Design Space Exploration of Shell Structures Using Quality Diversity Algorithms

Konstantinos SFIKAS*, Antonios LIAPIS, Joel HILMERSSON[a], Jeg DUDLEY[a], Edoardo TIBUZZI[a], Georgios N. YANNAKAKIS

*Institute of Digital Games, University of Malta
konstantinos.sfikas@um.edu.mt

[a] AKT II

Abstract

Computer-aided optimization algorithms in structural engineering have historically focused on the *structural performance* of generated forms, often resulting in the selection of a single 'optimal' solution. However, *diversity* of generated solutions is desirable when those solutions are shown to a human user to choose from. Quality-Diversity (QD) search is an emerging field of Evolutionary Computation which can automate the exploration of the solution space in engineering problems. QD algorithms, such as MAP-Elites, operate by maintaining and expanding an archive of diverse solutions, optimizing for quality in local niches of a multidimensional design space. The generated archive of solutions can help engineers gain a better overview of the solution space, illuminating which designs are possible and their trade-offs. In this paper we apply Quality Diversity search to the problem of designing shell structures. Since the design of shell structures comes with physical constraints, we leverage a constrained optimization variant of the MAP-Elites algorithm, FI-MAP-Elites. We implement our proposed methodology within the Rhino/Grasshopper environment and use the Karamba FEA solver for all structural engineering calculations. We test our method on case-studies of parametric models of shell structures that feature varying complexity. Our experiments investigate the algorithm's ability to illuminate the solution space and generate feasible and high-quality solutions.

Keywords: evolutionary algorithm, evolutionary computation, quality diversity, optimisation, multi objective optimisation, conceptual design, shell structures, spatial structures.

1. Introduction

A frequently encountered problem within the AEC industry is the design of efficient shell structures. Such structures - in which architectural form is inextricably linked to structural behaviour and performance - require a close collaboration between architect, structural engineer and other AEC consultants to find a shape that is acceptable to all parties and their separate demands. Historically, many of these shells were conceived as form-active structures, and thus their final shape was ascertained through form-finding techniques–first physically [1],[2], then digitally [3]. These digital simulations are remarkably flexible, and able to incorporate a range of physical input constraints, such as ensuring structural efficiency (e.g. axial loading only) or geometric rationalisation (e.g. avoiding panel warp). However, for a single set of inputs (materials, loading and support conditions) form-finding algorithms only generate a singular solution, rather than a range of potential solutions. As such, it can be difficult for the designer to evaluate potential trade-offs between the desired criteria and propose new weightings of them.

We see similar limitations in evolutionary computation (EC) algorithms, such as Divide-and-Conquer or Hill-Climbing. These algorithms are Single Objective Optimisation (SOO) approaches, and as such they too suffer from a difficulty in weighting and combining output values to achieve a well-balanced evaluation of Fitness for the variants. More sophisticated EC approaches include Multi Objective Optimisation (MOO) or Pareto optimisation algorithms, such as SPEA-2 and HypE. These MOO algorithms have been implemented in AEC digital design toolkits for several years now [4],[5], and allow substantially more detailed exploration of the solution space.

Quality Diversity (QD) search is a family of EC algorithms that simultaneously maintain the quality and diversity of solutions [6] by rewarding *divergence* of the population according to some ad-hoc measure of difference between the actual artifacts rather than the genotypic information. *Quality* can be a secondary objective to this divergence [7], act as a constraint on minimal quality [8], or as a survival criterion between similar artifacts [9]. Algorithms such as MAP-Elites can illuminate the search space, maintaining and evolving in parallel different solutions in different behavioural niches (i.e. with similar design characteristics orthogonal to quality). MAP-Elites and its variants has been a popular algorithm for problems where the "optimal solution" is subjective, such as evolutionary art [10] and game content generation [11].

Considering structural engineering as a response to (or extension of) an architectural design process, we argue that designing shell structures is a subjective search process as well. Any singular "optimal" design is likely to be rejected by other AEC stakeholders, while as with any client-based interactions, showing multiple (good) alternatives can solicit more meaningful feedback. This paper, therefore, argues that QD search algorithms that illuminate the solution space and show trade-offs between different designs will help engineers interface with the subjectivity of the architectural process.

This paper presents a constrained variant of the MAP-Elites algorithm for QD search and tests it in a set of shell structure optimization tasks ranging from simple to complex. The FI-MAP-Elites algorithm, proposed here, leverages two archives evolving in parallel: one archive of solutions that fail a constraint on maximum allowed displacement, and another archive of solutions that satisfy this constraint and search for optimal elastic energy of the structure. The algorithm is integrated into Grasshopper and allows engineers to specify the structure of possible solutions and the dimensions towards which FI-MAP-Elites should explore. Comparisons against a baseline SOO algorithm available in Grasshopper show that the proposed constrained QD search algorithm produces far more diverse solutions, while SOO typically converges towards a single type of solution.

