



## **PrismArch**

### **Deliverable No D2.1**

#### **Initial version of parametric design space**

<b>Project Title:</b>	PrismArch - Virtual reality aided design blending cross-disciplinary aspects of architecture in a multi-simulation environment
<b>Contract No:</b>	952002 - PrismArch
<b>Instrument:</b>	Innovation Action
<b>Thematic Priority:</b>	H2020 ICT-55-2020
<b>Start of project:</b>	1 November 2020
<b>Duration:</b>	24 months
<b>Due date of deliverable:</b>	Month 7, 31 May 2021
<b>Actual submission date:</b>	13/07/2021
<b>Version:</b>	1.0
<b>Main Authors:</b>	Antonios Liapis, Konstantinos Sfikas, Georgios N. Yannakakis



Project funded by the European Community under the H2020 Programme for Research and Innovation.



Deliverable title	Initial version of parametric design space
Deliverable number	D2.1
Deliverable version	Final
Contractual date of delivery	31 May 2021
Actual date of delivery	13 July 2021
Deliverable filename	PrismArch_D2.1_0.98
Type of deliverable	Report
Dissemination level	PU
Number of pages	120
Workpackage	WP2
Task(s)	T2.1: Parametric definition of the solution space using the principles, restrictions and rules of each design discipline T2.2: AI-assisted content creation and design suggestions based on evolutionary algorithms
Partner responsible	UoM
Author(s)	Antonios Liapis (UoM), Konstantinos Sfikas (UoM), Theodore Galanos (UoM), Georgios N. Yannakakis (UoM)
Editor	Konstantinos Sfikas (UoM)
Reviewer(s)	Helmut Kinzler (ZHA), Daria Zolotareva (ZHA), Risa Tadauchi (ZHA), Aleksandra Mnich-Spraiter (ZHA, Jeg Dudley (AKT), Edoardo Tibuzzi (AKT), Arun Selvaraj (Sweco), Dinos Ipiotis (Sweco), Oussama Yousfi (Sweco)

Abstract	The objective of this document is to define the parametric space that artificially intelligent algorithms will explore, as well as mathematical formulas for measuring constraint satisfaction, efficiency, and diversity in the different design disciplines contributing to PrismArch. The document surveys how current approaches in AI for architecture define representations, functional objectives and constraints, and identifies formulas, code libraries, and software that can be used for the purpose of AI-assisted content creation in PrismArch. This document takes input from the user and functional requirements of PrismArch described in D1.1 (“Report on current limitations of AEC software tools, leading to user and functional requirements of PrismArch”).
Keywords	Artificial Intelligence, Parametric Design, Possibility Space, Design Intelligence, Spatial Analytics, Fitness Functions, Constraints, Evolutionary Algorithms, Diversity Measures

## Copyright

© Copyright 2020 PrismArch Consortium consisting of:

1. ETHNIKO KENTRO EREVNAS KAI TECHNOLOGIKIS ANAPTYXIS (CERTH)
2. UNIVERSITA TA MALTA (UOM)
3. ZAHA HADID LIMITED (ZAHA HADID)
4. MINDESK SOCIETA A RESPONSABILITA LIMITATA (Mindesk)
5. EIDGENOESSISCHE TECHNISCHE HOCHSCHULE ZUERICH (ETH Zürich)
6. AKT II LIMITED (AKT II Limited)
7. SWECO UK LIMITED (SWECO UK LTD)

This document may not be copied, reproduced, or modified in whole or in part for any purpose without written permission from the PrismArch Consortium. In addition to such written permission to copy, reproduce, or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

All rights reserved.

## Deliverable history

Version	Date	Reason	Revised by
0.1	15/4/2021	Table of Contents	Antonios Liapis (UoM)
0.8	15/5/2021	Beta version for review	Antonios Liapis (UoM)
0.9	25/6/2021	Revised version for review	Konstantinos Sfikas (UoM)

0.95	25/6/2021	Revised version for review	Georgios N. Yannakakis (UoM)
0.97	5/7/2021	Revised version for review	Konstantinos Sfikas (UoM)
0.98	11/7/2021	Revised version for review	Konstantinos Sfikas (UoM)
1.0	13/7/2021	Proof reading	Spiros Nikolopoulos (CERTH)

## List of abbreviations and Acronyms

Abbreviation	Meaning
AI	Artificial Intelligence
PCG	Procedural Content Generation (in Games)
EC	European Commission
IP	Intellectual Property
ToC	Table of Contents
WP	Workpackage
AEC	Architecture, Engineering and Construction
BIM	Building Information Modelling
CAD/CAM	Computer-Aided Design & Computer-Aided Manufacturing
VR	Virtual Reality
MEP	Mechanical, Electrical, Plumbing

## **Executive Summary**

This deliverable, grounded in a review of the state-of-the-art in the pertinent areas, aims to map out the parametric space that artificially intelligent algorithms can explore, as well as mathematical formulas for measuring constraint satisfaction, efficiency, and diversity in the different design disciplines contributing to PrismArch. The document surveys how current approaches in AI for architecture define representations, functional objectives and constraints, and identifies formulas, code libraries, and software that can be used for the purpose of AI-assisted content creation in PrismArch. This document takes input from the user and functional requirements of PrismArch described in D1.1 (“Report on current limitations of AEC software tools, leading to user and functional requirements of PrismArch”).

## Table of Contents

<b>1 INTRODUCTION</b>	<b>11</b>
<b>1.1 PrismArch Overview</b>	<b>11</b>
<b>1.2 Context of this Deliverable</b>	<b>12</b>
1.3 Method for Collecting Data in this Deliverable	13
1.4 Structure of the Deliverable	13
<b>2 DEFINING THE REPRESENTATION AND THE CONCEPTUAL SPACE</b>	<b>16</b>
<b>2.1 Survey</b>	<b>16</b>
2.1.1 Design Problem Solving	16
2.1.2 Grid-based Representations	17
2.1.3 Graph-based Representations	19
2.1.4 Polygon-based Representations	20
2.1.5 Pixel-based Representations	20
2.1.6 Indirect Representations	21
2.1.7 Subdivision-based Representations	24
2.1.8 Representations based on Boolean Operations	24
2.1.9 Dual Representations	26
2.1.10 Ad-hoc Representations	28
<b>2.2 PrismArch applications</b>	<b>29</b>
2.2.1 Problem representation	30
2.2.2 Solution representation	30
2.2.3 Generation and mutation operators	30
2.2.4 Representation and user interaction	31
2.2.5 Data Collection during User Interactions	32
<b>2.3 Software, Tools and Algorithms</b>	<b>33</b>
<b>3 FUNCTION EVALUATIONS AND CONSTRAINTS</b>	<b>36</b>
<b>3.1 Survey</b>	<b>36</b>
3.1.1 Inherent constraints and function of architectural design	36
3.1.2 Structural Engineering	40
3.1.3 MEP engineering	45
Coordination between MEP disciplines:	45
MEP Systems optimization:	46
MEP service life optimization:	46

3.1.4 Sustainability	47
3.1.5 Conclusions	50
<b>3.2 PrismArch applications</b>	<b>50</b>
3.2.1 Constraints Arising from Inter-Disciplinary Collaboration	51
Inter-disciplinary constraints from the perspective of architectural design:	51
Inter-disciplinary constraints, from the perspective of structural engineering:	52
Inter-disciplinary constraints, from the perspective of MEP:	53
Inter-disciplinary constraints, conclusion:	53
3.2.2 Project-specific constraints	53
3.2.3 Fitness functions	54
Structural engineering:	54
MEP engineering:	55
Sustainability:	55
Conclusion:	56
3.3 Software, Tools and Algorithms	56
<b>4 DESIGN EXPLORATION DIMENSIONS</b>	<b>60</b>
<b>4.1 Survey</b>	<b>60</b>
4.1.1 General visual properties	60
4.1.2 Aspects of Space	63
4.1.3 Architectural space heuristics	63
<b>4.2 PrismArch applications</b>	<b>68</b>
4.2.1 Input from ZHVR:	69
4.2.2 Input from AKT II:	69
4.2.3 Input from Sweco:	69
4.2.4 Proposed dimensions of diversity:	70
Direct geometric evaluations:	70
Isovist and Visibility Graph - based evaluations:	70
4.2.5 Conclusion	70
4.3 Software, Tools and Algorithms	71
<b>5 REALIZING QUALITY-DIVERSITY AND DESIGNER MODELING IN PRISMARCH</b>	<b>72</b>
5.1 Algorithmic Background	72
5.1.1 Quality Diversity Algorithms	72
5.1.2 Designer Modeling	73



5.2 Vision arising from Workshops with AEC industry partners	74
5.2.1 Envisioning QD in PrismArch - ZHVR	74
5.2.2 Envisioning QD in PrismArch - AKT II:	76
5.2.3 Envisioning QD in PrismArch - Sweco:	78
5.2.4 ZHVR: Context of design parameters	79
5.2.5 ZHVR: Context of constraints:	79
5.2.6 ZHVR - context of optimization:	80
<b>6 CONCLUSIONS AND FUTURE STEPS</b>	<b>83</b>
<b>7 REFERENCES</b>	<b>86</b>
<b>APPENDIX A: QUESTIONNAIRE AND RESPONSES</b>	<b>100</b>
A.1 Original Questionnaire provided to partners	100
Questions:	100
A.2: Responses from ZHVR	102
A.3: Responses from AKT II	111
Responses from Jeg Dudley (AKT II)	111
Responses from Edoardo Tibuzzi (AKT II)	116
A.4: Responses from Sweco	118



## ▪ 1 INTRODUCTION

### ○ 1.1 PrismArch Overview

The overarching goal of the PrismArch project is to achieve a “prismatic blend” between aesthetics, simulation models and meta-information that can be presented in a contextualized and comprehensive manner in virtual reality (VR). This blend will allow for collaborative manipulation of the design and accurate assessment of new design decisions. This goal passes through intuitive interactions in a VR world with blended graphics across various types of simulation software that satisfies the needs of all types of designers in parallel. PrismArch will create a VR-aided design environment that supports the major disciplines that are typically engaged in an architectural project (architects, structural and MEP engineers), facilitates the effective realization of an architectural project, and enhances the overall decision making process through an action and reaction paradigm.

As editing single items in VR with controllers is time consuming, PrismArch expects to leverage artificial intelligence (AI) as an assistive technology driven by the user in order to edit items collectively and in an informed manner. AI tools will primarily be incorporated into PrismArch for **assistive design** through the formulation of the design procedure as a parametric space problem. The overarching goal of a VR-aided environment for collaborative editing permitted by multiple users of different disciplines raises new challenges, as every modification can impact a wide variety of elements in an architectural project. An aspect of the AI algorithms developed in PrismArch is to **detect and address potential conflicts** and the failure of hard constraints on the design’s function. When such conflicts or constraints are detected, the AI tools will inform the corresponding author (architect or engineer) and suggest potential solutions. By obtaining the constraints that should overrule such a project, provided by the AEC experts, a set of feasible solutions can be formed. This will be achieved through parameterizing the design process for the targeted disciplines in order to define a multidimensional solution space, and then generate appropriate alternatives to the user’s designs and present them as suggestions which the users can take into account for adjusting their design. In order to best assist the designers in their decision-making, AI tools will visualize this solution space by defining **design exploration dimensions** and generate diverse solutions that satisfy the functional requirements (as hard constraints) but exhibit different styles in order to provide **more, and more diverse, suggestions** for a designer to choose from or be inspired by. PrismArch will take advantage of **evolutionary computation** [1] to search the solution space driven by measures of quality and diversity. By exploiting artificial evolution, the solution set will be confined and traversed fast and efficiently, allowing PrismArch to make real-time AI-assisted suggestions that can play a critical role either in conflict resolution or even serve as a driving force of inspiration. Finally, we recognize that not all disciplines or individual designers have the same needs, preferences, or style. The AI tools of PrismArch will incorporate models for each type of designer in order to adapt automatically the AI-generated content based on the preferences of the individual designer. This will lead to **personalized suggestions**, via machine learning and statistical models trained on real interaction data. The main dimension of adaptation will pertain to the design dimensions explored by the algorithm; by adapting stylistic priorities, different suggestions will be shown to each designer that better match their core preferences and style. Finally, a new form of designer modelling [2] will be explored in terms of different disciplines, which can also adjust the types of constraints prioritized depending on the discipline.

## ○ 1.2 Context of this Deliverable

This deliverable reviews the relevant literature in terms of the algorithmic representations of the design space of architectural projects, and defines in computational terms the solution space of each design problem (i.e., architectural, structural, or MEP). The deliverable follows the “prismatic blend” of PrismArch and identifies that design space representation, functional constraints, and designer preferences must be blended and integrated into an AI tool in order to facilitate the generation of suggestions that can be useful, feasible, diverse, and personalized. In order to identify the correct way to algorithmically represent the problem, to formulate the constraints, and to encompass a broad range of stylistic dimensions in a way that can be traversed by an evolutionary algorithm, we perform a survey of the related work on design and engineering optimization, as well as related fields on computer-aided design and machine learning.

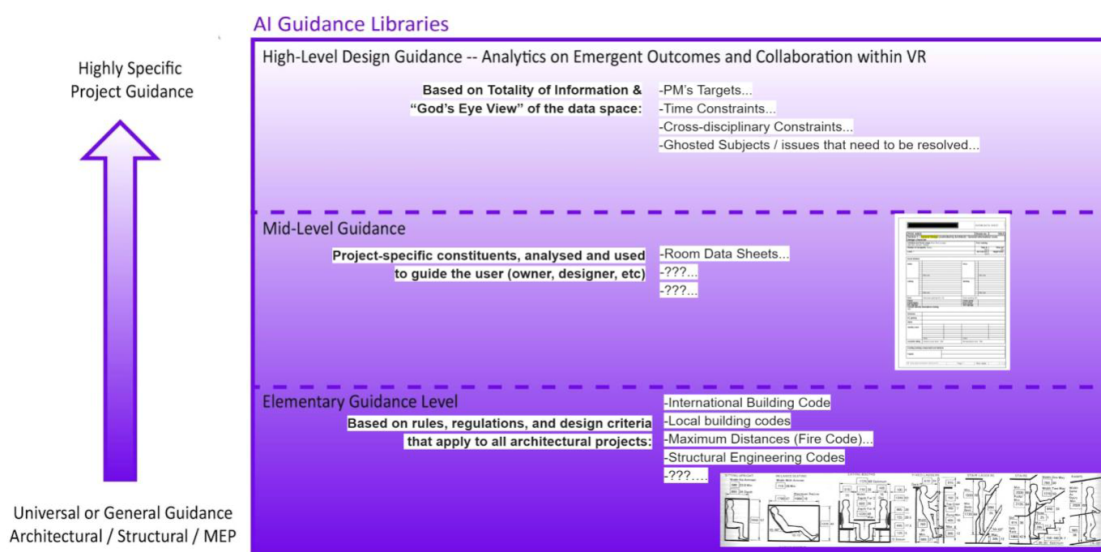


Figure 1: PrismArch AI guidance levels envisioned in D1.1.

The earlier deliverable D1.1 (“Report on current limitations of AEC software tools, leading to user and functional requirements of PrismArch”) reviewed the role of AI in the context of PrismArch and identified the different levels at which AI guidance can be deployed. As shown in Figure 1 (which is directly integrated from D1.1), there are varying levels of AI guidance that can be deployed, from low-level but general guidance on rule-based design criteria based on codes and regulations, to high-level but very project-specific guidance such as time constraints or cross-disciplinary constraints. In order to maximize the impact of AI guidance developed in PrismArch, D2.1 focuses mostly on the intermediate level (“mid-level guidance” in Fig. 1) which has the opportunity to be specific to a project by integrating designer-specified constraints in the form of room datasheets but also generalizable by e.g. swapping to new datasheets. However, constraints informed by regulations are also discussed in the context of functional constraints in D2.1. Moreover, D2.1 presents designer preferences that can be personalized through designer modelling [2]; this opens the potential of more project-specific guidance or author- and discipline-specific personalization of AI suggestions.

### ○ 1.3 Method for Collecting Data in this Deliverable

This deliverable studies past and current work on evolutionary computation for parametric design, revisiting studies from early years but mainly attempting to map out the state-of-the-art that can inform the PrismArch implementation of Quality-Diversity search and designer modeling. To collect the necessary materials to inform the survey conducted along three directions (representation, constraints, and exploration dimensions, as explained in Section 1.4), a thorough bibliographical search was conducted through Google Scholar, Semantic Scholar, Arxiv and our own networks of contacts in evolutionary computation. It should be noted that much of the state-of-the-art in both AI-assisted design and modeling originates from digital game development, and thus several of the surveyed papers come from academic conferences on Game Artificial Intelligence, in addition to traditional parametric design publications. Finally, relevant publications for generative and evolutionary art are explored, especially in Section 4, since they cover quantitative evaluations of stylistic choices and dimensions for exploration.

In order to better understand the expectations of the AEC partners of PrismArch and situate QD and designer modeling tools within their needs, a number of Masterclasses of the relevant AI technologies (see Section 5.1) were made and free-form workshops were held. These seminars and discussions were formalized through a questionnaire created by UM and completed by ZHVR, AKT II, and SWECO. The questionnaire, presented in full in Appendix A.1, aimed to collect information about practical constraints and exploration dimensions that arise during the process of a project, taking into account the requirements of cross-disciplinary collaboration. Partners' responses, provided in full in Appendix A, informed the PrismArch-specific suggestions for each of the three sections (Section 2, Section 3, and Section 4). Moreover, the questionnaire included more general questions (Questions 6 and 7 in Appendix A.1) regarding the grand vision of the application of the algorithms in question for each discipline. These responses warrant further analysis, and are discussed in Section 5.

### ○ 1.4 Structure of the Deliverable

This deliverable follows the premise of a “prismatic blend” between design space representation, functional constraints and designer preferences, and splits the survey of the state-of-the-art and the proposals for PrismArch AI-navigable design space into **three sections**. Section 2 focuses on the representation of the spatial and functional aspects of the design problem, and the operators that could allow evolution to modify and improve such aspects. The main survey of Section 2 is on the genotype-phenotype mapping [3] which can facilitate a compact representation of the design space in order for an evolutionary algorithm to explore, as well as on the genetic operators that can modify the genotype in controllable ways. Section 3 focuses on the functional aspects of the design problem and includes hard constraints on the requirements from a design or engineering perspective. Section 3 surveys related constrained engineering problems [4] and specifically architectural design, reviews relevant simulation-based approaches and functional metrics, and formalizes the constraints and mathematical formulations for bringing evolving designs closer to feasibility. Section 4 focuses on the stylistic preferences of a design project, relying as much on the specific disciplines of architecture or engineering as on broader cognitive aspects of perception and beauty [5]. This section identifies important visual features relevant to space, which can be quantified in a way that can help the evolutionary search towards diverse suggestions. The aggregation and weighting of these quantifiable formulations will also form the designer

models which can drive evolution towards personally interesting areas of the design space. Section 5 goes beyond the original intent of D2.1, which was to map out the design space, and discusses the algorithmic background and envisioned advances beyond the algorithmic state-of-the-art, based on workshops conducted by AI experts and AEC experts. The paper concludes with Section 6, while all questionnaires carried out for collecting the data from partners (see Section 1.3) are in Appendix A.



## ▪ 2 DEFINING THE REPRESENTATION AND THE CONCEPTUAL SPACE

This section focuses on the algorithmic representation of the design problem, in a way that an artificial intelligence can iterate upon and improve based on constraints or stylistic preferences presented in Sections 3 and 4 respectively. To a large extent, this section views problem representation from the perspective of evolutionary computation, which will be used for optimization and quality-diversity search [6,7].

Artificial evolution operates on a genotype, i.e. compressed information that is converted to a phenotype which is the final design artifact being evaluated and optimized. The representation thus largely concerns this **genotype-to-phenotype mapping** [3] and the ways in which the genotype can be modified towards better solutions in terms of its phenotype. These modifications to the genotype are referred to as **genetic operators** [1] and are largely grouped under **recombination** (when more than one genotypes exchange genetic material) and **mutation** (when one genotype changes stochastically). The representation of the genotype and the type of genetic operators that facilitate its evolution are critical to the performance of the algorithm and also depend heavily on the design problem at hand. Therefore, a survey of representations and operators from related fields of architecture, parametric design, and game level generation highlights the most prominent approaches.

### ○ 2.1 Survey

There is a broad variety of representations for spatial layouts and architecture. Extensive work has also explored evolutionary computation, genotype-to-phenotype mappings, and genetic operators in level design for computer games; such knowledge is directly transferrable to the goal of PrismArch. The following survey highlights a plethora of methods for representing and evolving content; however, it starts with a more general exposition of the thinking process that needs to be followed in order to define a problem space (Section 2.1.1).

#### ▪ 2.1.1 Design Problem Solving

In order to endow an artificial intelligence (AI) the ability to solve an architectural or engineering problem, a fundamental understanding of the principles of problem solving is necessary. Polya [8] identified four principles for problem solving: understanding the problem (i.e. restating it in terms understood by the problem solver), devising a plan (i.e. considering the steps needed to solve the problem as well as caveats and edge cases), executing the plan (until it is clear that it will not yield results) and reflecting on the outcomes (i.e. improving problem solving abilities based on successes or failures of the plan). The problem solving premise of Polya is well-suited for evolutionary search, where understanding the problem entails designing appropriate mathematical formulations of e.g. quality (fitness function) or constraints (feasibility evaluation), devising a plan refers to the design of the genetic operators which can move the candidate solutions closer towards a target outcome, executing the plan in the sense of performing the evolutionary run, and reflecting on the outcomes as the presentation to an end-user of the best evolved outcomes. Newell et al. [9] are even more explicit in their framing of problem solving as a search within a problem space. While Newell et al. focus on the conceptual processes of recognizing solutions, generate-and-test search and heuristic search, these terms and algorithmic processes are widely used in evolutionary content generation [10]. Importantly, Newell et al. do not associate problem



solving as random search, but instead as conducted by defining a set of operators for applying changes to a candidate solution, and choosing which operator to apply at which time. The evolutionary approach followed in PrismArch follows this method, defining genetic operators for changing a candidate solution; this Section describes the representation and genetic operators of such solutions.

Solving an architectural design problem, specifically, is a challenging endeavor which requires iteration and reflection. This is due to the fact that the design solution is “a result of negotiation between problem description and solution” [11]. Based on a chronological review of architectural design models, Lawson [12] suggested that such design problem solving processes follow an iterative, three-stage model of analysis, synthesis and evaluation. Problem understanding only occurs after multiple iterations of analysis, synthesis and evaluation; indeed, a conjecture of the design and the problem specification often proceed in parallel (influencing each other) throughout a design process [13].

Motta and Zdrahal [14] characterize a parametric design application as a mapping from a six-dimensional space to a set of solution designs. The six dimensions consist of (1) parameters, (2) value ranges of these parameters, (3) constraints, (4) requirements, (5) preferences, and (6) a global cost function. This formulation is especially relevant as framing for this deliverable, as Section 2 describes the parameters and value ranges, Section 3 describes the constraints, requirements, and global cost function while Section 4 describes the preferences of the problem.

### ▪ 2.1.2 Grid-based Representations

Grid-based representations divide the site into a grid of equal-sized cells and partition objects so that they can be located within a cell [15]. This is a popular representation for optimization because it is straightforward to encode in a computer program and is also predominantly used in computer games, where the grid consists of tiles of different types such as the gameworlds of Super Mario Bros. (Nintendo, 1985) or Sid Meier’s Civilization II (MicroProse, 1996). Grid-based representations have been especially prominent for automated content generation (in games and beyond) due to their straightforward encoding as a genotype.

Sentient Sketchbook [17] represents simple tile-based game levels in a grid-based representation, and encodes them directly in the genotype as a two-dimensional matrix of integers. Each integer is mapped to a specific tile type. This genotype-to-phenotype mapping is a direct encoding [3], as opposed to indirect encodings discussed in Section 2.1.6. Evolving such a genotype is straightforward, as small changes in the genotype result in small changes in the phenotype. In terms of genetic operators, Sentient Sketchbook uses both recombination and mutation (see Figure 2). Recombination uses a two-point crossover [1] which splits the map into three pieces and exchanges the middle piece with that of another individual. Mutation alters a small portion of the maps’ tiles: each tile has an equal chance of being swapped with a randomly chosen adjacent one, or changing the tile’s type. Other tile-based representations of levels follow a similar approach to Sentient Sketchbook. The evolutionary dungeon designer [18] represents rooms as an array of integers in the genotype and applies two-point crossover. Mutation in the case of the evolutionary dungeon designer is very high, however, and the mutation operators can either change a tile’s type or rotate the entire genotype by 180°.

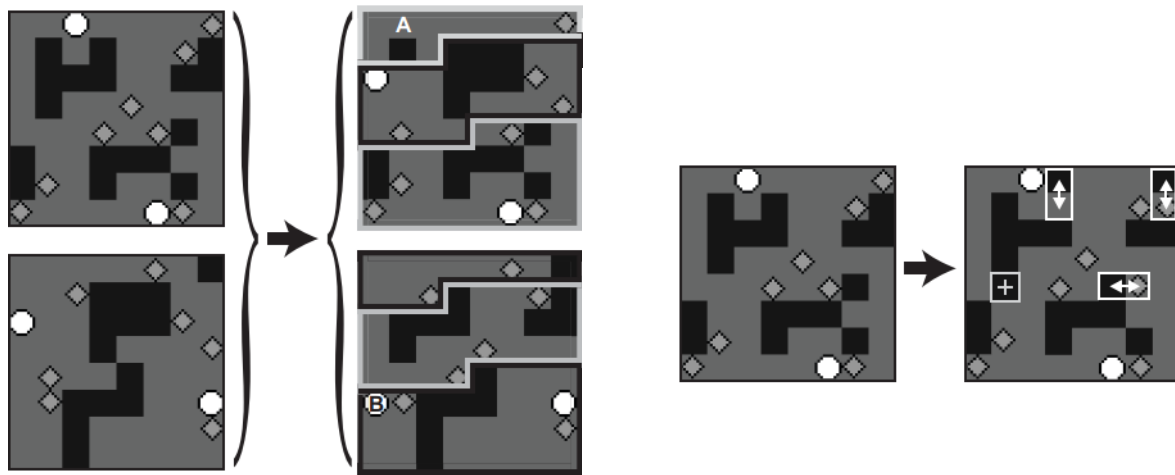


Figure 2: Genetic operators in Sentient Sketchbook [17]. Left: recombination through 2-point crossover results in two offsprings with parts of each parent level. Right: mutation may add or remove impassable tiles or swap adjacent tiles. Source: [157]

Ashlock et al. [19] highlight different ways of representing grid-based levels, including direct and indirect representations. Ashlock et al. differentiate between a direct representation where integers specify the type of each tile (at the minimum, a binary distinction between passable and impassable tiles), and a direct representation where integers specify the connectivity of each tile (i.e. which adjacent tiles can be accessed from this tile). The latter representation (termed “chromatic” by Ashlock et al.) allows for “walls” that do not take up an entire tile, and moreover allows for one-way paths. For the chromatic representation, Ashlock et al. use two operators: uniform crossover and uniform mutation. Uniform crossover iterates through two individuals’ genes simultaneously and has a probability of using one or the other individual’s genetic information in that location to produce an offspring. Uniform mutation iterates through one individual’s genes and has a probability of randomly assigning a tile’s type at that location.

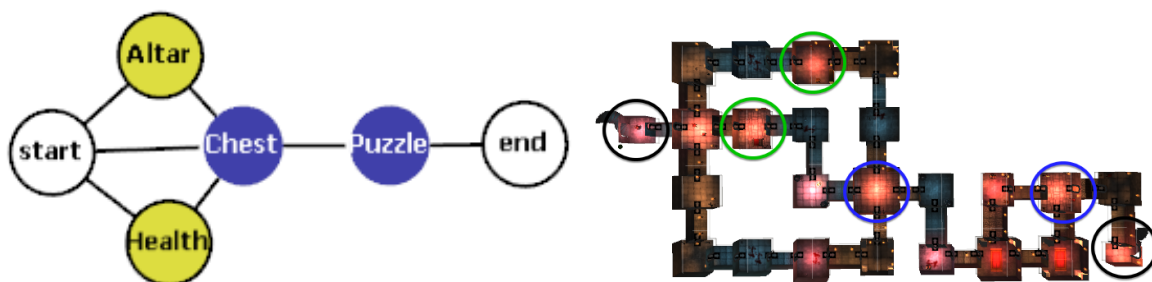
Sonancia [20] generates rooms for a horror game using a direct representation where each integer specifies the ID of the room. In addition to this direct tile-based representation, the genotype contains information about the location of doors (as tuple objects describing which rooms are interconnected) and monsters or quest items (as tuple objects describing their type (i.e. item or monster) and the room they are placed in). Sonancia evolves levels only via mutation: mutation can shift a room’s walls, divide rooms, connect rooms with doors or remove doors (two rooms can only be connected with one door), move monsters’ or items’ assigned rooms (ensuring one monster per room and one item per room) or add new monsters. After mutation is applied, a flood fill algorithm ensures that rooms are not disconnected and are sufficiently large; if not, the gene is repaired to assimilate small or disconnected rooms with adjacent ones, moving items or monsters as needed.

