Integrating Grasshopper and Matlab for Shape Optimization and Structural Form-Finding of Buildings

Nixon Wonoto[1] and Vincent Blouin[2]

[1]Clemson University, nwonoto@clemson.edu
[2]Clemson University, vblouin@clemson.edu

Corresponding author: Nixon Wonoto, nwonoto@clemson.edu

ABSTRACT

Structural optimization is a promising form-finding technique for the architectural schematic design phase of buildings. However, most published case studies tend to reduce practical design and analysis problems into simplified theoretical models in which materiality, geometry and loading conditions are over-simplified. This paper presents a structural optimization case study that allows the inclusion of complexity using Grasshopper and Matlab. The optimization process includes an automated update of structural size, shape and topology, material properties, and loading conditions. The method is applied to a parametric skyscraper design problem to demonstrate the use of Grasshopper to expedite the implementation of a complex problem and thereby facilitate the architectural schematic design phase.

Keywords: Structural optimization, Grasshopper, FEA, GA, Form-finding
DOI: https://doi.org/10.14733/cadaps.2019.1-12

1 INTRODUCTION

Structural optimization and form-finding use structural performance as the primary driver for selecting optimal candidates from the design space. Structural optimization was often exclusive to the engineering field but has recently swayed into the field of architecture. In line with the emergence of design computation in architecture, there has been an increasing number of publications focused on form-finding techniques in the architecture schematic design phase of buildings. Some of them include automated form-finding processes for designing large concrete roofs [18], domes [10], Voronoi's cell structures [6], trusses [5], tessellated structures [21], and Miura origami fold retractable roofs [22].

In spite of these advancements, there has not been many architectural structural optimization frameworks that incorporate methods that allow easy consideration of the complexity associated with realistic design problems. More specifically, design difficulties are related to the dependency on design variables of the loading conditions, material properties, size, shape and topology properties, and the utilization of multiple structural types. Thus, real design and analysis problems are usually reduced to over-simplified exercises.

Even though one may argue that the architectural schematic design phase does not require detailed structural definition, the authors believe that over-simplification is partly responsible for the renowned difficulties of the collaboration between architects and engineers during the building design process. We also suspect that the need for simplification is due to the difficulty of programming complex processes with the commonly used

software. The premise of this paper is to overcome these issues by encouraging the use of semi-automated CAD processes that capture realistic structural behaviors to reduce the programming burden. The paper presents a versatile method for efficiently carrying out architectural structural optimization that includes the complexity of a typical engineering problem in designing complex structures. In this problem, an analytic form of the objective function is not known but is evaluated through performance analysis. Such a problem is classified as a black-box simulation-based optimization problem [12],[19]. The method integrates Grasshopper [9] and Matlab [14] for carrying the structural optimization procedure.

Grasshopper is an algorithmic modeling plugin for the commercial CAD software Rhinoceros [17] used extensively by architects interested in structural form-finding. Grasshopper has recently become increasingly prevalent among architects due to its convenient visual programming interface for creating intelligent parametric models that can be integrated into performance analysis and optimization components. Also, Grasshopper facilitates interactive updates between the visual programming interface and the plot of the geometry which is uncommon in most architecture and engineering modeling software. That is to say, in the context of the architecture and engineering fields, Grasshopper, in many cases, can be used to conveniently deal with complex modeling and preliminary analysis tasks graphically in which most other software has to rely on writing programs to tackle the same issue. Some analysis and optimization plugins, such as Karamba [11] and Galapagos [7] for Grasshopper, were recently developed to allow architects to perform interactive structural optimization of highly complex models efficiently on a single platform. Some researchers use these programs to structurally optimize trusses, frames and modular structures [3],[8]. Unfortunately, the utilization of Grasshopper has not much proliferated among engineers, whilst having a common language is crucial in a collaboration process. Also, a secondary software is most likely required for performing a multitude of structural analyses. The method presented in this paper creates an interoperability between Grasshopper and Matlab for conducting structural optimization. The advantages of forming this interaction between the two software are to allow more versatility in configuring the optimization setup and to allow the use of a parallel computing environment. Facilitating the connection between architectural software (Grasshopper) and engineering software (Matlab) will then improve the architect-engineer collaboration.

The method is presented in Section 2 and then implemented in Section 3 with the example problem of the structural design and optimization of a skyscraper. The method and the results are discussed in Section 4 and Section 5 concludes the paper.

2 METHOD

The method consists of three interconnected parts: Grasshopper to generate automatically the structure, a finite element analysis (FEA) code [2], written in Matlab, to analyze the structural deformation and stresses, and a genetic algorithm (GA) [15], also written in Matlab, to optimize the structure.

The integration between Grasshopper, the customized FEA, and the customized GA optimizer is carried out in an iterative manner as shown in Figure 1. Grasshopper is used to generate a parametric model. It then manages and updates the structural analysis FEA input file based on the values of the design variables passed by Matlab during the optimization process. The output from Grasshopper includes nodal coordinates, element connectivity, and all information related to the structural analysis, including the loading conditions, boundary conditions, material properties, section property, element types, and type of analysis. The advantage of managing the structural analysis setup in Grasshopper is to have full control over the analysis configuration. Instead of predetermining the values of the analysis configuration, the method adjusts the analysis format automatically depending on the given geometry.

2.1 Grasshopper

Grasshopper is used as a form generator to create the building with all the information necessary to define all aspects of the structure. In addition, the model is written in a way that makes it easily accessible to architects. In the Grasshopper environment, several components shown in Figure 2 were developed to model the structure and facilitate the integration between Grasshopper and the customized FEA code and the GA optimizer in Matlab. Figure 2 shows five components and a “user component”. The user component differs from components 1 to 5 by the fact that it is the only component that needs to be created or altered by the user of this software who would define the size, shape, and topology of the building of interest. The user component includes the definition of the CAD model of the building structure while components 1 to 5 include all the commands specific to the analysis and optimization of the structure.