2. General Methodology for Constrained Quality Diversity Search

Feasible-Infeasible Multidimensional Archives of Phenotypic Elites (FI-MAP-Elites) is a hybrid of the MAP-Elites [11] and the FI-2Pop GA [12] algorithms. Its main goal is to combine the illumination capabilities of the former with the constraint-solving capabilities of the latter. Its operation is controlled by the following hyperparameters that are problem-specific:

- **Representation:** refers to how the algorithm stores the artifacts in a compact genotype [13].
- **Initialization:** refers to how the initial population is formed. In many cases initialization of the genotype is as simple as generating a random list of numbers. More complex representations, however, may require a more elaborate scripted initialization [14].
- **Variation:** refers to the genetic operators applied while evolution is carried out to produce offspring from selected individuals. While traditional EC operators include recombination and mutation [15], FI-MAP-Elites only uses mutation of one individual.
- **Infeasible fitness (F_i):** measures how many constraints are failed by the solution and to which degree. The infeasible population aims to minimize the distance to feasibility; to convert this

into a maximization task, we consider that an infeasible fitness below a threshold (usually $F_i < 1$) means that the individual is *infeasible* (i.e. at least one constraint is not satisfied). Ad-hoc heuristics are designed per constraint and added together to form the infeasible fitness score.

- **Feasible fitness (F_f):** is a problem-specific measure of the desirable property of the final artifact. Feasible fitness is only calculated if the individual is feasible, as in some problems the constraints render calculating the feasible fitness impossible [16].
- **Behaviour characterizations (BCs):** refer to quantifiable characteristics of the artifact. BCs are orthogonal to quality and artifacts of both high or low BC scores should be desirable. These BCs define the dimensions of exploration and diversity and should be of interest to the designer. While most MAP-Elites implementations use two BCs, the algorithm can operate with any number of BCs but this may impact the computational cost of evolutionary search.

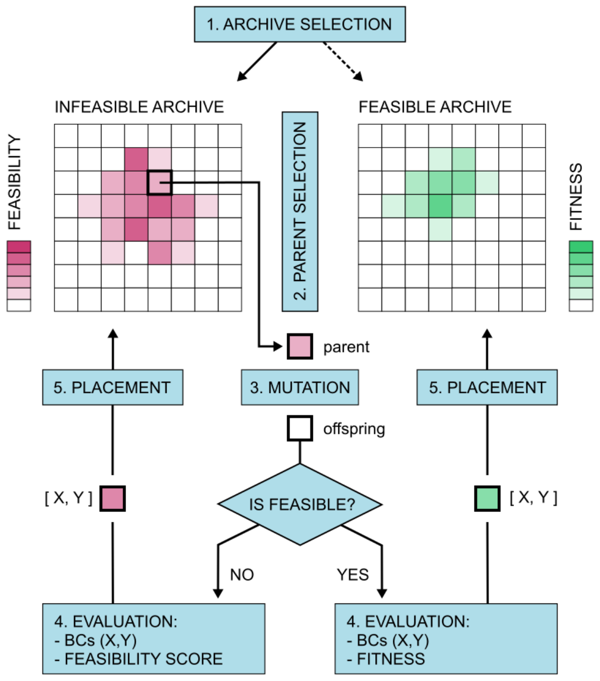


Figure 1. Functional diagram showcasing the operation of the FI-MAP-Elites algorithm, for two BCs.

The FI-MAP-Elites algorithm [14] (visualized in Fig. 1) operates by retaining two distinct archives of solutions, one for the feasible and one for the infeasible population. Both archives are partitioned into cells arranged in an n-dimensional grid, where n is the number of BCs of the problem. Each cell may contain a single individual (elite) with a specific set of BC values that fall within this cell's value range. An initial population seeds the two archives, based on the feasibility of individuals. During its operation, the algorithm selects a random individual either from the feasible or the infeasible archive (alternating between the two), mutates it, evaluates the offspring and places it in the proper archive and cell, based on its feasibility and its BCs respectively. If the cell is empty, the offspring is added to the archive; if not, the offspring replaces the current elite occupying that cell if its fitness (feasible or infeasible) is higher. Through many repeated iterations of this process, the algorithm will gradually fill both archives with solutions that tend to approach their maximum fitness (objective function or degree of feasibility). After several iterations, the algorithm typically manages to provide a diverse set of fit and feasible solutions for the end-users.

3. Applied Methodology & Experiment Protocol

To efficiently test FI-MAP-Elites in shell structure design tasks, a series of parametric shell models were defined in Rhino's visual programming interface, Grasshopper, based on SubDivision (SubD) surfaces. SubD surfaces have the advantage of being derived from a low polygon control mesh that is easy to manipulate through adjustment of its vertices (control points). This gives them geometric flexibility and makes them suitable for representing a wide range of forms. Furthermore, the resulting surfaces are generated through iterative Catmull-Clark subdivisions [17] of the input mesh which guarantees a smooth, continuous curvature. SubD-based approaches have been successfully utilised by AKT II on a series of built and unrealised projects over the last decade [18],[19].

4.1 Algorithmic Parameters

Based on Section 3.1, there are several problem-specific hyperparameters to the algorithm. We elaborate the parameters for the shell use case here.

