EasyChair Preprint

№ 5216

Achieving an Optimized Solution for Structural Design of Single-Storey Steel Buildings using Generative Design Methodology

Adriano Torres, Bardia Mahmoudi, A.J. Darras, Ali Imanpour and R.G. Driver

March 24, 2021

CSCE 2021 Annual Conference
Inspired by Nature – Inspiré par la Nature

26-29 May 2021

ACHIEVING AN OPTIMIZED SOLUTION FOR STRUCTURAL DESIGN OF SINGLE-STOREY STEEL BUILDINGS USING GENERATIVE DESIGN METHODOLOGY

Torres, A.[1,2], Mahmoudi, B.[1], Darras, A.J.[1], Imanpour, A.[1], and Driver, R.G.[1]
[1] University of Alberta, Canada
[2] atorres@ualberta.ca

Abstract: The significant capabilities of emerging technologies need to be studied to better understand how they can be used to enhance the efficiency of the structural design process. Software already used in the industry are evolving, and some applications are utilizing the power of machine learning and artificial intelligence. Various companies are starting to invest in these technologies and are searching for solutions to reduce component mass, improve structural performance, and minimize manufacturing process time. Currently, the Steel Centre at the University of Alberta is researching these technologies' applications towards typical structural designs. Industry consultation is being conducted to map out current industry practices and logistics. A literature review of various optimization algorithms and past studies on the application of generative design (GD) is being performed. In addition, a single-storey case study is being conducted that involves developing an automation tool in Grasshopper that generates warehouse geometry according to user inputs. S-Frame, an advanced structural analysis software, is being integrated into the design tool. Wallacei, an evolutionary solver, is being used to input design objectives and constraints, resulting in optimizing the key parameters. This automation tool aims to assist in developing a deep understanding of the possibilities of GD towards structural optimization, and specifically towards single-storey structures in Canada, which would lead to the creation of extremely efficient structures. Lastly, the case study preliminary results are highlighted in this paper along with future development and research.

1 INTRODUCTION

Single-storey buildings constructed using structural steel are commonly used in Canada for shopping centres, recreation facilities, and industrial buildings. During the design development phase of a project, engineers evaluate multiple design parameters to achieve the owner's objectives within a limited time before the detailed design begins. The current single-storey design workflow in design offices is mostly manual and tedious. Moreover, miscommunication and human error may occur during design due to overlapping tasks. Regardless of these disadvantages, the design process has the capability not only to become automated, but also to seek innovative new solutions (Rempling et al. 2019, Almusharaf and Elnimeiri 2010). GD is a specific application of artificial intelligence (AI) that can quickly generate thousands of high-performing design scenarios (McKnight 2017). It has been applied to architectural designs in Canada (Nagy et al. 2018), but has not yet been realized in structural design. One important factor that structural engineers prioritize during the design stage in order to achieve an optimum design is the total weight of the structure. While optimizing the weight cannot be said to not produce the lowest cost or optimal structure, it is used in this case study as a proxy for labour cost, material cost, and environmental effects, etc., caused by the manufacturing and fabrication processes.

In this paper, an automation design tool is developed using Rhino3D (Robert McNeel & Associates 1998), a 3D modeling software, and Grasshopper (Rutten 2007), an algorithmic modelling plugin for Rhino3D. The design tool takes the user inputs and feeds it to a metaheuristic single-objective optimization algorithm. The application and performance of various algorithms used in Grasshopper are studied and considered for future implementation. In this case study, the algorithm's sole objective is to reduce the structures steel tonnage. However, other essential objectives and constraints in the design of steel structures are studied and considered. For the structural analysis portion of this design tool, a link between Grasshopper and S-Frame (Casoli 1981) is being developed to incorporate an FEA solver into the tool. Enhancements to the geometry of the single-storey structure generated in Rhino3D are proposed. After discussing the preliminary results of this research, future steps to enhance the automation tool is outlined.

The main objective of this research is to gain a deep understanding of GD's possibilities towards structural optimization, which will lead to an automation tool that can design safer and lighter single-storey structures in Canada. The method has the potential to reduce material usage, minimize construction waste, and improve productivity in the Canadian construction industry. Furthermore, this research has strong potential to provide Canadian practitioners in the steel construction industry with an automated process to design single-storey buildings.

2 OPTIMIZATION IN STRUCTURAL DESIGN

2.1 Optimization Algorithms

An evolutionary solver used in this project's optimization tool is responsible for finding the optimal solution for the design problem by implementing a metaheuristic single-objective algorithm. Various evolutionary solvers are discussed in the literature. Most of these algorithms share the same concepts, as they are developed based on the group behaviour of different creatures in nature and how they evolve. The main advantage of these algorithms is that they are derivative-free. Whereas other mathematical approaches require a well-defined and differentiable objective function and attempt to find the optimal solution by computing the derivative of the objective function, metaheuristic algorithms search the domain just by assessing the objective function's value. Since optimization of engineering systems requires evaluating sophisticated objective functions that are not usually differentiable, metaheuristic algorithms have gained popularity among researchers. The key stages of the optimization process are illustrated in Figure 1 and summarized below. In this figure, N is the number of solutions considered for the first generation, and m is the total number of generations considered for limiting the loop of updating the generations. It is worth noting that the design variable considered here is only the spacing between the columns, and the objective function is the total weight of the structure. The penalty function increases the total weight of structure when the results obtained from structural analysis software do not pass the design codes' requirements.