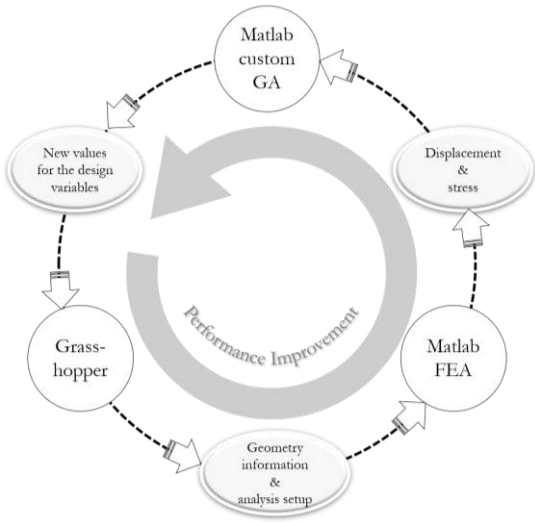

Figure 1: Workflow of the presented structural optimization method.

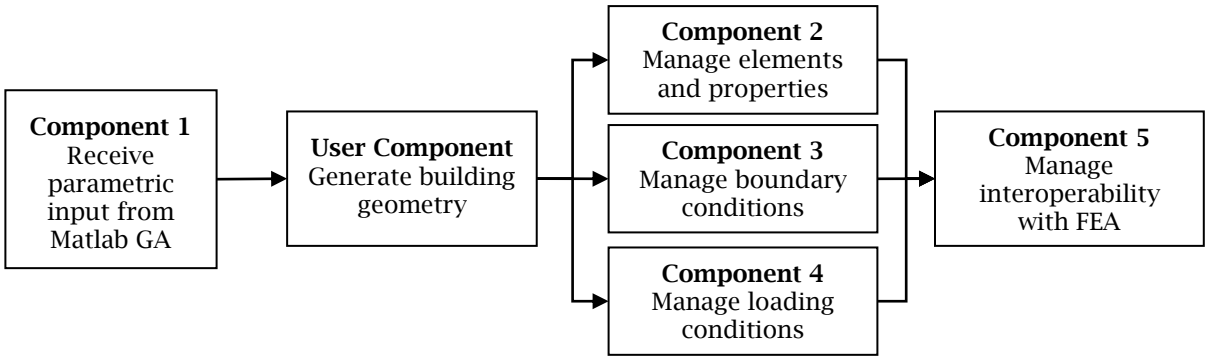

Figure 2: Grasshopper components for managing interoperability.

Figure 3 shows an excerpt of the entire Grasshopper code, which is a cluster of components that consist of a multitude of interconnected modules made of predefined functions and/or scripts. Each module has one or more input and output that are then connected to other modules. In order to explain how the building is translated into the CAD model in Grasshopper, each component is described in more details as follows.

Component 1 takes the input from Matlab for the parametric description of the building. The list of parameters include: number of floors (N), discretization parameters of each floor (noted n and p), twist angle of each floor (alpha), floor-to-floor height (h), shape of floors (circle, oval, rectangle), radii of bottom, mid, and upper floors, types of elements (truss or beam), material properties (Young's modulus of elasticity, Poisson's ratio, density), cross-sectional properties (moments of inertia, area), wind direction and velocity, and wind force parameters define by code. Note that among this list of parameters, only a few are selected as design variables for the optimization. In the example presented in Section 3 of this paper, the four design variables are the radius of the floor located at the mid-height of the building, the radius of the top floor, the total twist angle of the tower, and the number of floors. All other parameters are fixed for a given design problem.

As mentioned above, the user component includes the parametric description of the structure (shown in Figure 4) and is executed using the following steps: (1) Define N floors separated by a distance h (floor-to-floor height). In this example, the periphery of each floor is defined as a circle. (2) Discretize the circle into an n-gon. In the example, n = 6 such that each floor is a hexagon, as shown in red in Figure 4. (3) Define the twist angle of each floor, alpha. In this example, alpha is assumed to be the same for each floor. (4) Define the radii of the

bottom, mid, and top floors and calculate the radius of all floors as interpolated values of bottom, mid, and top floors. (5) Discretize each side of each floor into p smaller segments. In the example, p = 4. As a result, each floor is discretized into 24 vertices connected by 24 segments. (6) Connect adjacent floors by creating vertical columns at the hexagon's vertices shown in green and vertical and diagonal bracings shown in blue in Figure 4. As a result of this parametric description, a virtually infinite number of structures with similar properties can be defined under this class of buildings.

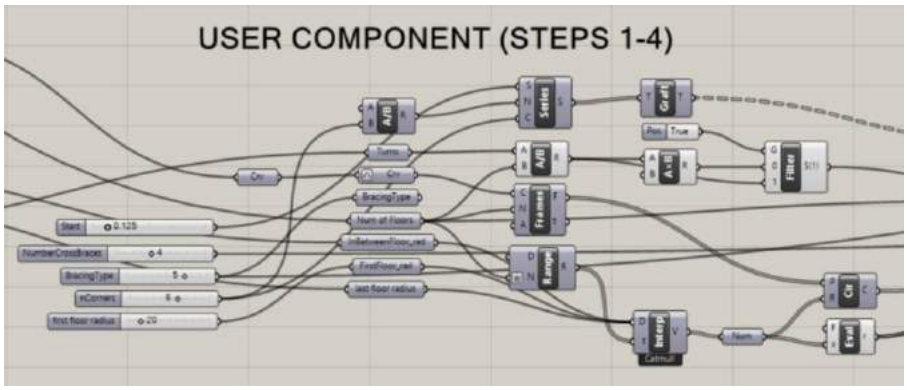

Figure 3: Excerpt of CAD model in Grasshopper environment.

