Constrained Multi-Criteria Optimization for Integrated Design in Professional Practice

Claire Duclos-Prevet, François Guéna, Mariano Effron

Constrained Multi-Criteria Optimization for Integrated Design in Professional Practice

Claire Duclos[1], François Guéna[1], Mariano Efron[2]

[1] MAP-MAAC, Ecole Nationale Supérieure d'Architecture de Paris La Villette, France
c.duclos.prevet@gmail.com
francois.guena@email.com
[2] Architecture-Studio, France
lm.efron@architecturestudio.fr

Abstract. To design sustainable architecture, theory encourages architects to rely on automated exploration processes. In practice, the problems encountered are often multicriteria and under constraint. This paper compares different constraint handling strategies, approachable to designer, for processes involving evolutionary algorithms. Four methods are tested on a case study from professional practice. Two methods rely on parametric models: the penalty function method and the use of hyperparameters. The others involve the use of generative techniques: a rule-based method and a repair algorithm that takes the form of an agent-based model. This study highlights the significant impact of the choice of the constraint management method on exploration performance. Among other results, it appears that models involving the use of generative techniques are more efficient than those using parametric models. This calls for the development of dedicated tools.

Keywords: building envelope design, generative design, agent-based modeling, constrained multiobjective evolutionary algorithm, daylighting simulation

1 Introduction

The global climate change emergency requires architects to deal with the environmental issues earlier and earlier in the architectural design process. Nowadays, visual programming facilitates interoperability and multidisciplinarity, making environmental simulation and evolutionary optimization approachable to architects.

These new tools have allowed the emergence of new computer-aided design methods; namely, the exploration of architectural solutions where many variants can be generated, evaluated, sorted, and finally selected. These performances-based approaches, known as integrated design methods, have been the subject of a significant research effort (see Shi, X et al., 2016 for a comprehensive review).

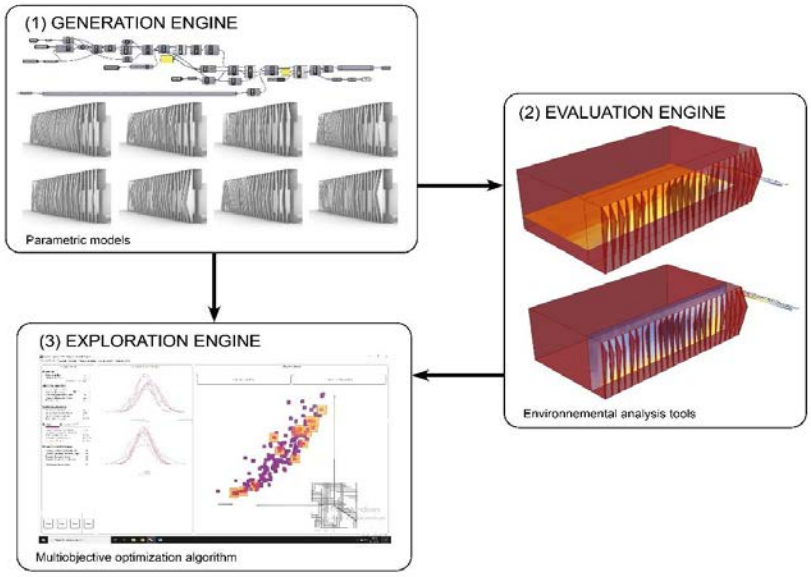

Figure 1. The 3 components of integrated design methods with optimization algorithm. Source: Author, 2021.

Among these methods, a large part focused on building envelope design with the aim of optimizing visual and thermal comfort (Eltaweel, Yuehong, 2017). These methods are associated with different names in the scientific literature but rely on the same approach. As illustrated in Figure 1, these methods require 3 components: a generative engine for shape modeling (often a parametric model), an evaluation engine for environmental performance analysis, and an engine for solution space exploration, (mostly evolutionary algorithms).

Despite their growing popularity in the scientific community, integrated design methods remain largely unused in the practice of architectural offices (Li, et al., 2020). Indeed, these theoretical approaches are usually not easy to implement because problems encountered by practitioners are often both multi-criteria and under constraints. This makes an already complex problem formulation even more difficult (Zhao, S. and De Angelis, E., 2018). However, a few papers report applications on professional projects (Haymaker J., et al.,2018; Shen, X. et al., 2018), some showing that these methods allow experimented architects to achieve design improvements (Bernal, M. et al., 2020).