1) Stage 1: the algorithm creates a set of random solutions by varying design variables associated with the problem. This set is also referred to as the first generation. Each solution returns a specific value for the objective function defined for the optimization problem and by inspecting these values, different solutions can be ranked against each other.
2) Stage 2: in every optimization problem, solutions are subject to different constraints with a feasible space defined. The algorithm applies these constraints to the solutions by a penalty function. If a solution meets all constraints, the value of its objective function remains the same. However, if the solution's design variables violate these constraints, it would get penalized by a multiplier in its objective function so that it would not be able to compete with feasible solutions of the generation when it comes to rank them based on their objective function.
3) Stage 3: once the set is sorted based on the objective function of the solutions, the algorithm applies certain mathematical functions to the generation and adjusts their design variables, leading to a new set of solutions (i.e., the next generation). The mathematical functions vary for different algorithms. For instance, particle swarm optimization (PSO) algorithm updates the solutions with the velocity function (Kennedy and Eberhart 1995) and genetic algorithm generates new solutions with mutation and crossover functions (Goldberg 1989). The algorithm's main goal is to modify the

solutions by generating and guiding the solutions toward the global optimum of the problem. While these functions focus on obtaining the best solutions of each generation and improving them in subsequent generations, randomness is also formulated within them, which helps the algorithm search the entire feasible domain of the problem and prevent getting trapped in zones where local optima are located.

4) Stage 4: it has been proven mathematically that the functions responsible for generating new solutions help the optimization algorithm converge at the end if it undergoes a sufficient number of iterations. There are two ways to specify when an algorithm should terminate the loop of creating new generations and bypass performing the second and third stages. The preferred approach is to consider a total number of generations for the algorithm before it has stared generating solutions. The second approach involves checking the convergence at each iteration by comparing the best solutions of the last two generations with each other. If the difference between the value of the objective function of these two solutions is less than the specified tolerance, it is assumed that the algorithm is no longer capable of finding a better solution, so it is allowed to stop generating new ones. The latter approach might not be appropriate because sometimes the algorithm might get stuck around a local optimum. Terminating the loop does not let the randomness considered in mathematical functions help the algorithm discover new regions in the domain that may contain more optimal solutions.