Component 2 converts each segment into either a truss or a beam element and defines the cross-sectional properties (i.e., moments of inertia and surface area) and material properties (i.e., modulus of elasticity, Poisson's ratio, and density) for each element.

Component 3 manages the boundary conditions by extracting the vertices of the first floor and fixing their translational and rotational degrees of freedom.

Component 4 manages the loads applied on the structure with the following steps. (1) Calculate the weight of each floor based on its thickness, radius, and density of the material and distribute the weight of the floor slab to the n*p nodes of that floor. (2) Calculate the dead weight of each structural element and distribute equally to each node of that element. (3) Knowing the wind direction, calculate the wind loads based on codes as three-dimensional force vectors applied to the exterior surfaces of the building. (4) Convert the surface forces to concentrated force vectors applied to the n*p*N vertices of the structure. In this example, assuming 22 floors, a total of 528 force vectors are created and distributed throughout the structure.

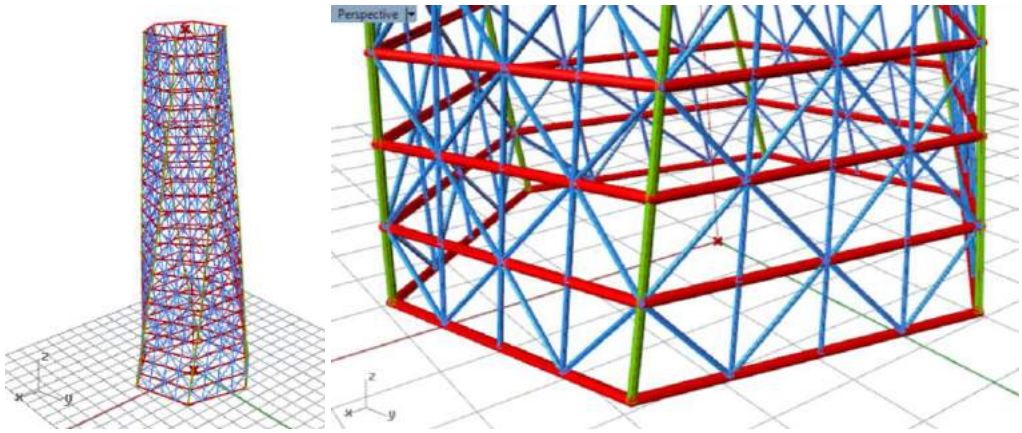

Figure 4: Grasshopper CAD model of structural elements showing the periphery of the floors (red), the columns (green), and the vertical and diagonal bracings (blue).

2.2 Customized Matlab FEA Code

A customized FEA code was programmed in Matlab using the direct stiffness approach [4]. The code can deal with any combination of three-dimensional truss and frame systems. The code was made versatile for dealing with problems with multiple loading conditions and section properties. The maximum nodal displacement and maximum elemental Von Mises stress are calculated and can be used for constructing constraint or objective functions for the structural optimization process. The code was validated using the commercial FEA software Abaqus [1]. Using the customized FEA is preferable over using another software, such as Abaqus, in order to save computational time when dealing with structures modeled as trusses and frames. Note that Abaqus could be used in lieu of the Matlab customized FEA for more complex structural problems that involve shells and solid elements, friction and contacts, nonlinear material behaviors and more advanced types of analysis.

2.3 Customized Genetic Algorithm

Genetic Algorithm (GA), invented by Holland in 1960, is a mechanism of natural adaptation implemented into a computer system [15]. GA is based on the principle of natural selection and genetically inspired operations such as crossover and mutation. A set of alternative designs of the structure are initially defined randomly. This set of designs is called a population of individuals or parents. At each iteration, pairs of individuals are selected and mated using crossover and mutation operations to generate offsprings, also called children that will be used as parents in the next iteration. Each individual has a fitness value correlated to the objective function of the optimization problem. The GA iterative process is designed to find structures with increasingly higher fitness values and converge towards the optimum structure.

In this particular work, a customized GA code was programmed in Matlab. The code was programmed to be versatile to handle any number of discrete and continuous design variables. Values of the design variables are encoded into binary to facilitate the genetic modification and mutation processes. Single point crossover or uniform crossover can be used to produce the offsprings. The mutation rate in the mutation process can be adjusted to further enrich the variation among the children for the next iteration. The concept of elitism is also used in which the strategy is vastly utilized to ensure the improvement of the convergence in the individuals' fitness in each subsequent iteration [13].

3 IMPLEMENTATION: STRUCTURAL DESIGN AND OPTIMIZATION OF A SKYSCRAPER

The developed structural optimization method is used for designing a twisted skyscraper, shown in Figure 5, under both dead and wind loads. The twist of the building is assumed to be the most important architectural feature. The case was designed such that the skyscraper employs the exterior diagrid for the braced tubular structural system, using the combination of frames and trusses at the outside perimeter as the primary structure. Beam elements are positioned at each corner of the building as columns extending from the foundation to the top of the tower. Spandrel beams at each story tie the columns. The diagonal truss elements are used to increase the lateral stiffness further, allowing more spacing for the vertically spanned truss members. A dead load of floor slabs is taken into account. The additional stiffness that the floor plates provide was taken as further aspect contributing to the increase in structural integrity.