This gap between the state of the art and the observed practice have motivated our research project, which follows an *action research methodology* and tries to identify the technical obstacles that may explain this discrepancy. Regarding the methods applied to envelopes, we have noticed that they can involve large solution spaces and require the inclusion of constraints. Therefore, the objective of our research is to identify and compare the

performances of different techniques allowing to deal with these constraints in a multicriteria optimization problem observed.

2 Multi-criteria optimization under constraints

2.1 Constrained problems

In optimization, constraints are different from optimization criteria because they are, by definition, inviolable. If the constraint is not satisfied, then the solution is deemed infeasible, and evaluating its performance would be useless and time consuming. Regarding envelope design problems, the constraints encountered can be functional, aesthetical, or purely practical – such as collision issues in solar shading systems for the later.

Moreover, envelope exploration can be conducted on two distinct scales: at the module level (Fathy and Fareed, 2017; Jayathissa, P. et al., 2018, Tabadkani, A. et al, 2019), or globally – the entire facade – (Yi, Y.K., 2019; Erkan and Elias-Ozkan, 2016, Negendahl and Nielsen, 2015). Global scale implementation involves many decision variables, which dramatically increase the size of the solution space and, consequently, computation times. Regarding visual and thermal comfort, working at the global scale appears to be mandatory in three circumstances: (i) when the building form is complex, (ii) when the building is placed in a heterogeneous urban context, making access to the sun uneven, or (iii) when the architectural concept relies on complexity, irregularity or random effects.

In order to reduce the size of the solution space, it is possible to perform a sensitivity analysis to order the decision variables and reduce their number by keeping only the most relevant (Kheiri, F., 2018). However, it is possible that this method does not sufficiently reduce the solution space, and this is more likely when the number of criteria to maximize increases and the purpose of these criteria diverge. Therefore, it might be necessary, to reduce the solution space, to add constraints that could be, for instance, aesthetical (Chatzikonstantinou et al., 2019).

2.2 Constraint management methods with evolutionary algorithms, in theory

Most algorithms used for exploration purposes in architecture are evolutionary algorithms and most problems are multi-criteria (Li et al., 2020). According to Coello, Carlos A. (2002) and Michalewicz, Z. and Schoenauer M. (1996), there are different methods to handle constraints with evolutionary algorithms. These methods can be classified into 5 categories:

(1) Penalty functions, which consist in degrading the score of the objective function according to the degree of constraint violation. There are several types

of penalty, such as static penalty, when penalty factors do not evolve over generations, or dynamic penalty when they do.

(2) Special representations and operators such as homomorphic maps, or the decoder approach, designed for some particularly difficult problems.

(3) Repair algorithms, which consist in modifying – *repairing* – solutions that do not respect the constraints to make them feasible. There are two types of repair algorithms (Salcedo-Sanz, S., 2009): Lamarckian algorithm where the modification of the solution leads to a modification of both the phenotype and the genotype; and the Baldwinian algorithm that modifies only the phenotype.

(4) Methods that separate constraints and objective functions as Co-evolution or behavioral Memory. A very accessible method consists in transforming a constraint into an optimization function (Coello, C. A. 2017), so the problem is no longer a constrained problem.

(5) Hybrid methods that mix evolutionary algorithms and other optimization techniques as Nelder and Mead's simplex method or ant colony optimization.

Some of these techniques from the categories (1), (3) and (4) can be easily implemented by the designer, others require computer programming skills because they involve a partial modification of evolutionary algorithms, or are techniques designed for specific problems.

2.3 Constraint-handling, in practice

In practice, designers who explore solution space are usually not computer scientists. Some are visual programmers and use optimization solvers available in their basic version. The most popular visual programming platform in the architecture field (Grasshopper) has two multi-criteria optimization solvers (Octopus and Wallacei). They do not handle constraints except for a filtering system. This system allows to use a death penalty technique which consists in deleting the non-feasible solutions and run until the desired number of complying solutions is reached. This method is not recommended unless the quantity of infeasible solutions is limited, otherwise the search may stagnate (Coello, C. A. 2002).