- **Representation:** The parametric shell model is manually defined by the designer in Rhino/Grasshopper [20]. This includes a definition of a SubD surface, several constraints on its control points' variability, as well as a number of control parameters that affect the points' movement along specified vectors. Any point on the SubD control polygon selected as *support* by the designer is defined as 'creased' to ensure that the point that is constrained exists on the smooth shape. The genetic algorithm only stores (and manipulates) the values of the control parameters.
- **Initialization:** When initializing the population for evolution, control parameters are assigned a random value within a designer-specified value range.
- **Variation:** Variation occurs solely through the mutation operator that iterates over all the control parameters defined in the model and changes their values according to two parameters: *Max Step Size* (i.e. the maximum distortion allowed per mutation step) and *Mutation Rate* (i.e. the probability that this control parameter will be changed).
- **Infeasible fitness:** The only constraint in this use case is an upper limit on displacement (d), i.e. the distance between the point in the original model and the deformed model. The upper limit on displacement is specified by the designer (C_d). Equation (1) calculates the infeasible fitness for this problem (F_i), which is bound in the value range of [0,1] and feasible individuals always have $F_i = 1$.

$$F_i = \begin{cases} \frac{C_d}{d} & \text{, if } d > C_d \\ 1 & \text{, if } d \leq C_d \end{cases} \tag{1}$$

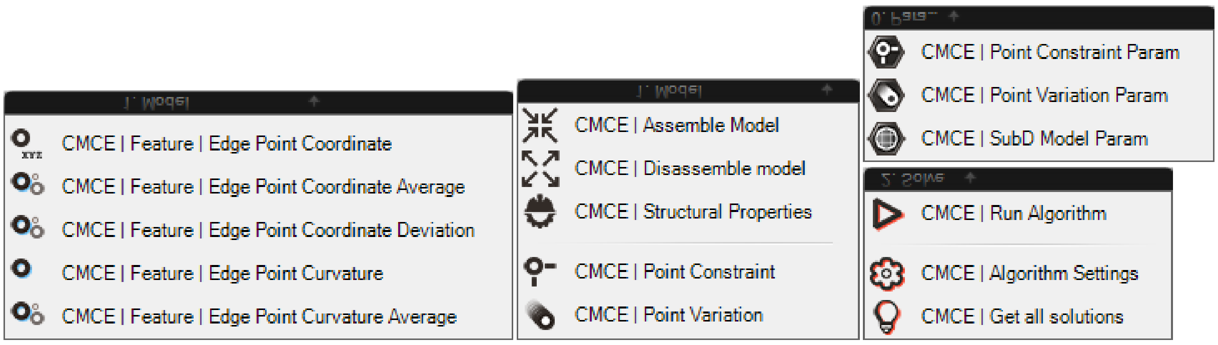
- **Feasible fitness:** For feasible individuals, quality is ascertained based on the total elastic energy (U) of the structure. The value for U measures how much energy is stored in the structure based on its elastic deformation. The feasible fitness (F_f) is only calculated on individuals that satisfy the constraints ($d \leq C_d$); the score of F_f=0 for infeasible individuals is calculated only for the baseline single-objective algorithm in the experiments of this paper.

$$F_f = \begin{cases} 0 & \text{,if } d > C_d \\ \frac{1}{1+U} & \text{,if } d \leq C_d \end{cases} \tag{2}$$

- **Behaviour characterizations (BCs):** Two formulas based on control point coordinates are explored as BCs in this work: (a) *Edge Point Coordinate Average*, as the average coordinate value of a set of parametric points on a series of edges in the SubD geometry (either X, Y, or Z, determined by the user); (b) *Edge Point Coordinates Deviation*, calculated as the difference between the average value and the coordinate (X, Y, or Z, determined by the user) of each point on a series of edges in the SubD geometry, divided by the number of points.

4.2 Rhino Implementation

In order to apply FI-MAP-Elites to structural engineering problems, we exploit the parametric design capabilities of the Rhino/Grasshopper design environment. Within this environment, we developed a set of components which allow the structural engineer to (1) define the structural parametric model, (2) select the algorithmic parameters and (3) run the FI-MAP-Elites algorithm and generate an optimized (in terms of quality and diversity) set of variations of the parametric model.


CMCE | Feature | Edge Point Coordinate
CMCE | Feature | Edge Point Coordinate Average
CMCE | Feature | Edge Point Coordinate Deviation
CMCE | Feature | Edge Point Curvature
CMCE | Feature | Edge Point Curvature Average
CMCE | Assemble Model
CMCE | Disassemble model
CMCE | Structural Properties
CMCE | Point Constraint
CMCE | Point Variation
CMCE | Point Constraint Param
CMCE | Point Variation Param
CMCE | SubD Model Param
CMCE | Run Algorithm
CMCE | Algorithm Settings
CMCE | Get all solutions
Figure 2. Grasshopper toolbars for the FI-MAP-Elites implementation of shell design.

Customization of the algorithm is done through parameters (on the point constraints, its variation range, and parameters of the SubD model) and through details on the physics model and BC characterizations (see Fig. 2 for a summary of the components available). The solver toolbars provide the settings and components to run the process, along with tools to retrieve results. The components can be combined to produce a pipeline to initialize and evaluate solutions.

As soon as the algorithm's operation is over, a Grasshopper component outputs the data of all elites in both archives, as well as their positions on the archive's grid (based on their BCs). Each one of the generated variations includes various meta-information, including: (1) its Karamba [21] model, (2) its Rhino geometry, (3) its genome (parameter values), (4) its elastic energy and (5) its displacement values. Based on these outputs, the archive's contents can be visualized in Rhino's viewport, in a customized manner. Fig. 6. illustrates an example of such a visualization of the feasible archive, which includes a visual (by vertex colours) indication of the parametric model's fitness.