Another instance of direct representation from the field of games is the work of Karavolos et al. [21] on evolving multi-floor game levels. Due to the need to maintain elevation information, each grid location was assigned two integers on the genotype: one representing the floor it was on, and one representing whether it contained a game-specific item pickup. Moreover, while the phenotype is 20x20 tiles, the genotype represents it as a 4x4 grid of cells that each contain 5x5 tiles. Recombination is implemented by randomly picking a cell from either parent at each position of the cell grid. When applying mutation, each cell may be

mutated by one of the following variants: Move Powerup to another cell, Grow Cell or Erode Cell (in terms of either first-floor tiles or walls), Place Stairs next to a random first-floor tile, Place Block or Dig Hole (to add ‘chunks’ of tiles of the same elevation). An additional repair function is applied after recombination and mutation to ensure that unreachable areas (e.g. an island of first-floor tiles without a stair) becomes reachable.

### ▪ 2.1.3 Graph-based Representations

Understanding the most basic, high-level constraints and properties of an architectural design has often been accomplished through graph visualizations [22]. Graphs can contain information for each room and convey the connectivity with other rooms and spaces. While geometric graphs [23] (or spatial networks) contain information about the coordinates, sizes, and distances of the physical spaces, other graph representations can convey the high-level connections without having to conform to the spatial constraints of the problem [24]; we refer to the latter as non-geometric graphs to avoid confusion. Since non-geometric graphs regarding rooms and their connections are often provided by the client, there is limited work in actually optimizing such high-level abstractions. However, work in game content generation has explored an evolutionary graph expansion strategy in order to produce an abstraction of a video game level. In this work, Karavolos et al. [25] begin with a minimal graph representing the start and the end of a level and through evolution increase the graph by adding nodes with different in-game functions (e.g. puzzle room nodes, boss fight room nodes). The genotypes stores the information as a set of nodes and edges, with nodes storing the type of room and edges as tuples of the IDs of the two connected nodes and whether the edge is two-sided (i.e. allows players to go in either direction between connected rooms). Genetic operators included are only for asexual mutation, with options of adding or deleting a node, changing a node’s type, and adding and deleting edges between nodes. The resulting graph was then transformed into a playable level through a layout solver [26]. The layout solver ensures that the correct connections between rooms in the graph is maintained, but may add some or many empty rooms in-between in order to ensure that the spatial constraints are met (see Figure 3).



*Figure 3: Graph-based level generation by Karavolos et al. [25]. Left: the mission graph, evolved via graph evolution. Right: a spatial layout of the left graph, based on a constraint solver. The yellow nodes are shown in green and the blue nodes in blue. As noted, additional rooms are added in the spatial layout to account for geometric constraints. Source: [25].*

As noted above, geometric graphs have been particularly useful within the domain of architecture and especially floorplan layouts. Quality assessments regarding optimal connections, sizes, and distances based on a graph layout have been proposed by [27], although that work did not actually optimize these algorithmically. A variant of the geometric graph where the nodes are replaced with rectangular rooms of appropriate size has been

used in automated floorplan generation [28]. In this work by Arvin and House [28], rooms of different sizes are initially represented as appropriately sized circles and their connections as graph edges. Rather than optimize this layout, the authors performed a physics simulation by assuming that the edges were springs drawing all the nodes together, while the circles were solid and stopped rooms from coming too close together. This animation was deemed valuable for its “responsive design” [28], and constraints were inherently satisfied by making compact designs (due to springs drawing rooms together) while maintaining the connections between rooms. Once the physics simulation was completed, the circles were replaced with rectangular rooms of appropriate size centered around the circles’ locations.

#### ▪ 2.1.4 Polygon-based Representations

Ultimately, most of the results of automated design are rendered as polygons or pixels (see Section 2.1.5). However, polygons are not often used directly as genotypic information. For instance, much of the work on floorplan generation encodes the (polygonal) space as space partition trees that subdivide a larger space [29, 30, 31, 32, 33], an example of which (by Doulgerakis) is included in Section 2.1.7.

An intuitive approach for representing the space as polygons is by encoding the coordinates of rectangles that compose the space: for instance, Keatruangkamala and Sinapiromsaran [34] encode each room as a rectangle with the genotype containing its top left corner’s coordinates and its height and width. Michalek et al. [35] used a more complex way of representing rectangular “units” using an arbitrary reference point, the distance to each wall from that point (North, South, East, West) and the size of the windows on each of the four walls. Michalek et al. distinguished units into rooms (living spaces), boundaries (which contain other units), hallways (which do not have walls and connect other walled rooms), and accessways which are special types of hallways that intersect two units. It should be noted that the genotype does not contain information about the type of unit it represents (except implicitly based on which parameters are present, e.g. no walls for hallways) and instead the order in which values appear on the genotype are used to instantiate rooms based on a pre-authored mapping. Therefore, the genotype can control the dimensions and window placement of rooms but not the number or types of rooms. In both cases examined here, the fixed length of the genotype ensures that the right number and type of rooms is present as intended by the designer. However, the results are very constrained by the rectangular nature of the rooms and especially by the fact that the algorithm can not add or remove e.g. hallways when needed. A more indirect representation with a genotype of variable length would be preferable if more substantial changes to the layout would be desirable or when the optimal number of rooms and halls is not known in advance.

#### ▪ 2.1.5 Pixel-based Representations

Many of the 2D representations discussed in this Section are ultimately rendered as 2D images; however, the generator rarely operates on the level of pixels and usually operates on a higher level such as a grid representation (see Section 2.1.2). Recent advances in deep learning tend to exploit pixel regularities that they have discovered on a corpus of images they are trained on and generate new content at the pixel level [36]. While these approaches are not directly tied to evolutionary computation, which is the focus of this deliverable, it is worthwhile to survey a couple of examples that generate designs at the pixel level.

The first application for generating pixel-based representations of floorplans was by Huan and Zheng [37], who used a Pix2Pix generative adversarial network (GAN) to produce colored images that represented rooms, doors and windows of a layout. A dataset of plan drawings of apartments collected from a property website was converted into colorful images of colored “blocks”. Each color represented a room type, or walkways, doors, windows, and balconies. The authors explored the use of different inputs and desired outputs, such as the plan drawing as input and generating the color labeled map as output or vice versa. It should be noted that the resulting images were not used for a design goal, such as converting it to a CAD drawing or developing it further through AI or human intervention. Chaillou [38] introduced archiGAN, a multi-step pipeline via deep learning which translated an empty canvas to a building footprint, followed by a room split, and completing it with furnishings. The training set included over 700 annotated floor plans. Each transformation in the pipeline was handled via a generative adversarial network which was trained independently. It should be noted that results rendered as pixels often included artifacts (although the general shapes and colors were visible, and additional steps for rendering via additional GANs or through scripts for vectorization [39] could convert the result into an editable and usable result.

The final relevant deep learning approach for generating layouts at the pixel level is Graph2Plan [40]. Unlike the previous examples, Graph2Plan uses the RPLAN large-scale floorplan dataset containing more than 80,000 human-designed samples. These human-designed samples are processed in a scripted fashion to derive a graph representation by finding the size and position of each node (and its edges) from the actual floorplan. Each node also contains information on the type of room it represents, while the resulting image consists of colored blocks similar to Huang and Zheng [37] (but without extra elements such as doors and windows). The deep learning model uses as input the geometric graph (see Section 2.1.3) and the building boundary. The model produces a pixel-based representation of the floorplan, which is then processed through another network to convert the freeform colored pixel forms into boxes that more accurately represent rooms. This final step ensures that the tool can be integrated into a user interface that allows the user to create and edit floorplans by providing only an input boundary and a partial graph or a few adjacencies.

#### ▪ 2.1.6 Indirect Representations

An indirect representation encodes the necessary information to produce the phenotype in a compact genotype. For instance, if a phenotype is represented as a grid of tiles, a direct representation would contain information about each tile (see Section 2.1.2); if a phenotype is represented as a set of polygons, a direct representation would contain the coordinates of each point of each polygon. On the other hand, an indirect representation can produce the above phenotypes without storing all information of the phenotype.

The most popular and expressive indirect representations within evolutionary computation at large are arguably **genetic programming** and **neural networks**.

Genetic programming [41] produces computer programs, usually represented in syntax tree structure, that must be run in order to evaluate their quality. Genetic programming has been applied to many different tasks such as image processing [42, 43], game playing [44, 45], and hardware component design [46, 47]. GP has also been applied to evolve spatial layouts. The premise of modularity in designs by Le Corbusier has been an inspiration for genetic programming methods. A notable example is by Asojo [48], who encoded program rules based on the main elements of Le Corbusier’s architectural style and used genetic

programming to combine them into a specific 3D structure. This example uses shape grammars that are well-defined by the (expert) developer. Similarly, Coates and Makris [49] encoded the Dom-ino house by LeCorbusier as a set of “boolean (constructive solid geometry) operations” [49] and used genetic programming to create combinations of 13 copy and move functions to produce 3D “variant” Dom-ino houses through interactive evolution [50]. Moving from 3D structures to 2D floorplan optimization, Jagielski and Gero [51] use Genetic Programming to evolve floorplans on multiple floors through an ad-hoc mapping between genotype and phenotype and is further discussed in Section 2.1.10. Moreover, Doulgerakis applied genetic programming to create floorplans through space subdivision which is discussed in Section 2.1.7.

Artificial neural networks (ANNs) are function approximators which can represent any output as a non-linear function of its input. In generative art and music, neural networks have been applied to produce images by assigning colors using the pixel coordinates as inputs [52] or by using an underlying human melody to produce drum or instrument accompaniments [53]. Specifically, the ANNs used for such generative art tasks encoded additional activation functions than those common in e.g. machine learning or deep learning. The compositional pattern-producing network [3] (CPPN) includes activation functions that can represent patterns such as symmetry, reflection, or repetition. ANNs and CPPNs as genotypes have a very indirect mapping with the phenotype; indeed, changing the set of inputs that are processed by the ANNs can still lead to valid results and make the representation capable of infinite resolution [54]. CPPNs and ANNs are usually evolved through the neuroevolution of augmenting topologies approach [55] (NEAT), which includes specialized recombination and mutation operators. Recombination can only be applied between networks that are structurally similar, and merges the two network structures (increasing the size of both individuals) while inheriting randomly between the two individuals if they have edges or nodes that match. Mutation can add edges, nodes, change the weights of edges, or randomly assign a new activation function on a node. In terms of design problems, ANN-based representation has been applied to generate terrain, 2D and 3D models; these are the most relevant representations to PrismArch and are discussed below.

ANNs have been used to create heightmaps for game terrain in Sentient World [56]. A heightmap operates on a grid layout and specifies the height of the terrain in each tile (see Figure 4); it can be mapped in two dimensions by creating colored bands (e.g. low height values can be mapped as water, very high values can be mapped as mountaintops, etc.) or as a 3D terrain. In Sentient World, the ANN representation could be mapped to any resolution of game terrain by specifying the grid size of the phenotype. Specifically, the  $\{x,y\}$  coordinates of each tile’s midpoint was passed as input to the ANN, and a single output determined that tile’s height. Sentient World was a design tool, allowing the designer to specify the high-level patterns (on a low-resolution grid) and tasked the ANN to replicate as many of these patterns as possible in a higher-resolution grid. The tool thus took advantage of the infinite resolution potential of ANN to the fullest. ANNs were evolved towards novelty via the NEAT algorithm, and then trained via backpropagation to match the patterns of the designer’s canvas.

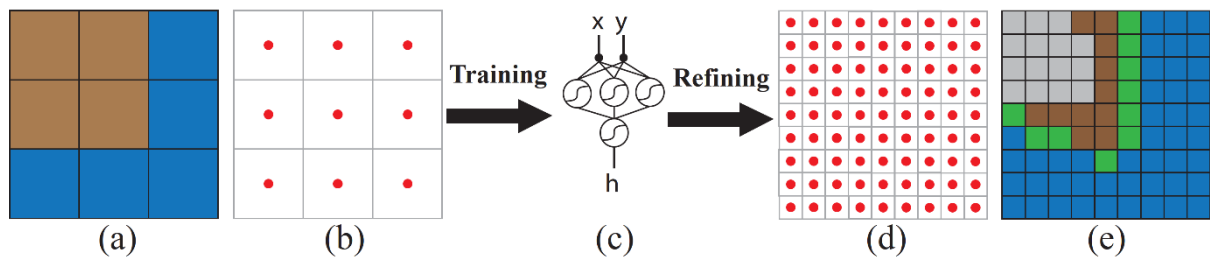


Figure 4: Terrain generation based on artificial neural networks in *Sentient World* [56]. The tiles' centers of the low-resolution grid (a and b) are processed through a neural network (c) to predict the terrain's height (low as blue, high as brown). The trained ANN can produce a higher-resolution image (e) if tiles' centers at a higher resolution are provided. Source: [56]

CPPNs were used to create polygons in 2D for a game-specific problem of spaceship design [130]. The spaceship was represented as a polygonal mesh, with specific constraints regarding e.g. that the mesh is contiguous and has no holes (see Figure 5). A similar approach was followed to generate polygons of flowers [57], with an even stricter generative representation to ensure symmetries.

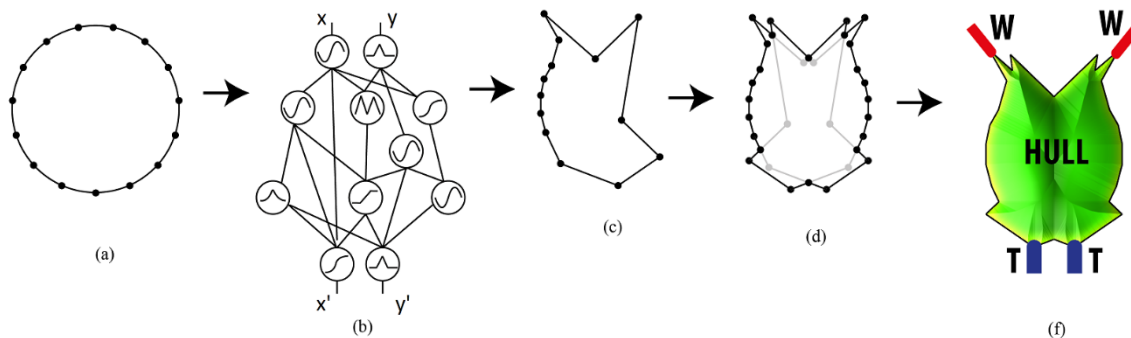


Figure 5: Spaceship polygon generation by Liapis et al. [130]. The vertices of a circle are processed by a CPPN which produces new coordinates for them. Then, a reflection is merged with the previous polygon to create a symmetrical shape (d), and thrusters and weapons are added (e). Adapted from [130].

CPPNs were used to create 3D structures which were evolved interactively on a public website named *Endless Forms* [58]. The mapping from CPPN to the phenotype (which is a lattice of voxels that can be empty or solid) is similar to *Sentient World*: the  $\{x,y,z\}$  coordinates of a predefined lattice is given as input to the CPPN, with the output being a single real number which maps to solid or empty depending on whether it is above or below an ad-hoc threshold. The 3D structures are on a  $10 \times 10 \times 20$  voxel lattice, and thus appear blocky: a marching squares algorithm is applied to create a smoother transition and add curves to the final 3D artefact shown to the users. Through the *Endless Forms* website, users can choose others' creations and evolve them further via interactive evolution [50] and then submit them for 3D printing and produce a physical artifact of their work. Grbic et al. followed a similar approach via an evolving ANN, using the  $\{x,y,z\}$  coordinates of a predefined 3D lattice and receiving as output the type of block of a *Minecraft* (Mojang, 2011) building to place at that position. In this case, the ANN returned multiple outputs (one per block type or per block type rotation, in case orientation of the blocks mattered) and the block type with the highest value in the output was chosen.

Another indirect representation for generating terrain heightmaps is introduced in [59]. This generative representation evolves a set of rules for cellular automata [60] which can

recursively subdivide a space into progressively smaller segments and define the height at their center-points. Operators for this representation included 2-point crossover for recombination, and a point mutation that had a chance to change every rule by tweaking one of the nine parameters that defined it.

#### ▪ 2.1.7 Subdivision-based Representations

An indirect form of encoding that is so widely used that it requires its own section is space partitioning or subdivision. As Koenig and Knecht define it, “a subdivision algorithm is understood as the recursive division of an area into smaller rectangular areas by edge-parallel slicing” [30]. The point and direction of the slicing can be randomized, determined by rules, or encoded in the genotype. The vast majority of the literature focuses on subdivisions along a horizontal or vertical axis, and always dividing a rectangular bounding box into smaller rectangles. Evolutionary computation has often been used with subdivision algorithms [29, 30, 31, 32, 33], as it is straightforward to encode the slices’ coordinates and direction into a genotype and apply them in the order that they appear in the genotype. We focus instead on certain subdivision approaches that introduce some innovations.

Doulgerakis applied genetic programming to iteratively partition a predetermined space into smaller subdivisions. Unlike most other approaches, the representation of Doulgerakis included non-orthogonal slashes that could create more interesting layouts. The subdivisions were assigned as rooms; the purpose of each room and the general quality of the layout was determined from a pre-authored specification file describing the types of rooms required, their connections, and their surface ratios. The genotype of Doulgerakis was modified via recombination or mutation. Recombination followed the premise of one-point crossover by selecting a random breakpoint in each tree-structure and exchanging the sub-trees between individuals. Mutation chose a random break-point in the tree-structure, removed the sub-tree and replaced it with a random tree-structure at that location.

Another interesting partition of the space (although not, in essence, subdivision) comes from Schoenauer [61], which encoded a space as a set of Voronoi cells. The variable-length genotype consisted of tuples with the coordinates of the Voronoi midpoint and a binary value defining it as solid or empty. This representation was not applied for floorplan generation in this example, but instead for the engineering problem of the cantilever plate [61]. Inoue and Takagi use a similar approach, saving only the centers of the rooms and applying a flood-fill style expansion to expand each room in four directions. While Inoue and Takagi use a grid map for the phenotype, the genotype is a compact indirect representation (where any floorplan resolution can be mapped to a preset number of rooms’ central coordinates) which is deterministic and will always produce the same layout based on the scripted expansion rules. Similar to subdivision algorithms, the floodfill by Inoue and Takagi ensures that the entire space is occupied by rooms; unlike subdivision, however, non-rectangular rooms can occur in this way. It should be noted that Sonancia [20] (see Section 2.1.2) expands rooms outwards in a similar fashion, however this expansion is performed by a genetic operator and thus a direct representation of all tiles of the floorplan is required in the genotype.

#### ▪ 2.1.8 Representations based on Boolean Operations

Another indirect form of encoding that in some ways is the opposite of subdivision combines primitive shapes (in 2D or 3D) together to produce a complete layout. In this encoding, the



genotype contains descriptions of shapes that are iteratively added or subtracted from a “canvas” (which can begin as solid or empty, depending on the encoding).

A straightforward instance of this type of encoding deals with floorplan generation and operates on a two-dimensional canvas (and a space that is often, but not always, partitioned as a grid). Ashlock et al. [19] suggest two indirect representations for the problem of maze generation. The first assumes that the canvas starts as empty space, and the genotype places “lines” of variable length and direction which can be additive (i.e. add solid space) or subtractive (i.e. remove solid space). The genotype stores information about the placement of each line as two long-form integers: one integer is the index of the tile where the line starts, and the second integer is bit-sliced to determine (a) whether the line consists of passable tiles or impassable, (b) the direction and (c) the length of the barrier. Another variable is included in the genotype that determines the number of impassable tiles that can be added in this way; if this number is reached the remaining barrier commands are ignored. The second indirect representation by Ashlock et al. assumes that the canvas starts off as fully solid and is “carved out” into rooms or corridors represented in the genotype as pairs of long integers. The first integer represents the coordinates of the tile where the placement begins, while the second integer is bit-sliced to determine whether (a) it represents a room, a horizontal corridor or a vertical corridor, and (b) to determine its dimensions.

Following up on this earlier work, Ashlock and McGuinness [62] evolved dungeons for role-playing modules using a solid initial canvas and large integers for an indirect representation. In this case, the representation is generative and involves several iterations of slicing the integers and filling remaining unfilled dungeon tiles with other integers. Genetic operators included a two-point crossover (more details in Section 2.1.2) while the mutation operator chose a few random integers and randomized them. Unlike direct representations, these mutations could have unforeseen consequences in the phenotype (dungeon) due to the generative representation.

Cachia et al. [63] represented large, two-floor game levels in two indirect ways, one per floor. In this case, the canvas for both floors starts as fully solid. The genotype contains information for both the ground and the first floor, however these are applied to produce the phenotype in sequence (see Figure 6). First, specifications of the ground floor carve out the space at all heights, essentially making an open “hole” with view of the sky. Second, specifications of the second floor carve short “tunnels” into the solid space and place floating platforms on open space. For the ground floor, the representation is defined by corridors and rooms. Rooms are represented by their central coordinates and size, corridors are represented by their coordinates and the corridor’s width and length (negative length aligns corridors vertically rather than horizontally). For the first floor, a random digger approach is used: an agent converts impassable space into passable by moving and turning at random (guided by probabilities) and adds stairs to the ground floor. The digger begins at the center of the map and is represented in the genotype as five probability values: the probability to move forward, to turn left, to turn right, to move onto a visited cell, to place a flight of stairs. To avoid a low locality in the search space by having a digger create different second floor “trails” from one generation to the next, a seed for all diggers’ randomness is set at the start of evolution. The genotypes evolve only via mutation, which adjusts the current parameters in the genotype by increasing or decreasing them by a random value.



Figure 6: Two-floor game level generated by Cachia et al. [63]. Left: the bottom floor is generated by subtracting rectangles (rooms and corridors) from a solid space. Right: the top floor is generated with a digger agent that adds or removes solid space depending on the content of the bottom floor. Stairs are also added by the digger agent (shown in brown).

In architecture, Wang et al. [64] explored the generation of the massing volumes of buildings through an additive or subtractive approach. The subtractive algorithm created a building massing design by removing several parts from a maximal mass. Since subtractive elements were always rectilinear prisms, when such subtractive elements overlapped, the result was a jagged edge (facade) on the building. The additive algorithm follows an inverse operation, and creates the building massing by combining several mass elements (additive elements). The subtractive approach, which started from a simple maximal mass properly aligned to the boundaries of the building space was deemed preferable to the additive approach which could result in chaotic arrangements, huge overhands, and free-floating masses.

Bao et al. [65] use an additive approach to represent a building layout as a union of a set of (overlapping) boxes. These boxes are optimized through constrained gradient descent, rather than evolutionary computation. An interesting addition of Bao et al. however is in the display of the alternatives of this process, which is performed in the form of graphs that cluster good but diverse layouts as single points that the user can visit through a user interface. The alternative designs are shown in relation to the user's currently viewed design, showing how similar they are based on the distance of the nodes on the graph.

### ▪ 2.1.9 Dual Representations

In many instances of automated solvers for architectural or engineering problems, multiple representations are required. As a simple example, a top-down layout is one representation while a 3D version of this layout is another. There are ways to automate the conversion from one representation to another: relevant examples include the generation of a 3D BIM model [66] from a 2D CAD drawing through scripted rules, the generation of a floorplan from a connectivity graph through deep learning [40] or constrained programming [25]. Note that in indirect embryogenies, a compact set of instructions generates a complex phenotype deterministically (always returning the same phenotype from the same phenotype). Instead, in these cases the genotype produces deterministically an interim phenotype (such as a room connectivity graph) which can be inspected by a human user before proceeding to the transformation into a more complete structure (the final phenotype). This transformation

from interim phenotype to final phenotype is often stochastic, and therefore it can also be optimized.

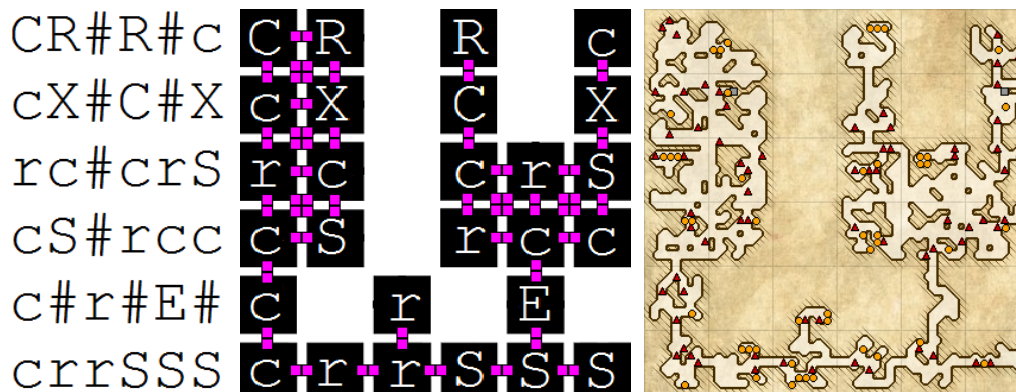


Figure 7: Evolution at two levels of representation for dungeons by Liapis [67]. Left: the dungeon segments' properties are represented as characters (e.g. E: empty segment) while the presence of walls (#) determine the connectivity of segments (magenta connections in middle figure). Right: each segment is evolved separately based on specifications from the left figure, and stitched together to create a complex, high-resolution dungeon. Source: [67]

While many transformations from one format to another could be included in this analysis (indicatively, extrusion of walls from a layout to a 3D model or cellular automata for generating a more complex heightmap from a map sketch [17]), this section focuses on instances where evolutionary computation is applied both to create the interim phenotype and –in a second optimization round– to transform the interim phenotype into the final representation. This form of multiple rounds of optimization has been pioneered in game level generation and specifically to generate dungeons. A first instance was introduced by Liapis [67] where the dungeon was first represented as a low-resolution grid of a few segments and then as a high-resolution grid of tiles (with a grid of tiles allocated per segment). Representation on the low-resolution grid is as a matrix of integers: each integer can be impassable (thus disabling connections between adjacent grids) or describe the types of contents and patterns that should be in that segment (e.g. high challenge, moderate reward, empty, etc.). This low-resolution grid is evolved through 2-point crossover and a mutation that changes the type of segment. Once the low-resolution grid is produced, each segment is evolved separately based on its connections with adjacent segments (which determines the walls and connection tiles at the edge of the segment, which are not evolved but impact the constraints and evaluation of the segment) and the high-level segment type. For instance, a “moderate challenge” tile has between 3 to 5 monster tiles and between 1 to 2 treasure tiles, and the evaluation of its quality follows a different fitness function than e.g. a “moderate reward” segment. Each segment is initialized and evolved in a separate population, and the fittest outcome per segment is used to instantiate the high-resolution dungeon. A similar approach is taken by the Evolutionary Dungeon Designer [18] (described in Section 2.1.3) where the designer specifies the size and connections of rooms while the evolutionary algorithm optimizes –often in tandem with the designer– each room. It should be noted that the Evolutionary Dungeon Designer operates on a dual representation and makes both visualizations available to a human designer, but AI optimization takes place only on the second, detailed representation. Figure 7 shows both levels of representations for a level.

Although not explicitly using evolutionary computation for optimization on both (or even in any) of the representations, it is worth noting here that design problems are often

represented as both a connectivity graph and as a floorplan, similar to the ones described above. The Graph2Plan algorithm uses a connectivity graph of rooms to create pixel-based images of floor layouts and is described in Section 2.1.5. The work of Doulgerakis [32] uses a connectivity matrix and evolves the floor layout based on genetic programming and is described in Section 2.1.6. The work of Karavolos et al. [25] evolves the connectivity graph itself and relies on constraint satisfaction to generate the level layout; this is described in Section 2.1.3. Besides graph-to-layout transformations, ArchiGAN [38] uses multiple (three or more) representations and applies deep learning to transform one representation into the other; this is described in detail in Section 2.1.5.

#### ▪ 2.1.10 Ad-hoc Representations

Certain representations are so problem-specific that they can not be categorized under the broad terms used previously. Such representations are usually tailored to a design problem that has a strictly defined possibility space.

An indicative example is the Interlace, a large residential project in Singapore [68] consisting of rectangular apartment blocks stacked in an interlocking brick pattern, with voids between blocks. Each stack of blocks is rotated around a set of vertical axes, thereby creating a complex interlocking configuration. In an experiment for automated optimization of the layout of Interlace [69], the representation followed a decision chain where the first block's placement impacted the next blocks' placement etc. The blocks had specific ways in which they could be placed due to the design of Interlace, based on the "joints" where blocks were interlocking and based on 12 permitted angles of placement. In this fashion, the representation was a simple array of real-valued numbers which was mutated and recombined as per normal numerical optimization problems.