Given these characteristics observed in practice, only three solutions, identified in the literature on constraint management methods with evolutionary algorithms, are accessible to designers: (i) static penalty functions, (ii) transforming constraints into optimization criteria, and (iii) designing a Baldwinian repair algorithm. The first two methods imply the evaluation of unsatisfactory solutions, consequently they are not suitable when the simulation computation time is long. The other methods presented in section 2.2 are not included in our list of methods easily implementable by designers because they would require a modification of the optimization solver, or the use of alternative algorithms that are not currently implemented in the visual programming platform.

Importantly, in numerous case studies originated from scientific literature on integrated design methods it appears that other methods are used to reduce

the solution space only to solutions that comply with the constraints by adjusting the parametric model. This can be achieved through intermediate parameters – *hyperparameters* – which control several decision variables and thus reduce their number (Chatzikonstantinou et al., 2019; Negendahl and Nielsen, 2015). Instead of using a parametric model, there is also the possibility to develop models, so-called generative models, capable of generating complexity from simple rules using loops (Caetano et al., 2019), which rules can be akin to constraints. In the relevant literature, different generative techniques are used with integrated approaches such as cellular automata (Kim J., 2015), or agent-based models (Gerber et al., 2017; Zarrabi et al., 2018).

While they are not always used to handle constraints, agent-based modeling (Macal, 2016) allows, starting from an original solution, autonomous architectural elements (agents) to interact until they reach a steady state, which is a compromise where all agents achieve their goals. In this way, an agent-based modeling can be used to transform an infeasible original solution into a complying solution. It can thus operate as a repair function.

In order to eventually guide designers in the use of these techniques, we sought to compare their effectiveness and manageability by testing them on an envelope design case.

3 Case study

3.1 Context and objectives

Our case study is based on a real-world project designed by Architecture Studio. It is a medical research institute. The objective was to optimize the solar protection system of the laboratories in order to maximize indirect daylighting. The system consists of large triangular vertical blades. The shapes of the triangles vary to create a pattern in the facade.

A first study had been carried out in a professional context. To meet the deadlines, we simplified the problem and used a basic method with a simple parametric model and a genetic algorithm for single-criteria optimization (hours of sunlight). Four decision variables were analyzed, one for the orientation of the blades of each facade. Thus, the problem had neither constraints nor a large solution space.

In a second step, with an academic perspective, we have reconsidered this problem without reducing its complexity. Thus, it becomes a multi-criteria problem under constraint, where the natural interior illuminance must be maximized while the direct and diffuse solar radiation on the glazed parts must be minimized. Also, the number of decision variables is much larger: each blade can take a different orientation and the shape of the blade can also vary. Thus, it becomes possible to use the reflected light radiation to maximize indirect daylighting, as illustrated in Figure 2. The problem then contains feasibility

constraints to avoid collisions between the blades, and aesthetic constraints to avoid jagged effects for the pattern in the facade.

3.2 Constraints-handling methods

In order to study and compare the different methods, we have experimented several approaches to solve this constrained multi-criteria optimization problem, which we have tested on a piece of facade reduced to 30 blades. The "death penalty" method does not work on this problem since it has too many infeasible solutions. Thus, we implement a first method (1) with 60 variables (2 per blade for 2.3E+55 solutions). It relies upon a penalty function to internalize the collision constraints and introduces a third optimization criterion that focuses on the smoothness of the curve drawn by the blades. The penalty function evaluate the degree of constraint violation based on the collision area between each blade. The advantage of this method is that it does not exclude any solution except the infeasible ones. Moreover, this method is easy to implement in visual programming since it is a simple parametric model. The difficulty lies in the definition of the penalty coefficients (Coello, C. A. 2002). To deal with this issue, it is necessary to evaluate a sample of the solution set beforehand to calibrate the model.

A second method (2) consists in using hyperparameters that are the coordinates of the control points of two NURBS curves (22 variables for 4.9E20 solutions). One curve is used to vary the orientation, and the other to modify the shape of the blades. The advantage of this method is that it generates, by construction, only aesthetically pleasing solutions and that the space of solutions to be explored is greatly reduced. Also, this method is easy to implement because it is a parametric model. However, the number of control points of the curves is necessarily fixed, hence many satisfactory scenarios are excluded from the exploration.