4.3 Hyperparameters and Baseline Algorithms

For all experiments with FI-MAP-Elites in this paper, we use an initial population of 100 random individuals, a mutation rate of 50% and mutation step of 20%, a grid of elites of 16 by 16 (BC1 by BC2) for both feasible and infeasible archives, and results are collected after 50,000 evaluations.

As a baseline to test the performance of FI-MAP-Elites, we use a single-objective optimizer (SOO) of the Galapagos library within Grasshopper. This simple algorithm aims to maximize a single-objective fitness (F_b) computed as the average of the infeasible and feasible fitness; see Eq. (3). Therefore, it overlooks diversity dimensions but tackles constrained optimization.

$$F_b = \frac{1}{2}(F_i + F_f) \tag{3}$$

The evolutionary solver of the Galapagos optimizer within Grasshopper is used for this baseline, with the following settings: a *population* of 256 individuals, an *initial boost* of 1 (the initial population size is 256), an *elitism* of 5% (the 13 fittest individuals are retained across generations) and an *inbreeding rate* of 50% (slightly favors the selection of similar parents during crossover). The population evolves for a number of generations resulting in a comparable number of evaluations as FI-MAP-Elites. We collect solutions from the final population on the last generation to derive SOO performance metrics.

4.4 Hypothesis and performance metrics

Our hypotheses in the experiments described in Chapter 5 can be summarized as follows:

- The two-population approach of FI-MAP Elites will discover more feasible solutions faster.
- Evolution will focus on the feasible population earlier and lead to higher quality solutions in FI-MAP-Elites compared to single-population approaches.
- As a QD algorithm the variety of solutions (in terms of the chosen BCs) will be higher.
- It is expected that more high-quality solutions that are structurally different will emerge from the QD search process that favours local competition within the same behavioural niche.

Following the literature in QD search [9], we evaluate the following metrics: (a) **maximum performance**, as the lowest elastic energy (U_{min}) score and lowest displacement (d_{min}) among feasible individuals of the SOO population or the feasible archive, (b) **coverage**, as the ratio of all cells in the feature map that are occupied by feasible solutions[1], (c) **QD score**, as the sum of feasible fitness scores among all elites in the feasible archive (of feasible individuals in the SOO population). The maximum performance assesses whether at least one solution is well-performing, while coverage assesses whether a large behavioural variety of solutions is found. Finally, the QD score assesses a combination of the above, with higher scores achieved when the algorithm produces many behaviourally diverse individuals of high quality. In addition, we measure the **fitness deviation** (σ_U and σ_d) as the standard deviation in terms of the elastic energy and displacement (respectively) of all feasible solutions. This measure captures the uniformity of designs when exploring along different BCs.

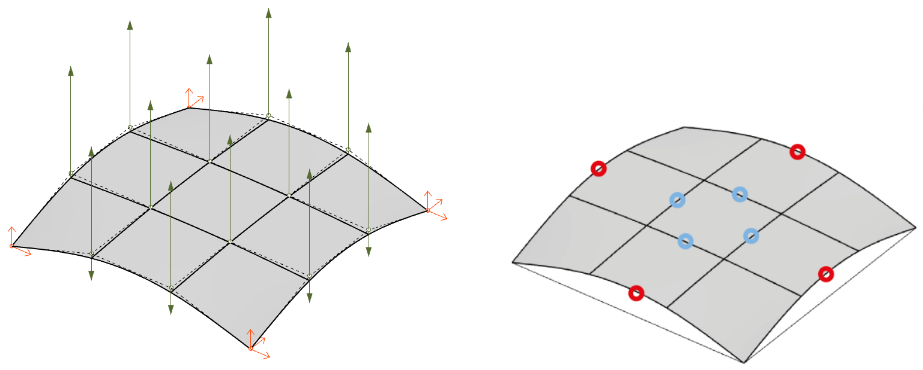
Moreover, we assess constraint satisfaction and constraint optimization performance through the following metrics: (a) **feasible ratio**, as the ratio of solutions that satisfy all constraints against all solutions in the final population(s); (b) **infeasible coverage**, as the ratio of all possible cells in the feature map that are occupied by infeasible solutions (i.e. the infeasible archive for FI-MAP-Elites).

5. Results

In order to evaluate the proposed algorithm's performance, we use three different test cases: two variants of a simple test case, and a complex test case comparable to real-world engineering problems.

5.1 Simple test case

Fig. 3 shows the parametric surface model of the simple test case, based on a square SubD surface subdivided into 9 patches and 16 points. The algorithm varies the height of each point in order to optimize the shape. Each point is given a variability in Z-direction from 0 to 5, which is represented by the green arrows at each control point.



[1] For SOO, all solutions are processed and only feasible ones are placed on the feasible feature map along the two BCs of FI-MAP-Elites (with only the fittest solution retained in each cell).

Figure 3: The square base test case, with the height variability range (left), and the BCs calculated based on the average height of the centre points (blue dots in right image) and height difference between arches (red dots in right image).

Diversity dimensions (BCs) focus on two target areas, namely the central region and the arches (blue and red dots, respectively in Fig. 3). BC1 is taken as the *average Z coordinate* of the edges of the central face, to capture the height at the middle of the shell. BC2 is the *Z coordinate deviation* between the midpoints of the side arches, in order to capture asymmetric aspects of an otherwise symmetrical problem. High performing solutions for this form are largely known beforehand, so these two case studies act as benchmarks to ensure that the solutions provided by the QD algorithm are as expected.