Another example where the designer hard-codes their specific requirements into the representation of the genotype is presented by Gero et al. [70]. In this case, a 4-floor building with identical floors and pre-made exterior and interior walls is used. The genotype is an array of integers mapping to different types of offices, and it is mapped onto the phenotype by placing these offices along a predetermined path that iterates through each location on the 4-floor building in a zig-zag pattern (also zig-zagging between floors). In this case, the phenotype is evaluated based on proximity of distances between offices and activity interactions between offices, which is calculated based on the actual physical space. However, the indirect encoding used for the genotype is based on the order that the designer has determined that the spaces would be filled in. Any minor change (such as a wall partition being removed) would impact the ad-hoc zig-zag pattern and the genotype would create a very different phenotype; in that case, evolution would have to be carried out from scratch.

Ad-hoc representations for the genotype can also be based on the specifics of the tool used for optimization. Since tools for Computer-Aided Design nowadays come with evolutionary optimization plugins (e.g. Galapagos for Rhino), it is often simpler to use the internal variables used by the tool to describe the design as the genotype. Doe and Aitchison, for example, use seven parameters from Grasshopper's sliders for a specific design problem created by the authors and optimize the values of these parameters via the Octopus plugin for Rhino. These seven parameters (with predefined, custom value ranges) defined various translations and rotations of two types of pre-authored habitation modules, and the possible configurations they could encode was up to 768. This very constrained representation, which is opaque for any user except the developer of this design formulation, was sufficient for the goals of the

task at hand but could not be generalized or translated for larger or different problems (such as another type of habitation module).

## ○ 2.2 PrismArch applications

As discussed in Section 2.1.1, there is a difference between the problem space (i.e. what problem we are trying to address, and what constitutes a valid solution) and the –algorithmic– solution space. In theory, the problem space is defined a priori and influences the algorithmic design. In architectural design, however, often the problem space itself is defined through multiple iterations of analysis, synthesis and evaluation [12]. Given this fluidity where design and problem specification often proceed in parallel [13] throughout a design process, the algorithmic representation should also be flexible, accommodating and adapting to the human designers' input. At the same time, the algorithmic representation should be as “expressive” as possible [71], being able to represent a large variety of topological arrangements and geometrical forms, in a relatively compressed manner.

Based on the survey of methods for encoding and evolving genotypes that can represent architectural or related designs, there is a broad variety of formats with some caveats in terms of how controllable they can be in terms of evolutionary operators and in terms of epistasis. Epistasis describes the “situation where one gene pair masks or modifies the expression of another gene pair” [72] and can impact the performance of evolutionary computation as small changes in one gene may impact the general structure of the phenotype. This is especially concerning in a creativity support tool where suggestions should be close to the user's own designs, since a small change in the genotype could cause very different designs. Moreover, the designer should ideally be able to control aspects of the design even during evolution, such as “freezing” a room in an evolving floorplan so that it does not change further. To achieve this, direct representations are more amenable to low epistasis and tighter control over the genotype by e.g. blocking the mutation of certain genes. On the other hand, most of the direct representation formats surveyed focus on grid-based layouts which can be somewhat constraining in terms of the types of design patterns that can emerge. Grid-based, partition-based and rectangular representations are ubiquitous in both commercial computer-aided design (CAD) software and in optimization methods, but they are restrictive in terms of the types of designs they can present to a designer.

For the purposes of the AI generation of suggestions for a human designer, therefore, PrismArch attempts to bridge the gap between a controllable, direct encoding in the genotype, and a more expressive representation that does not limit itself to “boxy” designs. Therefore, the artefacts generated by the system are in the form of Voronoi diagrams, but with additional elements beyond the function that each cell is assigned to. The benefit of Voronoi diagrams is in their flexibility, as they can represent rectangular and polygonal regions, as well as capture curves along walls. By choosing an ad-hoc position for the Voronoi centers, a rectangular grid representation can still be achieved; additional grid representations such as a hexagonal grid are also possible with the same format. If evolution is not allowed to modify the positions of these Voronoi centers, then the evolutionary algorithm can create the familiar rectangular layouts, if a designer wishes. However, if the evolutionary algorithm also controls the coordinates of the Voronoi centers, many different forms are possible; the preference towards specific forms (indicatively, curves or straight angles) can be encoded as soft constraints or as aesthetic dimensions (see Section 4 for more details). If the designer wants to constrain the representation to e.g. rectangular grids then

that can be achieved by simply turning off the mutation of Voronoi centers. Additional interim mappings are possible, e.g. starting from a rectangular grid but allowing evolution to slightly change it, or setting specific portions of the floorplan as amenable to evolutionary mutation and others as “frozen” [73].

### ▪ 2.2.1 Problem representation

In the context of AI assistive design for PrismArch, we can define the problem representation as an abstract description of the solution that captures a number of topological characteristics such as, for example, connections between specific rooms, as well as a number of other important features, such as the target surface area of each space unit, but without specifically describing how those features are to be attained. Alternatively, the problem representation can be viewed as a set of hard constraints which differentiate solutions into feasible (the ones that satisfy them) or infeasible (the ones that do not satisfy them). By incorporating this approach, we consider that designers themselves should define the problem and negotiate it with a potential client, by analyzing the problem definition itself or potential concrete solutions that it produces.

The problem representation could be captured by a large number of possible specifications and constraints. Indicatively, the definition of a list of space units and their prescribed connections could be one of the main ingredients. Furthermore, a number of more detailed prescriptions could be included, such as the area and elevation per space unit. Finally, geometrical constraints such as a specific boundary or other site-related characteristics may also be important to take into account.

### ▪ 2.2.2 Solution representation

As shown in Figure 9, the representation operates on a hierarchical fashion with three layers currently envisioned: 1) the Voronoi tessellation of a 2D plane, 2) the utilization of specific cells by specific rooms and 3) the “details”, such as placing doors, windows etc.

The genotype contains one gene per Voronoi cell. Since mutation can add or remove Voronoi cells (see Section 2.2.3) except in special cases such as rectangular or hex grids, the chromosome can thus have variable length. Each gene (Voronoi point) contains the 2D or 3D coordinates of the point (in case of multi-floor layouts), an integer that determines its space unit, and additional identifiers in the case of openings, i.e. the type of opening (door or window) and the cell it connects to. Additional variables will be considered as development progresses to increase the expressivity of the solution representation.

### ▪ 2.2.3 Generation and mutation operators

There are many approaches for the generation and transformation of solutions with the chosen representation. On the one end of the spectrum, one could apply a completely random generation and mutation which would generate results with a very small chance of being feasible (i.e. satisfying at least the hard constraints). On the other end of the spectrum, one can design specially crafted generation and mutation methods that guarantee feasible solutions with the caveat of more computationally intensive processes and less variety in results. Our chosen approach lies somewhere in the middle of those edge cases, as an attempt to get the best of both.

The generation process starts by subdividing the space via a random voronoi diagram, then utilizes the diagram's cells (assigns them with a specific function) while considering some connectivity constraints and finally adds details such as placing doors and windows. The mutation process is a bit more complex. It operates in a hierarchical fashion and can occur in one of three ways: mutating the voronoi points (which may affect the cells' utilization and placement of doors and windows), mutating the cells' utilization (which may affect the placement of doors and windows) or affecting the placement of doors and windows (which does not affect anything else). Both the generation and the mutation processes do not guarantee feasible solutions. The discovery of feasible solutions is endured through the broader operation of the evolutionary algorithm.

Finally, the evaluation of the evolving content (represented as a Voronoi diagram) as well as aspects of the genetic operators and initialization strategies, are dependent on an interim connectivity graph or adjacency matrix. As noted in Section 2.1.3 and expanded upon in Section 3.1, connectivity graphs are vital simplifications of the design problem and have often been used to assess the quality of floorplans in terms of proximity and doorways between rooms. As noted in Section 2.1.9, usually such connectivity graphs are provided by the client and are not open to negotiation; this is the assumption for the current prototype of PrismArch's AI algorithms. However, the graph itself could be evolved as discussed in Section 2.1.3 as a step prior to the Voronoi tessellation. Some constraints on what could be evolved and the ability for human control over aspects of the graph (e.g. not allowing the removal of certain nodes/rooms or edges/connections) would be necessary in this case. These constraints can easily be encoded in the genetic operators, for instance by blocking remove node or change mutations for nodes marked as "frozen" [73].

#### ▪ 2.2.4 Representation and user interaction

This section presents several ways that a designer could become more involved in the design process, apart from describing the problem and the algorithm settings and receiving the results at the end.

**Direct interaction with the solution representation:** There are several ways in which the user may directly interact with the solution representation, in the context of the QD assistive system. First of all, they may generate their own solutions without even utilizing the evolutionary algorithm at all. Alternatively, they may choose to directly modify an existing solution generated by the system. While generating or modifying solutions, a designer can receive all the available "analytics" (as described in sections 3.2 and 4.2) during their design process and thus make more informed decisions or observations about their designs. Furthermore, user-generated designs can be used as the initial population of the evolutionary algorithm, thus providing a potentially better starting point for evolution instead of randomly generated solutions.

Alternatively, a designer may interfere with the generative process by imposing specific restrictions on what parts of the solution representation the algorithm can modify. Examples of this type of intervention are the following: 1) A designer may choose to "freeze" the initial arrangement of Voronoi points. This way they may restrict the search in, for example, rectangular or hexagonal geometries. 2) A designer may choose to "freeze" specific parts of the solution such as the arrangement of a specific room or set of rooms. This way, the algorithm will keep discovering variations of the rest of the design, while leaving intact the part that the designer considers to be of value.

Finally, the designer may even revisit the problem definition, after having examined a set of solutions. Slightly changing the problem definition can either be treated as part of a completely new process of search or, alternatively, as part of a continuous process of search that does not discard the previously generated solutions, despite the fact that they were generated through different feasibility criteria. Changing the problem definition is likely to cause infeasibilities in the existing set of solutions, however the evolutionary process (through mutation and selection) should be able to gradually correct the solutions, based on the new constraints.

**Indirect interaction with the solution representation:** Apart from a direct involvement of the user with the design process, his interaction may also be indirect. In this case the user does not actively alter or constrain the solution space, but is guiding the evolutionary process through their preferences.

The first way of doing so, is through interactive evolution [50]. In this case, the user is presented with a set of possible solutions and asked to choose the ones that they prefer. Their choice may be based on any type of subjective criterion, or take into account the analytics and measurements that the system provides for each one of them. As soon as the user selects a subset of solutions, they are treated as the initial population for the generation of the next set. This way, the user's preference is directly treated as a form of fitness function, in the algorithm's operation.

In line with the work planned in WP2, the most ambitious and general way in which the designer can indirectly control the solution representation is through a designer model. The user's preferences may be captured in the form of a designer model [2], via supervised machine learning (or, potentially, unsupervised learning when clustering groups of designers). After such models have been generated, they can be utilized as fitnesses, constraints, or custom representations [158] for the evolutionary algorithm that adjust various qualities of the generated content towards a specific style, or set of preferences that mimic the style of a designer persona.

#### ▪ 2.2.5 Data Collection during User Interactions

Apart from the intrinsic aspects of the Quality Diversity assistive system, the algorithm will also exist within a specific context of operation, where human designers interact with it, controlling some or all of its parameters and interfering with its operation in various ways.

The data-records of this human-AI interaction will form the basis for the generation of designer models which can represent (predict, reproduce) the behaviour and/or preferences of the designers that have used the system for a period of time. Data collection will have to adhere to the overall data organization of the PrismArch application, respecting issues of intellectual property and providing access to the data only to their respective owners. Any potential aggregation of data in the form of analytics or in the context of designer modeling that may depend on data of mixed ownership will have to be agreed upon between the respective owners.

As described in Section 2.2.4, there are several ways in which the designer can interact with the solution space and specifications, which also influences the type of data that can be collected regarding this interaction.



At its most basic, data collection will be in the form of timestamped events that describe instances of actions of specific subjects within the design space. In other words, every event describes “who did what, when”.

For example, if a user is directly creating an asset through the design interface of PrismArch, the system may be automatically collecting all of the actions that a designer takes while designing a “solution” from scratch. Depending on the context of use, it is possible that this approach can generate an unnecessary amount of data. In this case, the data collection can occur in predefined time-intervals, so as to reduce their volume but still retain a compressed (lossy) overview of the design process. This mode of data collection could be then used in order to train neural networks that can “mimic” the designer’s behavior, i.e. design in a style that resembles that of a specific designer or a set of different designers.

In the second mode of interaction, instead of the hands-on design activity, the recorded data should include the designer’s preferences. In other words, the data collection could keep track of which solutions (out of a larger set of solutions) the designer found to be preferable. This mode of data collection could be used in order to model a specific designer’s “preference”, i.e. a model that can predict the selections that a specific designer would make. As soon as such a model is trained, it can then be “embedded” in the algorithm’s operation, effectively generating a hybrid AI that mimics some aspects of human preferences.

### ○ 2.3 Software, Tools and Algorithms

*Table 1: Software that could be used within the scope of problem and solution representation within PrismArch*

Name	License	Description	Possible use
Voro++: A Three-dimensional Voronoi Library in C++ <a href="http://math.lbl.gov/voro++/about.html">http://math.lbl.gov/voro++/about.html</a>	modified BSD license, that makes it free for any purpose	An open source software library for the computation of the 3D Voronoi diagram, a widely-used tessellation that has applications in many scientific fields.	State representation of the underlying spatial structure of architectural designs.
Triangle: A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator <a href="https://www.cs.cmu.edu/~quake/triangle.html">https://www.cs.cmu.edu/~quake/triangle.html</a>	copyrighted by the author; may not be sold or included in commercial products without a license	Triangle generates exact Delaunay triangulations, constrained Delaunay triangulations, conforming Delaunay triangulations, Voronoi diagrams, and high-quality triangular meshes. The latter can be generated with no small or large angles, and are thus suitable for finite element analysis.	State representation of the underlying spatial structure of architectural designs.
Triangle.Net: a C# port of Jonathan Shewchuk’s Triangle software <a href="https://github.com/Geri-Borbos/Triangle.NET">https://github.com/Geri-Borbos/Triangle.NET</a>	MIT License	C# port of Jonathan Shewchuk’s Triangle software	State representation of the underlying spatial structure of architectural designs.
SharpNEAT: Evolution of Neural Networks	MIT License	Neuroevolution of Augmenting Topologies (NEAT) is an evolutionary algorithm devised by Kenneth O. Stanley. Sharp-NEAT is a complete	CPPNs evolved with NEAT as (part of the) state representation of 2D/3D structures.

<a href="https://github.com/colgre/en/sharpneat/blob/master">https://github.com/colgre/en/sharpneat/blob/master</a>		implementation of NEAT written in C# / .NET created by Colin Green.	
sferes2: <a href="https://github.com/sferes2/sferes2">https://github.com/sferes2/sferes2</a>	MIT License	Sferes2 is a high-performance, multi-core, lightweight, generic C++ framework for evolutionary computation. It is intently kept small to stay reliable and understandable.	May be used as an engine for performing evolutionary computation, novelty search, quality diversity search in the context of generative design.



## ▪ 3 FUNCTION EVALUATIONS AND CONSTRAINTS

Evolutionary computation or other optimization techniques have been used by practically every engineering principle. In the context of PrismArch we are particularly interested in the engineering some principles that are related to the design of buildings and even more so those who are directly affecting the building's form. We select two engineering principles which we find to be the most interesting in that sense: structural engineering and sustainability. Structural engineering is a very important aspect of architectural design which is absolutely necessary. Above everything else it defines the minimum criteria (constraints) for the safety of the structure against potential earthquakes, strong winds and other factors and, consequently, the security of the buildings' inhabitants.

From these two engineering principles, we focus mostly on their scientific aspect, i.e. the way that they can be used to evaluate the quality of a specific architectural design, in the terms that they pose. For structural engineering, this would be whether a proposed structure fulfils some minimal quality criteria (constraints) and then to what degree it does so in an optimal manner. For sustainable design, the criteria are usually multi-objective and the problem is usually how to balance the values of different qualities by exploring the design space.

### ○ 3.1 Survey

This section presents an overview of how certain problems that relate to architectural design have been addressed by AI methodologies. Section 3.1.1 presents the types of constraints and function metrics that have been taken into account in systems of automated architectural design. Section 3.1.2 focuses on the perspective of structural engineering and its relation with architectural design. A number of indicative case studies are presented, showcasing how structural engineering can form a synergistic relationship with architectural design through the use of computational approaches such as evolutionary computation and parametric design models. Finally, section 3.1.3 presents how the field of sustainability design has explored the relation between multi-objective optimization of sustainability - related metrics with the buildings' form, ranging from examples that only affect the buildings' facades, to examples that are optimizing the complete form of a building.

#### ▪ 3.1.1 Inherent constraints and function of architectural design

Architectural design is a complex problem that has intricate interconnections with engineering, economical and social problems and approaches. Apart from its interconnections with "external" perspectives, however, the practice of architectural design also poses its own, inherent, functional constraints and evaluation criteria that are mainly related to the organization of space in regard to its planned utilization by humans. Such constraints and evaluations are usually addressed in the early stages of the design process, where the architects have more freedom to investigate variations of a design. The following paragraphs describe several approaches where such constraints and functional metrics have been utilized in automated generation. Those studies can help us to first distinguish a number of constraints and functions and second to examine how they have been utilized in the context of computational optimization.

As reported in the work of Rodrigues et al. [77], the problem of automated architectural plan generation has been addressed computationally by many studies ever since the 1990s. Those studies use a variety of AI algorithms and computational methods for addressing the problem,

including genetic algorithms (GA), genetic programming (GP), see also Section 2.1.6), evolutionary strategies (ES), simulated annealing, sequential quadratic programming, L-systems, Voronoi diagrams, Dijkstra's algorithm and stochastic hill climbing. The addressed dimensions of optimization include geometrical and topological constraints and features, heating, cooling and lighting - related metrics, as well as walk distance minimization. Finally, the considered design variables include wall dimensions, interior and exterior doors, windows, space units, floor levels, equipment/furniture, building boundaries, adjacent buildings, openings orientation and spaces' adjacency. As the authors mention, most of the published work until their investigation focuses on a rather small subset of variables so as to minimize the problem's complexity.

In one of the early related studies, in 1994, Charman [78] treats architectural floor plan design as "an activity of determining subspaces of a given space according to defined or implicitly understood requirements and conflicting criteria" (as previously proposed in [79]). Charman [78] also explains that from a mathematical point of view, a floor plan problem can be expressed as a system of non-linear constraints on continuous domains, whose formal analysis is very complex. He proposes a methodology which treats the problem as a Spatial Constraint Solving Problem and uses a continuous representation (in contrast to discrete representations, such as grid subdivisions) for defining the shape and location of rooms. In the way that he formulates the problem, there are two types of constraints: First, implicit constraints, i.e. constraints that are present in all floor-plan problems and second explicit constraints, i.e. constraints that correspond to the specifications of the problem at hand. Implicit constraints include, for example, the fact that rooms must not overlap and that rooms must be inside the assigned plan area. Explicit constraints, on the other hand, correspond to the specifications of the problem at hand. Those include, for example, the dimensions of the rooms or the adjacencies between rooms. Such constraints are supposed to be defined by the user/designer, although from an implementation point of view there is no difference between explicit and implicit constraints. Despite the fact that this example has some limitations (for example the shape of every room must be a rectangle), it proposes a concrete methodological approach which is very similar to what we can find in much more recent works.

Medjdoub and Yannou [80] propose a computational methodology for the problem of architectural layout planning that mainly focuses on constraint solving. Similar to [78], they make a distinction between implicit and specification constraints. In their approach, implicit constraints are mainly utilized to reduce the combinatorial complexity of the problem (for example by restricting it to orthogonal geometries). Specification constraints, on the other hand, belong to the "functional diagram" and are explicitly declared by the architect. The latter include geometrical and topological constraints: Geometrical constraints relate to the surface area, length or width, ratio, or spatial orientation of rooms (for example Fig. 10), while topological constraints relate to the adjacency between pairs of rooms, adjacency between rooms and the perimeter, non-adjacency or proximity (for example Fig. 10). They point out that the latter distinction has a clear analogy to the way in which architects approach the design process: starting from sketches whose aim is to reach an optimal topology and then moving on to the specification of a more precise geometrical implementation. Interestingly, as they also point out, a preliminary optimization at the topological level (before dealing with geometrical constraints) is of particular value as the number of different topological solutions is relatively small. Consequently an architect can even have a complete overview of the space of topological possibilities for a given problem and thus make more informed decisions.

Finally, their software solution, ARCHiPLAN, allows a designer to define their own topological and geometrical constraints to explore the space of possibilities of their design problem. Their software can be used to design architectural layouts in orthogonal spaces, although they point out that it should be extended to less restricted geometries.

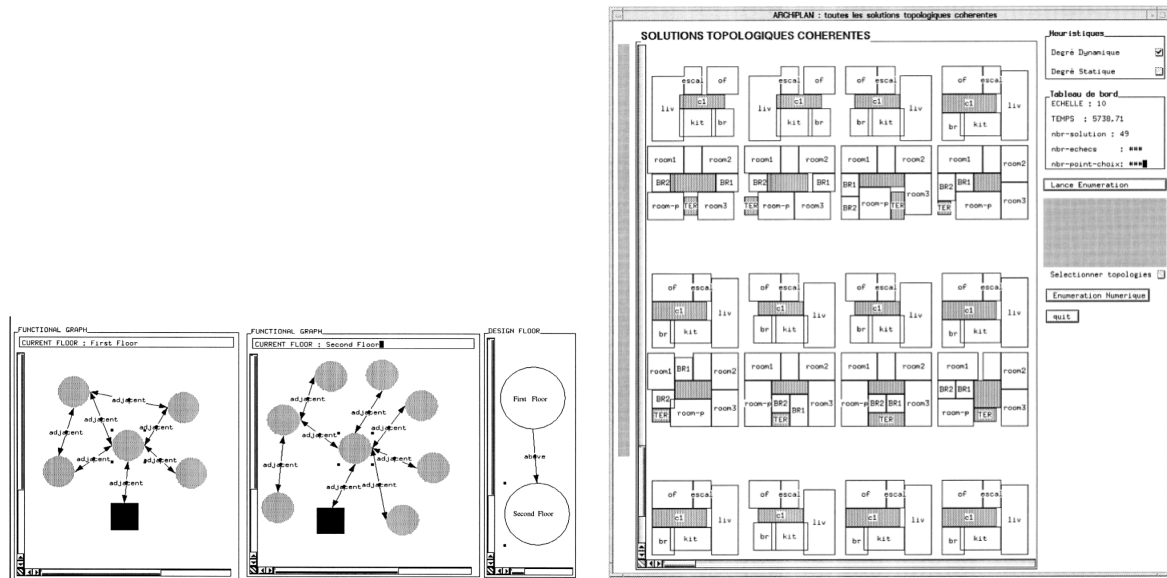


Figure 10: Example illustrations from the work of Medjdoub and Yannou [80]. Left: Functional diagram of a house with two floors. Right: Some geometrical solutions of the house on left. Source: [80]

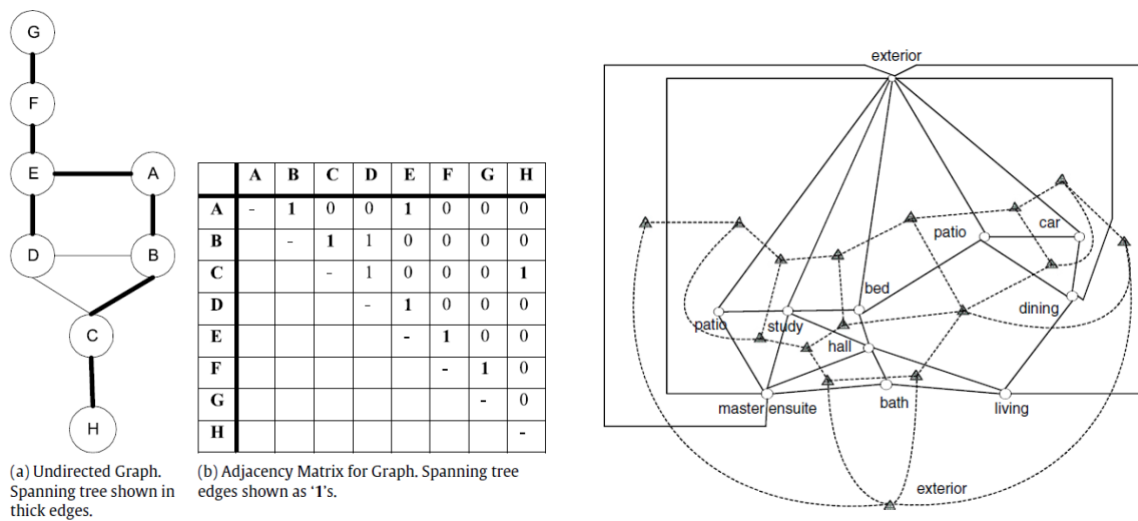


Figure 11: Example illustrations from the work of Wong and Chan [76] which focuses on the topological optimization of architectural plans. Left: Directed graph representation illustrated. Right: Dual graph (dotted lines and triangular nodes) construction from the solid line graph. Source: [76]

Wong and Chan [76], in 2009, present a methodological approach to architectural layout planning which follows roughly the same principles as [80] but especially focuses on the topological representation of layouts and the optimization that can be applied to it before even moving on to the geometrical implementation. They propose the use of adjacency matrices (Fig. 11) as an efficient representation of connectivity graphs which is compact, human-understandable and easy to manipulate in the context of evolutionary computation

(i.e. it is quite straightforward to apply mutation operations on the matrices, or crossover operations between them). At the level of constraints, their methodology is based on a very important aspects of graphs: the distinction between planar and non-planar graphs, i.e. graphs that can be projected in the geometrical space without overlapping edges and graphs that cannot. This distinction poses a hard constraint that can eliminate infeasible solutions. Planar graphs can be further analyzed and evaluated, based on their set of dual graphs, that describe the boundaries between spaces, instead of their connectivity (see Fig. 11). The feasible topological solutions are then optimized according to the degree to which they satisfy the client's preferences. More specifically, they propose that a large proportion of the client's preferences can be translated into "adjacency preferences" which can be then utilized directly as evaluations of generated solutions. Other factors, like the available budget and other design constraints are also taken into account.

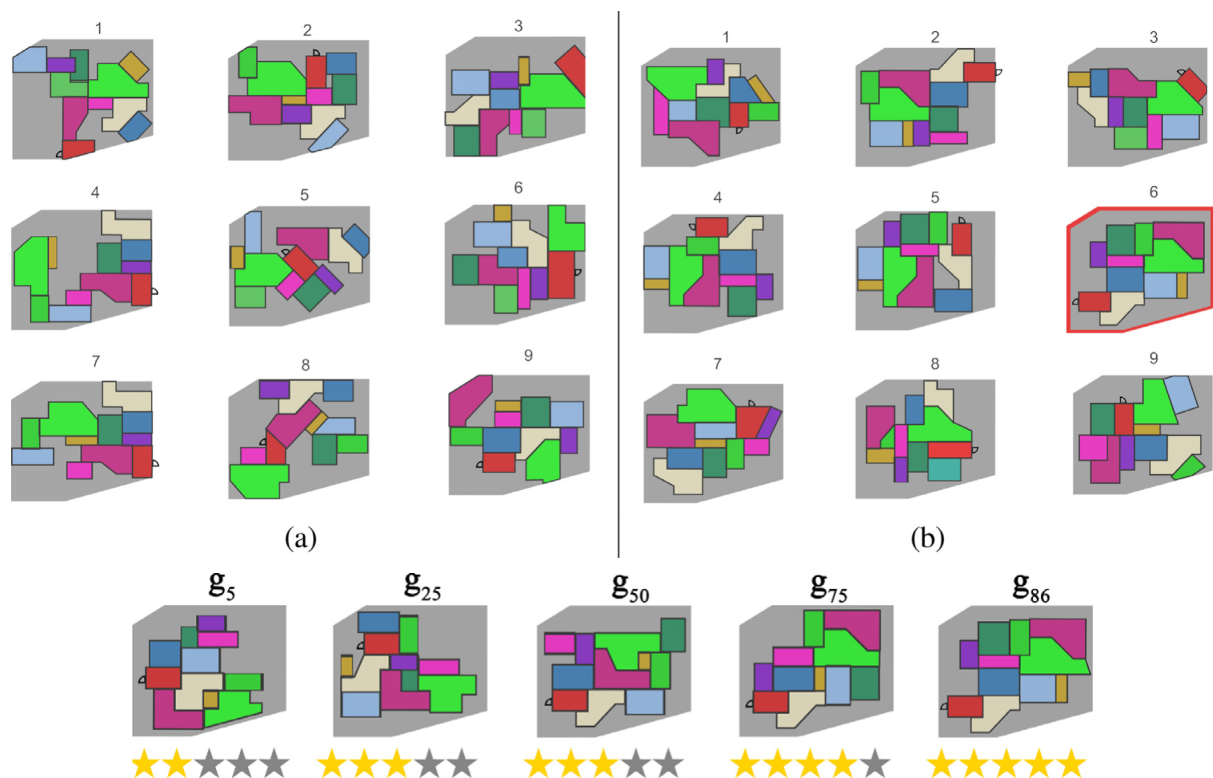


Figure 12: Example illustrations from the work of Bahrehmand et al. [81] on optimizing architectural plan layouts based on user preferences. Top left: a screenshot of nine layouts of first generation. Top right: a screenshot of nine layouts of 86<sup>th</sup> generation. Bottom: Five layouts that were rated by user in different generations. Source: [81]

Bahrehmand et al. [81], in 2017, propose a hybrid approach for the generation and optimization of architectural layouts that takes into account a number of constraints, functional evaluations and designer preferences. They use a state representation which consists of space units (which may refer to rooms or other types of spaces) whose shape is not restricted to rectangular polygons. On the contrary, they allow for those regions to be of any type of irregular polygons, which can be deformed throughout the optimization process, thus allowing for a much larger diversity of possible solutions (as shown in Fig. 12). The user's preferences are considered directly, during the optimization process. More precisely, they implement a system of interactive evolution, where the user can both select and intervene (modify) the available set of solutions at each step. Their methodology considers several

constraints: First, they consider geometrical constraints that a designer may impose on the solution. Those may be specific requirements about the shape of the whole building or parts of it, or they may relate to the building's surroundings which form specific geometrical boundaries. Second, they consider connectivity constraints, mainly by making sure that there are no disconnected rooms, i.e. that all rooms are reachable. Third, opening constraints consider all types of openings, such as doors, windows and main entrances and compare them against the user specifications. Apart from these hard constraints, they also use several functional metrics, in the form of soft constraints, which may be used for the optimization of the produced solutions. First, the "overflow quality function" is a metric that assesses the degree to which various boundaries are respected by the solution. Second, the "topological quality function" is a metric that evaluates the adjacencies of rooms, according to a predefined adjacency matrix. Third, several "spatial quality metrics", including circulation, privacy and compactness. Overall, their proposed methodology especially prioritizes the designer's involvement in the optimization process.