3.1 Design variables

The four design variables defined in the example problem are the radius of the floor located at the mid-height of the building (s1), the radius of the top floor of the building (s2), the total twist angle of the tower in radians (s3), and the number of floors (s4), which controls the overall height of the building since the height of each floor is assumed to be constant. The variables s1, s2, and s3 are continuous, while the variable s4 is discrete. The radius of each floors between the bottom, middle and top is defined on a smooth cubic polynomial interpolation of s1 and s2. Since the maximum stress is supposed to be at the bottom of the building, the thickness of the tubular cross-section of the structural members decreases with the floor number and depends on the total number of floors (s4). The optimization problem is taken as a mixed non-linear and integer programming problem involving changes in the number of connections and elements. Other variables such as the number of corners of the floor plan and the bracing type are fixed, but can also be used for the optimization, depending on the design intent. In this case, hexagonal floorplans are used with the bottom floor radius of 20 meters. The height of each floor is assumed to be 5 meters. Figure 5 shows several variations of the parametric model of the skyscraper.

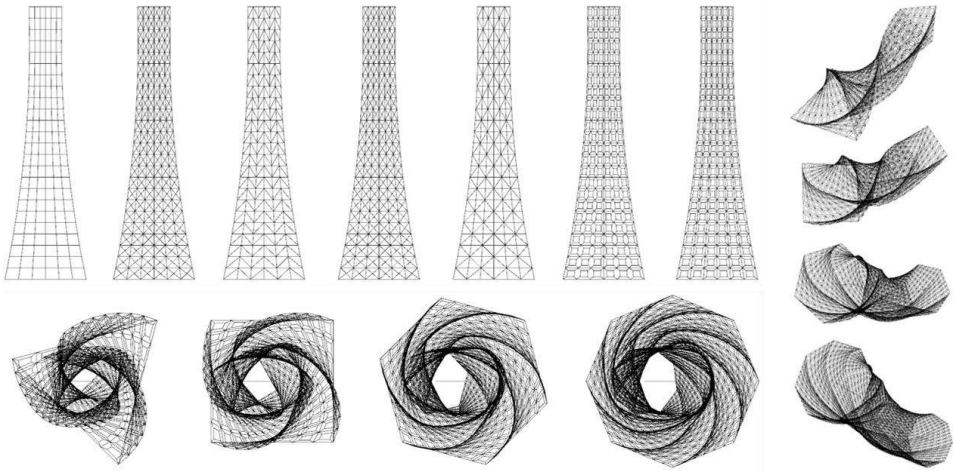

Figure 5: Variability of the parametric model of the skyscraper.

The design space is defined by the lower and upper bounds of each variable as defined in Table 1. The corresponding values of the maximum displacement, maximum elemental Von Mises Stress, and the total floor area of the building are also included in Table 1 in order to provide the order of magnitude of each performance variable. Figure 6 shows the geometry and corresponding deformation of the lower and upper bound configurations.

Boundary	S1 (m)	S2 (m)	S3 (rad)	S4 (#floors)	Disp. (m)	Stress (MPa)	Total area (m^2)
LB	4.0	3.0	0	18	0.054	32.7	4,132
UB	20.1	20.1	5.03	26	1.42	1083.9	28,281

Table 1: Lower and upper bounds of the optimization.

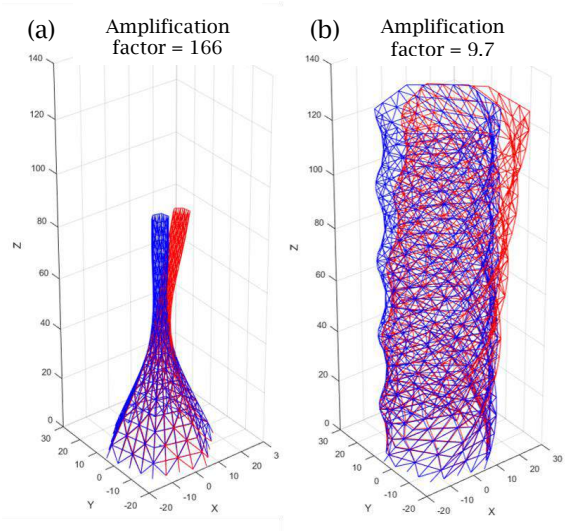

Figure 6: Displacement plots of (a) the lower bound state and (b) the upper bound state.

3.2 Loading Conditions

The structure is subjected to both dead and wind loads. The dead loads take into account the weight of the concrete floor slabs and the weight of the structural members. Thus, the dead load is a function of the four design variables. The dead load depends on the cross-section of structural elements such that the amount of material decreases with the height of each floor. As a result of this feature, the total load applied on the ground is reduced significantly without exceeding the maximum allowable stress along the structure. The dead load applied to any given floor corresponds to the weight of all floors above it and is uniformly distributed at all nodal connections of that floor.

The lateral load due to the wind pressure is a more complicated aspect compared to the dead load. When the building is twisted, each node in the structure carries a unique magnitude of the load. There are two determining factors used in calculating the pressure of the wind load, including the wind velocity pressure (qz), and the leeward and windward coefficient (cp). The value of qz is the function of the wind speed, and the height of the skyscraper or in this case the number of floors (s4). Based on the Saffir-Simpson scale, the wind speed of category three ranges from 49.6 to 58.1 m/s, simulating a structure under major hurricane [20]. In this case, the wind speed of 53.6 m/s is used. The topographic factor of 1 is used to assume that the structure is built on an unprotected site and fully exposed to the wind. A factor of safety of 1.15 is used to account for the substantial hazard to human life in the event of failure. Other than the height, larger surface area of the façade can significantly increase the resultant of the pressure loads applied to the nodes due to qz, which is a function of s1 and s2. The value of each wind coefficient, cp, corresponding to each node is a function of the twist (s3). The value of cp is divided into x and y components and tabulated into matrices that are updated at each iteration of the optimization process to give the proper loading condition corresponding to the particular design alternative. Figure 7 shows cp values at a sample design alternative of the skyscraper. There is a total of 24 sections corresponding to four bracing segments at each edge of the hexagonal floor plan.