Experiments conducted with this structure assume a base of 16m by 16m, made of grade C16/20 concrete, and a thickness of 7cm, a mesh resolution of 2 subdivisions, and applied loads of 2kN/m^2. For the FI-MAP-Elites hyperparameters, we use an initial population of 100 random individuals, a mutation rate of 50% and mutation step of 20%, a grid of elites of 16 by 16 (BC1 by BC2) for both feasible and infeasible archives, and a total of 10^5 iterations of parent selection. The upper threshold for displacement (C_d) is set to 5cm.

We test two versions of this simple test case (Case 1 and Case 2). In both cases an identical input geometry is used, but they are given two different support constraints. Case 1 has all translations (X, Y, Z) constrained at the four corner points, while Case 2 has the X direction free. This means that Case 1 can transfer horizontal forces in two directions to the supports, while Case 2 can only transfer the horizontal forces in one direction. This should force the same patch into two different structural behaviours, thereby rendering different resulting shapes in the two maps.

Table 1. **Simple Test Case**: Comparing the final archive of QD with the final population of SOO, for Case 1 and Case 2. Results are averaged across 10 experiment repetitions. Error signifies a confidence interval of 95%.

	Case 1		**Case 2**	
	FI-MAP-Elites	**SOO**	**FI-MAP-Elites**	**SOO**
Feasible Coverage	94.3% ± 2.1%	0.6% ± 0.3%	88.9% ± 0.6%	0.7% ± 0.1%
Feasible QD-Score	110.37 ± 1.20	1.12 ± 0.51	75.32 ± 0.91	0.44 ± 0.11
Feasible U_{min}	0.38 ± 0.00	0.34 ± 0.01	0.72 ± 0.02	3.01 ± 0.74
Feasible σ_U	1.16 ± 0.08	0.00 ± 0.00	1.82 ± 0.04	0.02 ± 0.01
Feasible d_{min}	0.37 ± 0.04	0.13 ± 0.02	0.44 ± 0.06	2.08 ± 0.35
Feasible σ_d	0.62 ± 0.03	0.03 ± 0.01	0.76 ± 0.02	0.03 ± 0.01
Feasible Ratio	48.5% ± 0.6%	100.0% ± 0.0%	47.1% ± 0.2%	100.0% ± 0.0%
Infeasible Coverage	100.0% ± 0.0%	0.0% ± 0.0%	99.8% ± 0.3%	0.0% ± 0.0%

Table 1 shows the different performance metrics (see Section 4.4) for both Case 1 and Case 2. Results are averaged from 10 independent runs of both FI-MAP-Elites and SOO per case. A first observation for both cases is that the final SOO population contains exclusively feasible individuals, and those individuals have almost identical scores in elastic energy among them (very low σ_U). Moreover, based on the very low coverage when these solutions are mapped in the FI-MAP-Elites feature map, the solutions are geometrically very similar. Unsurprisingly, FI-MAP-Elites is designed to explore the BCs specified by the designer and reaches very high scores in both feasible and infeasible coverage. Case 2 seems to be more challenging for EC to optimize, as evidenced by the much higher elastic energy in the best solution for both SOO and FI-MAP-Elites. While for Case 1 the SOO manages to converge towards a solution with a *U* score lower than the best *U* score found in FI-MAP-Elites, this is not true in Case 2. We presume that SOO converges early to a *local optimum* and does not explore the search space enough in order to find a better solution. The QD algorithm unsurprisingly does explore along the BCs and thus finds (many) better solutions. However, both the lower coverage and lower QD-score of FI-MAP-Elites compared to Case 1 shows that the freedom of changing the corner points' coordinates makes the problem more difficult.

Fig. 6 shows how the feature map of some indicative runs of FI-MAP-Elites differ between Case 1 and Case 2. It is evident that better solutions (with lower elastic energy) can be found in different parts of the design space (for these BCs), thus indicating different trade-offs between geometric designs. It can be seen that the change in support conditions leads to different solutions for each location in the map, as the shapes with the lowest fitness go from 2-directional compression shells in Case 1 to single-span shell topologies in Case 2. This can be seen as the fittest solutions in Case 1 are the ones with a high middle height, regardless of symmetry, whereas in Case 2 2-axis symmetric solutions are no longer possible and the fittest individuals have moved to the spots with high single-axis symmetry (barrel vault forms). We also see a different class of fit individuals in the middle of the map as the lower middle height of these shells allows them to develop some synclastic curvature, which further increases stiffness.

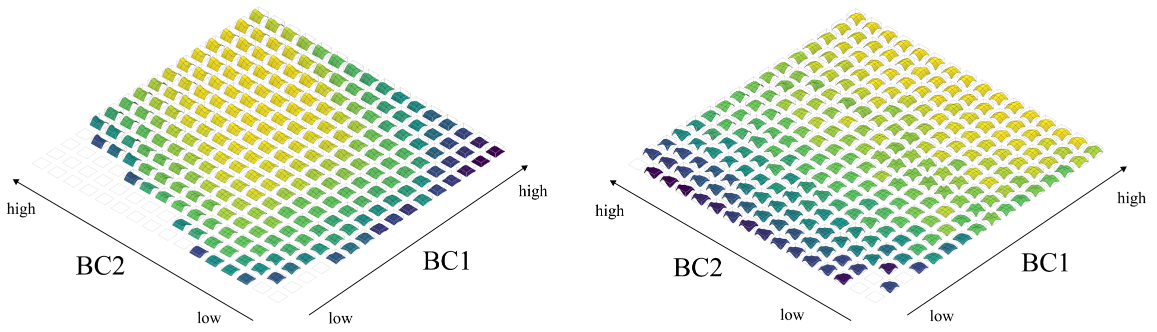


Figure 4: Indicative archives of feasible solutions from the first run of FI-MAP-Elites for Test Case 1 (left) and Test Case 2 (right), collected after 50,000 evaluations. Yellow shapes have higher feasible fitness.