As the related work suggests, the functional aspects of architectural design form a complex and open-ended problem. They include a large number of constraints and functional evaluations that an architect usually takes into account. Those constraints and qualities are expressed in either the topological or the geometrical representation of architectural solutions. In contrast to purely engineering perspectives (such as structural engineering and sustainability design), however, many of the functional aspects of architectural design include some degree of subjectivity. For example, in [76], the optimization of connectivity graphs is evaluated against a preference graph that represents the client's subjective preferences or demands. In [81], on the other hand, the designer's preferences or expertise are directly utilized in an interactive evolution approach [50]; purely analytical metrics are there mostly to inform the designer, rather than as absolute quality criteria. Nevertheless, as becomes clear in examples such as [81], the exploration of the design space in relation to inherent constraints and even subjective metrics can be enhanced by computational methods.

### ▪ 3.1.2 Structural Engineering

Structural engineering is inextricably linked with architectural design. At the very least, the structural design of a building guarantees its integrity against various static and dynamic forces, such as load bearing, earthquakes and strong winds. Structural optimization is approached via a number of constraints and metrics. The constraints relate to the minimal requirements for a structure to be considered safe. The optimization metrics relate to the minimization of cost, expressed through the quantity and quality of the used material, versus the maximization of structural integrity. The "medium" through which the structural integrity is "negotiated" against the cost is, of course, the design space of structural problems. This is expressed through a topological and a geometrical perspective and is of large interest within the context of PrismArch, as a building's structural design is also coupled with other perspectives including, but not limited to, architecture. The relationship between architectural design and structural engineering is addressed in various ways in practice and, as reported in [82], it often leads to conflicts. However, parametric models and optimization algorithms can be a medium for a better negotiation of the constraints and qualities of the two approaches. The following paragraphs offer examples of optimization studies so as to explain the nature of the problem, as it is expressed through specific constraints, metrics and approaches, both within the field of structural engineering as well as on the verge between structural engineering and architectural design. As explained in the detailed survey of Kicing



et al. [83], structural problems have been addressed through evolutionary computation ever since the mid-seventies and early eighties (e.g. [84, 85]). Structural optimization problems can be divided in the following three categories (Fig. 13):

- **Topological Optimum Design** is usually applied in the conceptual design stage and includes methods of searching for an optimal material layout of a structural system. TOD problems, being quite complex in nature, have been most successfully approached using heuristic methods, including simulated annealing [86] and EAs [87, 88, 89].
- **Shape Optimization** is usually applied during the embodiment design stage and includes methods of searching for the optimal contour, or shape, of a structural system whose topology is fixed.
- **Sizing Optimization** encompasses searching for optimal cross-sections, or dimensions, of elements of a structural system whose topology and shape are fixed and is applied during the detailed design stage. Such problems are usually addressed successfully using formal methods, such as Mathematical programming [90] and optimality criteria methods [91].

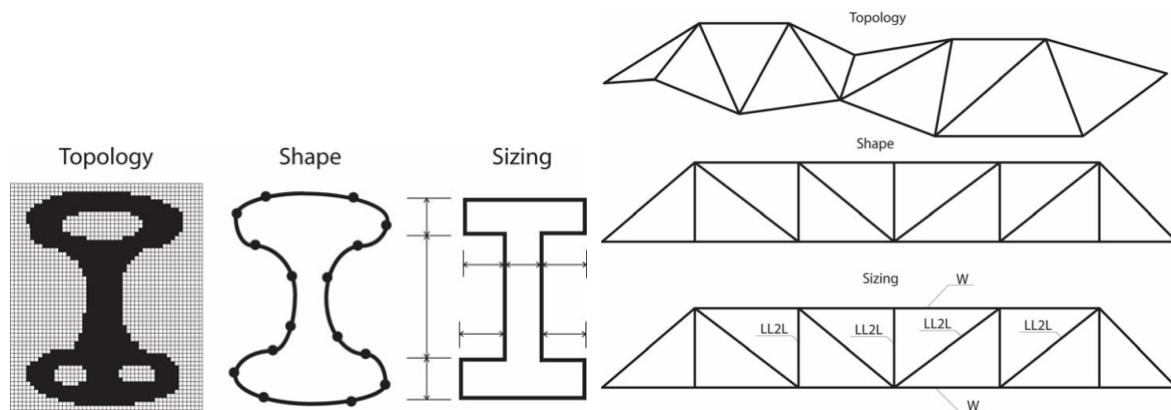


Figure 13: Topology, shape, and sizing optimization for continuous (a) and discrete (b) structural design problems. Left Topology, shape, and sizing optimization for continuum structural design problems. Right: Topology, shape, and sizing optimization for discrete structural design problems. Source: [83]

Out of those categories, Topological Optimum Design and Shape Optimization are probably the most relevant, as they are addressed in the earlier stages of design, where the largest part of “negotiation” between different disciplines takes place. The following paragraphs present examples of how computational methods have been used to address structural optimization of various problems, ranging from purely technical ones to others that are much more interconnected with architectural design.

Most of the early examples of evolutionary structural optimization are focusing on problems of relatively small complexity. However, such examples are still useful as they expose the methodological approaches in relatively simple terms. For example Chapman et al. [89], in 1994, use a genetic algorithm for optimizing the topology design of cantilevered plates (which, as reported in [61], is a classic benchmark for optimization in structural engineering). They use a binary representation which corresponds to a discretized 2D plane whose cells are either void or filled with structural material. The fitness function they use (the degree of stiffness divided by the plate’s weight) allows them to promote “structures which best

combine light weight and load-carrying ability". A genetic algorithm guided by that fitness function can discover families of topologies that best satisfy the problem.

Schoenauer [61], in 1997, examined how various representations of the same problem (i.e. the genotypic description of a cantilever plate) affect the genetic algorithm's ability to find optimal solutions. Schoenauer's case studies include a bit-array representation, a Voronoi representation and an H-representation. The three representations are tested using a classic genetic algorithm (GA), as well as two different evolutionary strategies (ES, GP). The bit-array representation, similarly to [89], represents a 2D grid whose cells can either be void or filled. The Voronoi representation consists of a set of points that generate a Voronoi tessellation. The Voronoi cells can then be void or filled. Finally, the H-representation uses a set of holes (described as rectangles in the 2d plane) that represent the void areas (holes) of the material plane. Results show that both the Voronoi representation and the H-representation are outperforming the bit-array representation both in terms of computational performance and in terms of quality of the results, with the H-representation yielding the best results overall. The choice of algorithm (between GA, ES and GP) seems to have almost no effect on the results, with few exceptions. The author explains that both the H-representation and the Voronoi representation are exhibiting a high degree of epistasis (i.e. the expression on the phenotype of one gene is influenced by other genes in the genotype). Epistasis is highest in the case of Voronoi cells, as the influence of one site in the physical space is modulated by all neighbour sites which, as Schoenauer [61] suggests, may be the reason for its reduced performance. Overall, his study showcases the importance of representation for evolutionary approaches which seems to have a stronger impact than, for example, the choice of algorithm.

Kicinger et al. [92], in 2005, apply evolutionary computation to the problem of designing steel structural systems of tall buildings. Their solution representation subdivides the frame into cells, each of which can be utilized by a specific typology of bracing elements. Consequently, the evolutionary process gradually finds optimal distributions of those modules along the frame (fig. 10a). The researchers use a specialized software called "Inventor 2001" in order to generate, evaluate and evolve their designs. As they report, the software offers a number of features that could be used as fitness functions, including the total weight, the weight of bracings, weight of beams, weight of columns and number of bracings. In their approach, they only take the total weight into account, for a number of reasons: First, it has been traditionally used as an indicator of quality. Second, the total weight is simultaneously a good indicator of the design's cost, as well as its novelty (usually novel design concepts are introduced to reduce the weight of a structure). Finally, the use of a utility-based fitness function would introduce bias and thus reduce the objectivity of the results. Apart from fitness, the main constraint imposed on the generated designs was their structural integrity under a variety of loads that resemble a realistic scenario, including "dead" and "live" loads and wind forces. As the authors report, they chose to not over-constrain their approach by taking into account constructability issues, as those might have stood in the way of the evolutionary process. Additionally, the authors seem to be especially interested in the issue of novelty or creativity, within the constraints of their problem. Although they perform a typical, single-objective evolutionary optimization and not a specialized multi-objective or quality-diversity [6] approach, they do take some precautionary measures so as to avoid local optima (such as "starting from rather poor parents" [92]) and promote novelty. Their findings suggest that evolutionary computation (which they apply through the Inventor 2001 software) "is useful

for exploring design representation space of steel skeleton structures in tall buildings, and for searching for novel design concepts, which may gradually emerge from simpler substructures being evolved by the system” [92].

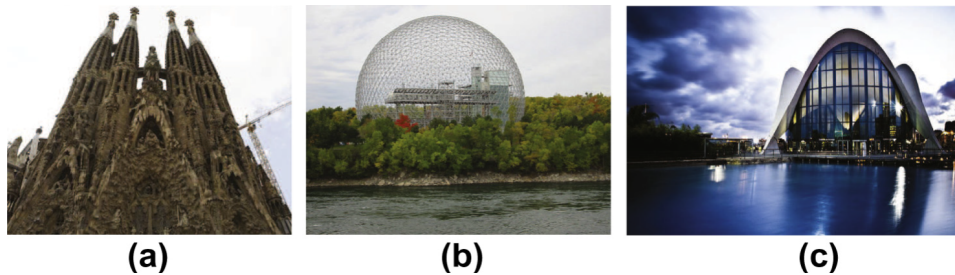


Figure 14: Historical examples of structures by architects with strong and innovative engineering concepts: (a) Antonio Gaudi ([93]), (b) Buckminster Fuller ([94]), (c) Felix Candela ([95]). Source: [82]

Beghini et al. [82] point out that structural topology optimization can be a link for connecting structural engineering with architectural design. Their study points out that often the aesthetic and philosophical requirements of architects are in conflict with the stability and performance requirements of civil and structural engineers. However, they also present a number of historical examples whose aesthetic qualities are particularly intertwined with their structural function: namely Antonio Gaudi’s “Basilica de la Sagrada Familia” [93], Buckminster Fuller’s “American Pavilion of Expo 67” [94] (also known as the Montreal Biosphere), and Felix Candela’s “The Oceanographic” [95] (an oceanarium in Valencia, Spain), all of which are shown in Fig. 14. These examples, of course, were the works of extremely talented architects with a very strong structural sensibility. For example, Antonio Gaudi used physical models as an integral part of his design process in order to calculate the sophisticated structures that define the form and function of his intricate works. A much more recent approach that has produced astonishing results in regard to this perspective is the use of parametric models as a common ground between architects and engineers. The authors present a number of case studies (for example Fig. 15) that showcase how, through the use of parametric models, they were able to explore the design space much more efficiently, while taking into account the buildings’ aesthetic qualities and their structural characteristics simultaneously. They use a topology optimization method, described in [96] which allows them to optimize the distribution of material within a specified region, by evaluating the stiffness of the resulting structure. Their study suggests that topology optimization software can be used as a means of communication between architects and engineers, showcasing a variety of possible engineering solutions with ease, or even constraining the engineering solution space with aesthetic criteria. This way, the design of form and function can become more integrated and the exploration of the solution space can become more efficient, taking into account both perspectives at once.

While focusing on the use of parametric models and optimization techniques as a middle ground between form-finding and the structural engineering perspective, it would be a huge oversight not to mention the work of the Block Research Group (BRG), at the Institute of Technology in Architecture of ETH Zurich, led by Dr. Philippe Block and Dr. Tom Van Mele. Their huge volume of work was initially inspired by the historical heritage of architecture which provides us with countless examples of large structures that operate only under tension (i.e. they are not reinforced). Some of the foundations of the group’s research have been set in Block’s PhD dissertation [97] which proposed a novel computational methodology for

generating compression-only vaulted surfaces and networks. The methodology uses projective geometry, duality theory and linear optimization, providing a graphical and intuitive method, adopting the same advantages of techniques such as graphic statics, but offering a viable extension to fully three-dimensional problems. Ever since then, as reported in [98], the group's expertise has expanded from compression-only problems to other structural systems such as thin concrete shells, "bending-active" membrane structures, fabric formwork systems, and general spatial systems of forces with applications in bridge design and large-span roofs. One of the main areas of study of the group relates to computational form-finding and optimization. They are the developers of the RhinoVAULT software [99], an interactive design system in the form of a plugin for McNeel's Rhino CAD software which allows the designer to create and explore compression-only structures, using the Thrust-Network-Approach. As explained in [98], their research is largely project-driven and focuses both on computational innovations, as well as fabrication innovations, such as 3D-printing or other techniques.

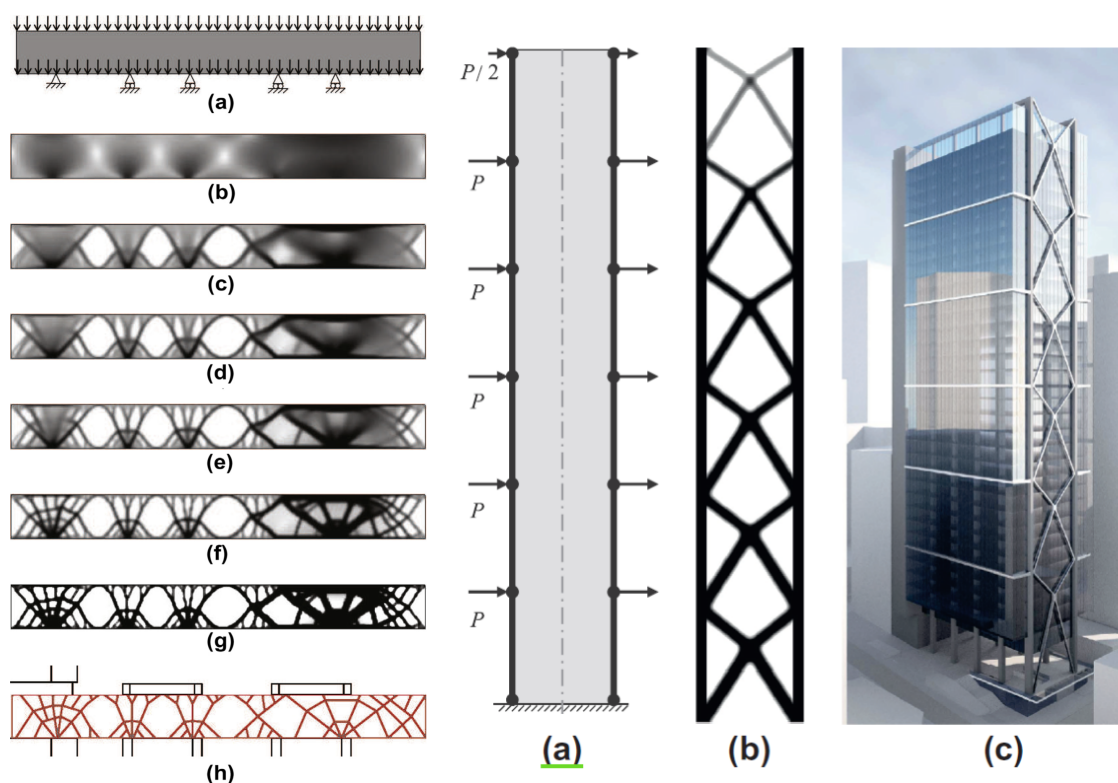


Figure 15: Case studies showcasing the use of topology optimization for large-scale problems related to architectural design. Left: Topology optimization for design of the upper "beam" spanning several towers for the Zendai competition. Right: Illustration for the concept design of a 288m tall high-rise in Australia, which shows the engineering and architecture expressed together: (a) problem statement, (b) results of the topology optimization, (c) renderings of the design. Source: [82]

As the relevant literature suggests, there is a lot of room for interaction between the constraints and functions of structural engineering and other aspects that relate to architectural design, such as the aesthetic and functional aspects of a building's form. Both evolutionary computation (or other optimization methods) and the use of parametric models seem to be capable of supporting this interaction, by undertaking the heavy computational work which is necessary for the efficient search of the design space. By utilizing such systems,

architects and engineers can cooperate more efficiently and find novel solutions that better satisfy the different points of view. Including optimization methods and parametric models in the conceptual stage of the architectural design process would be an important tool for architects, as it would allow them to make more informed decisions in their form-finding approaches.

### ▪ 3.1.3 MEP engineering

As explained in [184], “Mechanical, electrical, plumbing and related systems (MEP systems) have become one of the bigger contributors to the building construction costs. They are also heavy contributors of the energy consumption in buildings” [184]. In the academic context, MEP systems have been addressed through two lenses: The first one relates to the coordination between different involved disciplines and the second relates to the (design) optimization of MEP systems, in relation to measurable aspects such as energy efficiency, sustainability and service-life . Both of those aspects are very important in the context of PrismArch.

#### **Coordination between MEP disciplines:**

MEP coordination is an important aspect that influences the overall success and performance of building projects [193, 194]. MEP systems are required to ensure a comfortable indoor environment, distribute electric power and communication networks, provide potable water and dispose of waste water [195]. Thus, the coordination of MEP systems is critical for a successful / well-performing project [196]. As further explained in [185], a significant amount of research has been conducted in relation to MEP coordination.

According to Hassanain et al. [185] BIM technologies have greatly helped MEP coordination and offered a huge improvement over the traditional process of “sequential comparison overlay process”. However, BIM technologies are not without limitations. Their main drawback is that they mainly identify hard (physical) clashes, excluding many coordination-related aspects such as the complexity of building systems, limited budget, limited installation schedule, limited building space [194] and the necessary technical knowledge for the various systems’ design, installation and maintenance [186]. In response to this observation, the authors of [185] are proposing a coordination methodology that identifies a checklist of 63 “items” that are split in four categories: documentation coordination, operation and maintenance coordination, within discipline coordination and cross-disciplinary coordination. Their study reveals that “that careful consideration and communication between the mechanical, structural and architectural design teams was paramount in achieving proper mechanical coordination. Furthermore, it was found that constant communication between the electrical and the other design teams was necessary, to avoid electrical design conflicts. Finally, fire safety consideration was found to be most important in plumbing systems’ coordination”.

Other relevant studies also emphasize the importance of cross-disciplinary coordination in MEP. For example: Yarmohammadi and Ashuri [197] studied BIM-based MEP coordination in the USA and showcased that the experience level of the design team, the quality of preliminary design and the complexity of MEP systems influence the quality of project coordination. Riley and Horman [198] sought to correlate the effects of design coordination on project uncertainty and established empirical evidence that design coordination reduces the overall project costs. Korman [195] proposed rules and guidelines for MEP coordination using the BIM software. Finally, Khanzode et al. [199] presented the benefits and challenges

associated with the implementation of BIM/VDC tools and processes for MEP coordination and Khanzode [200] presented an integrated, “virtual design and construction” and Lean method for the coordination of MEP systems.

This part of the survey showcases the recognized importance of MEP coordination, as well as the difficulties in addressing it in a systematized, methodical way.

#### **MEP Systems optimization:**

Although, in theory, any engineering problem can be treated as a problem of optimization (and addressed through computational methods) our literature investigation reveals that out of the engineering fields involved in MEP, only certain aspects of HVAC systems have been addressed through computational methods of optimization.

As far as energy efficiency is concerned, HVAC systems have been addressed as problems of optimization in many cases. For example, they have been approached through multiobjective optimization [187, 189] or fuzzy logic [188], or even through reinforcement learning [190, 191].

In more detail, Li et al. [187] are focusing on the multiobjective problem of maximizing the level of thermal comfort and indoor air quality while minimizing the system energy consumption, in a case study of interior office spaces. The computational approach that they use in order to obtain the pareto front of the multiple objectives is the “non-dominated particle swarm optimization”, a variation of the “particle swarm optimization” algorithm. Apart from that, however, the authors emphasize the need for a surrogate model that can approximate the simulation results efficiently. They use the Kriging method as a means of learning to predict the simulation results, based on available samples and provide a fast evaluation of the solutions in the context of the optimization algorithm.

The aforementioned methods are focusing on the control aspects of design systems. However, HCAV performance is also related to the systems’ geometrical / topological design. A relevant study that approaches optimization from this perspective can be found in the work of Manuel et al. [192], where the authors apply topological optimization via “computational fluid dynamics” simulations for the automated design of duct layouts.

#### **MEP service life optimization:**

Another aspect of optimization for MEP systems is their “service life”. As explained by Kwon et al. [183], the maintenance of buildings’ MEP systems has been recognized as an important issue, as the number of deteriorating buildings around the world increases. They further suggest that the service life of building components needs to be estimated in advance and propose a methodology that leads to proactive measures, instead of reactive which is the usual case. They propose a hybrid approach which combines a genetic algorithm with case-based reasoning methodologies in order to tackle a set of maintenance - related problems for MEP components.

In [183], the authors propose a model for estimating the service life of MEP components, based on previous data. Their model comprises of four different modules: (1) data collection, (2) attribute selection, (3) attribute weighting, based on GA and (4) case retrieval. Their data base includes historical data on buildings provided by the Korean Land and Housing Corporation that were carefully processed. A set of selected attributes were then selected. The selected attributes were weighted through a GA, in order to evaluate their “significance” in the outcome, which is the estimation of the building’s service life quality. Their results

showed that “the developed estimation model can support more systematic building maintenance than current approaches because the outcomes are determined based on reliable data extracted from previous cases”. The surveyed study is a simplistic form of machine learning, where a model is derived from data, allowing for the prediction of certain properties of design. Another important characteristic of this approach is that it focuses on high-level features of the MEP systems, and not on their detailed implementation. This high-level view of the MEP system provides an interesting direction for applications in PrismArch which are intended to be useful (at a high-level) even in early stages of design and coordination.

#### ▪ 3.1.4 Sustainability

Sustainable design is a field of engineering closely related to architecture where computational methods have been widely researched for problem solving, dating back at least thirty years from now (some of the early relevant studies can be found in [100, 101]). The need for AI solvers arises from the complexity of the problems within this field, since multiple (potentially conflicting) criteria must be taken into account.

As the comprehensive review of Evins [102] showcases, computational optimization methods including direct search, evolutionary methods and other bio-inspired algorithms have been successfully applied to various problems in the context of sustainable building design. In the context of sustainable design, a building’s envelope is a very important part as it defines the building’s relation to the “exterior” environment, in relation to heat, light and air. The construction details of the building’s envelope, including its selected materials, insulation and glazing have been addressed as an optimization problem by many studies (for example [100, 103, 104, 105, 106, 107, 108]). Most of those studies focus on the multi-objective nature of the problem and the balancing of conflicting parameters. Other studies have attempted to expand the search space so as to include variations of the building’s form (for example [101, 109, 110, 111, 112]), thus touching on a much broader number of problems, including architectural, aesthetic, structural and sustainability factors at once. Another relevant design aspect that has been addressed computationally is the specific technology of double-skin facades of buildings (for example [113, 114, 115]). Those facades consist of two glass layers with an air gap in between, introducing problems of control and air flow and posing a special case of construction and form optimization. Finally, as reported in [102], there are two more broad categories of related problems that have been addressed by optimization algorithms. The first one relates to systems, such as the Heating, Ventilation and Air Conditioning (HVAC) systems as well as artificial lighting and its relation to energy consumption. The second one is energy generation systems, such as combined heat and power (CHP) systems, solar technologies and ground energy and storage systems. These types of design problems have also been addressed as optimization problems (for example [116, 117, 118, 119, 120]), although mostly taking the form of the building as a given and focusing on the technical control and design issues that are mostly unrelated to what is directly perceived by the buildings’ inhabitants/users.

In one of the earlier relevant studies (2003), Marsh [109] describes how the problem of shading for an element or a whole building can be approached as a geometrical problem and addressed computationally. As he explains, taking natural light into account in the early stages of the design process can be used as a means of making informed decisions related to form-finding. As he also points out, taking the solar position into account manually, when designing

shading devices, can be a laborious task. He suggests that shading problems can be addressed much more efficiently by using specialized raytracing software. His proposed methodology is explained on a relatively simple problem of designing a shading device for a window. The problem can be simply stated as finding the proper shape for the shading surface, so that it guarantees that the window will remain shaded (i.e. will not receive direct sun light) throughout a specified set of dates and times. The approach to solving this problem is rather simple, in principle. It utilizes the ray-tracing software to generate simulations of where the light would reach, according to date and time and then extends or reshapes the geometry in such a way that it provides the necessary shading. As March explains, “the simple case methodology can quickly become cumbersome when dealing with relatively complex window shapes with many vertexes or areas of polygonal concavity” [109]. However, this methodology serves as a very good example of the relation between form and function, in the context of sustainability design. Furthermore, nowadays there are more specialized tools for lighting simulation and similar problems could be addressed computationally with much more elaborate techniques for optimization.

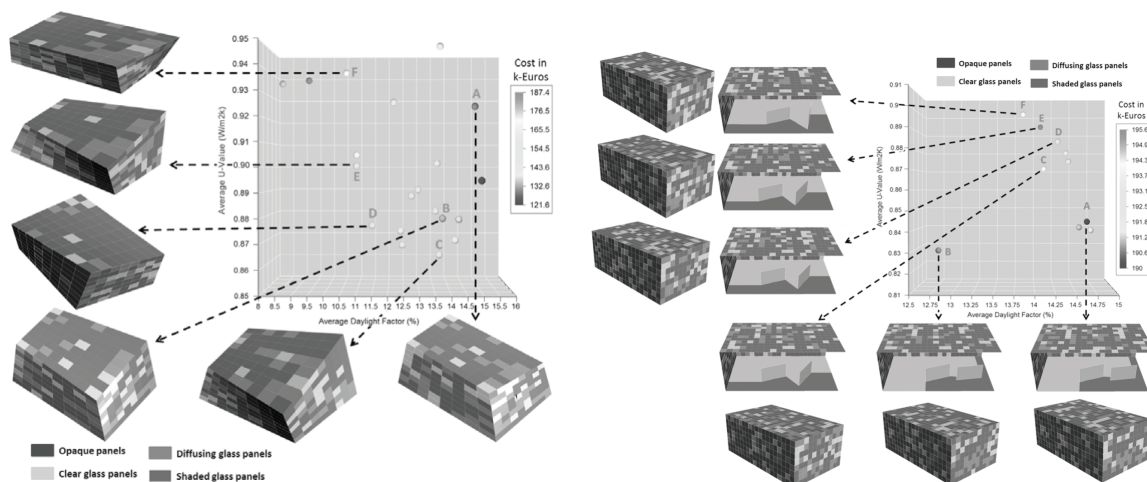


Figure 16: Results from the work of Kaushik and Janssen [121], showcasing the use of sustainability criteria for the design of a building's envelope and interior layout. Left: Study of Pareto optimal solutions: generated variations of the shape of the building's envelope and the distribution of panels. Right: Study of Pareto optimal solutions: generated variations of the distribution of panels on the building's envelope and the interior layout. Source: [121]

In a more recent study, published in 2012, Kaushik and Janssen [121] showcase how multiple sustainability-related metrics can be used to drive a number of aspects of architectural design. This work uses multi-objective evolutionary optimization to automatically explore variations of a building's form. They strongly focus on the building's envelope and the distribution of panels of different materials on it, but also examine variations of its interior structure, as shown in Figure 16. Their overall optimization strategy is defined in the following objectives: 1) Maximize the daylight factor at response points, 2) Achieve desired sun hours at response points, 3) Minimize average U-Value of the panels, 4) Minimize overall cost of the panels and 5) Achieve desired height at each of the five internal spaces. They use the Dexen [122] evolutionary design system in SideFX Houdini [123] while the evaluations are performed through a specialized simulation software (Radiance). Their results showcase how a designer can apply evolution to efficiently search a very large amount of variations and balance conflicting objectives.