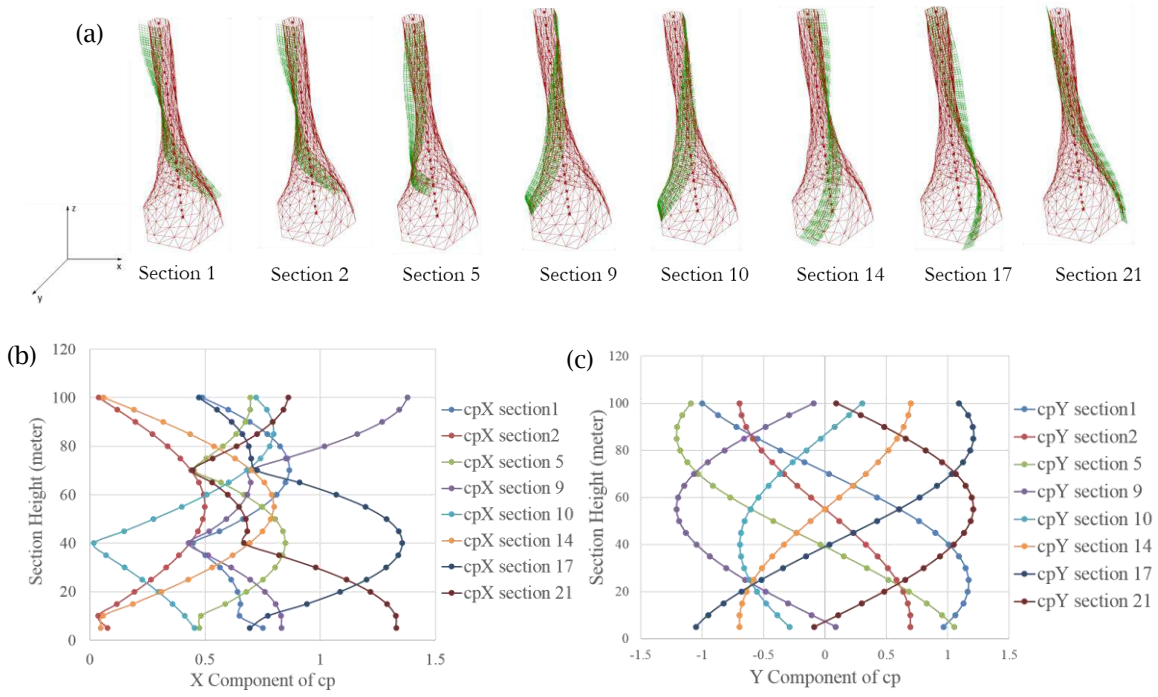

Figure 7: (a) Sampled sections of the structures at a sampled design alternative, (b) X component of CP vs. the sampled sections, and (c) Y component of CP vs. the sampled sections.

3.3 Constraint and Objective Functions

The problem was formulated to maximize the total twist (s3) of the skyscraper subjected to constraints on stresses, deformation, and total floor area. The maximum elemental Von Mises stress is restricted to be less than 250 MPa for a feasible solution. The maximum nodal displacement in the structure has to be less than 0.2 percent of the building's height [23]. The total floor area has to be between 12,000m^2 and 14,000m^2. By imposing these three constraints, both the lower and upper bound configuration listed in Table 1 become infeasible. Geometrically, as the twist (s3) increases, the building's lateral stiffness decreases because slanted peripheral columns provide smaller bending stiffness compared to the conventional vertical columns [16]. As a result, not only the maximum deflection at the peak increases but the maximum stress may increase.

The objective of the optimization problem consists of maximizing the twist angle of the building. For the design intent, aesthetically the twist should increase to the point where one or more constraints become active. Reducing the height of the tower (s4) can significantly reduce the stress and displacement contributions due to the wind velocity pressure, and reducing both s1 and s2 decreases the exposure of the façade to the wind pressure. However, simply reducing s1, s2 and s4 would eventually violate the total floor area constraint. As a result, it is predicted that the solutions that maximize the twist of the tower should locate s1, s2, s3, and s4 somewhere in between the lower and upper bounds of each design variable for which some of the constraints are nearly active.

3.4 Structural Optimization Result

The computational time of the analysis of each individual of the GA optimization process is mostly the execution time of Grasshopper and the FEA. In the particular case of the skyscraper design, Grasshopper and the FEA require an average computational time of 46 seconds and 53 seconds, respectively.

The performance of the genetic algorithm generally depends on the population size and the total number of iterations for carrying out the structural optimization. The results reported in this paper were obtained using a population size of fifty individuals and forty iterations. As shown in Figure 8, the maximum fitness value, which corresponds to maximizing the twist angle of the building increases drastically from the first to the third iteration. The further slight increase occurs in the 20th iteration. The trend line of the mean fitness value (i.e., average over the population) in each iteration increases during the first 20 iterations and then decreases slightly. The fluctuation in the mean fitness value shows that the stochastic process involving the crossover and mutation in the algorithm perform as desired.

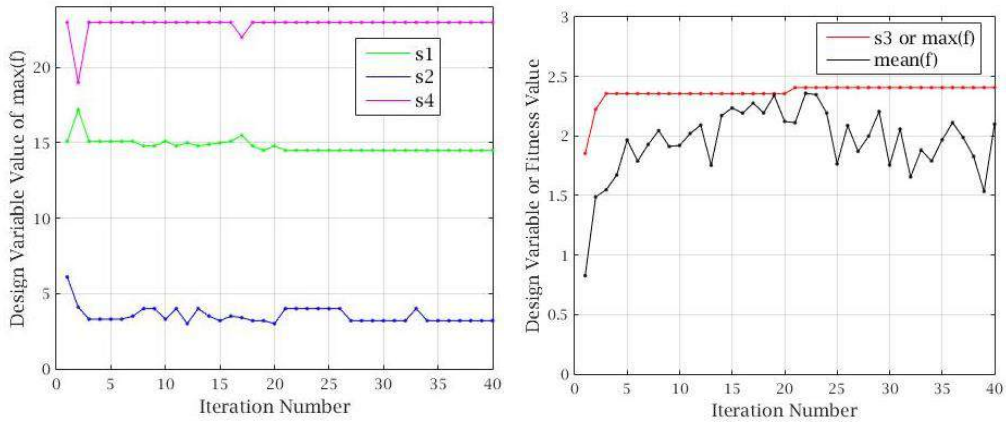

Figure 8: Evolution of design variables and fitness function during GA optimization process; (a) s1, s2, s4, (b) s3 (or max(f)) and mean fitness.