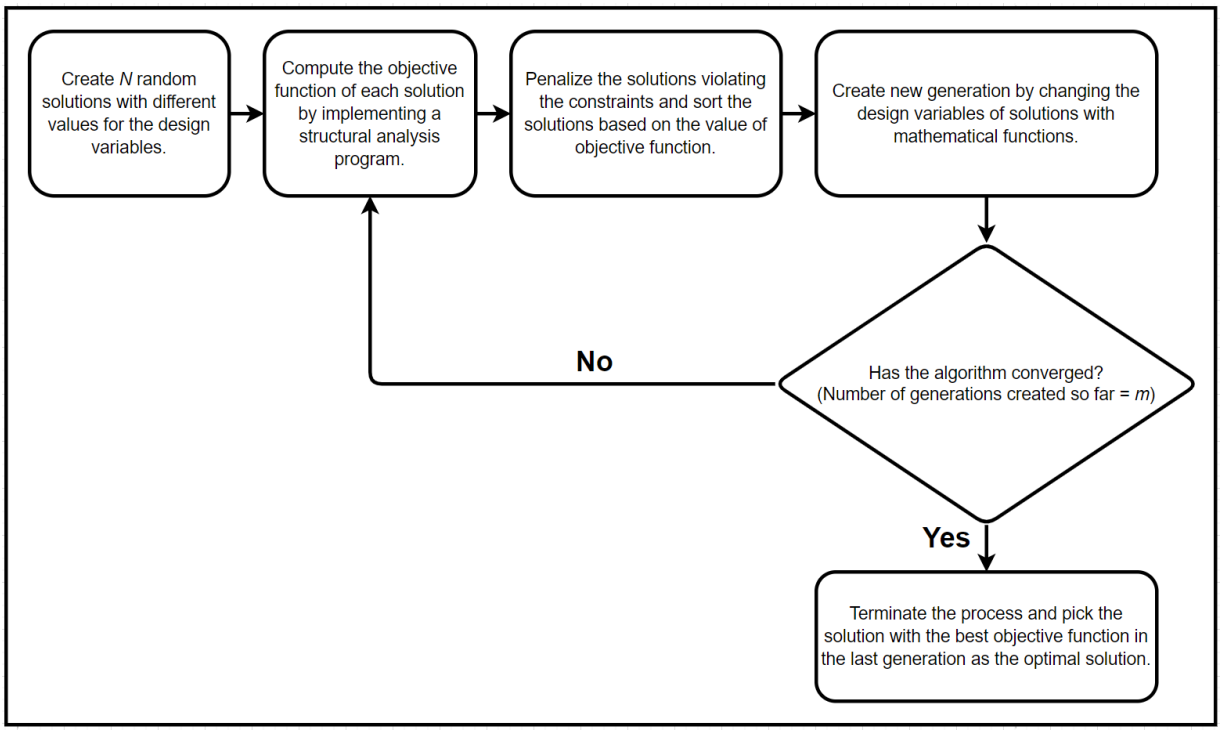


Figure 1: Optimization process stages

2.2 Grasshopper Algorithms

The efficiency of optimization algorithms strongly depends on the number of variables, constraints, and objective functions. Past studies showed that Optimus, a tool based on the jEDE algorithm, outperforms several other single-objective optimization tools of Grasshopper in the optimization of a frame structure (Cubukcuoglue et al. 2019). The following is a list of the tools that Optimus was compared against, and the algorithms that they use: Galapagos (Rutten 2013), based on the genetic algorithm; SilverEye (Cichocka et al. 2017), using the PSO algorithm; and Opossum (Wortmann 2017), using an RBFOpt algorithm (Costa and Nannicini 2018). In this study, the performance of Wallacei (Makki and Showkatbakhsh 2018), a tool

based on the NSGA-II algorithm, which is primarily developed for solving multi-objective optimization problems, with the four Grasshopper optimization tools introduced above to determine which tool yields the best result for optimizing single-storey buildings. By introducing the new nondominated sorting concept, the nondominated sorting genetic algorithm II (NSGA-II) allows us to solve optimization problems with more than one objective function with the help of fundamental components of the genetic algorithm, which can only be used for solving single-objective problems (Deb et al. 2002).

2.3 Wallacei

Wallacei is an evolutionary multi-objective optimization and analytic engine. This evolutionary solver can consider several objective functions simultaneously to determine the optimum solution. In this case study, there is only one objective function, reducing steel tonnage. However, the ability to run several objective functions is a highly valuable property considering the automation tool requires more objectives, as mentioned in Section 3.1. In addition, the solver allows the user to store and save arbitrary data for each iteration of the design. Compared to other Grasshopper components such as Galapagos, Wallacei has specific features that give the user better control over the optimization, graphs, and plots to follow the optimization (Granberg and Wahlstein 2020). The basic interface for the Wallacei component in Grasshopper is shown in Figure 2.