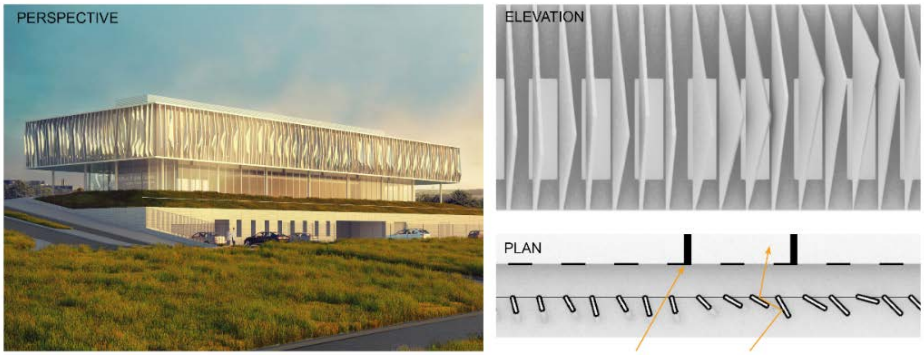

Figure 2. Perspective, plan and elevation of the solar shading system. Source: Architecture Studio, 2015 and Author, 2019

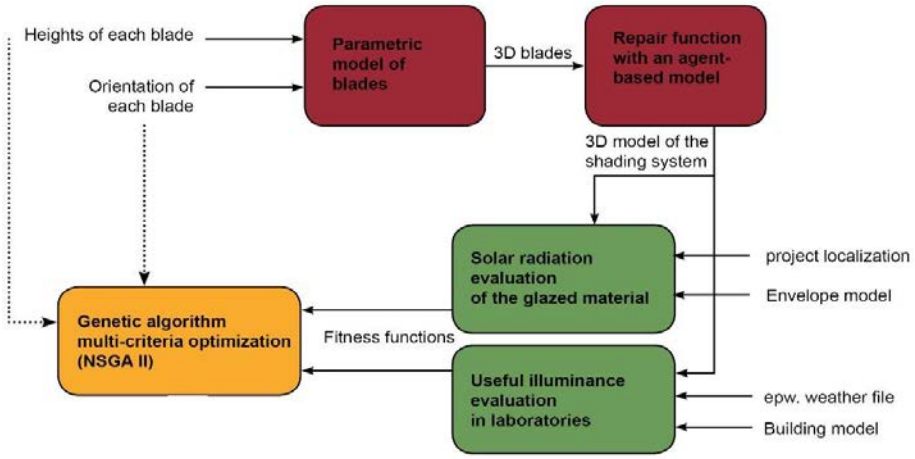

Figure 3. Framework for the implementation of the constrained multi-criteria optimization method with repair function. Source: Author, 2021

A third method (3) consist in modifying the parametric model to exclude unsatisfactory solutions using a rule-based generative process. Our algorithm generates blades one after the other in a way that respects the collision and curvature constraints with 60 variables for 2.1E+37. The number of decision variables remains unchanged but the number of values that each of them can take has been greatly reduced. This method allows to generate aesthetically satisfying solutions with curves having a variable number of control points. The disadvantage of this particular algorithm is that it contains a stochastic dimension to ensure that the curve is continuous. This generates a discrepancy between genotype and phenotype. That can disturb the search of the genetic algorithm. Also, this technique requires the use of recursion and is not easily accessible with visual programming.

Finally, a fourth method (4), depicted in Figure 3, is to use the simple parametric model of method (1) with the same solution space and add a repair function that acts as an agent-based model. In this model, the blades are the agents and have two attributes: their shape and orientation. The behavior of each agent varies at each iteration depending on these two closest neighbors to avoid collisions and jagged effects. At each iteration, the blades evolve until the system finds an equilibrium where these two constraints are satisfied. It is thus a system of self-organization of the blades. The advantage of this method is to be able to generate only aesthetically satisfying solutions without overly restricting the field of possibilities and without the need to use a random function. The disadvantage is that it requires some programming knowledge to implement this type of model. Also, the relevant algorithm that will allow to respect the constraints without distorting too much the initial solution is hard to find.