5.2 Complex test case

The complex case study focuses on a more detailed four-arch vault shell which is initially mirror symmetrical in the XZ and YZ planes (as seen in Fig. 5). This form comprises 16 patches, and has 29 control points adjusted by the optimisation algorithms. The shell has a base boundary of 20m by 10m, is formed in grade C16/20 concrete with a thickness of 10cm, a mesh resolution of 3 subdivisions, and only self-weight (no applied loads). The upper threshold for displacement (Cd) is set to 6cm.

Diversity is defined using two *Edge Point Coordinates Deviation* BCs: BC1 is the Z-height deviation between midpoints of the four external arches, BC2 is the Z-height deviation between midpoints of the four axial iso-curves that meet at the centre of the vault.

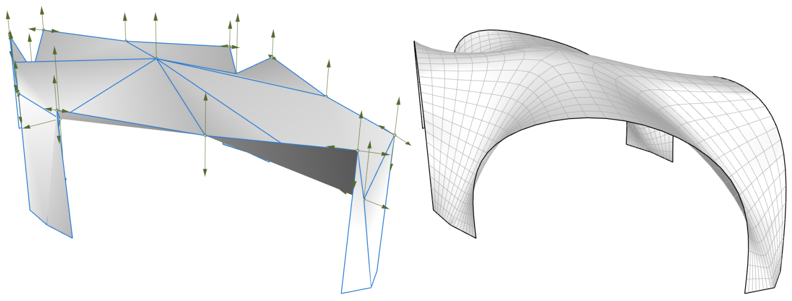


Figure 5: Inputs for complex case study: control mesh and control point variation vectors (left, vectors in green), and resulting smoothed mesh that is structurally analysed (right).

Table 3 shows the performance metrics on the complex test case, averaged from 8 independent runs. The increased complexity of this test-case has had an impact on the feasible coverage of the

FI-MAP-Elites approach. However, more than half of the examined domain has been covered with feasible solutions in this approach (64%). On the other hand, the SOO's feasible coverage remains at a very low value, suggesting that the solutions discovered in SOO are very similar to each other in every run. Furthermore, the feasible population discovered by the QD approach exhibited better performance in terms of both elastic energy and displacement, in comparison to SOO. This observation is aligned with the general notion that diversification in evolutionary approaches is a means of achieving better fitness.

Table 3. **Complex Test Case**: Comparing the final archive of QD with the last population of SOO. Error signifies a confidence interval of 95%. Displacement (*d*) is reported in centimetres.

	FI-MAP-Elites	SOO
Feasible Coverage	66.9% ± 1.5%	0.8% ± 0.3%
Feasible QD-Score	110.66 ± 2.18	1.46 ± 0.56
Feasible U_{min}	0.34 ± 0.01	0.45 ± 0.02
Feasible σ_U	0.21 ± 0.02	0.01 ± 0.00
Feasible d_{min}	0.15 ± 0.01	0.27 ± 0.02
Feasible σ_d	0.11 ± 0.01	0.01 ± 0.00
Feasible Ratio	49.5% ± 0.4%	100.0% ± 0.0%
Infeasible Coverage	68.3% ± 1.7%	0.0% ± 0.0%

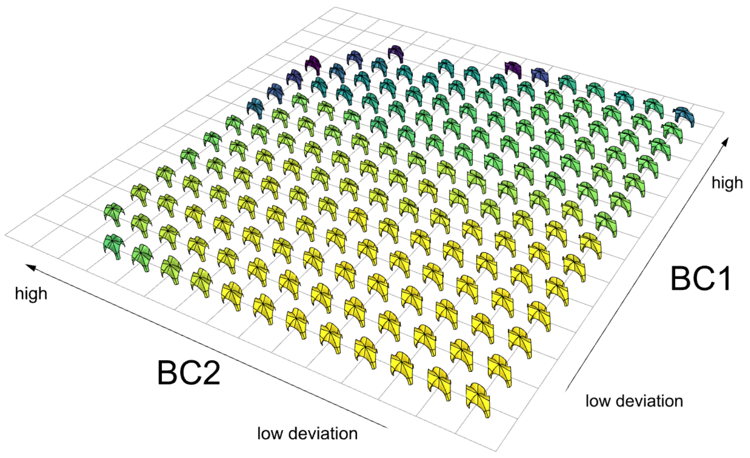


Figure 6: Visualization of all feasible solutions found in a single run of the FI-MAP-Elites algorithm on the Complex Test Case. Meshes in yellow have a higher fitness (i.e. lower elastic energy)

As seen in Fig. 6, the highest-fitness solutions are concentrated near the regions of low deviation for both sets of control points. Furthermore, we observe a clear gradient that starts from this point and diminishes the further away we move from it. This observation is aligned with an engineering intuition that more symmetric solutions will exhibit better structural properties. Despite this fact, however, it is worth noting that all presented solutions are feasible and locally optimized, rendering them potential candidates for further investigation.