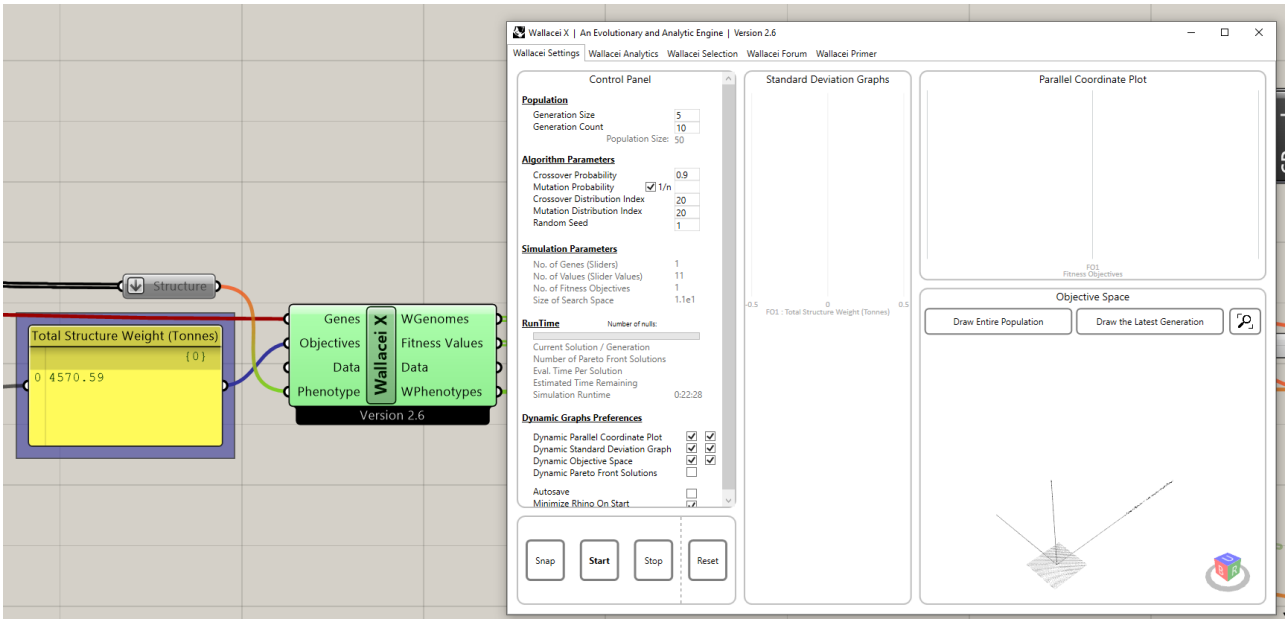


Figure 2: Wallacei interface in Grasshopper

3 PROPOSED AUTOMATION TOOL FOR STRUCTURAL DESIGN

3.1 Description of Automation Tool

GD methodology developed for the purpose of single-storey building design can generate a large number of layout options according to the designer's specific requirements. The project's GD workflow has three main components: generate, evaluate, and evolve (shown in Figure 3). The designer can specify the length, width, and height of the building inside the Grasshopper script, shown in Figure 4a. This generates the first design option in real-time within Rhino 3D, as shown in Figure 4b. The evolutionary solver Wallacei is then used to produce a large number of options by varying the equal column spacing used in each direction to obtain the most cost-effective option, taking into account the weight of structural steel only, a standard method implemented by fabricators in approximating the cost. This process leads to various plausible

design options with respective design data, aiding the designer to make a judgment call on which options to proceed with.

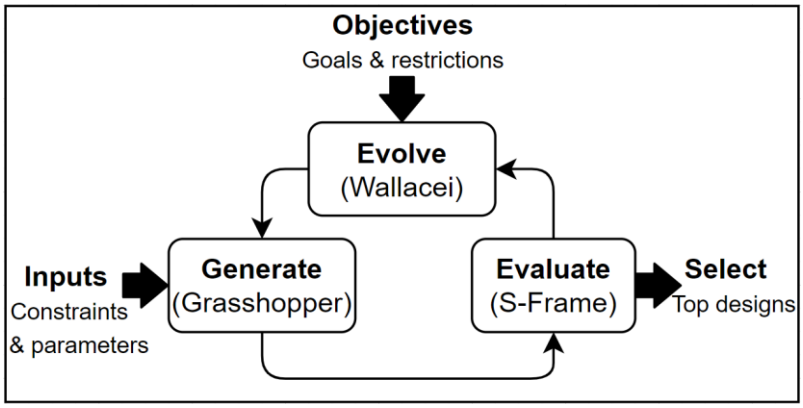
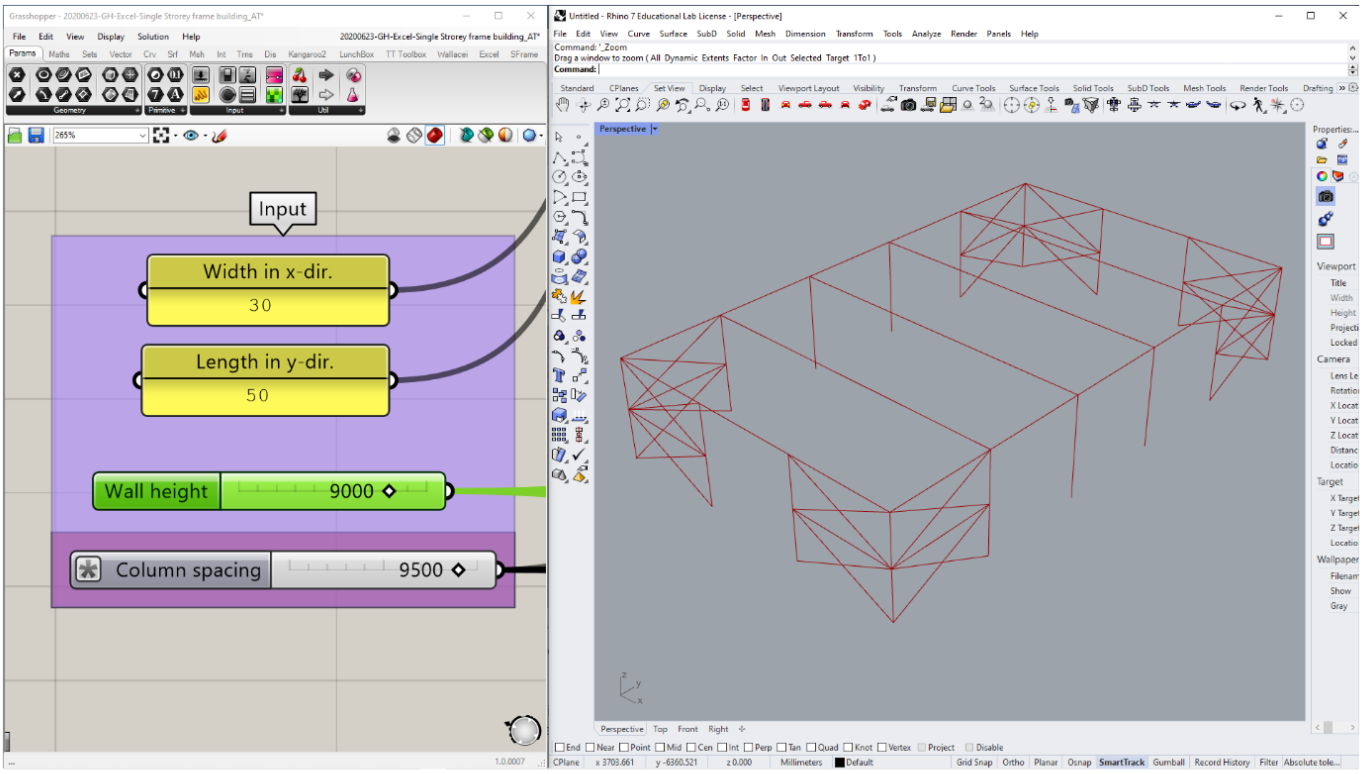