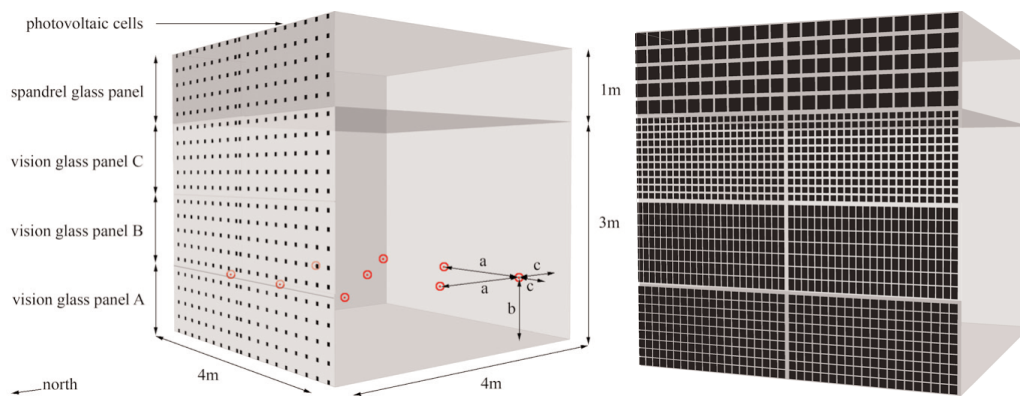


Figure 17: Snapshots from the work of Choo and Jansen [75], with the problem definition (left) and the optimized design (right). A simulation model describes a surface of photovoltaic cells, a simple interior layout, as well as a number of sensors at the building's interior. Source: [75]

Choo and Jansen [75] address the problem of designing integrated photovoltaic facades for buildings through an evolutionary optimization approach of “iterative refinement through simulation”, proposed by Jansen and Kauchik [124]. Their work focuses on the optimization of energy savings, which requires a balance of three conflicting sub-objectives: the maximization of energy generation, the minimization of cooling loads, and the maximization of daylight savings. The authors test their methodology on a relatively simple problem (shown in Fig. 17) which they address computationally with the Galapagos Evolutionary Solver [125] and manage to find good optimized solutions. Their method and results showcase the benefit of applying evolutionary computation to this type of problem, in contrast to the tedious manual exploration of design variations.

Chen et al. [74], in 2018, address the optimization of two conflicting criteria: 1) the efficiency of a building's cooling system and 2) optimizing the daylight access in the building. The “conflict” between those parameters becomes more significant in the context of their case study which is situated in a tropical zone, characterized by intense sunlight and high temperatures. They address the problem by evolving variations of the building's form and envelope. Their work is a very good example of how evolutionary optimization can work together with parametric models. More precisely, they use a parametric model which loosely defines building topology as a trapezoidal shape with a courtyard in the middle. By changing the model's parameters, a large number of geometric implementations (variations) can be produced. The model's parameters are effectively used as the “genotype” and a multi-objective evolutionary algorithm is searching for the best combination of those parameters, so as to optimize the building's cooling efficiency and daylight access simultaneously.

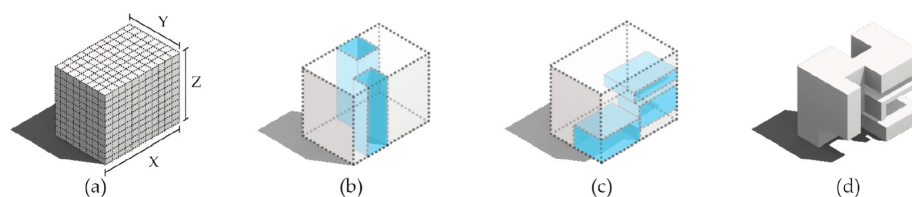


Figure 18: Example illustrations from the work of Wang et al. [126] on performance-based optimization of buildings' massing design based on horizontal and vertical subtractions. Source: [126]

Wang et al. [126], in 2019, showcase how performance-based optimization incorporating parametric modelling and evolutionary optimization can allow architects to leverage building massing design to improve energy performance. Their approach focuses intensely on the topological and geometrical variability which their system can generate. More precisely, they use a subtractive building massing approach which allows them to generate a very large variety of building forms (as shown in Fig. 18). Their work focuses mostly on the specific details of their proposed representation and how they affect the ability of the algorithm to discover novel topological formations.

### ▪ 3.1.5 Conclusions

The above survey showcased how many different perspectives pose functional criteria, in the form of constraints or fitness functions, in the process of evaluating design solutions related to architecture. The related works were purposefully chosen to frame the problem through a lens of analysis, optimization and design automation. This way we did not only distinguish several important functional criteria, but also examined them directly in a relevant context, from a computational perspective.

In the case of Architectural Design (section 3.1.1) the functional criteria are mostly related to the organization of space, from a topological and geometrical perspective. As suggested in the relevant studies, a strong emphasis is given to the subjectivity of the relevant constraints that depend on the client's preferences and on the designer's expertise and preferences. Despite this subjective element, it appears that architectural design optimization tools can still be very useful, as they enhance the designer's ability to search the design space, especially at the conceptual stages of the design process. Structural engineering (section 3.1.2) focuses mostly on the optimization of structural integrity versus cost, however there is a lot of room for exploration of variations in relation to other factors, including a building's aesthetic and utilization criteria. As explained in [82], the use of evolutionary computation or parametric models can be a very useful link for enhancing the collaboration between structural engineers and architects. Finally, in the field of Sustainability Design (section 3.1.3) we can detect a large amount of relevant work. Optimization techniques in that field are widely used for addressing the hard, multi-objective problems that arise in that discipline. Apart from the purely technical aspects of that discipline, there are also exemplary works that showcase how the engineering aspects can be utilized in the architectural form-finding process (for example [126, 121, 74]).

It should be noted that in the literature review there was little consideration of the relationships between disciplines and how an automated generative system can take into account constraints for different disciplines. We instead rely on new findings collected by AEC partners and described in Section 3.2.1. The PrismArch application will be the first to address these concerns through an QD assistive tool.

### ○ 3.2 PrismArch applications

Despite the large amount of constraints and criteria that can be taken into account in the process of architectural design, the design space of architectural problems is so vast that even after taking into account all of the constraints and optimizations, there is still a lot of room for exploration. As discussed in Section 2.1.1 there is usually no single best answer to an architectural problem. Combining the analytical and processing power of computers (under the proper algorithmic approaches) with the overview of human designers and engineers

seems to be a very promising approach of addressing architectural design, because it can promote a more efficient search of the space of possibilities while taking into account the architectural and engineering constraints and allowing for other functional criteria that may be subject to a client's requirements or a designer's preferences.

One of the goals of PrismArch is to exemplify ways in which various engineering approaches can share a common framework during the design process. From a computational perspective, this goal could be addressed via a generative system that can provide useful suggestions to a designer, by taking into account a large number of possible constraints and functions.

As the literature suggests, there are many facets of architectural design that can be addressed computationally. Some examples include floor plan layouts, overall building volumes, the arrangement of elements of facades, the form and function of structural elements. Floor plan layouts and building volumes are the problems with the highest impact factor, in the sense that they embed a large proportion of the architectural process, at least at the initial stages of design.

As discussed in Section 2.2, we suggest that a system of abstract generative architecture should be first and foremost as unrestricted as possible, in terms of the topological and geometrical characteristics of the solutions that it can generate. While the representation can be expressive, constraints and fitness functions can control the types of results that such a system produces. The following sections describe a number of constraints as well as functions that can be utilized in the context of an evolutionary exploration of the design space of specific architectural plan layouts.

### ▪ 3.2.1 Constraints Arising from Inter-Disciplinary Collaboration

Interdisciplinary constraints are an aspect of the design process that is especially important in the context of large-scale design projects. The examples of evolutionary design that can be found in the relevant literature, however, are usually focused on problems of relatively small scale or on problems of a larger scale but from a relatively narrow perspective (within the bounds of a single discipline). Consequently, the best way to identify these constraints was through feedback from the AEC industry partners of PrismArch, each from their own perspective (architecture, structural engineering, MEP engineering). This section reports on the most important ones.

More specifically, in the questionnaire (see Section 1.3) this question was posed in two parts. "**Question 3**" asks: "What are the most critical constraints that are imposed on your discipline by other disciplines?" and "**Question 4**" asks: "What are the most critical constraints from your discipline that you feel other disciplines should respect?". This section summarizes the responses of PrismArch partners to those questions and attempts to capture the interdisciplinary constraints landscape in this way.

#### **Inter-disciplinary constraints from the perspective of architectural design:**

In response to Question 3, ZHVR partners point out two critical constraints: The first one is that the "Efficiency and responsiveness of the design process" (i.e. the design team's ability to turn around a design revision in a short time) is key for client satisfaction. The second one is that "The other disciplines' ability to highlight any potential risks and propose possible solutions early on in the project" is crucial throughout a project.

Given the non-standard nature of ZHA's design approach and as became apparent through discussion with ZHVR partners, we are given to understand that these constraints are to be seen in the context of projects that involve a significant amount of risk. Diverging from the typical and taking the risk to explore new regions of the design space (in conceptual and / or technological terms) requires an increased amount of coordination both internally and in relation to external factors such as the relation with the client and the cooperation with other involved disciplines.

In response to Question 4, ZHVR partners point out three key aspects of architectural design that should constrain the design activity of disciplines. The first one is the definition of habitable spaces: any modification to the habitable spaces should always happen in coordination with the architects, so as to ensure that it does not cause problems in the experience of the end user. The second constraint is the uniqueness of the project / design solution. ZHVR partners suggest that one way of respecting this constraint is that MEP and structural engineers are designing within the bounds of specific envelopes, provided by the architects. The third and final constraint is the overall "narrative that defines a project, from its large gesture / big picture to the detail level. As ZHVR point out, there is nothing worse than a design intervention that "damages or breaks the golden thread, or breaks the design history".

The constraints raised by ZHVR are essentially measures for protecting the "design intent", as it is the main axis of framing the end users' experience and serving their expectations. Given the particular nature of ZHA's design approach, which more often than not deviates from the typical in terms of geometry, technology and concept, all of these constraints become especially important and intertwined with the associated risks and the dynamic nature of the interdisciplinary cooperation.

**Inter-disciplinary constraints, from the perspective of structural engineering:**

AKT II partners recognize four main types of constraints that originate from other disciplines and external factors and affect the design process for structural engineers: First, they recognize cost as the main driver in the design process and report that structural design is directly constrained by it. Second, they report that various types of architectural requirements often impose geometric constraints to a solution and, as a consequence, non-optimal structural solutions are often selected for the benefit of architectural function. Third, they explain that MEP requirements often impact structural elements by requiring openings that result in increasing the structural element size and cost. Finally, the physical aspects of the site such as the ground condition, existing services etc can be a big part of the overall constraints, influencing the structural element position or maximum loading allowed.

Furthermore, AKT II partners point out that the most important structural constraints that all other disciplines should respect are the ones that relate to safety and structural integrity. Namely, the site constraints, fire requirements, seismic requirements and performance requirements. As they explain, those constraints are universally respected by all other disciplines and the only requests that arise in the context of collaboration usually just relate to ensuring that the building is "working as hard" as possible, i.e. that it has been optimized so as to reduce carbon emissions, cost or construction times, within the bounds of the aforementioned constraints.

**Inter-disciplinary constraints, from the perspective of MEP:**

Sweco partners reported that the design process of MEP engineers is often driven by decisions made by structural engineers and architects. Constant collaboration with the other disciplines is considered to be very important, so as to coordinate the MEP services and save costs from errors that may occur on the construction site. Various types of negotiations around possible changes or tolerances may occur during the design process.

On the other hand, Sweco partners point out that other disciplines should make sure to provide the necessary space for installation and maintenance of MEP systems. These may include plant areas, corridor ceiling voids, apartment ceiling voids or even roof space for installation and maintenance of units.

**Inter-disciplinary constraints, conclusion:**

To conclude, all AEC partners were able to recognize the fact that their design activity is constrained by external factors (including the activity of other disciplines), as well as the fact that their domain knowledge and responsibility introduce constraints that other disciplines must recognize and respect. Furthermore, all disciplines recognize the importance of effectively communicating those constraints throughout all the stages of the design process and, in some cases, a large emphasis is placed on the initial stages of design, where a number of important and high-level design decisions are made.

**▪ 3.2.2 Project-specific constraints**

In addition to the constraints arising during interdisciplinary collaboration highlighted by the AEC partners during workshops and in their responses, there are also project-specific constraints which are specified by the designer/expert in cooperation with the client. These constraints are stored in a formal structure that describes certain aspects of the problem at hand. The proposed application can operate on a large set of such constraints, each of which can be clearly specified. The complete list is described in full detail in 2.2.1, and includes the following constraints: 1) A list of space units that enumerates all the rooms and other spaces that the solutions should incorporate, 2) A connectivity matrix that describes the desired connections between pairs of space units that should be directly accessible, 3) The optimal areas per space unit, or acceptable ranges of areas, 4) the floor / level per space unit, in case the layout spans more than one floors, 5) the differentiation between interior and exterior spaces, 6) the imposition of specific geometrical boundaries, 7) the proximity preference matrix, that describes the preference for geometrical or topological proximity between rooms and 8) the optimal surface area and orientation of windows for each space unit. As explained in 2.2.1, some of those constraints are entirely optional, while others are integral to guide the generation and evaluation of the evolving floorplans.

The role of all those constraints is to abstractly define the solution, by providing some of its topological and geometrical characteristics so that the evolutionary design system can “fill in the gaps” and generate concrete geometrical implementations that satisfy them. In other words, they are the rules that differentiate feasible from infeasible solutions in the search space [159]. From a technical perspective, apart from stating the constraints, one must also declare the method for a) inspecting whether a specific solution satisfies them and b) finding ways to gradually correct the infeasible solutions and turn them into feasible. As far as inspecting constraint satisfaction is concerned, we have already implemented the necessary algorithms for testing the satisfaction of most proposed constraints. Those algorithms are

manually developed to take advantage of the topological and geometrical data structures of the solution representation. For “solving the problem” and actually generating a set of feasible solutions, this is undertaken through specialized constrained optimization algorithms such as the feasible-infeasible two-population family of algorithms [159,157]. These methods simultaneously solve the problem of feasibility, while optimizing along some selected fitness and generating a diverse population along a number of “behavioral” dimensions [160].

Finally, we should clarify that the proposed list of constraints does not necessarily need to be utilized completely by the designer. Some of the proposed explicit constraints are absolutely necessary for the formulation of an architectural problem (such as the list of space units, the connectivity matrix and the optimal area constraints), while others could be treated as options. For that reason, the QD assistive tool is developed so that it can operate even with a partial subset of the proposed constraints.

### ▪ 3.2.3 Fitness functions

Within the broader practice of architecture, engineering and construction, there are countless features that can be treated as problems of optimization. The literature survey (section 3.1) presents a range of relevant examples, covering a historical perspective as well as the state of the art across the disciplines of architecture, structural engineering and sustainability design. The main purpose of this survey is to showcase the broad spectrum of applications of design optimization, as well as to point out a number of technical aspects that need to be taken into account.

In the context of PrismArch, we believe that it is crucial that we combine the existing theoretical knowledge that the literature survey encapsulates with the expertise of PrismArch partners from the AEC industries, so as to distinguish the most valuable aspects of design optimization (in the broader context of the PrismArch application) and focus on them. During a series of workshops relating to Quality Diversity, AEC partners distinguished the most important aspects of design optimization (each from their own perspective/ discipline) and formalized them in their answers through a relevant questionnaire. This section presents a number of possible ways forward, by taking into account both the state of the art, as well as the specific input of AEC partners.

#### **Structural engineering:**

As far as the field of structural engineering is concerned, feedback from AKT II partners via the questionnaire and workshops (see Section 1.3) spans over two distinct aspects that relate to construction: Structural Optimization and Fabrication optimization.

As AKT II partners explain, many types of structural optimization are included in the process of structural engineering. Some of the most basic examples include the automated selection of section profiles from predetermined groups of options, based on FEA analysis. Other than that, they report that one of the most important features of optimization is the structure weight. This is usually approached via parametric tools that apply simulations on the designed geometry in order to optimize it. As they explain, optimizing (minimizing) the structure weight subsequently optimizes cost and sustainability. On the other hand, AEC partners report that the design process is often driven by cost appraisal. The limitations posed by the client’s budget are used to assess the structural impact and may necessitate the review of alternatives that combine different structural options, different grid opportunities (spacing between structural elements) as well as their relation to architectural and MEP constraints.

Fabrication optimization is another aspect of design optimization that was brought up by AKT II partners. As they report, one way of achieving fabrication optimization is through the regularization of discrete elements in some way. An example would be to make all of the connections in a structure conform to a limited set of typologies, or making all of the member lengths identical / one of a limited set of different sizes. According to AKT II partners this type of fabrication optimization is especially important when working with non-standard structures. Another relevant problem is to ensure the planarity of elements, so as to simplify the forms and ensure that they can be constructed through planar sheets of material that do not require expensive processing like hot forming.

**MEP engineering:**

The feedback received from the MEP partners (see Appendix A) suggests that there are several aspects of building services that can be treated as problems of optimization. Mentioned features include: (1) finding the best routes with less friction, (2) calculating and properly designing spaces and clearance areas, (3) effectively arranging services in corridors and/or risers and (4) adjusting the user interface based on the user.

From the perspective of a Quality Diversity (QD) design tool, the first feature (best routes with less friction) seems to be the most complex one. That is because it likely relies on a quite detailed data representation that exceeds the geometric properties of the routes and, furthermore, the degree of domain-specific knowledge required to implement them in QD may be beyond a realistic scope.

As far as (2) and (3) are concerned (designing spaces and clearance areas and arranging services in corridors / risers), those aspects seem to be much better candidates for an actual implementation in the context of QD. Both of them seem to be less detailed than (1) and the methods for generating and/or evaluating a specific arrangement are probably easier to grasp and implement. Consequently, these aspects could be selected as a “gateway” for integrating MEP optimization in the context of QD and PrismArch.

**Sustainability:**

Through the input of all AEC partners, it became evident that sustainability is an aspect that concerns all disciplines involved. All PrismArch partners from the AEC industry (ZHVR, AKT II, Sweco) mention sustainability as an important aspect of the optimization process of a design solution. PrismArch can be an ideal setting for addressing factors of sustainability in an interdisciplinary setting. The proposed Quality Diversity generative AI system can help towards this goal by providing the tools to perform multiobjective evolution and examine many (potentially conflicting) aspects of optimization, at once.

Based on the relevant literature review (section 3.1.3), we found that in the field of sustainability design, optimization problems are very often multi-objective. The main goal, in such cases, is to find optimal solutions that manage to address a rather large number of partially conflicting objectives. Energy efficiency is one of the main aspects of sustainable design that have been addressed via computational optimization. An example multi-objective problem in that context would be to maximize the insulation of a building in relation to a warm exterior environment (so as to reduce the energy consumption of HVAC systems for cooling), while at the same time maximizing the amount of natural light (so as to reduce the need for artificial light) and minimizing the material and construction cost. Many of the relevant fitness measures are tightly coupled with specific construction technologies such as,

for example, the technology of double-skin facades which, when properly designed, can offer multiple long-term benefits in terms of energy efficiency.

#### Conclusion:

In general, incorporating a number of engineering functions into the evolutionary design system will greatly increase its practical value. Thankfully, there are many tools available, some of which are reported in Section 3.3 that may be embedded or interconnect with the QD evolutionary loop. Such tools could be used directly within the evolutionary algorithm as an active part of the generative process or, alternatively, as analytic reports that are received on demand, for a smaller subset of solutions.

In any case, it would be best to focus on engineering functions that are relatively “light” from a computational perspective, especially if they become a part of the evolutionary design process (and not simply used as informative reports, at the end). Simulation-based evaluations tend to be very time consuming and, if incorporated without some attention, may cause serious side-effects in the algorithms. Furthermore, it would also be wise to select a set of engineering functions that can be calculated based on the solution representation that is proposed in Section 2.2. An example of a compatible engineering function would be to evaluate the quality of a generated plan in terms of its usage of natural light. Since the suggested representation precisely describes the layout, including the position, shape and size of windows, natural light coverage seems feasible that it can be directly evaluated from that perspective, using one of the available, open-source tools that we have catalogued.

On the other hand, some of the engineering perspectives rely on design aspects that are simply missing from the proposed solution representation. Structural engineering, for example, is calculated based on a specific technology of construction which may be a reinforced concrete structure consisting of columns, walls and slabs, or a metal skeleton structure. In order for generated architectural layouts to be evaluated from the perspective of structural engineering, an intermediate layer of generating those structural solutions should first be implemented or utilized, so that the result can then be evaluated. Overloading the solution representation with all necessary aspects to address any type of engineering problem would be overcomplicating the problem and lead to many negative side-effects both on the development process and on the runtime needs of the system. Those extra layers of information and processing can be regarded as parts of potential and specialized fitness function calculation methods.

Overall, the engineering features that will be incorporated should be very carefully selected, taking into account all the relevant technical issues and the way they can affect both the technical development and the operational efficiency of the AI application. As a following step we plan to examine the available methods and tools more closely, from that perspective, as well as consult with AEC partners so as to make the best decisions in that regard.

### ○ 3.3 Software, Tools and Algorithms

*Table 2: Software that could be used for function evaluation and constraint satisfaction tests within PrismArch*

Name	License	Description	Possible use
<b>Karamba 3D</b>	Commercial, closed source	An interactive, parametric finite element program. It lets you analyze the response of 3-dimensional beam	



<a href="https://www.food4rhino.com/app/karamba3d">https://www.food4rhino.com/app/karamba3d</a>		and shell structures under arbitrary loads.	
<b>Visualarq</b> <a href="https://www.food4rhino.com/app/visualarq">https://www.food4rhino.com/app/visualarq</a>	Commercial, closed source	VisualARQ adds flexible BIM features to Rhino, and speeds up the process of modeling an architectural project in 2D and 3D. VisualARQ provides tools to generate all project document information (plans, sections, elevations) and quantification (surfaces, dimensions, components, quantities) from the 3D model automatically.	
<b>Rhinovault: Funicular Form Finding</b> <a href="https://www.food4rhino.com/app/rhinovault">https://www.food4rhino.com/app/rhinovault</a>	Free	The Rhinoceros® Plug-In RhinoVAULT emerged from current research on structural form finding using the Thrust-Network- Approach to intuitively create and explore compression-only structures.	
<b>Rhinomembrane v2.0</b> <a href="https://www.food4rhino.com/app/rhinomembrane-v20">https://www.food4rhino.com/app/rhinomembrane-v20</a>	Trial	Rhino Membrane is one of the most powerful tools for form finding of tensile structures and yet most simple to understand and use.	
<b>Intralattice</b> <a href="https://www.food4rhino.com/app/intralattice">https://www.food4rhino.com/app/intralattice</a>	Free, open source, MIT license	This is the beta release of Intralattice, a plug-in for Grasshopper used to generate solid lattice structures within a design space.	
<b>Dodo</b> <a href="https://www.food4rhino.com/app/dodo">https://www.food4rhino.com/app/dodo</a>	Free, closed source	Dodo is a collection of tools for machine learning, optimization, and geometry manipulation.	
<b>Polyframe</b> <a href="https://www.food4rhino.com/app/polyframe">https://www.food4rhino.com/app/polyframe</a>	Free	PolyFrame is a geometry-based, structural form-finding plugin for Rhinoceros3d based on the principle of the equilibrium of polyhedral frames known as 3D/polyhedral graphic statics.	
<b>Ladybug tools</b> <a href="https://www.food4rhino.com/app/ladybug-tools">https://www.food4rhino.com/app/ladybug-tools</a>	Free, AGPL-3.0+	<p><b>Ladybug:</b> Import and analyze weather data in Grasshopper. Diagrams: Sun-path, wind- rose, radiation-rose, etc. Radiation analysis, shadow studies, and view analysis.</p> <p><b>Honeybee:</b> connects Grasshopper3D to validated simulation engines such as Energy-Plus/OpenStudio and Radiance for building energy, comfort, daylighting and lighting simulations.</p> <p><b>Dragonfly:</b> creation and manipulation of large-scale EnergyPlus and Radiance models by capitalizing on an abstracted 2D representation of building geometry, where all rooms are assumed to be extrusions of floor plates.</p> <p><b>Butterfly:</b> connects Grasshopper to the OpenFOAM engine, which can be used</p>	Daylight simulations, wind comfort and other types of simulations that may be used as function evaluations of a generated design.

		to run advanced computational fluid dynamic (CFD) simulations.	
<b>Air conditioning and ventilation design</b> <a href="https://www.food4rhino.com/resource/air-conditioning-and-ventilation-design">https://www.food4rhino.com/resource/air-conditioning-and-ventilation-design</a>	Free		
<b>Wind comfort prediction with computational fluid dynamics (by SimScale)</b> <a href="https://www.food4rhino.com/resource/wind-comfort-prediction-computational-fluid-dynamics">https://www.food4rhino.com/resource/wind-comfort-prediction-computational-fluid-dynamics</a>	Free	Wind comfort prediction with computational fluid dynamics (CFD)	



## ▪ 4 DESIGN EXPLORATION DIMENSIONS

This section focuses on the design exploration dimensions which can be explored through Quality-Diversity search. These dimensions are an important way to differentiate individual or general stylistic choices and preferences. We use “design exploration” here in an abstract manner, denoting properties that can differentiate a set of solutions but which are not criteria that need to be satisfied (or maximized). We rely as much on literature related to disciplines of architecture or engineering as on broader cognitive aspects of perception and beauty [5].

Design exploration dimensions are important to define in this Deliverable for two reasons. First, the AI algorithms developed for PrismArch largely rely on evolutionary search towards Quality-Diversity [6,7], which requires some variance without necessarily requiring a solution to have more or less of a specific design dimension. Showing a broad range of diverse choices (which are still high-quality as the AI optimizes the quality and constraint dimensions discussed in Section 3) is more likely to inspire the designer. The second reason is that these properties are likely to be subjective and depend on the idiosyncratic style or priorities of a certain phase of the design; therefore they are ideally suited for computational models which adapt to a specific designer based on prolonged interactions with the editing interface. Therefore, creating a personalized preference model that can evaluate content specifically towards a designer can provide generation that is stylistically appropriate without compromising function and constraints which remain unchanged.

### ○ 4.1 Survey

The structure of this survey follows a path from the general aspects of design, inspired by studies in cognitive psychology and neuroscience, to the specifics of a built and livable environment. Therefore, Section 4.1.1 provides a general overview of evaluation in related fields such as generative art and game visuals, while Section 4.1.2 focuses on aspects of space as identified by Christopher Alexander, and finally we focus on the human experience in Section 4.1.3 by viewing the built environment from the perspective of the person navigating through it.

#### ▪ 4.1.1 General visual properties

In his book *“Art and Visual Perception”* [127], cognitive psychologist Rudolf Arnheim observes the psychological impact of certain art pieces on the viewer by assuming a holistic perceptual processing of the scene. Introducing the term perceptual forces as the psychological and physical forces that guide the viewers’ attention at specific points and along specific axes on an object or scene, he attempts to identify the most important contributors to the creation of these forces: the simplest of those are balance and shape. For Arnheim, the main contributors of balance are weight and direction: weight refers to the pull of the viewer’s attention on specific areas and is influenced by location (with more importance given to the image’s center and the horizontal and vertical axis), while direction guides the viewer’s attention along specific axes. On the other hand, Arnheim approaches shape in the context of the minimal visual cues that can accomplish identification. He attributes the perception of shape to simplicity, subdivision, similarity and difference. Simplicity is achieved when structural features of the shape are arranged in an easily deductible and memorable pattern; such structural features “can be described by distance and angle” [127]. Subdivision refers to the human ability to group visual cues in order to dissect the whole into visually distinct parts.

Similarity can visually group distinct shapes or features into a single unit or pattern, while difference is perceived as an anomaly and grabs the viewer's attention.

From a different scientific field, neuroscientists Ramachandran and Hirstein [128] has also suggested "speculative and arbitrary" laws of art; these eight universal laws, grounded mostly on empirical studies of the brain, are: peak shift, isolation, grouping, contrast, perceptual problem solving, symmetry, abhorrence of coincidence and metaphor. Ramachandran and Hirstein identify these properties of visual perception as common in all brains and thus resistant to cultural influences.

