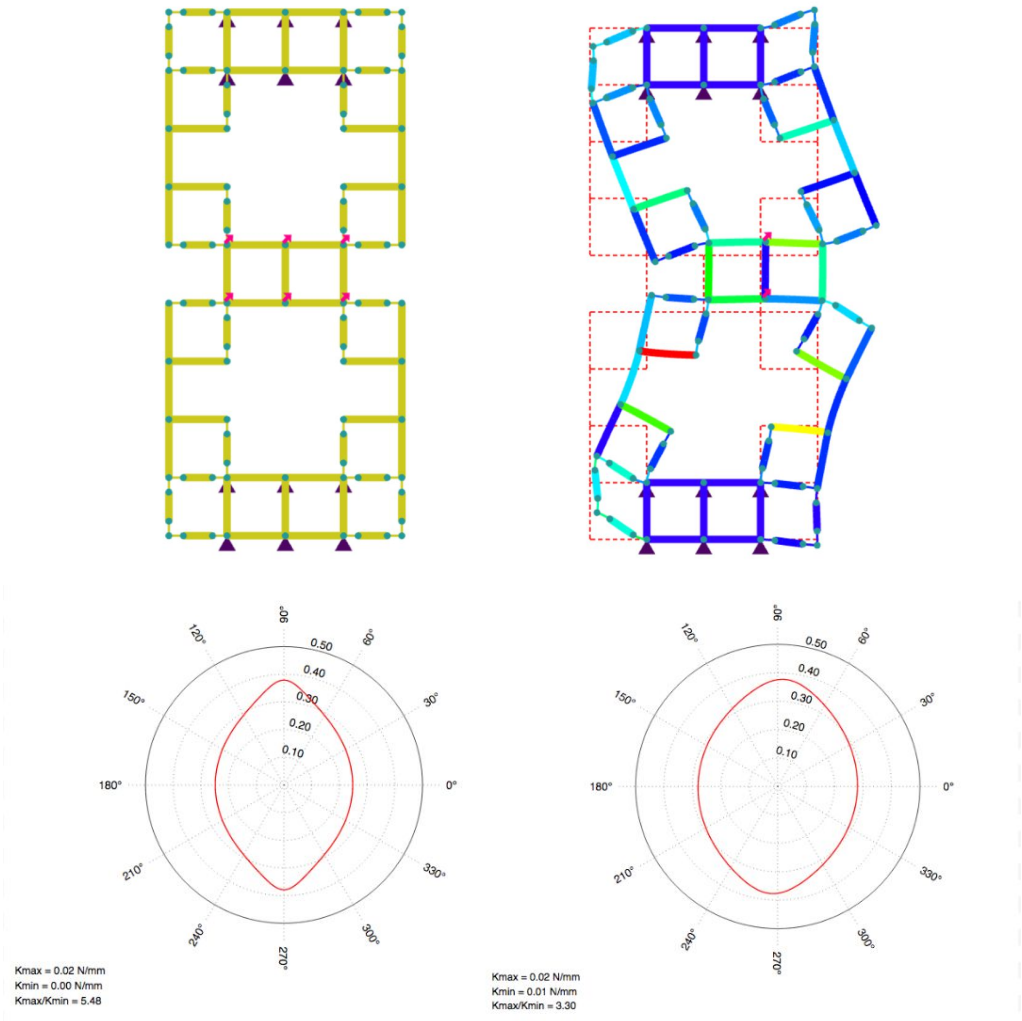

Figure 15. An optimized linear displacement mechanism is able to resist off-axis loads while maintaining compliance on-axis.

Conclusion

In this work, I aimed to create a design tool for discretely assembled mechanisms that can aid in developing a user's intuition of the discrete design space. I developed a nonlinear finite element plane-frame solver capable of quickly simulating discrete mechanism problems. Since the discrete design space is not well suited to traditional gradient based optimization methods, I developed an interactive heuristic for sequentially replacing parts to maximize displacement in a particular direction. Additionally, I developed a suite of interactive design tools that enable users to modify geometry and boundary conditions as well as visualize forces and displacements of the mechanisms. Finally, I developed a radial stiffness plot as an indicator that is especially well suited for the visualization the directionality of large displacement compliant mechanisms.

As future work, I would like to explore alternative solver implementations. I think solutions that do not rely on the assembly of an overall system stiffness matrix are particularly intriguing for the possibility of parallelizing the solve and enabling GPU acceleration. This would enable the simulation of much larger mechanisms and is a prerequisite to porting this design and simulation tool to three-dimensions.

Appendix:

All of the source code for this project is hosted on github:
https://github.com/langfordw/structural_optimization/tree/master/threejs_environment

A live version can be experienced at: http://www.willlangford.me/labs/mechsim/

References:

[1] W. Langford, A. Ghassaei "Automated Assembly of Electronic Digital Materials," *ASME MSEC*, 2016, pp. 1–10.

[2] W. Langford, A. Ghassaei "Hierarchical Assembly of a Self-Replicating Spacecraft," *IEEE Aerospace*, 2017, pp. 1–10.

[3] L. Cao, A. T. Dolovich, A. L. Schwab, J. L. Herder, and W. (Chris) Zhang, "Toward a Unified Design Approach for Both Compliant Mechanisms and Rigid-Body Mechanisms: Module Optimization," *J. Mech. Des.*, vol. 137, no. 12, p. 122301, 2015.

[4] W. C. O. Mara, R. B. Herring, and L. P. Hunt, "Handbook of Compliant Mechanisms."

[5] X. Z. Jin, Mohui, "A numerical method for static analysis of pseudo-rigid-body model of compliant mechanisms"

[6] R. Avilés, et al. "A procedure based on finite elements for the solution of nonlinear problems in the kinematic analysis of mechanisms"

[7] O. Sigmund, "On the Design of Compliant Mechanisms Using Topology Optimization"

[8] A. Chajes and J. E. Churchill, "Nonlinear Frame Analysis by Finite Element Methods," vol. 113, no. 6, pp. 1221–1235, 1987.

[9] D. L. Logan, E. Veitch, C. Carson, K. R. Burrell, V. Gould, E. Wagner, D. L. Logan, E. Veitch, C. Carson, K. R. Burrell, V. Gould, and E. Wagner, *A First Course in the Finite Element Method Fourth Edition*, vol. 147, no. 3. 2007.

[10] "Nastran | Stress Analysis Software | Autodesk", *Autodesk.com*, 2016. [Online]. Available: http://www.autodesk.com/products/nastran/overview. [Accessed: 20- Dec- 2016].