Figure 3: Project's GD workflow



(a) User inputs in Grasshopper script (b) Design option generated inside Rhino3D

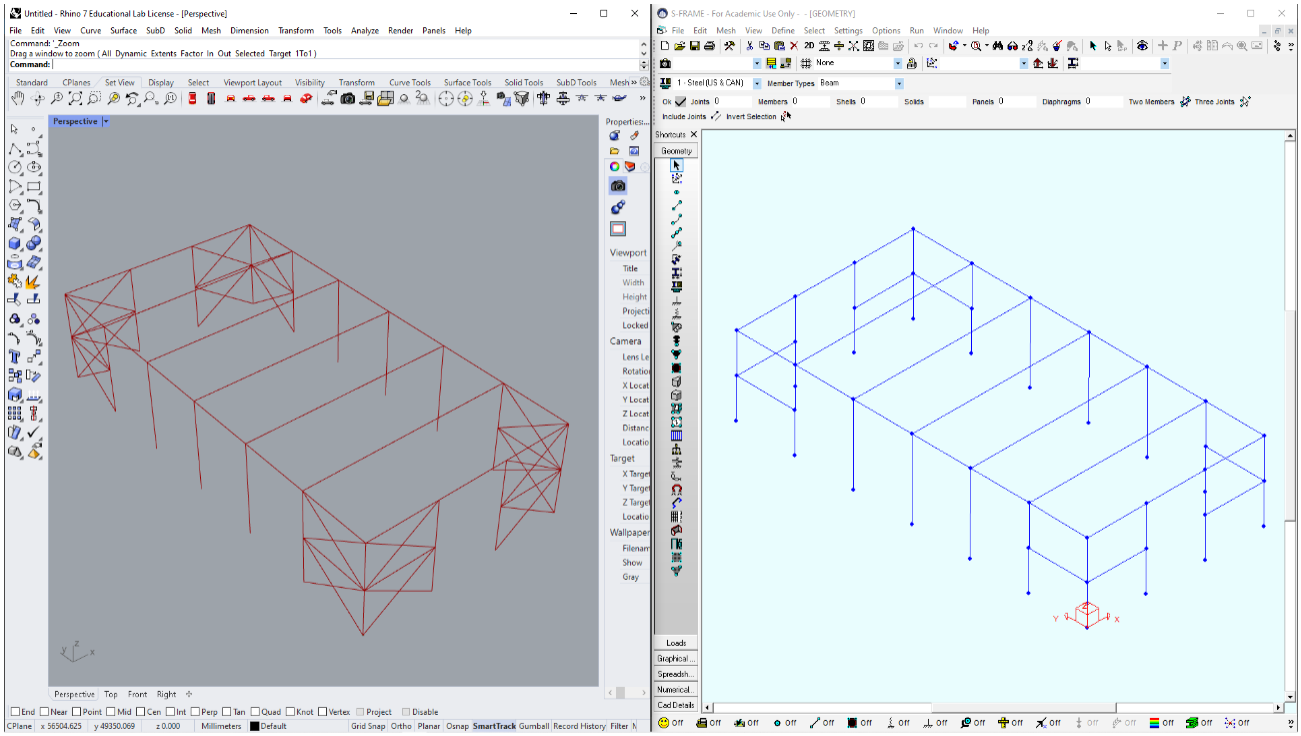
Figure 4: Single-storey structure automation tool

3.2 Structural Analysis Component

The current analysis method in the automation tool mentioned in Section 3.1 uses Excel. After Excel performs a simplified calculation to select the structure's members, a summary of the structure's weight is created. From this summary, the weight of the beams, struts, and bracing is totalled and represents the total weight of steel for the structure. This total steel weight is the driving factor for comparing various layouts that the script produces. However, using Excel is a very simplified method of structural analysis and needs to be replaced with a more advanced means of analysis.