Finally, in Fig. 7 we present the best-found individual solutions of the FI-MAP-Elites and SOO approaches, for a detailed examination. In these examples the QD algorithm found a lower deformation solution than the SOO. The QD optimal solution has sharper curvature at the two supports in the short-span axis, but achieves a shallower angle into the two long-axis supports. As these results suggest, the best range of solutions for high-dimensional problems like this can often not be easily intuited by the engineer, unlike the simple test cases.

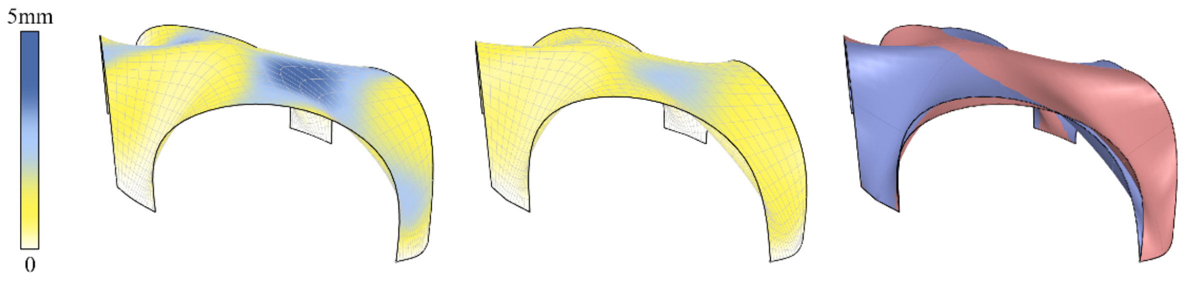


Figure 7: Best solution found by the SOO (left) and by FI-MAP-Elites (middle), with blue regions representing high displacement. Both solutions overlaid (right) with blue color for FI-MAP-Elites and red color for SOO, to highlight their difference.6. Discussion

This paper introduces a constrained Quality-Diversity evolutionary approach in structural engineering and tests it on the problem of designing shell surfaces. Unlike traditional evolutionary optimization, which converges towards similar solutions, our method produces many diverse solutions and improves them throughout evolution. It uses a two-population approach to explore the solution space efficiently and discover feasible solutions. It produces many individuals, with a high coverage of the designer-specified behavioural space of geometric properties. The produced visualization of all possible layouts can provide important feedback to engineers and inspire future iterations. As shown in some use-case example shell structures, this approach can offer many good alternatives for an engineer to choose from, compared to the single design produced by single-objective evolution.

The experiment setup included three test cases: two variants of a simple problem and a more complex problem closer to the real-world structural engineering problems encountered by AKT II [18],[19]. We compared the output of FI-MAP-Elites against designs collected from a traditional single-objective optimization (SOO) method available to all engineers through the Galapagos optimizer (in Rhino/Grasshopper). While the objective function included both the design goal and the constraints, through a form of fitness penalty typical in constrained optimization [16], other evolutionary approaches may perform differently. For instance, a two-population approach such as FI-2pop GA is expected to find feasible individuals even in the complex case study, where SOO struggled. Moreover, traditional genetic diversity preservation mechanisms such as the Fitness Uniform Selection Strategy [22] could also lead to more diverse results. Finally, a multi-objective approach [23] where the BCs are used as objectives alongside the SOO objective function (F_b) may lead to a more diverse population. On the other hand, since the best solutions were found at specific BC coordinates with low BC scores, optimizing towards high BCs may not lead to any improvements at a higher computation cost. Such more complex baselines will be explored in future work, where more complex structural engineering problems will be tested with FI-MAP-Elites.

Results of FI-MAP-Elites as a method of illuminating the design space and generating many good alternatives is promising for the structural engineering test cases, including the complex one which is close to real-world problems faced by professionals. However, there are many other problems within the AEC domain that could benefit from FI-MAP-Elites that should be explored in future work. Preliminary studies in illuminating the space of stick models for structural engineers with FI-MAP-Elites [24] have shown that the method is valuable for simple test cases but does not scale for more complex problems due to the additional effort for designers in defining the entire possibility space. The FI-MAP-Elites has also been tested for MEP engineering towards designing ventilation shafts [24], operating under strict design spaces provided by the designers; this allows designers to ensure that solutions satisfy their ad-hoc constraints but limits the potential of QD search to provide truly surprising solutions to the designers. Beyond testing FI-MAP-Elites in other engineering tasks, a more ambitious avenue for future work is testing the output of the algorithm (e.g. the feature maps of Fig. 4) in real-world, everyday workflows of a professional setting and their applicability as inspiration for engineers, for communicating between disciplines (e.g. showing a feature map of alternatives to other AEC stakeholders, such as architects), and for communicating with clients. This ambitious direction of inquiry would shed more light on the potential of Quality-Diversity search and the improvements needed to the algorithm, visualization, explainability [25], and tools (e.g. Rhino plugin).