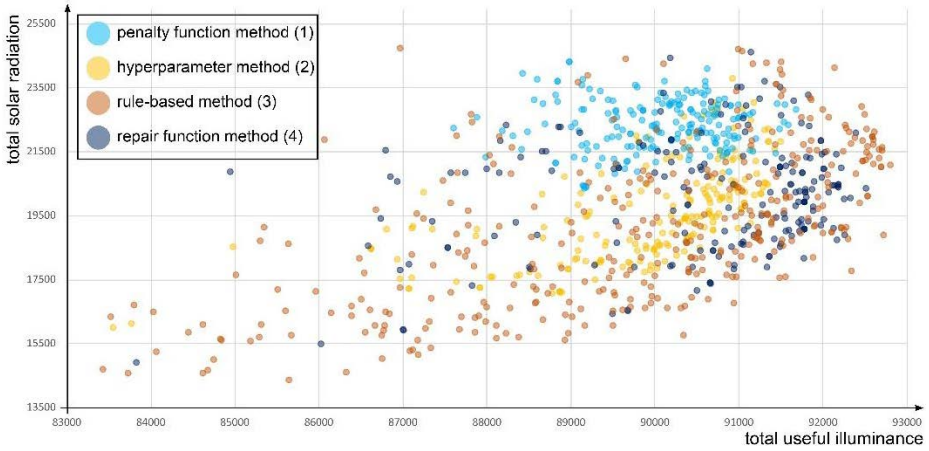

Figure 4. Graph with all the solutions evaluated for the 4 methods. Source: Author, 2021

3.3 Study parameters

The four methods were implemented on Grasshopper®, with the visualization being done on the Rhinoceros® modeler. The methods based on generative models were coded in Python. The experiments were carried out with a south-facing facade and with a weather file from the Paris Orly station (https://www.ladybug.tools/epwmap/). The Ladybug® plugin (Roudsari, 2013) was used for the evaluation of direct and diffuse radiation, and the Honeybee® plugin coupled with the Radiance lighting simulation software for the evaluation of useful illuminance. The total radiation value is the target to minimize. For the lighting simulation, the reflection parameters of the materials used are standard, they correspond to the HQE recommendations. The sum of the values of the useful illuminance with a 0.5 m grid is the objective to be maximized. The solver used for the optimization is Wallacei® with the evolutionary multi-objective algorithm (Deb, 2011) named NSGA-2 (Non dominated Sorting Genetic Algorithm II). The calculations are run on 14 generations of 25 individuals each. Once the simulations were completed, the results were assembled in an Excel® file using the Colibri® plugin and then analyzed with Excel® and the online visualization tool Design Explorer2®.

4 Results

The method (1) is the one that obtains the worst results in terms of performance with only 7 non-dominated solutions. The values for the total radiation oscillate between 19799.98 and 24299.93 kW/h, while the mean useful illuminance varies between 62.30% and 65.28%. Moreover, the 3rd optimization criterion

used to minimize the sawtooth effect on the façade is not met and the solutions are also less satisfactory from an aesthetic point of view.

Method (2) performs much better than (1), but less than (3) or (4). It has 21 non-dominated solutions and the scores vary between 15833.83 and 23771.28 kW/h for radiation, and between 59.42% and 65.00% for illuminance. They are also more satisfactory from an aesthetic perspective despite the emergence of discontinuity on the curve visible in Figure 5.