3.3 Implementation of Finite Element Method for Structural Analysis

To analyze the structure using the structural analysis program S-Frame, a link between the scripts in Grasshopper and S-Frame is needed to transfer the model's data from Rhino3D to S-Frame. Since there is no current API that exists to connect Grasshopper and S-Frame, a middleware text file is required to create this link between software. This text file will export the necessary information from Grasshopper and import it into S-Frame to create the structure model. To create such a text file, a template text file is made that holds all the semi-constant information that can later be filled out with the remaining data to match the desired model. Filling out the remaining data can be performed with a C# script that grabs all the needed geometry data from Grasshopper and inserts it into the text file. Once this is done, the text file can be opened with S-Frame, creating the S-Frame model and allowing the FEA solver to analyze the structure. This creation of this link is almost complete, as the only data left to transfer is the bracing geometry. The transfer of the model's data between software is shown in Figure 5.



(a) Structure model inside Rhino3D (b) Structure model inside FEA solver software

Figure 5: Transfer of model's data between Rhino3D and S-Frame

To create the C# script that fills out the needed text file, a custom Grasshopper component was created and shown in Figure 6. This custom component takes the user inputs such as the number of bays in each direction, the wall height, and bay spacing, which are used to generate all the remaining data required for the text file. This data consists of design codes, geometry, loads, section and material properties, etc. This is intended to create a fully-parametric tool that can update the text file for S-Frame as soon as one of the model's parameters changes inside Grasshopper.

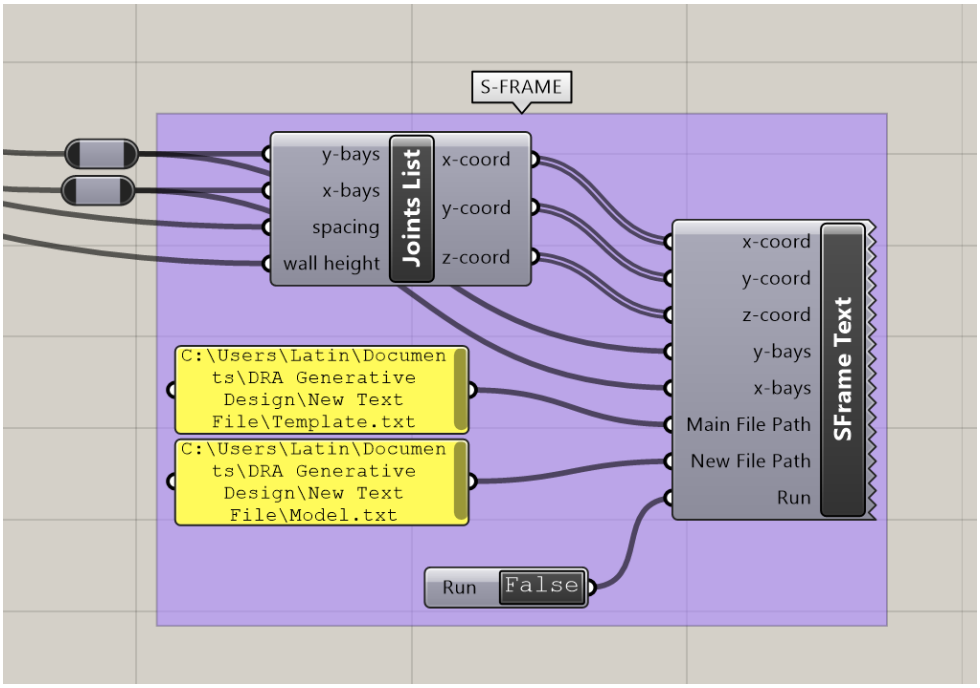


Figure 6: Custom Grasshopper component created from C# script

4 PRELIMINARY RESULTS AND FUTURE RESEARCH

4.1 Cost Estimate

In practice, designers estimate the steel structure cost in the design development stage based on steel tonnage. The steel tonnage can be considered a key metric to evaluate and compare design options. However, the total weight of steel can only provide an approximate estimate of the project's total cost. Other important variables are also required to improve the estimate's accuracy, since the total construction cost of structural steel framing is not necessarily a function of its weight (Ashworth and Skitmore 1983). The three primary components of the total cost are a function of the material, fabrication, and erection costs, as shown in Figure 7 (Barg et al. 2018). The material category is defined as structural shapes, plates, steel joists, steel deck, bolting products, welding products, painting products, and any other products purchased and incorporated into the project. The fabrication category includes the detailing and fabrication labour required to prepare and assemble the shop assemblies of structural shapes, plates, bolts, welds, and other materials. The erection category includes the erection labour needed to unload, lift, place, and connect the structural steel frame components. Lastly, other costs are defined as all cost items not specifically included in the three previous categories (Carter and Schlafly 2008). As shown in Figure 7, the material costs only constitute one-quarter of the total cost, and the majority of costs are associated with the fabrication and erection of structural steel framing. From the results shown in the mentioned studies, it is evident that the proposed automation tool needs to incorporate other cost estimating factors to provide a more realistic estimate of the project's total cost.