For the architectural design purpose, some feasible post-optimization individuals from some iterations can be picked and considered as selected design candidates for further architectural design development. The architectural design problem typically involves a complicated process containing several quantitative and qualitative objectives and constraints. Many performance functions related to multiple aspects such as structure, energy, and thermal comfort are quantifiable and can be included into both the sets of objectives and constraints. Aesthetic, for instance, is qualitative and subjective in nature. Thus, architectural structural optimization should

result in an improved and narrowed space of solutions instead of a single optimum. Variation of improved candidates provided by the genetic algorithm can be used to narrow down the design space, but still leave some room for further design tweaking and consideration. Table 2 shows some samples of the feasible design candidates that can be used for further design intervention.

Figure 9 shows the displacement and the load vectors of the best individual at iteration forty. Figure 10 shows the displacement and von Mises stress plots of the skyscraper design of the optimum at iteration forty. The maximum elemental von Mises stress occurs at the diagonal truss element at the third floor. Figure 11 shows the wireframe and the rendered mode of the structural elements at the optimum at iteration forty.

Iteration	S1 (m)	S2 (m)	S3 (rad)	S4 (#floors)	Disp (m)	Stress (MPa)	Total area (m^2)
1	15.1	6.1	1.855	23	0.146	221.07	13211
2	17.2	4.1	2.224	19	0.094	235.46	12296
17	15.5	3.4	2.356	22	0.124	247.82	12467.1
33	14.5	4	2.406	23	0.145	246.7	12193.4
40	14.5	3.2	2.406	23	0.133	239.28	12054

Table 2: Best individuals at selected iterations.

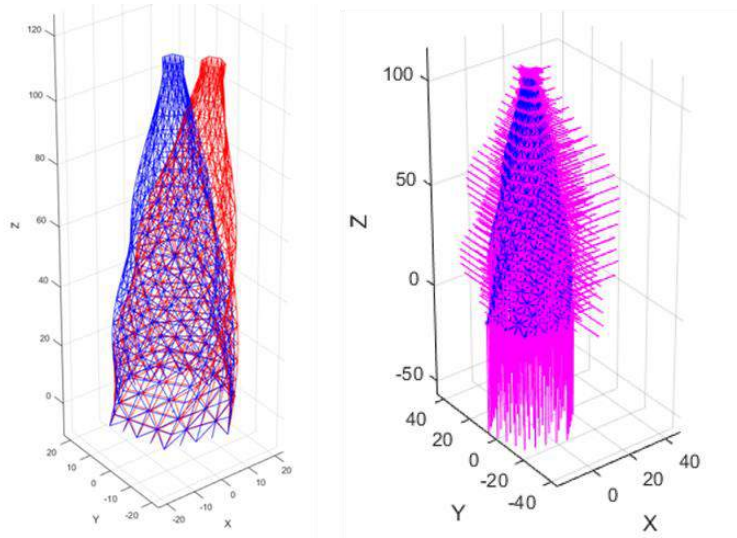

Figure 9: (a) Deformation and (b) Load vectors of the best individual at iteration forty.

Von Mises failure criterion was used to account for multiaxial loading conditions resulted from the combinations of lateral wind forces and dead loads of the floor slabs. Since the wind load is projected into the x and y component, the external force applied to each member is not a simple bending or axial load, but the combinations of shear, torsion, bending, and axial load. Those external loads generate the internal forces in each member, and the combinations of those generate both normal and shear stresses in each member. As opposed to normal stress criterion that accounts only the normal stress in the member, von Mises Stress takes into account all those stresses which are the off-diagonal stress components in the stress tensor. As opposed to only account for failure due to volumetric change, the culprit of the members' failure in this case study was suspected would be due to deviatoric stress under the multiaxial loading condition, and thus Von Mises failure criterion was selected.

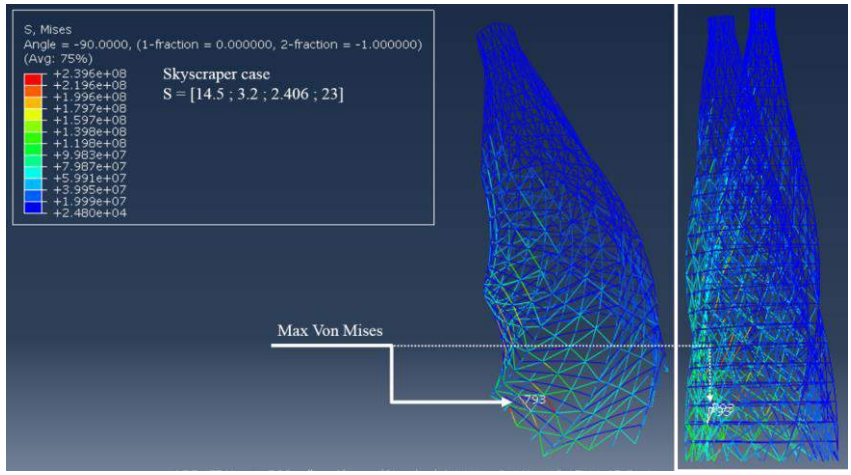

Figure 10: Displacement and Von Mises stresses of the optimum design at iteration forty.