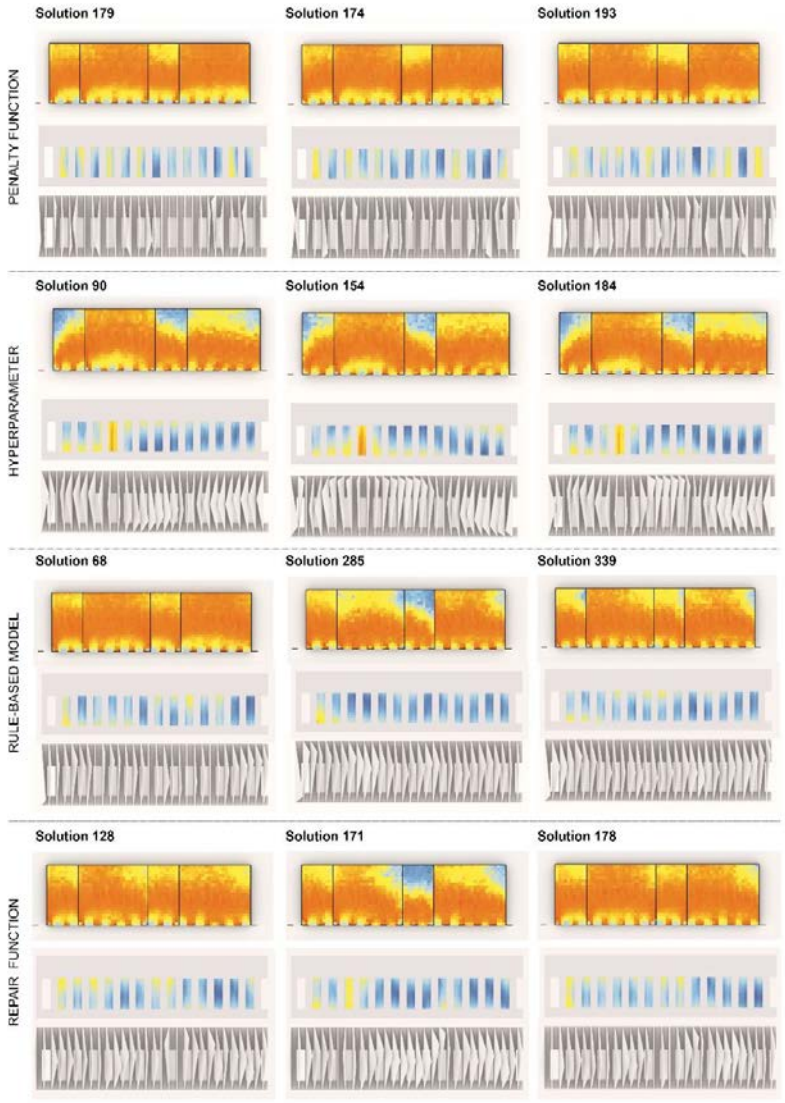

Figure 5. Extract of the solutions on the Pareto front for each method. Source: Author, 2021

Method (3) is the best performing of the 4 with only 13 non-dominated solutions. The values range from 14391.02 to 24720.07 kW/h, and from 59.34% to 66.00%, but the patterns generated in the facade tend to be monotonic.

Finally, method (4) performs slightly worse than (3), but is still much better than (1) and (2). It has 14 non-dominated solutions and values ranging from 12934.63 to 24608.58 kW/h for radiation, and between 55.92% to 65.86% for useful illuminance. Moreover, the solutions do not present any particular aesthetic problem.

5 Discussions

In practice, integrated methods based on the exploration of solution space often require the introduction of constraints, either because the project impose some constraints or because these constraints are necessary to reduce the size of the solution space. For the same problem, there are different ways to generate the geometry, which can have a strong impact on the exploration results. While the choice of optimization algorithm (Waibel et al., 2019) and the choice of evaluation method have already been studied (Carlucci, et al., 2015), little work, to our knowledge, focused on the impact of the choice of generative method.

We have identified 5 techniques available to designers to incorporate constraints into an optimization problem: formulate the constraints as a criterion, use a penalty function, a repair function, hyperparameters, or a generative technique. In our case study, the effectiveness of these methods diverges. The methods involving generative techniques (3 and 4) give much better solutions, but they are more complex to implement than the methods using parametric models (1 and 2). Thus, it appears necessary to develop dedicated tools allowing to democratize the use of generative techniques, in particular for the implementation of repair functions with a specific optimization solver allowing to use Lamarckian algorithms considered to be more efficient.

This study concerns a single scenario, the results could be quite different with other design problems. Indeed, it seems that the relevance of a method depends on the problem, and especially on the size of the feasible region. Indeed, death penalties are useful if a minority of the solutions in the space is infeasible. Penalty functions do not work well if a majority of solutions are infeasible (Coello, C. A. 2002). Also, it is not always easy or possible to define hyperparameters that makes it possible to get rid of the constraints; similarly, it is not always possible to create a rule-based generative model that includes decision variables or to find a repair function that works within the time constraints of professional practice. In sum, it would be interesting to experiment these methods on other design problems from practice in order to propose guidelines to designers that would orient their choice of generative method according to the type of problem encountered.

References

Bernal, M., Okhoya, V., Marshall, T., Chen, C., & Haymaker, J. (2020). Integrating expertise and parametric analysis for a data-driven decision-making practice. International Journal of Architectural Computing, 18(4), 424-440.

Caetano, I., Santos, L., & Leitão, A. (2020). Computational design in architecture: Defining parametric, generative, and algorithmic design. Frontiers of Architectural Research, 9(2), 287-300.

Carlucci, S., Causone, F., De Rosa, F., & Pagliano, L. (2015). A review of indices for assessing visual comfort with a view to their use in optimization processes to support building integrated design. Renewable and sustainable energy reviews, 47, 1016-1033.