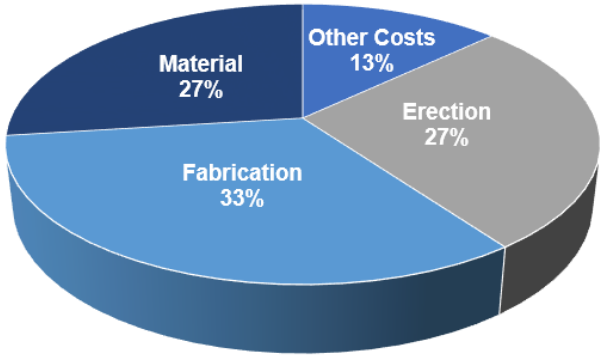


Figure 7: Distribution of the total cost of structural steel framing (Carter and Schlafly 2008)

4.2 Building Geometry

The current lateral load resisting system of the single-storey building consists of cross-bracing placed at the corners. However, other bracing configurations such as chevron bracing, V-bracing or single diagonal bracing can also be used in such low-rise buildings. The possibility of different bracing configurations (Figure 8), number of braced frames and bracing locations will be studied in the future. Note that the application of X-bracing, inverted V-bracing, and diagonal bracing options for a simple portal frame structure were evaluated in the past using a python script built in Grasshopper (Vasilev 2020). Furthermore, other steel frame geometric parameters such as height to eaves, the pitch of frame, and haunch length are so far missing in the design. Other research has included such parameters in optimizing steel frame buildings (Phan et al. 2013, Hernández et al. 2012). Adding different bracing configurations and other frame geometric parameters will widen the design space's scope and create a more realistic structure.

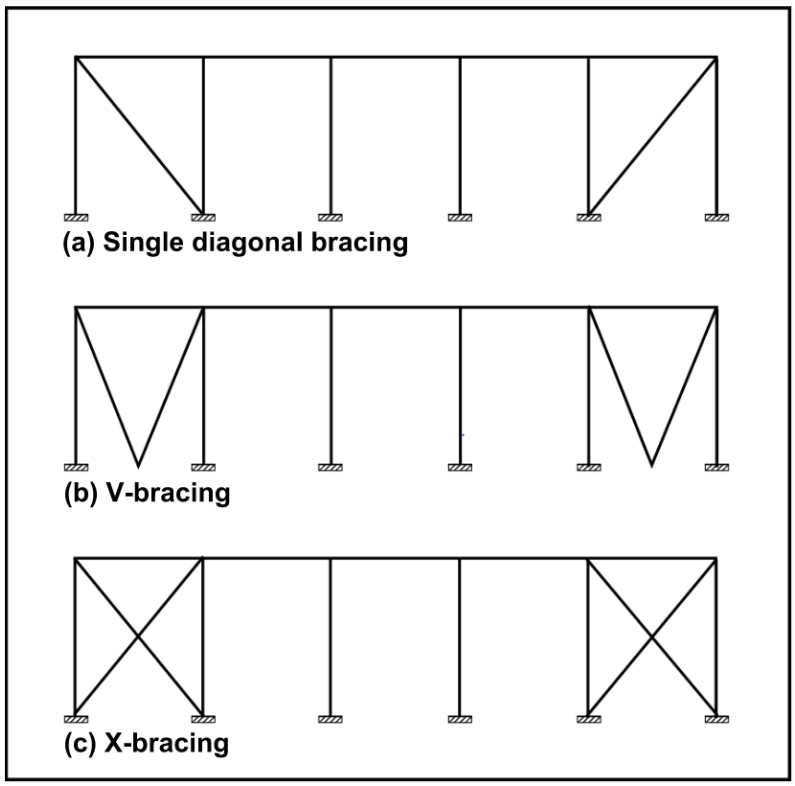


Figure 8: Typical bracing configurations for portal frames (“Portal Frames” 2021)

4.3 Future Direction

While the current study shows promising results obtained using the proposed automation tool, it is still rudimentary and requires further refinement and development before it could be implemented in practice. Further improvements to the proposed automation tool are as follows:

- Incorporate finite element analysis into the project's workflow by completing the link between Grasshopper and S-Frame.
- Incorporate additional factors in the overall cost estimation process to obtain an accurate value.
- Consult with fabricators and erectors to better understand their preferences in the construction of single-storey steel buildings and implement this industry knowledge into the optimization process.
- Expand the scope of the generated frame geometry to produce a more realistic structure and improve optimization.
- Implement multi-objective optimization algorithms by adding more design variables and objectives.
- Explore and compare different optimization algorithms used in structural applications and determine if better results can be obtained.
- Combine a connection design optimization tool with the current automation tool.

5 CONCLUSIONS

An automated optimization tool is proposed here for the structural design of single-storey steel structures by reducing steel tonnage in the design stage. The preliminary results show the evolutionary algorithm adopted can result in optimized design options in the design development stage to help the designer select an efficient, better design option. Future studies will incorporate other key design objectives including member availability, connection type, and different frame geometries into the proposed automated design tool to achieve a more accurate estimate of construction costs.