Principles such as balance and symmetry have been extensively used for content generation. Ochoa [129] evolved the expansion rules for Lindenmaier systems in order to generate plant-like structures according to five features that included the height of the structure and its bilateral symmetry as the ratio of nodes on the left side and the right side of the structure. Liapis et al. [130] used a similar method to measure weight in generated polygons representing spaceships by assessing the surface ratio of the bottom half and top half of the polygonal shape, as well as the surface ratio of the middle third of the shape along either the vertical or horizontal axis (see Fig. 19). While these surface ratio metrics assess balance, in the sense of weight and direction proposed by Arnheim [5], a stricter measure of symmetry was also evaluated by Liapis et al. as the ratio of the intersecting surfaces between a polygon and its reflection over the union of these two surfaces. Other measures of form highlighted by Arnheim were evaluated by Liapis et al. in the form of simplicity (as the perimeter of the polygon compared to the perimeter of an oval enclosed within the same bounding box) or jaggedness (the ratio of angles that were between 20o and 60o or 300o and 340o over all angles). In a related experiment, Liapis generated spaceship sprites by combining human-authored components, and evaluated the massing (i.e. the non-transparent parts) of the pixel images in terms of surface ratios (top half versus bottom half, middle half along X or Y axis versus exteriors, total surface of sprite versus surface of bounding box), outline, and connections between component sprites. Moving closer to the task of design of architectural spaces, Alvarez et al. [131] evaluate symmetry of a tile-based level representation (see Section 2.1.2) based on the presence of the same wall tiles in the closest of four different reflections (including horizontal, vertical, and diagonal symmetries) of the level. While Alvarez et al. [131] attempt to maximize symmetry, Sfikas et al. [132] explored the evaluation of two different symmetry metrics as either a diversity measure or as a quality measure for a MAP-Elites implementation of quality-diversity search [133]. The metrics of Sfikas et al. assessed symmetry along the horizontal axis (see Fig. 20) or bilateral symmetry based on presence of walls at the same positions in the original content and its reflection(s).

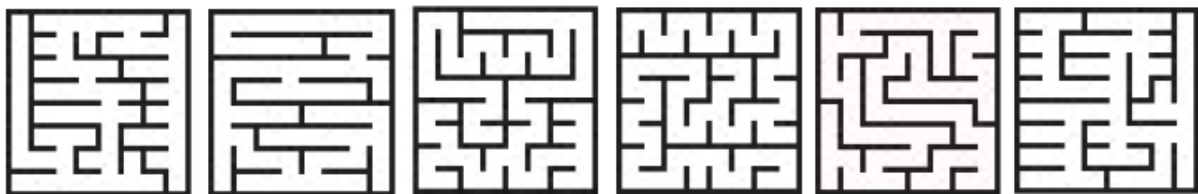


Figure 20: Mazes generated through MAP-Elites by Sfikas et al. [132]. Mazes to the left have a higher horizontal symmetry (with their reflection on the X axis) than those to the right.

In evolutionary art, symmetry and order has been studied and formalized in many different ways. In the form of mirror symmetry (i.e. perfect match between a pattern and its reflection), Gartus and Leder [134] assessed symmetry along four axes (horizontal vertical, main and

secondary diagonal) as the ratio of matching pixels between an image and its reflection. The four symmetry scores calculated in this way were then averaged to provide a general symmetry score between 0 and 1; this averaging form of aggregation is unlike the study of Sfikas et al. [132] where each symmetry was a different quality or diversity dimension, or the study of Alvarez et al. [131] where the highest score of the four was used as a measure of symmetry. Gartus and Leder generated black and white patterns consisting of triangular elements using MATLAB through the stochastic optimization process of simulated annealing [135]. These patterns were assessed by over 100 participants in terms of visual complexity, and showed a significant negative correlation between symmetry and perceived visual complexity. This means that viewers perceive symmetrical patterns as less complex than asymmetrical ones (which presumably seem “noisier”). Hubner and Fillinger [136] evaluated the mirror symmetry in pre-made patterns and users’ preference to them, although the correlations were not as strong as with other measures of balance (discussed below).

Other measures of alignment have also been tested, such as homogeneity as the similarity of number of black pixels in different subsections of a binarized image used by Hubner and Fillinger [136] in the aforementioned study, and “parallelism” as the similarity of orientation of different edges in the image [137]. Parallelism in this sense was assessed by Redies et al. based on the detected edges of an image process algorithm on a dataset of real-world images; a similar study on edge co-occurrence of real-world images was conducted earlier by Geisler et al. [138].

A broader measure of balance, rather than exact symmetry, has often been used as a design criterion and as an avenue of study for its psychological effects. Wilson and Chatterjee [139] generated a set of black and white images with black circles or hexagons distributed around a white canvas in order to assess how visual balance was assessed by human users and how it affected their preference. Similar to Liapis et al. [130], Wilson and Chatterjee assessed the generated patterns based on the ratio of pixels in each half of the image using 8 split criteria (horizontal, vertical, main diagonal, antidiagonal, and inner/outer divisions on the horizontal, vertical or two diagonal axes). Hubner and Fillinger [136] included a measure of balance as the Deviation of the Center of Mass, which “represents the center of perceptual “mass” in a picture and its deviation from the geometric center” [136].

Another important aspect highlighted by Arnheim is complexity, which has been assessed primarily through image processing techniques in evolutionary art and visual appreciation studies. Popular dimensions of this complexity are self-similarity (how similar the image as a whole is to its parts), anisotropy (difference in magnitude of changes in luminance or color across orientations in an image) and image compression (which assumes that simple images have redundant information and predictable data which can be compressed at higher rates than complex images). As such, complexity evaluation methods have mostly been applied to 2D images, rather than e.g. polygon representations, and more often than not these images were real-world photographs or paintings [140]. Complexity is often assessed based on the histogram of orientation gradients (HOG), which assesses the mean magnitude of changes in luminance or color in an image [141], or the Fourier slope [142] which is “an indicator of the strength of low spatial frequencies (representing coarse detail) relative to high spatial frequencies (representing fine detail) in the image” [143]. Machado et al. [144] use Zipf’s law-based metrics to assess the complexity of an image as the rank frequency and the size frequency. Zipf’s rank frequency is based on the number of occurrences of each pixel intensity value in the image and assessing their rankings based on a simple mathematical formula.

Zipf's size frequency assesses the difference between a pixel's value and the value of each of its neighbor pixels, processing it in terms of ranks as with the rank frequency. Machado et al. [144] also use alternative measures for complexity, such as a comparison in terms of pixel differences between the original, lossless image and its JPEG-encoded or GIF-encoded version. As noted above, the lossy conversions of JPEG and GIF would lead to more pixel changes or would not be able to create smaller filesizes. Both the pixel differences and the compressed files' size was used for this assessment of complexity.

#### ▪ 4.1.2 Aspects of Space

Beyond general aspects of style and design that permeate diverse facets of creativity such as visual arts and photography, Christopher Alexander laid out a set of fundamental properties in his book *"The Nature of Order"* [145]. Alexander uses "life" as the rhetoric vessel in this book, but builds on the author's 20 year experience following his own fundamental book on *"A Pattern Language: towns, buildings, construction"* [146]. The key to the idea behind *"The Nature of Order"* is that every part of space —every connected region of space, small or large— has some degree of life, and that this degree of life is well-defined, objectively existing, and measurable [145]. Alexander focuses on centers throughout this book, identifying centers those elements which appear within the larger whole as distinct and noticeable parts. In this context, a center can be defined only in terms of other centers, while centers are —and can only be— made of other centers. Alexander identifies 15 properties of centers relating to scale, strong centers, boundaries, alternative repetition, positive space, good shape, local symmetry, contrast, echoes, void, simplicity and inner calm, not separateness. Of these fifteen qualities, some can be more amenable to mathematical or computational formulation than others. Alexander is deliberately poetic in his description of many of these properties, and it was a challenge even to summarize them succinctly in the above list. However, a number of properties can be somehow formalized. Indicatively, the strong centers can be assessed in terms of their centrality in a connectivity graph, while moreover the presence of multiple strong centers linked together (and intensifying each other) can be assessed with more specialized graph algorithms such as PageRank. Alternating repetition can be naively assessed in terms of N-grams on metrics or usage of each space (thus finding consistent patterns). Roughness can be assessed in terms either almost but not perfect symmetry (e.g. as a desirable symmetry ratio in the symmetry metrics or via complexity (see Section 4.1.1).

#### ▪ 4.1.3 Architectural space heuristics

This section showcases analytical features of architectural space that relate to subjective elements of human perception. The following paragraphs describe relevant studies that revolve around notions of visibility and accessibility and attempt to correlate them with human experience and behavior. The purpose of reviewing those studies is to detect metrics which can be used in the context of quality-diversity, so as to generate different designs that take into account the human perception of the environments that those designs describe.

Benedikt [147], in 1979, introduces a methodology for the analysis of architectural space as an environment (rather than as an object). His study focuses on the relation between space and visibility and proposes an analytical framework that could form the basis for connecting those with human perception and experience. More precisely, he suggests that the human perception of architectural space can be analysed via a geometrical construct known as "isovist". An isovist is simply the set of surface-points that are visible from a specific point in

space. For a given architectural plan, there are many isovists that describe how it is perceived from different points of view (see Fig. 21). Benedikt points out that those isovists have specific characteristics that differentiate them, including their *area* (the amount of space that can be seen from a point), their *real-surface perimeter* (how much environmental (real) surface can be seen from a point), *occlusivity* (length of the occluding radial boundary), *variance* or *skewness* of their radials and *circularity* (as another measure of their complexity). While for a given environment there is an infinite set of isovists (since space is continuous), Benedikt suggests that its quality can be captured in a rather small subset of those. He eventually proposes the use of “isovist fields” (see Fig. 21), as a way of transitioning from the use of single isovists to the use of all the isovists along a specific path. Overall, his paper is an attempt to formulate a foundational analytical methodology that uses isovist fields as a directly measurable property of architectural environments which can then be analyzed via theories of visual perception and spatial description, thus providing a concrete analytical evaluation of those spaces. Despite its initial limitations, Benedikt’s methodology has posed a new approach for the analysis of architectural space which is continuously studied and expanded since.

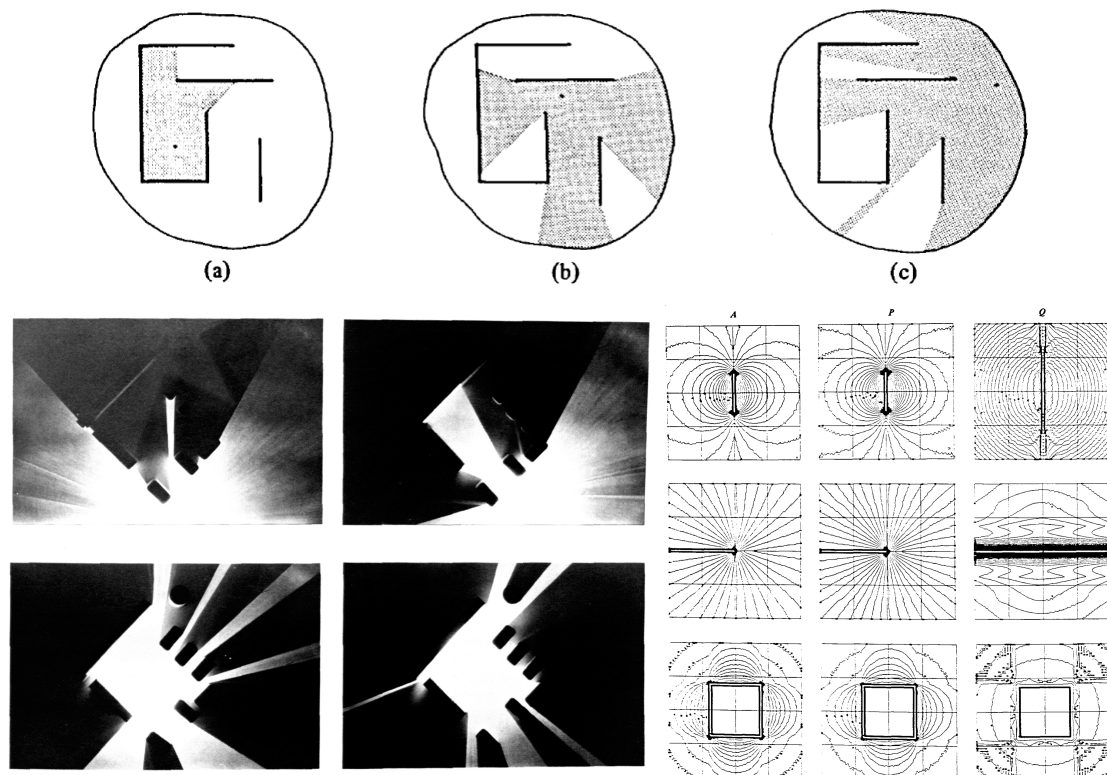


Figure 21: Example illustrations from the work of Benedikt [147] on isovists and isovist fields. Top: three isovists of an abstract architectural plan. Bottom left: Analog production of isovists along a path by point-source illumination of a model. Bottom right: Some examples of isovist fields for three (two-dimensional) simple environments: a free-standing wall, the end of a long wall and a free-standing square. Source: [147]

Turner et al. [148], in 2001, extend the idea that an architectural layout can be analyzed as a set of isovists, using a novel analytical methodology. They point out that a set of isovists can be also represented as a connectivity graph, in which the edges represent pairs of points that are visible to each other (see fig. 22). They show that this connectivity graph is equivalent to the set of isovists. Furthermore, they propose three metrics for analyzing the connectivity



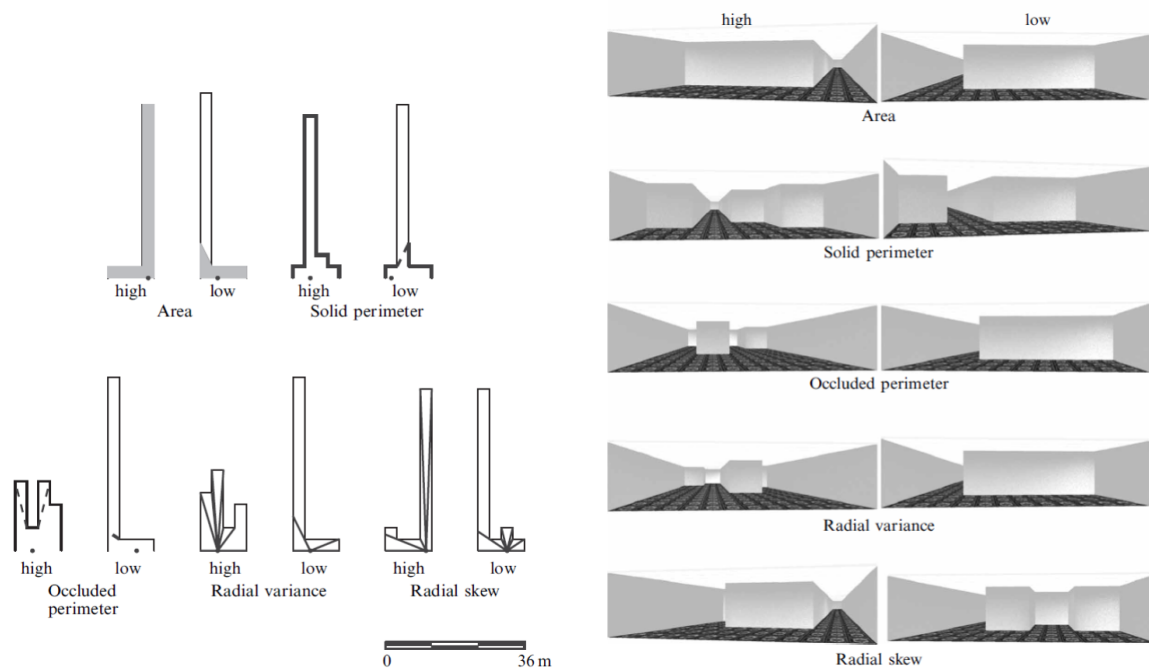
graph: the neighbourhood size, the clustering coefficient and the mean shortest path length. The *neighbourhood size* is a map that represents the number of active connections at each region of the analyzed space. The *clustering coefficient* is the total number of active connections divided by the total number of possible connections between cells (i.e. the number of edges the graph would have if it was fully connected). This is directly related to the convexity of a shape. For example, a convex polygon would have a clustering coefficient of 1. Finally, *the mean shortest path length* represents the average shortest path going from any node to any other node via the visibility connections. Applying these metrics on a real example generates a set of maps that characterize and differentiate its regions, as shown in Fig. 22.



Figure 22: Example results from the work of Turner et al. [148] on isovists and visibility graphs. Left: An example of a first-order visibility graph, showing the pattern of connections for a simple configuration. Right: Mies van der Rohe's Barcelona Pavilion showing neighbourhood size (left), visibility mean shortest path length analysis (middle), and accessibility mean shortest path length analysis (right). Source: [148]

Stamps [149], in 2005, offers a comprehensive review of studies relating the dependent variables of 'spaciousness' or 'enclosure' to independent variables calculated from the mathematical construct of an 'isovist'. As he explains, "enclosure is such an important feature of the environment that a specific region in the brain responds directly and selectively to it". His analysis is based on the original formulation of isovists, which takes into account the shape of the visible area from a vantage point and treats it as a polygon in 2D space. Based on this formulation, the main research question of his paper is: "how well do isovist measures predict responses of spaciousness or enclosure?". Secondly, he addresses the more technical aspect of: "how many measures are really needed to describe isovists?". His research is based on a number of case studies that include interior spaces, such as the hotel lobbies shown in Fig. 23, and exterior spaces. He conducts his experiments by analyzing the reported perceptions of spaciousness or enclosure on these case studies and examines their correlation with measurable properties of the corresponding isovists, including their horizontal size, boundary permeability, variation in distances to boundaries, concavity, boundary predictability and elongation, among others. As far as the main research question is concerned, his findings show that "impressions of enclosure are strongly related not only to actual horizontal size but also to variation in distance to boundary, to concavity, to elongation, to nearest distance, and to boundary predictability", which is aligned to the theory that "judgments of enclosure are actually judgments regarding the potential for safety or danger in a region". As far as the secondary research question is concerned, performing a principal component analysis showed that isovists can be effectively distinguished with two to perhaps

six properties. The most plausible properties are size and concavity, while mixed results were obtained for the other properties.



*Figure 23: Example illustrations from the work of Stamps [149], showcasing the case studies used to conduct his research. Left: examples of isovists with high and low values of area, solid perimeter, occluded perimeter, radial variance, and radial skew. The designs shown are floor plans for hotel lobbies of equal floor area, after [150]. Right: perspective sketches of the hotel lobbies shown in the left figure. Five pairs of hotel lobbies are shown. Each pair differs in one of five isovist properties: area, solid perimeter, occluded perimeter, radial variance, and radial skew. Source: [149]*

Koutsolampros et al. [151], in 2019, used Visibility Graph Analysis (VGA) and other related metrics so as to analyze the behavior of office workers, in the context of their working environment. The authors focus on two aspects of behaviour (movement and interaction) and investigate how the recorded behaviors relate to the properties of space and whether they can be predicted by the aforementioned analytical methods. Their methodology is based on a software tool called depthmapX [152], which offers 25 different analytical measures of space. They are roughly divided into six groups (also shown in Fig. 25): 1) the *Size* group, which includes Isovist Area, Connectivity and Isovist Perimeter, 2) the *Shape* group, which includes Compactness, Point First Movement, Point Second Movement, Min Radial and Max Radial Distance, 3) the *Potential to explore* group, which includes Drift Angle, Drift Magnitude and Occlusivity, 4) the *Potential to move* group, which includes Through Vision and Visual Clustering Coefficient, 5) the *Control* group, which includes Visual Control and Visual Controllability, 6) the *Global* group, which includes Angular Mean Depth, Metric Mean Shortest Path Angle, Metric Mean Shortest Path Distance, Metric Mean Straight Line Distance and Visual Mean Depth, 7) the *Normalized depth* group, which includes Visual Integration (including HH, P-value and Tekl) and 8) *Complexity*, which includes Visual Entropy and Visual Relativised Entropy. As they point out, some of these metrics have been disproportionately investigated in relevant research (the number of relevant citations for each one is shown in the table of Fig. 24). The goal of this research is to find how those metrics relate to movement and interaction of humans in office spaces, and results show that some metrics such as Visual

Mean Depth play an important role for understanding the effects of movement: more segregated floors and spaces tend to attract less movement. The authors also find that, of the two activities, movement is the easiest to predict, with many of the results applicable both to large-scale analysis but also on a per-site level.

Overall, as the reviewed studies suggest, making concrete correlations between isovist related metrics or visibility graph related metrics and specific aspects of human experience is not an easy task. The relevant research is still ongoing and many open questions remain. However, we believe that the utilization of some of the proposed metrics may be especially useful for the differentiation between architectural solutions. If nothing else, isovists and visibility graphs can be viewed as an encoding of an architectural environment that captures its complexity from the point of view of an observer that lies within it. Utilizing this aspect in the context of a quality diversity evolutionary can yield results where differences are much more meaningful than if they were relying directly on the top-down projection of a plan layout, for example.

Measuring	Metric	Extent	Units	Graph/Geometric	Citations
Size	Isovist Area	Local	Metric	Geometric	105
	Connectivity	Local	Topological	Graph	365
	Isovist Perimeter	Local	Metric	Geometric	28
Shape	Isovist Compactness	Local	Metric	Geometric	22
	Point First Moment	Local	Metric	Both	1
	Point Second Moment	Local	Metric	Both	2
	Isovist Min Radial	Local	Metric	Geometric	4
	Isovist Max Radial	Local	Metric	Geometric	8
Potential to explore	Isovist Drift Angle	Local	Metric	Geometric	5
	Isovist Drift Magnitude	Local	Metric	Geometric	7
	Isovist Occlusivity	Local	Metric	Geometric	21
Potential to move	Through Vision	Local	Topological	Both	19
	Visual Clustering Coefficient	Local	Topological	Graph	29
Control	Visual Control	Semi-Global	Topological	Graph	57
	Visual Controllability	Semi-Global	Topological	Graph	19
Global	Angular Mean Depth	Global	Angular	Both	11
	Metric Mean Shortest Path Angle	Global	Metric	Both	1
	Metric Mean Shortest Path Distance	Global	Metric	Both	8
	Metric Mean Straight Line Distance	Global	Metric	Both	3
	Visual Mean Depth	Global	Topological	Graph	39
Normalised depth	Visual Integration [HH]	Global	Topological	Graph	45
	Visual Integration [P-value]	Global	Topological	Graph	8
	Visual Integration [Tekl]	Global	Topological	Graph	5
Complexity	Visual Entropy	Global	Topological	Graph	16
	Visual Relativised Entropy	Global	Topological	Graph	4

Figure 24: Table from the work of Koutsolampros et al. [151] with the metrics provided by depthmapX [152] and the amount of relevant citations for each of them. Source: [151]



Figure 25: Example illustrations from the work of Koutsolampros et al. [151] on “dissecting” visibility graph analysis. Source: [151]

## ○ 4.2 PrismArch applications

This section presents a number of possible “modes of differentiation” between design solutions that can be used to define the dimensions of diversity in the Quality Diversity assistive AI tool, in the context of the PrismArch application. Similarly to 3.2, the survey of the

state of the art collected from the literature (Section 4.1) is combined with the input from AEC partners of PrismArch (data collection is described in Section 1.3).

- **4.2.1 Input from ZHVR:**

In the dedicated workshops and the questionnaire, ZHVR emphasizes the importance of treating a design solution as an emergent property that is an outcome of ongoing interactions between the designers and the client. Furthermore, ZHVR points out that during the initial stages of design, one of the most important aspects is the definition of the aspects that shape the boundaries of the solution space. This feedback is particularly useful to the broader scope and vision of the QD and designer modeling application as realized in PrismArch, and are discussed in greater detail in Section 5.2.

- **4.2.2 Input from AKT II:**

AKT II partners reported that the main characteristics that are explored during the initial stages of the design process usually include sustainable outputs, cost and structural performance. They point out that they use simulation software in combination with designer experience in order to evaluate these parameters both in isolation and in combination, so as to identify potential options to pursue in later stages. Furthermore, they report that all of the aspects that they address are quantifiable and can be at least approximately calculated during the initial stages of design. For example, cost can be approximated via directly measurable material quantities such as length, area, thickness or volume. Structural performance is calculated based on properties such as self-weight, weather and environmental loads, live loads from people, equipment, furnishings, while also taking into account the properties of the selected material and its structural properties. As far as sustainability is concerned, an important factor that structural engineers are focusing on is the embodied carbon, a property that encapsulates the carbon generated in the production, forming, transportation and maintenance of a material. Finally, another important aspect that relates to structural engineering is the fabrication process, especially in relation to 'non-standard' structures with unusual forms, irregular shapes, etc.

Most of the reported dimensions of exploration from AKT II can also be perceived as aspects of optimization. However, this does not prohibit a QD algorithm from using them as dimensions of exploration. As was showcased by AEC partners, using quantifiable measures for the exploration of design solutions is deeply integrated in the practice of structural engineering. Therefore, an important next step is selecting a set of specific features and integrating them in the quality diversity AI assistive tool.

- **4.2.3 Input from Sweco:**

Sweco partners reported that during the initial stages of design they explore several aspects. These include, for example, finding the best routes with less frictions, calculating the necessary spaces and clearance areas, effectively arranging services in corridors and / or risers, user interface adjustments based on the user among others.

Similar to structural engineering, several aspects of MEP engineering can influence the dimensions of search in the context of Quality Diversity search. During upcoming development steps the exact methodologies or algorithmic processes for evaluating the reported aspects, in cooperation with Sweco, will be identified and integrated.

- **4.2.4 Proposed dimensions of diversity:**

Based on the literature review of section 4.1, we propose a number of dimensions of diversity which can be investigated in the context of the Quality Diversity AI assistive tool. These dimensions provide geometric and/or topological differentiation between design solutions that are treated as rather “neutral” characteristics. Each one of them, in isolation, does not necessarily serve a specific purpose or design goal. However, utilizing a set of such neutral dimensions of diversity assists the algorithm in efficiently populating the design space (with samples that are geometrically and topologically different) and discovering the regions of that space where the objectives can be further maximized. Furthermore, the proposed set of dimensions can help human designers in their creative design process, by providing them with samples of solutions that are not slightly different from each other, but intensely and perceivably different.

**Direct geometric evaluations:**

The first set of proposed dimensions of diversity includes a number of direct geometric evaluations and is described in the following list. Of course this list can be updated or extended, based on experimental results and / or input from AEC partners.

**Plan Compactness:** A feature that expresses the degree to which the whole surface of a generated plan is “compact” or “dispersed”.

**Compactness per space unit:** The average of the compactness of each discrete space unit’s surface.

**Absolute Orthogonality:** The degree to which the boundary lines of the generated plan tend to be orthogonal, i.e. coinciding with either the X or Y axis of the coordinate system.

**Relative Orthogonality:** The degree to which the angles between consecutive boundary lines tend to be either square angles, or 180 degree angles (i.e. the two consecutive lines are co-linear).

**Symmetry and complexity:** The notions of symmetry and complexity are two broad aspects that have been extensively studied in the fields of evolutionary art and computational creativity [158] as modes of differentiation between generated artifacts. Applying such methods on generated designs is another way of abstractly diversifying solutions.

**Isovist and Visibility Graph - based evaluations:**

As the relevant literature suggests (section 4.1) through the analytical approaches of isovists and visibility graphs, a designed space can be tied to the human perception of environments. The work of Koutsolambros et al. [151] offers a condensed overview of a broad set of relevant methods for the analysis of architectural space. That same list is also supported by their open-source software, depthmapX [152], which makes such metrics even more accessible. We have selected two indicative metrics from this list, which we believe are good candidates as behavioral dimensions of the quality-diversity search algorithm. The first one (**Visual Control**) is one of the ways of associating visibility with control; the second (**Visual Entropy**) relates to the complexity of “travel” (or movement). More candidate metrics are worth exploring.

- **4.2.5 Conclusion**

This section presented a broad range of dimensions of diversity suitable for design exploration. Some of these dimensions are directly tied to specific objectives (such as the

ones proposed by engineering partners), while others are relatively independent and their purpose is mainly to increase diversity in a more abstract way which may be beneficial both for the algorithm's operation as well as for the creative/exploratory aspects of design. At this stage it is not possible to make strong statements about which dimensions will be the most effective or useful, because their effect has to be evaluated in their algorithmic context, as well as in the broader context of the PrismArch application.