Chatzikonstantinou, I., Turrin, M., Cubukcuoglu, C., Kirimtat, A., & Sariyildiz, S. (2019). A Comprehensive Optimization Approach for Modular Facades: The Case of PULSE Sunshading. International Journal of Design Sciences & Technology, 23(2), 159-185.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer methods in applied mechanics and engineering, 191(11-12), 1245-1287.

Coello, C. A. C. (2017, July). Constraint-handling techniques used with evolutionary algorithms. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 675-701).

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

Eltaweel, A., & Yuehong, S. U. (2017). Parametric design and daylighting: A literature review. Renewable and Sustainable Energy Reviews, 73, 1086-1103.

Ercan, B., & Elias-Ozkan, S. T. (2015). Performance-based parametric design explorations: A method for generating appropriate building components. Design Studies, 38, 33-53.

Fathy, F., & Fareed, H. A. (2017). Performance-driven Façade Design Using an Evolutionary Multi-Objective Optimization Approach. In International Conference for Sustainable Design of the Built Environment-SDBE London (p. 217).

Gerber, D. J., Pantazis, E., & Wang, A. (2017). A multi-agent approach for performance based architecture: design exploring geometry, user, and environmental agencies in façades. Automation in construction, 76, 45-58.

Haymaker, J., Bernal, M., Marshall, M. T., Okhoya, V., Szilasi, A., Rezaee, R., ... & Welle, B. (2018). Design space construction: a framework to support collaborative, parametric decision making. Journal of Information Technology in Construction (ITcon), 23(8), 157-178..

Jayathissa, P., Caranovic, S., Hofer, J., Nagy, Z., & Schlueter, A. (2018). Performative design environment for kinetic photovoltaic architecture. Automation in Construction, 93, 339-347.

Kheiri, F. (2018). A review on optimization methods applied in energy-efficient building geometry and envelope design. Renewable and Sustainable Energy Reviews, 92, 897-920.

Kim, J. (2015). Adaptive façade design for the daylighting performance in an office building: the investigation of an opening design strategy with cellular automata. International Journal of Low-Carbon Technologies, 10(3), 313-320.

Li, S., Liu, L., & Peng, C. (2020). A review of performance-oriented architectural design and optimization in the context of sustainability: dividends and challenges. Sustainability, 12(4), 1427.

Macal, C. M. (2016). Everything you need to know about agent-based modelling and simulation. Journal of Simulation, 10(2), 144-156.

Michalewicz, Z., & Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. Evolutionary computation, 4(1), 1-32.

Negendahl, K., & Nielsen, T. R. (2015). Building energy optimization in the early design stages: A simplified method. Energy and Buildings, 105, 88-99.

Roudsari, M. S., Pak, M., & Smith, A. (2013, August). Ladybug: a parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design. In Proceedings of the 13th international IBPSA conference held in Lyon, France Aug (pp. 3128-3135).

Salcedo-Sanz, S. (2009). A survey of repair methods used as constraint handling techniques in evolutionary algorithms. Computer science review, 3(3), 175-192.

Shen, X., Singhvi, A., Mengual, A., Spastri, M., & Watson, V. (2018). EVALUATING THE MULTI-OBJECTIVE OPTIMIZATION METHODOLOGY FOR PERFORMANCE-BASED BUILDING DESIGN IN PROFESSIONAL PRACTICE. ASHRAE and IBPSA, 646-653.

Shi, X., Tian, Z., Chen, W., Si, B., & Jin, X. (2016). A review on building energy efficient design optimization rom the perspective of architects. Renewable and Sustainable Energy Reviews, 65, 872-884.

Tabadkani, A., Shoubi, M. V., Soflaei, F., & Banihashemi, S. (2019). Integrated parametric design of adaptive facades for user's visual comfort. *Automation in Construction*, *106*, 102857.

Waibel, C., Wortmann, T., Evins, R., & Carmeliet, J. (2019). Building energy optimization: An extensive benchmark of global search algorithms. Energy and Buildings, 187, 218-240.

Yi, Y. K. (2019). Building facade multi-objective optimization for daylight and aesthetical perception. Building and Environment, 156, 178-190.

Zarrabi, A. H., Azarbayjani, M., & Tavakoli, M. Generative Design Tool: Integrated Approach toward Development of Piezoelectric Façade System.

Zhao, S., & De Angelis, E. (2018). Performance-based generative architecture design: A review on design problem formulation and software utilization. Journal of Integrated Design and Process Science, 22(3), 55-76.