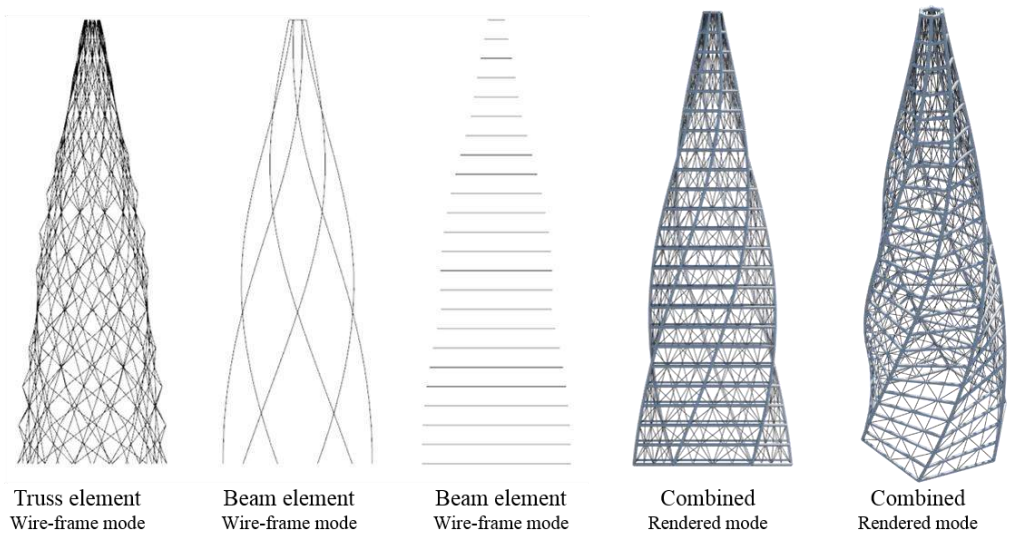

Figure 11: Wireframe and rendered mode of the optimum design.

4 RESULTS

The results of the structural optimization are consistent with the prediction, i.e., the best individuals that maximize the twist of the skyscraper are close to violating the total floor area constraint or the maximum stress constraint. Since the problem is a black-box, it is difficult to ensure that the best individual is the global optimum and it is impossible to predict whether the problem is multimodal or unimodal. An exhaustive search analysis of the design space was then performed to validate the results found by the genetic algorithm. The design space is discretized using 10 levels for s1 and s2 and 8 levels for s3 and s4. The full factorial analysis of the design space corresponds to a total of 6,400 analyses. To acquire a more accurate location of the optimum, the cosine interpolation method was used to generate smooth transitions between adjacent points within those discrete 6,400 points. Figure 12 shows that the location of the optimum found after interpolation is slightly shifted from the discrete points. The overall computational time of the exhaustive search is approximately ten times greater than that of the genetic algorithm. Table 3 shows the two best results obtained by the exhaustive search analysis.

S1 (m)	S2 (m)	S3 (rad)	S4 (m)	Disp (m)	Stress (Mpa)	Total area (m^2)
17.1	2.7	2.476	19	0.091	245.97	12005
16.0	2.0	2.564	21	0.106	248.51	12173

Table 3: Two best results obtained by the exhaustive search analysis.

Figure 12 is a graphical representation of the design space showing all feasible points and the optimum (highlighted by a red circle) among the experimental conditions of the full factorial analysis, with the variable s4 fixed as twenty-one floors. Comparing the results of the genetic algorithm and the exhaustive search analysis, the optimization problem seems to be multimodal, which means that the problem may have several local optima. Using the exhaustive search analysis, the global optimum was shown to have a twist of 2.56 rad and twenty-one floors. Despite not achieving the global optimum, the presented optimization method proves to be successful to find improved candidates in a much shorter time compared to the exhaustive search analysis, and thus is a viable form-finding technique for dealing with complex mixed integer nonlinear optimization problems during the architectural design process.

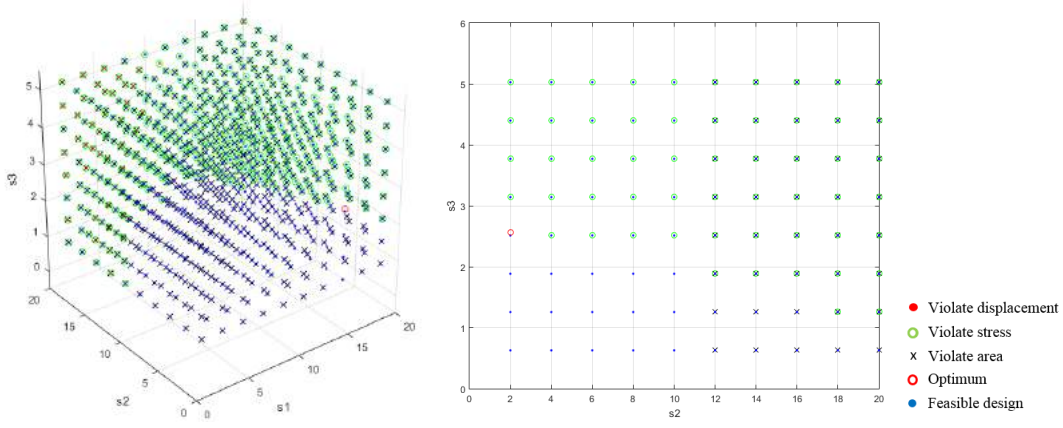

Figure 12: Graphical representation of the design space showing feasible designs and the optimum using the exhaustive search analysis: (a) s1 vs. s2 vs. s3 at s4=21; and (b) s2 vs. s3 at s1=16 and s4 = 21.