7. Conclusions

This paper presented a novel approach to structural engineering, testing a new algorithm that takes advantage of recent advances in constrained optimization and quality-diversity (QD) evolutionary search. The method shows promise in both simple and complex test cases and can produce a range of geometrically diverse high quality solutions, which can be beneficial during conceptualization stages and when exploring alternatives with different stakeholders. The algorithm is integrated into the Grasshopper visual programming environment within Rhino, which makes it both easily accessible and customizable by engineers, and also provides tools to intuitively visualize and inspect all generated solutions. Compared to a traditional single-objective optimization method currently available in Rhino, our constrained QD approach can find feasible solutions more easily and produce far more viable alternatives, compared with a single optimum that the single-objective approach converges to. Future work will explore converting FI-MAP-Elites into a plugin available for everyone via online repositories (e.g. food4rhino.com) and assessing its applicability further in everyday structural engineering workflows.

Acknowledgements

This research was carried out as part of *PrismArch*, a Horizon2020 research project. The project received funding from the European Union's Horizon 2020 research and innovation programme, under grant agreement No 952002. Konstantinos Sfikas is supported by the European Union's H2020 research and innovation programme (Grant Agreement No. 951911).

References

[1] F. Otto and B. Rasch, *Finding Form: Towards an Architecture of the Minimal*, Axel Menges 1996.

[2] M. Burry, *The Expiatory Church of the Sagrada Famíli*a, Phaidon Press 1993.

[3] D. Piker, "Kangaroo: Form Finding with Computational Physics", in *Architectural Design*, vol. 83 no. 2, 2013.

[4] R. Vierlinger, "Multi Objective Design Interface", Master's Thesis, TU Wien, 2013.

[5] M. Kyropoulou, "Shading Design for Outdoor Learning in Warm and Hot Climates Using Evolutionary Computation: A Case Study In Houston TX," in *Proceedings of the Annual Modeling and Simulation Conference (ANNSIM)*, 2022, pp. 682–693.

[6] J. K. Pugh, L. B. Soros, and K. O. Stanley, "Quality Diversity: A New Frontier for Evolutionary Computation," *Frontiers in Robotics and AI*, vol. 3, 2016.

[7] J. Lehman and K. O. Stanley, "Evolving a Diversity of Virtual Creatures through Novelty Search and Local Competition," in *Proceedings of the Annual Conference on Genetic and Evolutionary Computation*, 2011.

[8] A. Liapis, G. N. Yannakakis, and J. Togelius, "Constrained Novelty Search: A Study on Game Content Generation," *Evolutionary computation*, vol. 23, no. 1, pp. 101–129, 2015.

[9] J.-B. Mouret and J. Clune, “Illuminating search spaces by mapping elites,” *arXiv preprint 1504.04909*, 2015.

[10] M. C. Fontaine and S. Nikolaidis, “Differentiable Quality Diversity,” *arXiv preprint 2106.03894*, 2021.

[11] D. Gravina, A. Khalifa, A. Liapis, J. Togelius and G. N. Yannakakis: "Procedural Content Generation through Quality-Diversity," in *Proceedings of the IEEE Conference on Games*, 2019.

[12] S. O. Kimbrough, G. J. Koehler, M. Lu, and D. H. Wood, “On a Feasible–Infeasible Two-Population (FI-2Pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch,” *European journal of operational research*, vol. 190, no. 2, pp. 310–327, 2008.

[13] K. O. Stanley and R. Miikkulainen, “A Taxonomy for Artificial Embryogeny,” *Artificial life*, vol. 9, no. 2, pp. 93–130, 2003.

[14] K. Sfikas, A. Liapis, and G. N. Yannakakis, “A General-Purpose Expressive Algorithm for Room-Based Environments,” in *Proceedings of the 17th International Conference on the Foundations of Digital Games,* 2022.

[15] K. De Jong, “Evolutionary Computation: A Unified Approach,” in *Proceedings of the Conference [23]Companion on Genetic and Evolutionary Computation*, 2012, pp. 737–750.

[16] A. Coello Coello, “Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art,” 2002, *Computer Methods in Applied Mechanics and Engineering* 191(11-12):1245-1287.

[17] E. Catmull and J. Clark, “Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes,” in *Seminal Graphics: Pioneering Efforts That Shaped the Field*, Association for Computing Machinery, 1998, pp. 183–188.

[18] P. Schumacher, “The Congeniality of Architecture and Engineering”, in *Shell Structures for Architecture: Form Finding and Optimization*, Routledge 2014.

[19] J. Janssen and R. Parker, “Hybrid Shells”, in *Design Engineering Refocused*, Wiley 2017, H. Kara and D. Bosia, Eds.

[20] http://www.rhino3d.com

[21] http://www.karamba3d.com

[22] M. Hutter, “Fitness uniform selection to preserve genetic diversity,” in *Proceedings of the 2002 Congress on Evolutionary Computation*. pp. 783–788.

[23] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[24] K. Sfikas, A. Liapis, G. N. Yannakakis and J. Hilmersson, “Final revised version of parametric space of design, algorithms for AI assisted editing/design in VR, and algorithms for designer modelling”, Technical Report, https://prismarch-h2020.eu/download/497 .

[25] J. Zhu, A. Liapis, S. Risi, R. Bidarra and G. M. Youngblood: “Explainable AI for Designers: A Human-Centered Perspective on Mixed-Initiative Co-Creation,” in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2018.