6 REFERENCES

Almusharaf, A. and Elnimeiri, M. 2010. A Performance-based Design Approach for Early Tall Building Form Development. 5th International Conference Proceedings of The Arab Society for Computer Aided Architectural Design, Illinois Institute of Technology, Chicago, IL, USA, **1**: 39-50.

Ashworth, A. and Skitmore, M. 1983. The Effectiveness of Estimating in the Construction Industry, Master's Degree, The Charted Institute of Building, Englemere, Berkshire, England.

Barg, S., Flager, F., and Fischer, M. 2018. An Analytical Method to Estimate the Total Installed Cost of Structural Steel Building Frames During Early Design. *Journal of Building Engineering*, **15**: 41-50.

Carter, C., and Schlafly, T. 2008. "$ave More Money". *Modern Steel Construction*, American Institute of Steel Construction.

Casoli, G. S-Frame. V. Enterprise. S-Frame Software Inc. Windows. 1998

Cichocka, J.M.; Migalska, A.; Browne, W.N.; Rodriguez, E. SILVEREYE—The Implementation of Particle Swarm Optimization Algorithm in a Design Optimization Tool. In Proceedings of the International Conference on Computer-Aided Architectural Design Futures, Istanbul, Turkey, 10–14 July 2017; pp. 151–169.

Cichocka J.M., Migalska A., Browne W.N., Rodriguez E. (2017) SILVEREYE – The Implementation of Particle Swarm Optimization Algorithm in a Design Optimization Tool. In: Çağdaş G., Özkar M., Gül L., Gürer E. (eds) Computer-Aided Architectural Design. Future Trajectories. CAADFutures 2017. Communications in Computer and Information Science, vol 724. Springer, Singapore

Costa, A., Nannicini, G. 2018. RBFOpt: An Open-Source Library for Black-box Optimization With Costly Function Evaluations. *Mathematical Programming Computation*, **10**(4): 597–629

Cubukcuoglu, C., Ekici, B., Tasgetiren, M.F., and Sariyildiz, S. 2019. OPTIMUS: Self-Adaptive Differential Evolution with Ensemble of Mutation Strategies for Grasshopper Algorithmic Modeling. *Algorithms*, **12**(7): 141.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. 2002. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE *Transactions on Evolutionary Computation*, **6**(2): 182-197.

Goldberg, D.E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st. ed., Addison-Wesley Longman Publishing Co., Boston, MA, USA.

Granberg, A. and Wahlstein, J. 2020. *Parametric Design and Optimization of Pipe Bridges*, Master's Degree, KTH Royal Institute of Technology

Hernández, S., Brebbia, C.A., and De Wilde, W.P. 2012. *Computer Aided Optimum Design in Engineering XII,* WIT Press, Southampton, UK.

Kennedy, J. and Eberhart, R. 1995. Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, IEEE, Perth, WA, Australia, **4**: 1942-1948

Makki, M. and Showkatbakhsh M. 2018. Wallacei. Wallacei.

McKnight, M. 2017. Generative Design: What It Is? How Is It Being Used? Why It's A Game Changer. *The International Conference on Design and Technology*, KnE Engineering, Geelong, Australia, **2**: 176-181.

Microsoft. C#. V. 7.3. Microsoft. Windows. 2000.

Nagy, D., Villaggi, V., and Benjamin, D. 2018. Generative Urban Design: Integrating Financial and Energy Goals for Automated Neighborhood Layout. SpringSim: *Spring Simulation Multiconference*, Society of Computer Simulation International, San Diego, CA, USA, **25**: 1-8.

"Portal Frames". 2021. www.steelconstruction.Info. https://www.steelconstruction.info/Portal_frames.

Phan, D.T., Lim, J.B.P., Sha, W., Siew, C.Y.M., Tanyimboh, T.T., Issa, H.K. and Mohammad, F.A. 2013. Design Optimization of Cold-Formed Steel Portal Frames Taking into Account the Effect of Building Topology. *Engineering Optimization*, **45**(4): 415-433.

Rempling, R., Mathern, A., Ramos D., and Fernández, S. 2019. Automatic Structural Design by A Set-Based Parametric Design Method. *Automation in Construction,* **108**: 102936.

Robert McNeel & Associates. Rhinoceros 3D. V. 7.0. Robert McNeel & Associates. 1998

Rutten, D. Grasshopper. V.1.0. Robert McNeel & Associates. Windows. 2007.

Rutten, D. 2013. Galapagos: On the Logic and Limitations of Generic Solvers. Archit Design, 83: 132-135.

Vasilev, L. 2020. Parametric Modeling in Structural Design, Double Bachelor's Degree, LAB University of Applied Sciences.

Wortmann, T. Opossum: Introducing and Evaluating a Model-based Optimization Tool for Grasshopper. In Proceedings of the CAADRIA 2017, Hong Kong, China, 5–8 July 2017.