In terms of scalability of the method, potential issues may arise when increasing the size of the structure, the number of design variables, and the number of objective functions and constraints. As the size of the building structure increases, the computational time of both Grasshopper and the FEA increases significantly. Similarly, when the number of design variables increases, the population size and/or the number of iterations of the GA should increase. However, as parallel computing becomes increasingly accessible, the computational time will not be an issue in the future. When increasing the number of objective functions, the weighted sum method can be incorporated into the mathematical model which induces issues that are inherent to all optimization methods and are not exclusive to this method. Finally, constraints are handled by imposing penalty terms to the fitness function, therefore increasing the number of constraints would not require more computational power and would not increase the complexity of the problem. Based on these arguments, the proposed method is considered easily and sufficiently scalable to handle problems of much higher complexity.

5 CONCLUSIONS

This paper presents a form-finding architectural, structural optimization method that couples the architectural visual programming language, Grasshopper, and a high-performance computing language, Matlab. An instance of a parametric model of a twisted skyscraper was developed to demonstrate the capability of the method to carry out a mixed integer nonlinear programming problem. The presented method can conveniently deal with problems that demand high geometric complexity. By using the presented workflow, the wind loads, dead loads, section

properties, building's floor areas, and the number of structural connections can be taken as functions and automatically updated by Grasshopper during the optimization process. The current version of the tool requires a multiplatform scenario between Grasshopper and Matlab. Future development of the method includes a strategy to incorporate solid and shell elements into the functionality of the presented structural optimization method in order to simulate buildings that include structural shear walls. The ideal scenario, to avoid lengthy computational time, is to avoid adding the third platform, such as Abaqus, for performing the shell analysis and optimization. It is shown that the difficulty of dealing with these complex features can be alleviated by using Grasshopper as a convenient parametric CAD interface and may allow architects to include realistic constraints in their design process during the schematic design phase.

Nixon Wonoto, http://orcid.org/0000-0002-2007-6614
Vincent Blouin, http://orcid.org/0000-0001-5552-7625

REFERENCES

[1] Abaqus Unified FEA, https://www.3ds.com/products-services/simulia/products/abaqus/latest-release/, Dassault Systèmes.
[2] Adams, V.; Askenazi, A.: Building Better Products with Finite Element Analysis, OnWord Press, 1998.
[3] Curletto, G.: Parametric modeling in form-finding and application to the design of modular canopies, Mobile and Rapidly Assembled Structures 4, 2014.
[4] Felippa, C.A..: A historical outline of matrix structural analysis: a play in three acts, Computers and Structures, 2000.
[5] Felkner, J.; Chatzi, E.; Kotnik, T.: Architectural feedback in the structural optimization process, Structures and Architecture: Concepts, Applications, and Challenges, 2013.
[6] Friedrich, E.: The voronoi diagram in structural optimization. M.S. Thesis, Bartlett School of Graduate Studies, 2008.
[7] Galapagos, http://www.grasshopper3d.com/group/galapagos, David Rutten.
[8] Gerbo, E.; Saliklis, E.: Optimizing a trussed frame subjected to the wind using Rhino, Grasshopper, Karamba, and Galapagos, International Association for Shell and Spatial Structures (IASS), 2014.
[9] Grasshopper, www.grasshopper3d.com, David Rutten.
[10] Haddad, T.: Integrating structural feedback into design processes for complex surface-active form. M.Arch. Thesis, Georgia Institute of Technology, 2006.
[11] Karamba, http://www.karamba3d.com/, 3dKaramba parametric engineering.
[12] Ky, V.; Ambrosio, C.; Hamadi, Y.; Liberti, L.: Surrogate-based methods for black-box optimization. International Transactions in Operational Research 24(3), 2015, 393-424.
[13] Liang, Y.; Leung, K.: Genetic algorithm with adaptive elitist-population strategies for multimodal function optimization, Applied Soft Computing, 11, 2011, 2017-2034. https://doi.org/10.1016/j.asoc.2010.06.017
[14] Matlab, https://www.mathworks.com/products/matlab.html, Mathworks.
[15] Mitchell, M.: An Introduction to Genetic Algorithm, MIT Press, 1999.
[16] Moon, K.: Braced tube structures for complex-shaped tall buildings, Advanced Materials Research, 450-451, 2012, 1584-1587. https://doi.org/10.4028/www.scientific.net/AMR.450-451.1584
[17] Rhinoceros, https://www.rhino3d.com, Robert McNeel and Associates.
[18] Sassone, M.; Pugnale, A.: Evolutionary structural optimization in shell designs, Advanced Numerical Analysis of Shell-like Structures, 2007.
[19] Shan, S.; Wang, G.: Survey of modelling and optimization strategies to solve high-dimensional design problems with computationally-expensive black box functions. Structural and Multidisciplinary Optimization, 41(2), 2010, 219-241. https://doi.org/10.1007/s00158-009-0420-2
[20] Vickery, P.; Skerlj, P.; Twisdale, L.: Simulation of hurricane risk in the U.S. using empirical track model, Journal of Structural Engineering, 126(10), 2010, 1222-1237. https://doi.org/10.1061/(ASCE)0733-9445(2000)126:10(1222)
[21] Von Buelow, P.; Falk, A.; Turrin, M.: Optimization of the structural form using a genetic algorithm to search associative parametric geometry, Proceedings of the International Conference on Structures Architecture, 2010. https://doi.org/10.1201/b10428-93
[22] Wonoto, N.; Baerlecken, D.; Gentry, R.; Swarts, M.: Parametric design and structural analysis of deployable origami tessellation, Computer-Aided Design and Applications, 10(6), 2013, 939-951. https://doi.org/10.3722/cadaps.2013.939-951
[23] Zhu, X.: Wind load analysis on a high-rise square plan building. M.S. Thesis, Arizona State, 2014.