### ○ 4.3 Software, Tools and Algorithms

Table 3: Software that could be used for evaluation of aesthetic dimensions in PrismArch

Name	License	Description	Possible use
Dlib: <a href="http://dlib.net/">http://dlib.net/</a>	Boost Software License (open source licensing for use in any application, free of charge)	Extensive C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. Includes image processing functions for feature extraction (e.g. SURF, HOG, etc.)	Possibly useful in case we need to apply machine learning techniques in our processes. For example, we could train neural networks to replace complex algorithmic processes so as to increase the computational efficiency of our system.
depthmapX <a href="https://github.com/SpaceGroupUCL/depthmapX">https://github.com/SpaceGroupUCL/depthmapX</a>	GPLv3 <a href="https://www.gnu.org/licenses/gpl-3.0.html">https://www.gnu.org/licenses/gpl-3.0.html</a>	Multi-platform spatial network analyses software	Performing spatial analysis on generated architectural plans, so as to evaluate them along a number of dimensions, in the context of quality-diversity search.
Dispersive Flies Optimization algorithm	open-source, free	A search algorithm which was proposed by al-Rifaie et al. [154] and then modified by al-Rifaie et al. [153] so as to detect arbitrary types of global or local symmetries.	Detecting arbitrary types of symmetry in images of generated architectural plans, in the context of quality-diversity search.
Thornton Tomasetti's Design Explorer <a href="http://core.thorntontomasetti.com/design-explorer/">http://core.thorntontomasetti.com/design-explorer/</a> <a href="https://tt-acm.github.io/DesignExplorer/">https://tt-acm.github.io/DesignExplorer/</a>	open source, MIT License	An open source tool for exploring design spaces on the web. Design Explorer is an interface that lets you visualize and filter groups of iterations – sets of design solutions that are both intimately related, and potentially scattered across a vast, high-dimensional possibility space.	This software can be used as a reference point in relation to back-end capabilities and user interface.
Project Refinery <a href="https://dynamobim.org/introducing-project-refinery/">https://dynamobim.org/introducing-project-refinery/</a> <a href="https://www.keanw.com/2018/11/project-refinery-implement-generative-design-workflows-for-aec.html">https://www.keanw.com/2018/11/project-refinery-implement-generative-design-workflows-for-aec.html</a>	closed source, commercial	Project Refinery lets you create design options through optimization. It operates as a plugin for Autodesk Revit Dynamo.	This software can be used as a reference point in relation to back-end capabilities and user interface.

## ▪ 5 REALIZING QUALITY-DIVERSITY AND DESIGNER MODELING IN PRISMARCH

As noted in Section 1, artificial intelligence in WP2 aims to provide an assistive technology driven by the user in order to edit items collectively and in an informed manner. As described in Prismarch DoW, the main dimensions for this assistive technology is envisioned through the algorithms of **Quality-Diversity Search** and **Designer Modeling**. The needs of quality diversity for a set of quality dimensions (as hard constraints and as “soft” objective functions) and diversity dimensions (as exploration incentives) have largely shaped the writeup of this deliverable, while Section 4 is also highly relevant as dimensions that can receive higher or lower importance in different designer models. This section provides a high-level overview of the principles of Quality-Diversity (QD) search and Designer Modeling in Section 5.1, and reports on the internal workshops undertaken for WP2 with AEC industry partners in Section 5.2 in order to highlight the grand vision of PrismArch’s contribution within QD and designer modeling.

### ○ 5.1 Algorithmic Background

#### ▪ 5.1.1 Quality Diversity Algorithms

Evolution Towards Quality-Diversity. Inspired by the extreme diversity of high-performing creatures found in nature, the quality diversity (QD) paradigm [6] aims to discover the largest possible set of diverse and high-quality solutions in one evolutionary run. Rather than employ genotypic diversity mechanisms [161], QD stresses the importance of searching for diverse solutions first and then maximize their quality. QD draws inspiration from the idea of rewarding divergence to find the necessary stepping stones [162] towards high performing areas of the search space. In divergent search, artificial evolution is not guided by a fitness tied to the ultimate objective of the problem, but instead rewards directly the diversity of solutions, based on notions such as novelty [163], surprise [164] or curiosity [165]. QD search combines divergent search with localized convergence by (a) partitioning the conceptual space into separate behavioral niches and/or (b) rewarding local competition between individuals within the same niche and/or (c) enforcing minimal constraints of quality for feasible solutions. In order to handle constraints, two-population approaches were previously used in constrained optimization [157] to divergent search, and have since been used for QD search as well [160]. In terms of local competition, its first implementation [166] rewarded a solution based on the number of behaviorally similar solutions it outperformed; local competition then acts as one objective with novelty [166] and/or surprise [167] acting as the other objective in a multi-objective fashion.

Finally, partitioning the search space has been introduced by the MAP-Elites algorithm [133], where the behavior space is discretized along  $d$  dimensions of behavior (features) and stored as a grid or a feature map of cells. MAP-Elites enforces local competition by allowing only the best individual in each cell to occupy it; however, some variants include multiple elites per cell [6]. The fitness can be either mathematically formulated [168], derived from simulations [133,160], or predicted from models trained on a corpus [169]. The behavior dimensions are usually defined by the user; however, they can also be discovered automatically through dimensionality reduction methods applied before or during evolution [170]. Features are usually linear but can also be categorical, e.g. every bin storing an image that best matches a



different class [171]. In terms of how the space is partitioned along these features, most MAP-Elites variants use uniformly distributed bins along two feature dimensions [133], although ad-hoc partitions based on Voronoi regions [172] and dynamic partitions that adapt based on the current populations' characteristics [170] have been explored. Using multiple feature maps has also been investigated [6]. Parent selection is also an important aspect that can affect the performance of the algorithm, and different variants have been explored to promote exploration of the feature space [173,132].

QD search algorithms, and MAP-Elites specifically, have primarily been tested in deceptive problems with applications to robotics. Some of the early applications include maze navigation testbeds [6,167] (where the proximity to the goal may not actually lead to good strategies) and robot ambulation [133]. However, a number of real-world applications have also been addressed through QD search. Among the most promising ones, Surrogate-Assisted Illumination (SAIL) [169] leveraged machine-learned models that could predict the performance results of lengthy physics simulations to inform the quality dimensions of a MAP-Elites QD Search method. The surrogate-assisted MAP-Elites implementation was tested in 2D airfoil design [169], using two of the parameters of the design itself as dimensions for exploration. Moreover, QD algorithms have been applied for game content generation in the form of game levels [132], enemy behavior and abilities [160] and much more.

### ▪ 5.1.2 Designer Modeling

Designer modeling was introduced by Liapis et al. [2] as a method for capturing a designer's intentions and preference and accommodating them via a computer-aided design tool. Designer modeling is therefore envisioned as an instance of user modeling applied to computer-aided design. The term can incorporate any computational model which recognizes the goals, preferences or process of a human designer. Such a designer model can be useful for personalized, responsive computer-aided design tools in the game industry or elsewhere. Initially introduced for assistive design in game development tasks, designer models could be considered similar to player modeling [175]; however, the designer model should be considered distinct as it needs to incorporate the designer's intentions to satisfy the user, and can be seen as second-order user modeling.

A successful designer model should recognize the preferences, process and goal of a designer interacting with the tool, although accomplishing each of these aspects may hinge on different algorithmic processes. Modeling the preference or overall style of a designer falls under the category of preference learning [176], and requires extensive information on a designer's choices, rankings or ratings among alternatives. Such adaptive models of taste have been trained based on a user's choice of one artifact over others [130] or from a user's rankings of artifacts in order of preference [177]. On the other hand, modeling the goals or intentions of a designer may be achievable via goal recognition [178], e.g. by suggesting possible next steps via a probabilistic model of cognitive associations based on past interactions [179]. However, such a method will likely only present previously seen (or created) concepts and thus stifle the designer's creativity. Finally, modeling the designer's process can be seen as a short-term plan recognition problem [180]. Although past interaction data can inform the model of trends in the design process, a process can be highly situational: a designer may focus on fine-tuning different properties of their creations at different stages of the process, without necessarily looking at the bigger picture until the very end. A model

of process could therefore be more accurate if it learned solely from the designer's current actions rather than from a large pool of past interactions.

While user modeling [181] and player modeling [175] have a plethora of applications, designer modeling remains a nascent field of inquiry. Sentient Sketchbook [17] explored different computational models (and modelling algorithms) in order to capture and adapt the generated suggestions based on a designer's overall style (storing persistent weights of features in a database), their current process (measuring the differences between their current canvas and their previous iteration), and their goals in terms of symmetry (hard-coding the representation in case the user's current creation was mostly symmetrical)) [158]. The directions explored in this research showcase how different levels of preferences (general, temporary, or session-specific) can inform the assistive AI through different data structures and algorithms (persistent databases or changes to representation). Alvarez and Font [182] introduce a designer preference model which takes advantage of the MAP-Elites visualization of a two-dimensional feature map. The solution selected by a human designer in the feature map is given absolute preference, and a decaying preference is given to neighboring solutions to the selected one (based on the two-dimensional distance on the grid of shown suggestions, as specified by the exploration dimensions of MAP-Elites). These preferences are used to train a small neural network that uses the entire map layout as input, and learns to predict the designer preference based on the shown (selected and unselected) suggestions. Using the entire level layout differs from the approach of Sentient Sketchbook [158], which used the precomputed fitness functions and adapted their weights directly. In a follow-up study, Alvarez et al. [174] used an existing dataset of designer sessions, and clustered each interim canvas into categories such as "initial room shapes", "complex wall mazes" or "dense" layouts. They then followed each session's trace through the clusters, in order to identify how designers explore the space and move from one design state to the other. While there is no explicit machine learning or adaptation of the generated suggestions in this work, it is an interesting way of processing large datasets of user logs and can be a foundation for work in PrismArch if the volume of data collected during user studies is sufficiently large.

## ○ 5.2 Vision arising from Workshops with AEC industry partners

As a follow-up to the workshops conducted for the preparation of the deliverable (see Section 1.3), all AEC partners were asked to envision the use of Quality Diversity in the context of PrismArch (based on the example applications that were presented during the workshops) and to provide relevant ideas and examples.

The following sections present the summarized vision from all three AEC partners (ZHVR, AKT II, and SWECO) and a preliminary reflection of the envisioned characteristics and features. The full responses are included in this deliverable under Appendix A.

### ▪ 5.2.1 Envisioning QD in PrismArch - ZHVR

The following table summarizes the response of ZHVR to the 7th question of the questionnaire, on the envisioned use of QD in the context of PrismArch.

Summarized response of ZHVR to question 7:
--

1. Architects are purposeful curators of information. The ideal focus of QD would be in reducing and focusing information for the numerous workflows and scenarios illustrated in D6.1.
2. The ability to select, group, and sphere items, introducing a timestamp and other associated data, should be made as intuitive as possible.
3. As far as the UI/UX for the datasphere is concerned, the following are of importance:
  - a. Customization of personal space (avoid information overload)
  - b. Locating optimal visual representation (2D, 3D, multiple media)
  - c. Help with work basics
4. Key parameters for UI/UX:
  - a. Level of staff, team organization
    - i. Information optimization / curation (idiosyncratic disciplinary preferences)
    - ii. Information flows in teams and collectively

Regarding the first suggestion, QD is not only an optimization algorithm but at the same time a very powerful tool for the “illumination” of certain properties of a (design) search space. This aspect of QD effectively turns it into a powerful tool for the analysis and better understanding of a design problem, as well as a medium through which the designers may analyze their prior assumptions, or negotiate their different perspectives. Moreover, QD can be used (and has been used) in ways that bring it very close to the ideal utilization that ZHVR describe, where the automated, evolutionary search process is intertwined with the broader process of (architectural) design in which many different processes take place. At a high level, two exemplary approaches towards this direction include interactive evolution [50] and user modeling [181]. Both of these approaches blend human agency (knowledge, skill, experience) with the brute force of the search algorithm, with the potential of generating an outcome that is superior to either. Apart from those examples, PrismArch offers an opportunity to explore novel ways of human-AI interaction, especially suited to the special needs of the problem at hand and the algorithmic paradigms of QD, user modeling and adaptation. Such innovative ways of human-AI interaction will have to be approached gradually, on top of a functioning prototype and with the help of a proper, and specifically designed, user interface.

Regarding the second suggestion, there has always been an expectation that users’ interaction with the QD assistive system should be tracked and time-stamped (see Section 2.2.5), as well as presented in an intuitive way. In order to support the generation of designer models, data collection of the user’s interaction with the system is necessary. This data will encode various aspects of their activity, including the selection between a set of available solutions, the direct modification of an existing solution, or the generation of a design from the ground up. Time-stamping these interactions is necessary in the context of designer modeling and - as ZHVR argues - useful in the broader context of the PrismArch application.

The third and fourth suggestions largely concern the interface between the user and the broader PrismArch system. These interface requirements do not directly affect the QD assistive system but will affect how users interact with, understand, and collaborate with each other and with the AI system. For example the ability to customize one's personal space is by definition a matter of personal preference within the capabilities of modification that the UI permits. The same can be said about the selection between various modes of visual representation of a design (such as switching between 2D and 3D modes, or presenting various relevant layers of information).

▪ **5.2.2 Envisioning QD in PrismArch - AKT II:**

The vision of AKT II, in regard to the application of QD in the context of PrismArch are captured in the following statements, which are based on the questionnaire that can be found in section A.3.

Response of AKT II to question 7:

**1. In response to the provided example: “The system keeps track of the user’s interaction and learns to make suggestions that are more suitable to the user’s preferences”:**

Some problems are too specific to generalize and applying machine learning (designer modeling) to them may be problematic. For these kinds of problems, perhaps it could be best to allow the user to follow their experience and knowledge in order to generate viable solutions. Designer-generated solutions may be then used in the context of training the system.

**2. In response to the provided example: “The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process”:**

Even applying the ideas in (1) above, it seems likely that - even just in the first few iterations - the system will still not be generating ideal/ viable options, and thus giving the user the ability to at any stage step into the process and redirect the ongoing evolution seems very sensible.

**3. Exploration of non-geometric design spaces:**

Some of the most significant challenges posed in projects are related to the logistics of cross-disciplinary collaboration: sharing and maintenance of communications - sketches, site photos, reports - as well as the expected digital models and drawings. Any Tools that can highlight when these resources are not being used optimally would be hugely beneficial across the lifespan of a project.  
Some examples:

- a. Offer suggestions to tag and/ or attach documents/ emails/ sketches to an item that the system has flagged as related.
- b. Tag prior iterations of a design artifact to the current version.

**4. Model Synchronisation and Diffing:**

As a more general topic: Something that emerges naturally from the multi-disciplinary design process is the complexity of ensuring that the (necessarily) abstracted structural analysis model is an accurate reflection of the architectural model. Problems arise if one of these models is changed, and that does not carry through to the other. There might

therefore be value in producing tools to highlight when elements have changed substantially enough in the architectural model to potentially need adjustment in the structural version too.

Regarding the first suggestion, it is indeed likely that modelling a designer's behavior or experience for some very special projects, or even across a domain of very different projects, will require a large amount of data and may, perhaps, be impractical. The generality or data requirements of such designer models will have to be evaluated in experiments throughout WP2 (and reported in D2.3). On the other point, the ability of the designer to intervene in the evolutionary process, either in the form of selecting between generated solutions [50] or in the form of directly modifying aspects of a specific solution are more than welcome features that will even be necessary for the collection of interaction data and the training of the aforementioned models. To conclude, the modes of interaction with the QD assistive AI system are relatively unbounded as far as the back-end AI functionality is concerned and only restricted by the available resources for the design and development of the necessary user interface.

Regarding the second suggestion, it is true that the first iterations of an evolutionary approach may not be generating ideal/viable options, but it is also not necessary for a user to constantly interact with the system. Before "stepping in" and manually guiding the design process, the user could first of all let the system operate for more iterations and simply observe the solutions becoming better/more viable. Should the system fail to deliver the desired outcome, there are many options for the designer to modify e.g. parameters of the algorithm itself or the dimensions of diversity. If these adjustments still do not produce good results, the user could include a manually generated solution (or a set of solutions) as a good "seed" for the re-initialization of the evolutionary process.

The third suggestion warns that content that relates to the design process (including plans, sketches, 3D models, renderings, pictures, as well as communication assets, like emails or other forms of written communication) are often "lost" during cross-disciplinary collaboration and indeed, may also be the case with human-AI collaboration. Allowing the AI to maintain the provenance of the artefacts (previous versions, e.g. in the form of a genotypic lineage) or meta-data attached by a user such as e-mails and sketches is both manageable and desirable for a human-AI collaboration. On the other hand, developing an AI system that can predict whether two source materials are related based on their raw data has different requirements and requires vast amounts of data to train on. A simple system that tracks past relationships and maintains them in future iterations is a more attainable and less data-intensive solution instead.

The fourth suggestion raises the issue of changes that one discipline applies on the design generating incompatibilities with the design that is being developed by another discipline. Admittedly, this can be exacerbated when the QD assistive AI can be considered another "agency" involved in the design process. It should be noted, however, that human designers will definitely retain the high-level decision making ability and responsibility. The AI solution proposed, moreover, unburdens human designers both from having to detect such problems of incompatibility as well as from having to solve them. As long as the designers are operating within the context of the QD assistive algorithm, it will be not only feasible for the system to detect such incompatibilities, but also necessary. After all, the constraints that classify a

design solution as “feasible” include the non-existence of such incompatibilities. Having said that, it will be very useful to highlight some categories of such incompatibilities, as well as methods for addressing them, during the development of the QD - prototype.

### ▪ 5.2.3 Envisioning QD in PrismArch - Sweco:

As a response to Question 7, Sweco partners described a number of features that capture their vision for the utilization of the Quality Diversity assistive AI system, as well as the broader context of the PrismArch application.

Response from Sweco to Question 7:

1. The system can propose design solutions after given a set of parameters and restrictions.
2. The system can identify potential coordination errors and highlight to the user.
3. The system keeps track of interactions, learns from them and adapts the experience to better suit the user’s needs. This may contain predefined scenarios for the user to choose while logging in (e.g. coordination scenario, design scenario etc).
4. The system is used to provide effective UI to the logged in users.
5. The system can perform smart search based on the given keywords and present relevant results which will be driven from the user interactions.

Some of their envisioned features are especially relevant to the specific operation of the QD assistive AI system: they suggest that the system should be able to propose design solutions, given a set of parameters and restrictions, which is aligned with the general purpose of the assistive AI envisioned in WP2. Once the user defines the problem specification (see problem representation, at section 2.2.1), the system will automatically generate a set of solutions and organize them along the dimensions of diversity that the user has selected. Second, they propose that the system should be able to identify potential coordination errors and highlight them to the user. Indeed, as long as the relevant (implicit, explicit or interdisciplinary) constraints have been properly encoded in the system, it should be able to detect whether they are “satisfied”, as well as display the relevant information to the user. The distinction between feasible solutions (that satisfy all the necessary constraints) and infeasible ones is a key part of the QD algorithm’s application on design problems in general, as well as in the specific context of PrismArch. Third, they propose that the system should keep track of user interactions, learn from them and adapt the experience to better suit the user’s needs. This may contain predefined scenarios for the user to choose while logging in (e.g. coordination scenario, design scenario etc). This is very well aligned with the goals of designer modeling, where the traces of a designer’s interaction with the tool can be used to train models that can predict the designer’s behavior, experience or preference.

The remaining characteristics envisioned by Sweco relate to the broader UI and front-end behavior of the PrismArch application, in general. This includes the potential of a smart search based on the given keywords that can present relevant results driven from the user

interactions. Although such a functionality can be useful, it can easily be supported by a simple database querying system and no AI is necessary for such tasks.

#### ▪ 5.2.4 ZHVR: Context of design parameters

The following table summarizes the response of ZHVR to questions 1 and 2, which relate to the definition of design parameters that are useful to explore, especially during the initial stages of the design process:

Summarized response of ZHVR to questions 1 and 2 (Design parameters)
<ol style="list-style-type: none"> <li>1. A design/solution is an emergent property that arises from the interactions between the designers and the client or between partners.</li> <li>2. The boundaries that shape the solution space are negotiated/discovered throughout the design process.</li> </ol>

Regarding the first takeaway, it is clear that the system should allow the user to modify the “problem specification”, during the evolutionary search process. The problem specification includes a set of aims and constraints that are specific to the design problem at hand. Those aims and constraints should be negotiable and editable. One of the ways to evaluate a given problem specification would be to examine the solution space that arises from it. In other words, a given sample of solutions generated by the QD assistive system could be one of the aspects of negotiating and reshaping the problem specification itself. This way, the design problem, as well as its potential solutions can be updated based on the designer’s interactions with the client, or with other partners.

Similarly, the second takeaway is very well aligned with QD search processes as the user can select new dimensions of diversity (differentiation) between solutions, even during the course of the evolutionary search process. The design dimensions are an important part of both the subjective criteria and the domain knowledge that accompanies a solution space. Based on this feedback, it is important to retain a flexibility in the potentially included set of such dimensions.

#### ▪ 5.2.5 ZHVR: Context of constraints:

The following table summarizes the response of ZHVR to questions 3 and 4, which relate to the distinction of cross-discipline constraints:

Summarized response of ZHVR to questions 3 and 4 (Constraints)
<ol style="list-style-type: none"> <li>1. Efficiency and responsiveness of the design process (ability to turn around a design revision in a short time)</li> <li>2. Good coordination with other disciplines, so as to highlight risks and propose solutions.</li> </ol>

Regarding the first takeaway, a QD evolutionary approach tackles this issue at its core. One of the main aspects of QD algorithms is to generate a diverse set of solutions.

The second takeaway can be taken into account so that it also becomes central to the system's operation. More specifically, the QD-system could be utilized in such a way that it provides a "common ground" between different design disciplines, effectively enabling a dynamic interaction between them. This can be achieved by embedding design parameters and constraints from all disciplines in the same system and selectively exposing some of them to the users, depending on their "point of view", but also allowing them to letting the users select the ones that they are most interested in, but without completely excluding the rest from the search-space.

▪ **5.2.6 ZHVR - context of optimization:**

The following table summarizes the response of ZHVR to questions 5 and 6, which relate to the aspects of optimization in the context of (architectural) design.

Summarized response of ZHVR to questions 5 and 6 (Optimization)
<ol style="list-style-type: none"> <li>1. Mission statement (with respect to social and environmental impacts)</li> <li>2. Client satisfaction (usability of space, keeping cost within budget, design intent, etc.)</li> <li>3. Designer work-force must be effectively used: saving time and effort is key.</li> <li>4. Tracking and understanding information that flows throughout the design process.</li> </ol>

Regarding the mission statement, the aspects that belong to the sphere of community and society (at a high level) are not directly quantifiable and therefore should better be managed by the expertise and sensibilities of human designers. Certain relevant aspects, however, such as sustainability metrics, can be addressed as problems of design optimization in the context of quality-diversity, as is elaborated in sections 3.2.3 and 4.2.

Regarding the second statement, client satisfaction also includes many aspects where the designer's sensibilities, judgement and communication skills cannot be bypassed. However, this aspect still includes a number of relevant features that can be treated as problems of QD exploration and assistance. For example, optimizing the spatial arrangement of a layout or the construction cost in relation to the quantity of materials, are examples of the many possible relevant aspects.

Regarding the third statement, quality diversity can be used to assist the designer in exploring the design space more efficiently. It can reveal the distribution of solutions along a number of selected dimensions of diversity, while also taking into account implicit, explicit and interdisciplinary constraints. All of these characteristics make it especially relevant to the notion of productivity.

Regarding the fourth statement, in its original version it includes many aspects that relate to the user interface and user experience in the broader PrismArch application. As far as the quality diversity and designer modeling system is concerned, tracking and understanding the



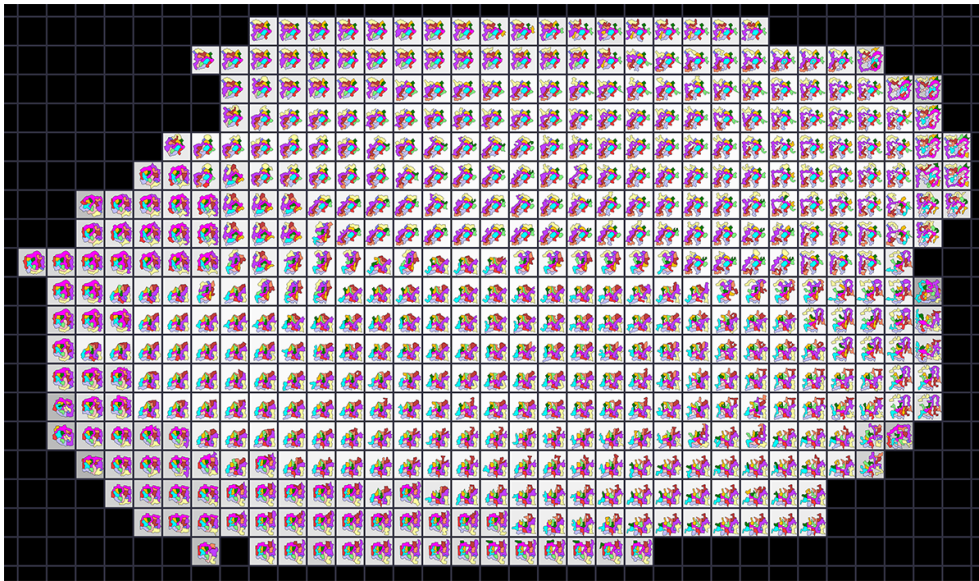
interaction between the system and the user is necessary and crucial. As explained in section 5.1.2, designer modeling depends on the collection of such data and can utilize them so as to recognize the preferences, process and goal of a designer.



## ▪ 6 CONCLUSIONS AND FUTURE STEPS

This Deliverable has laid out the principles and core concepts that underlie the design of the PrismArch algorithms. The fact that the goal of AI integration in PrismArch is not to replace human designers but rather to support and assist them requires a different approach from many of the fully automated optimization work surveyed in this Deliverable. Much of the work surveyed focuses on specific domains and artefacts of an AEC process, overlooking the complexities of AEC collaboration and the constraints that can emerge when working with different disciplines. To address the shortcomings and limits of the current literature, a number of seminars, masterclasses and workshops were conducted with AEC partners of PrismArch and a questionnaire was circulated. The responses to the questionnaire have provided vital data points that have assisted in expanding the deliverable to better capture the intricacies of actual practitioners' priorities and to shape a grand vision for the integration of assistive tools in a multidisciplinary collaboration setting as the one realized in PrismArch.

The fact that the QD assistive system generates alternatives to a human user's design necessitates that the representation chosen can allow for some control on the part of the designer. Moreover, the envisioned system should not provide entirely new designs that are likely to be too distant to the user's current frame of thinking/designing [156], as these suggestions will likely be rejected by the human user. Partners' input also highlighted the need for a human user to adjust, control, and override the AI initiative, which requires a representation that is easy to manually edit. This necessitates a representation that is both controllable and also expressive, and can capture –visually or functionally– many different possible designs. On the other hand, the fact that the AI will have to operate on many different phases of the design process and with many different end-users (e.g. architects, engineers) means that the representation should follow some *iterative refining* [56] and *hierarchical structure*. Specifically, during early conceptualization sketches the low-level details (such as the presence of windows, sustainability or visibility) can be omitted and high-level suggestions that focus on the building's massing can be provided. During later stages when the high-level concepts are finalized the representation can include details (and evaluate content on more grounded functional constraints) while not modifying the greater picture such as rooms' dimensions or connectivity. Importantly, the parametric space, functional constraints and design dimensions highlighted are designed for their integration into quality-diversity algorithms such as MAP-Elites [133] in order to best visualize the suggestions to human designers (see Figure 26 for an early example visualization). The visualizations that MAP-Elites naturally offer as part of the search process can be shown to a designer, even while the algorithm is still running (in theory). Upcoming work in WP2 (under Task 2.2./2.3 and D2.2/D2.3) will explore how these algorithms, parametric definitions, constraints, and preferences can be integrated with the VR tool developed and how best to take advantage of AI in the service of human design.



*Figure 26: A feature map evolved via MAP-Elites to minimize distance from a designer's ideal room areas, while also mapping out the conceptual space along two diversity dimensions.*

It should be noted that D2.1 surveyed a broad range of academic and commercial applications of optimization, machine learning, evolutionary computation, and parametric modeling. Some important related domains were also included as pertinent, specifically content generation in computer games (as there is extensive work on evolutionary level design in this field) and evolutionary art (as the exploration dimensions explored there could be repurposed for architecture). While thorough, the literature review is not conclusive but provides a broad range of examples for how spatial design problems have been represented and evaluated. These examples, and many other papers reviewed in preparation of this deliverable, inform some of our choices for the directions that PrismArch problem spaces (and solution spaces) should follow. Additional seminars and questionnaires from AEC partners provided vital insights that have largely been left unexplored in the academic literature. The problem of AI suggestions in PrismArch will follow the same iterative process of architectural problem definitions pointed out by Lawson [12]. We will have to iterate on these current directions highlighted in D2.1 and revise them based on multiple iterations of analysis, independent experiments, integration with the tool, consultations with AEC partners and expert designers, and reflection. D2.1 will provide a roadmap for this iterative process, as it lists the requirements and priorities of PrismArch and collates best practices, resources and software that can facilitate the realization of a context-aware QD co-creator of human designers operating in a VR space.



## ▪ 7 REFERENCES

- [1] K. A. De Jong, *Evolutionary computation: a unified approach*. MIT Press, 2006.
- [2] A. Liapis, G. N. Yannakakis, and J. Togelius, "Designer modeling for personalized game content creation tools," in *Proceedings of the AIIDE Workshop on Artificial Intelligence & Game Aesthetics*, 2013.
- [3] K. O. Stanley and R. Miikkulainen, "A taxonomy for artificial embryogeny," *Artificial Life*, vol. 9, no. 2, pp. 93–130, 2003.
- [4] Z. Michalewicz, D. Dasgupta, R. L. Riche, and M. Schoenauer, "Evolutionary algorithms for constrained engineering problems," *Computers & Industrial Engineering*, vol. 30, pp. 851–870, 1996.
- [5] R. Arnheim, *Art and visual perception: a psychology of the creative eye*. University of California Press, revised and expanded ed., 2004.
- [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, p. 40, 2016.
- [7] A. Cully and Y. Demiris, "Quality and diversity optimization: A unifying modular framework," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 245–259, 2018.
- [8] G. Polya, *How to solve it: a new aspect of mathematical method*. Princeton University Press, 2004.
- [9] A. Newell, J. Shaw, and H. Simon, *Elements of a theory of human problem solving*. Rand Corporation, 1957.
- [10] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, 2011.
- [11] R. Hudson, *Strategies for Parametric Design in Architecture: An application of practice led research*. PhD thesis, University of Bath, 2010.
- [12] B. Lawson, *How designers think: The design process demystified*. Architectural Press, 2006.
- [13] B. Hillier, J. Musgrove, and P. O'Sullivan, "Knowledge and design," in *Environmental design: Research and practice* (W. Mitchell, ed.), University of California, 1972.
- [14] E. Motta and Z. Zdrahal, "Parametric design problem solving," in *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based System Workshop*, 1996.
- [15] C. A. Baykan and M. S. Fox, "Constraint satisfaction techniques for spatial planning," in *Intelligent CAD Systems III* (P. J. W. ten Hagen and P. J. Veerkamp, eds.), (Berlin, Heidelberg), pp. 187–204, Springer Berlin Heidelberg, 1991.
- [16] R. Sharpe, B. Marksjo, J. Mitchell, and J. Crawford, "An interactive model for the layout of buildings," *Applied mathematical modelling*, vol. 9, no. 3, pp. 207–214, 1985.

- [17] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient sketchbook: Computer-aided game level authoring," in Proceedings of the 8th Conference on the Foundations of Digital Games, pp. 213–220, 2013.
- [18] A. Alvarez, S. Dahlskog, J. Font, J. Holmberg, C. Nolasco, and A. Osterman, "Fostering creativity in the mixed-initiative evolutionary dungeon designer," in Proceedings of the International Conference on the Foundations of Digital Games, 2018.
- [19] D. Ashlock, C. Lee, and C. McGuinness, "Search-based procedural generation of maze-like levels," Computational Intelligence and AI in Games, IEEE Transactions on, vol. 3, pp. 260 – 273, 10, 2011.
- [20] P. Lopes, A. Liapis, and G. N. Yannakakis, "Targeting horror via level and soundscape generation," in Proceedings of the AAAI Artificial Intelligence for Interactive Digital Entertainment Conference, 2015.
- [21] D. Karavolos, A. Liapis, and G. N. Yannakakis, "A multi-faceted surrogate model for search-based procedural content generation," IEEE Transactions on Games, vol. 13, no. 1, pp. 11–22, 2021.
- [22] B. Hillier and J. Hanson, The Social Logic of Space. Cambridge University Press, 1984.
- [23] H.-J. Bandelt and V. Chepoi, "Metric graph theory and geometry: a survey," Contemporary Mathematics, vol. 453, p. 49–86, 2008.
- [24] B. Medjdoub and B. Yannou, "Separating topology and geometry in space planning," Computer-Aided Design, vol. 32, pp. 39–61, 2000.
- [25] D. Karavolos, A. Liapis, and G. N. Yannakakis, "Evolving missions to create game spaces," in Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG), 2016.
- [26] R. Linden, R. Lopes, and R. Bidarra, "Designing procedurally generated levels," in Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2013.
- [27] Y.-C. Hsu and R. J. Krawczyk, "Space adjacency behavior in space planning," in Proceedings of the CAADRIA 2004 Conference, 2004.
- [28] S. Arvin and D. House, "Modeling architectural design objectives in physically based space planning," Automation in Construction, vol. 11, pp. 213–225, 2002.
- [29] J. C. Damski and J. S. Gero, "An evolutionary approach to generating constraint-based space layout topologies," in Proceedings of CAAD futures 1997, pp. 855–864, Springer, 1997.
- [30] R. Koenig and K. Knecht, "Comparing two evolutionary algorithm based methods for layout generation: Dense packing versus subdivision," AI EDAM, vol. 28, no. 3, pp. 285–299, 2014.
- [31] J. P. Duarte, "A discursive grammar for customizing mass housing: the case of siza's houses at malagueira," Automation in construction, vol. 14, no. 2, pp. 265–275, 2005.
- [32] A. Doulgerakis, "Genetic and embryology in layout planning," Master's thesis, University of London, 2007.
- [33] K. Shekhawat, "Algorithm for constructing an optimally connected rectangular floor plan," Frontiers of Architectural Research, vol. 3, no. 3, pp. 324–330, 2014.

- [34] K. Keatruangkamala and K. Sinapiromsaran, "Optimizing architectural layout design via mixed integer programming," in *Computer Aided Architectural Design Futures 2005*, pp. 175–184, Springer, 2005.
- [35] J. Michalek, R. Choudhary, and P. Papalambros, "Architectural layout design optimization," *Engineering optimization*, vol. 34, no. 5, pp. 461–484, 2002.
- [36] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Proceedings of the International Conference on Neural Information Processing Systems*, 2014.
- [37] W. Huang and H. Zheng, "Architectural drawings recognition and generation through machine learning," in *Proceedings of ACADIA*, 2018.
- [38] S. Chaillou, "Ai + architecture: Towards a new approach," Master's thesis, Harvard GSD, 2019.
- [39] S. Chaillou, "Space layouts & GANs: GAN-enabled floor plan generation." <https://medium.com/spacemaker-research-blog/space-layouts-gans-2329c8f85fe8>, 2020. Accessed 27 April 2021.
- [40] R. Hu, Z. Huang, Y. Tang, O. V. Kaick, H. Zhang, and H. Huang, "Graph2Plan: Learning floorplan generation from layout graphs," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2020)*, vol. 39, no. 4, pp. 118:1–118:14, 2020.
- [41] W. Langdon, R. Poli, N. Mcphee, and J. Koza, "Genetic programming: An introduction and tutorial, with a survey of techniques and applications," *Studies in Computational Intelligence*, vol. 115, pp. 927–1028, 2008.
- [42] L. Trujillo and G. Olague, "Using evolution to learn how to perform interest point detection," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR 2006)*, pp. 211–214, 2006.
- [43] A. Khan, A. S. Qureshi, N. Wahab, M. Hussain, and M. Y. Hamza, "A recent survey on the applications of genetic programming in image processing," 2020.
- [44] Y. Azaria and M. Sipper, "Gp-gammon: Using genetic programming to evolve backgammon players," in *Proceedings of the EuroGP Conference (M. Keijzer, A. Tettamanzi, P. Collet, J. van Hemert, and M. Tomassini, eds.)*, pp. 132–142, 2005.
- [45] Y. Shichel, E. Ziserman, and M. Sipper, "Gp-robocode: Using genetic programming to evolve robocode players," in *Proceedings of the EuroGP Conference*, 2005.
- [46] L. D. Lohn J, Hornby G, "Evolutionary antenna design for a NASA spacecraft," in *Genetic Programming Theory and Practice II*, p. 301–315, O'Reilly, 2004.
- [47] J. Koza, S. Al-Sakran, and L. Jones, "Automated re-invention of six patented optical lens systems using genetic programming," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1953–1960, 2005.
- [48] A. O. Asojo, "Exploring algorithms as form determinants in design," in *Proceedings 3rd International Space Syntax Symposium*, 2001.
- [49] P. Coates and D. Makris, "Genetic programming and spatial morphogenesis," in *Proceedings of the Symposium on Creative Evolutionary Systems*, 1999.



- [50] H. Takagi, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation," *Proceedings of the IEEE*, vol. 89, no. 9, pp. 1275–1296, 2001.
- [51] R. Jagielski and J. S. Gero, "A genetic programming approach to the space layout planning problem," in *CAAD futures 1997*, pp. 875–884, Springer, 1997.
- [52] J. Secretan, N. Beato, D. B. D'Ambrosio, A. Rodriguez, A. Campbell, J. T. Folsom-Kovarik, and K. O. Stanley, "Picbreeder: A case study in collaborative evolutionary exploration of design space," *Evolutionary Computation*, vol. 19, no. 3, pp. 373–403, 2011.
- [53] A. Hoover, P. Szerlip, and K. Stanley, "Functional scaffolding for composing additional musical voices," *Computer Music Journal*, vol. 38, pp. 80–99, 12 2014.
- [54] J. Secretan, N. Beato, D. B. D'Ambrosio, A. Rodriguez, A. Campbell, and K. O. Stanley, "Picbreeder: Evolving pictures collaboratively online," in *Proceedings of the Computer Human Interaction Conference*, 2008.
- [55] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, p. 99–127, 2002.
- [56] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient world: Human-based procedural cartography," in *Proceedings of Evolutionary and Biologically Inspired Music, Sound, Art and Design (EvoMusArt)*, vol. 7834, LNCS, pp. 180–191, Springer, 2013.
- [57] S. Risi, J. Lehman, D. B. D'Ambrosio, R. Hall, and K. O. Stanley, "Petalz: Search-based procedural content generation for the casual gamer," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 8, no. 3, pp. 244–255, 2016.
- [58] J. Clune and H. Lipson, "Evolving 3d objects with a generative encoding inspired by developmental biology," *SIGEVolution*, vol. 5, p. 2–12, Nov. 2011.
- [59] D. Ashlock and C. McGuinness, "Landscape automata for search based procedural content generation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2013.
- [60] N. Shaker, A. Liapis, J. Togelius, R. Lopes, and R. Bidarra, "Constructive generation methods for dungeons and levels," in *Procedural Content Generation in Games: A Textbook and an Overview of Current Research* (N. Shaker, J. Togelius, and M. J. Nelson, eds.), pp. 31–55, Springer, 2016.
- [61] M. Schoenauer, "Shape representations and evolution schemes," in *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, 05 1997.
- [62] D. Ashlock and C. McGuinness, "Automatic generation of fantasy role-playing modules," in *Proceedings of the Computational Intelligence in Games Conference*, 2014.
- [63] W. Cachia, A. Liapis, and G. N. Yannakakis, "Multi-level evolution of shooter levels," in *Proceedings of the AAAI Artificial Intelligence for Interactive Digital Entertainment Conference*, 2015.
- [64] L. Wang, K. Chen, P. Janssen, and G. Ji, "Algorithmic generation of architectural massing models for building design optimisation - parametric modelling using subtractive and additive

form generation principles,” in Proceedings of the 25th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), 2020.

[65] F. Bao, D.-M. Yan, N. J. Mitra, and P. Wonka, “Generating and exploring good building layouts,” *ACM Transactions on Graphics*, vol. 32, no. 4, 2013.

[66] J. Lim, P. Janssen, and R. Stouffs, “Automated generation of BIM models from 2D CAD drawings,” in Proceedings of the 23rd International Conference of the Association for Computer-Aided Architectural Design Research in Asia, 2018.

[67] A. Liapis, “Multi-segment evolution of dungeon game levels,” in Proceedings of the Genetic and Evolutionary Computation Conference, 2017.

[68] OMA, “The interlace.” <https://oma.eu/projects/the-interlace>. Accessed 21 April 2021.

[69] P. Janssen and V. Kaushik, “Decision chain encoding: Evolutionary design optimization with complex constraints,” in Proceedings of the EvoMusArt Conference, 2013.

[70] J. Gero and V. A. Kazakov, “Evolving design genes in space layout planning problems,” *Artificial Intelligence in Engineering*, vol. 12, no. 3, pp. 163–176, 1998.

[71] G. Smith and J. Whitehead, “Analyzing the expressive range of a level generator,” in Proceedings of the FDG workshop on Procedural Content Generation, 2010.

[72] Y. Davidor, “Epistasis variance: A viewpoint on ga-hardness,” in *Foundations of genetic algorithms*, vol. 1, pp. 23–35, Elsevier, 1991.

[73] T. Galanos, A. Liapis, G. N. Yannakakis, and R. Koenig, “Arch-elites: Quality-diversity for urban design,” in Proceedings of the Genetic and Evolutionary Computation Conference, 2021.

[74] K. Chen, P. Janssen, and A. Schlueter, “Multi-objective optimisation of building form, envelope and cooling system for improved building energy performance,” *Automation in Construction*, vol. 94, pp. 449–457, 07 2018.

[75] T. S. Choo, P. Janssen, S. Roudavski, and B. Tuncer, “Maximise energy savings using evolutionary multi-objective optimisation,” in *Open Systems: Proceedings of the 18th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2013)*, p. 127–136, 2013.

[76] S. S. Wong and K. C. Chan, “Evoarch: An evolutionary algorithm for architectural layout design,” *Computer-Aided Design*, vol. 41, no. 9, pp. 649–667, 2009.

[77] E. Rodrigues, A. R. Gaspar, and A. Gomes, “An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, part 1: Methodology,” *Comput. Aided Des.*, vol. 45, pp. 887–897, 2013.

[78] P. Charman, “A constraint-based approach for the generation of floor plans,” *Proceedings Sixth International Conference on Tools with Artificial Intelligence. TAI 94*, pp. 555–561, 1994.

[79] L. B. Kovacs, “Knowledge based floor plan design by space partitioning: A logic programming approach,” *Artif. Intell. Eng.*, vol. 6, pp. 162–185, 1991.

[80] B. Medjdoub and B. Yannou, “Separating topology and geometry in space planning,” *Computer-Aided Design*, vol. 32, no. 1, pp. 39–61, 2000.

- [81] A. Bahrehmand, T. Batard, R. Marques, A. Evans, and J. Blat, "Optimizing layout using spatial quality metrics and user preferences," *Graphical Models*, vol. 93, pp. 25–38, 2017.
- [82] L. L. Beghini, A. Beghini, N. Katz, W. Baker, and G. Paulino, "Connecting architecture and engineering through structural topology optimization," *Engineering Structures*, vol. 59, pp. 716–726, 2014.
- [83] R. Kicinger, T. Arciszewski, and K. Jong, "Evolutionary computation and structural design: A survey of the state-of-the-art," *Computers & Structures*, vol. 83, pp. 1943–1978, 2005.
- [84] A. Hofler, U. Leysser, and J. Wiedeman, "Optimization of the layout of trusses combining strategies based on michell's theorem and on the biological principles of evolution," in *AGARD Second Symp. on Structural Optimization 8 p(SEE N 74-15596 06-32)*, 1973.
- [85] M. Lawo and G. Thierauf, "Optimal design for dynamic, stochastic loading-a solution by random search," *Optimization Methods in Structural Design*, p. 346, 1982.
- [86] G. Anagnostou, E. M. Rønquist, and A. Patera, "A computational procedure for part design," *Applied Mechanics and Engineering*, vol. 97, pp. 33–48, 1992.
- [87] C. Kane and M. Schoenauer, "Topological optimum design using genetic algorithms," *Control and Cybernetics*, vol. 25, 03 1997.
- [88] P. Hajela and E. Lee, "Genetic algorithms in truss topological optimization," *International Journal of Solids and Structures*, vol. 32, pp. 3341–3357, 11 1995.
- [89] C. D. Chapman, K. Saitou, and M. Jakiela, "Genetic algorithms as an approach to configuration and topology design," *Journal of Mechanical Design* 1994, 12 1994.
- [90] L. A. Schmit, "Structural synthesis - its genesis and development," *AIAA Journal*, vol. 19, pp. 1249–1263, 1981.
- [91] N. Khot and L. Berke, "Structural optimization using optimality criteria," 02 1984.
- [92] R. Kicinger, T. Arciszewski, and K. DeJong, "Evolutionary design of steel structures in tall buildings," *Journal of computing in civil engineering*, vol. 19, no. 3, pp. 223–238, 2005.
- [93] "Sagrada fam´ilia." [https://en.wikipedia.org/wiki/Sagrada\\_Familia](https://en.wikipedia.org/wiki/Sagrada_Familia). Accessed: 2021-27-04.
- [94] "Montreal biosphere." [https://en.wikipedia.org/wiki/Montreal\\_Biosphere](https://en.wikipedia.org/wiki/Montreal_Biosphere). Accessed: 2021-27-04.
- [95] "L'océanografic." <https://en.wikipedia.org/wiki/L%27Oceanografic>. Accessed: 2021-27-04.
- [96] L. L. Stromberg, A. Beghini, W. Baker, and G. Paulino, "Application of layout and topology optimization using pattern gradation for the conceptual design of buildings," *Structural and Multidisciplinary Optimization*, vol. 43, pp. 165–180, 2011.
- [97] P. Block, *Thrust Network Analysis: Exploring Three-dimensional Equilibrium*. PhD thesis, Cambridge, MA, USA, May 2009. PhD dissertation.
- [98] "Block research group (brg)." <https://www.block.arch.ethz.ch/brg/about>. Accessed: 2021-27-04.

- [99] M. Rippmann, L. Lachauer, and P. Block, "Rhinovault - interactive vault design," *International Journal of Space Structures*, vol. 27, pp. 219–230, December 2012.
- [100] N. Bouchlaghem and K. Letherman, "Numerical optimization applied to the thermal design of buildings," *Building and Environment*, vol. 25, pp. 117–124, 1990.
- [101] J. Gero, N. D'cruz, and A. Radford, "Energy in context: A multicriteria model for building design," *Building and Environment*, vol. 18, pp. 99–107, 1983.
- [102] R. Evins, "A review of computational optimisation methods applied to sustainable building design," *Renewable & Sustainable Energy Reviews*, vol. 22, pp. 230–245, 2013.
- [103] V. Zegarac Leskovic and M. Premrov, "An approach in architectural design of energy-efficient timber buildings with a focus on the optimal glazing size in the south-oriented facade," *Energy and Buildings*, vol. 43, no. 12, pp. 3410–3418, 2011.
- [104] D. Coley and S. Schukat, "Low-energy design: combining computer-based optimisation and human judgement," *Building and Environment*, vol. 37, pp. 1241–1247, 2002.
- [105] D. Tuhus-Dubrow and M. Krarti, "Genetic-algorithm based approach to optimize building envelope design for residential buildings," *Building and Environment*, vol. 45, no. 7, pp. 1574–1581, 2010.
- [106] M. Sahu, B. Bhattacharjee, and S. C. Kaushik, "Thermal design of air-conditioned building for tropical climate using admittance method and genetic algorithm," *Energy and Buildings*, vol. 53, pp. 1–6, 2012.
- [107] S. Bambrook, A. Sproul, and D. Jacob, "Design optimisation for a low energy home in Sydney," *Energy and Buildings*, vol. 43, no. 7, pp. 1702–1711, 2011.
- [108] J. Holst, "Using whole building simulation models and optimizing procedures to optimize building envelope design with respect to energy consumption and indoor environment.," in *Proceedings of the Eighth International IBPSA Conference*, 2003.
- [109] A. Marsh, "Computer-optimised shading design," in *Proceedings of the building simulation conference 2003*, 01 2003.
- [110] M. Turrin, P. V. Buelow, and R. Stouffs, "Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms," *Adv. Eng. Informatics*, vol. 25, pp. 656–675, 2011.
- [111] J. M. Gagne and M. Andersen, "A generative facade design method based on daylighting performance goals," *Journal of Building Performance Simulation*, vol. 5, pp. 141 – 154, 2012.
- [112] L. Caldas, "Generation of energy-efficient architecture solutions applying gene arch: An evolution-based generative design system," *Adv. Eng. Informatics*, vol. 22, pp. 59–70, 2008.
- [113] C.-S. Park, G. Augenbroe, and T. Messadi, "Daylighting optimization in smart facade systems," in *Proceedings of the Eighth International IBPSA Conference*, 01 2003.
- [114] R. Evins, P. Pointer, and R. Vaidyanathan, "Multi-objective optimisation of the configuration and control of a double-skin facade," 11 2011.
- [115] D. Saelens, B. Blocken, S. Roels, and H. Hens, "Optimization of the energy performance of multiple-skin facades," in *Proceedings of the Ninth International IBPSA Conference*, 2007.

- [116] M. Palonen, A. Hasan, and K. Siren, "A genetic algorithm for optimization of building envelope and HVAC system parameters," Proc. IBPSA'09, pp. 159–166, 2009.
- [117] K. F. Fong, V. I. Hanby, and T.-T. Chow, "HVAC system optimization for energy management by evolutionary programming," Energy and Buildings, vol. 38, no. 3, pp. 220–231, 2006.
- [118] W. Huang and H. Lam, "Using genetic algorithms to optimize controller parameters for HVAC systems," Energy and Buildings, vol. 26, no. 3, pp. 277–282, 1997.
- [119] R. Evins, P. Pointer, and R. Vaidyanathan, "Optimisation for chp and cchp decision-making," in Proceedings of the Building Simulation 2011 Conference, Sydney, Australia, pp. 1335–1342, 2011.
- [120] O. Shaneb, P. Taylor, and G. Coates, "Optimal online operation of residential  $\mu$ chp systems using linear programming," Energy and Buildings, vol. 44, pp. 17–25, 2012.
- [121] V. Kaushik and P. Janssen, "Multi-criteria evolutionary optimisation of building envelopes during conceptual stages of design," in Beyond Codes and Pixels: Proceedings of the 17th International Conference on Computer-Aided Architectural Design Research in Asia, p. 497–506, 01, 2012.
- [122] P. Janssen, "Dexen: A scalable and extensible platform for experimenting with population-based design exploration algorithms," Artificial Intelligence for Engineering Design, Analysis and Manufacturing : AI EDAM, vol. 29, pp. 443–455, 11 2015.
- [123] "Houdini." <https://www.sidefx.com/products/houdini/>. Accessed: 2021-27-04.
- [124] P. Janssen and V. Kaushik, "Iterative refinement through simulation: Exploring trade-offs between speed and accuracy," in Proceedings of the 30th eCAADe Conference, pp. 555–563, 2012.
- [125] "Galapagos evolutionary solver." <https://grasshopperdocs.com/addons/galapagos.html>. Accessed: 2021-27-04.
- [126] L. Wang, P. Janssen, K. Chen, Z. Tong, and G. Ji, "Subtractive building massing for performance-based architectural design exploration: A case study of daylighting optimization," Sustainability, vol. 11, p. 6965, 12 2019.
- [127] R. Arnheim, Art and visual perception: a psychology of the creative eye. University of California Press, revised and expanded ed. (2004) ed., 2004.
- [128] V. S. Ramachandran and W. Hirstein, "The science of art: a neurological theory of aesthetic experience," Journal of consciousness Studies, vol. 6, pp. 15–51, 1999.
- [129] G. Ochoa, "On genetic algorithms and lindenmayer systems," in Parallel Problem Solving from Nature — PPSN V (A. E. Eiben, T. Back, M. Schoenauer, and H.-P. Schwefel, eds.), (Berlin, Heidelberg), pp. 335–344, Springer Berlin Heidelberg, 1998.
- [130] A. Liapis, G. N. Yannakakis, and J. Togelius, "Adapting models of visual aesthetics for personalized content creation," IEEE Transactions on Computational Intelligence and AI in Games, vol. 4, no. 3, pp. 213–228, 2012.
- [131] A. Alvarez, S. Dahlskog, J. Font, J. Holmberg, and S. Johansson, "Assessing aesthetic criteria in the evolutionary dungeon designer," in Proceedings of the 13th International Conference on the Foundations of Digital Games, pp. 1–4, 2018.

- [132] K. Sfikas, A. Liapis, and G. N. Yannakakis, "Monte carlo elites: Quality-diversity selection as a multi-armed bandit problem," in Proceedings of the Genetic and Evolutionary Computation Conference, 2021.
- [133] J.-B. Mouret and J. Clune, "Illuminating search spaces by mapping elites," ArXiv, vol. abs/1504.04909, 2015.
- [134] A. Gartus and H. Leder, "Predicting perceived visual complexity of abstract patterns using computational measures: The influence of mirror symmetry on complexity perception," PLoS One, vol. 12, no. 11, 2017.
- [135] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671–680, 1983.
- [136] R. Hubner and M. G. Fillinger, "Comparison of objective measures for predicting perceptual balance and visual aesthetic preference," Frontiers in psychology, vol. 7, p. 335, 2016.
- [137] C. Redies, J. Hasenstein, and J. Denzler, "Fractal-like image statistics in visual art: similarity to natural scenes," Spatial vision, vol. 21, no. 1-2, pp. 137–148, 2008.
- [138] W. S. Geisler, J. S. Perry, B. Super, and D. Gallogly, "Edge co-occurrence in natural images predicts contour grouping performance," Vision research, vol. 41, no. 6, pp. 711–724, 2001.
- [139] A. Wilson and A. Chatterjee, "The assessment of preference for balance: Introducing a new test," Empirical Studies of the Arts, vol. 23, no. 2, pp. 165–180, 2005.
- [140] G. Mather, "Visual image statistics in the history of western art," Art & Perception, vol. 6, no. 2-3, pp. 97–115, 2018.
- [141] C. Redies, S. A. Amirshahi, M. Koch, and J. Denzler, "Phog-derived aesthetic measures applied to color photographs of artworks, natural scenes and objects," in European conference on computer vision, pp. 522–531, Springer, 2012.
- [142] C. Redies, A. Brachmann, and G. U. Hayn-Leichsenring, "Changes of statistical properties during the creation of graphic artworks," Art & Perception, vol. 3, no. 1, pp. 93–116, 2015.
- [143] E. Van Geert and J. Wagemans, "Order, complexity, and aesthetic appreciation," Psychology of Aesthetics, Creativity, and the Arts, vol. 14, no. 2, p. 135, 2020.
- [144] P. Machado, J. Romero, M. Nadal, A. Santos, J. Correia, and A. Carballal, "Computerized measures of visual complexity," Acta psychologica, vol. 160, pp. 43–57, 2015.
- [145] C. Alexander, The nature of order. Taylor & Francis, 2004.
- [146] C. Alexander, A pattern language: towns, buildings, construction. Oxford university press, 1977.
- [147] M. Benedikt, "To take hold of space: Isovists and isovist fields," Environment and Planning B: Planning and Design, vol. 6, pp. 47–65, 01 1979.
- [148] A. Turner, M. Doxa, D. O'Sullivan, and A. Penn, "From isovists to visibility graphs: A methodology for the analysis of architectural space," Environment and Planning B: Planning and Design, vol. 28, pp. 103 – 121, February 2001.

- [149] A. Stamps, "Isovists, enclosure, and permeability theory," *Environment and Planning B: Planning and Design*, vol. 32, pp. 735 – 762, 2005.
- [150] M. L. Benedikt and C. A. Burnham, "Perceiving architectural space: From optic arrays to isovists," *Persistence and change*, pp. 103–114, 1985.
- [151] P. Koutsolampros, K. Sailer, T. Varoudis, and R. Haslem, "Dissecting visibility graph analysis: The metrics and their role in understanding workplace human behaviour," in *Proceedings of the 12th Space Syntax Symposium*, 2019.
- [152] "depthmapx." <https://varoudis.github.io/depthmapX/>. Accessed: 2021-27-04.
- [153] M. M. al Rifaie, A. Ursyn, R. Zimmer, and M. J. Javid, "On symmetry, aesthetics and quantifying symmetrical complexity," in *EvoMUSART*, 2017.
- [154] M. M. al Rifaie, "Dispersive flies optimisation," *2014 Federated Conference on Computer Science and Information Systems*, pp. 529–538, 2014.
- [155] A. Turner, *Depthmap 4: a researcher's handbook*. Bartlett School of Graduate Studies, University College London, 2004.
- [156] A. Liapis, G. N. Yannakakis, C. Alexopoulos, and P. Lopes, "Can computers foster human users' creativity? Theory and praxis of mixed-initiative co-creativity," *Digital Culture & Education (DCE)*, vol. 8, no. 2, pp. 136–152, 2016.
- [157] 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.
- [158] A. Liapis, G. N. Yannakakis, and J. Togelius, "Designer modeling for Sentient Sketchbook," in *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, 2014.
- [159] 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.
- [160] A. Khalifa, S. Lee, A. Nealen, and J. Togelius, "Talakat: Bullet hell generation through con-strained MAP-Elites," in *Proceedings of The Genetic and Evolutionary Computation Conference*, pp. 1047–1054, ACM, 2018.
- [161] M. Preuss. *Multimodal optimization by means of evolutionary algorithms*. Springer, 2015.
- [162] E. Meyerson and R. Miikkulainen. Discovering evolutionary stepping stones through behavior domination. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 139–146, 2017.
- [163] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2), 2011.
- [164] D. Gravina, A. Liapis, and G. N. Yannakakis. Surprise search: Beyond objectives and novelty. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2016.

- [165] C. Stanton and J. Clune. Curiosity search: Producing generalists by encouraging individuals to continually explore and acquire skills throughout their lifetime. *PLoS ONE*, 11(9), 2016.
- [166] J. Lehman and K. O. Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218, 2011.
- [167] D. Gravina, A. Liapis, and G. N. Yannakakis. Quality diversity through surprise. *IEEE Transactions on Evolutionary Computation*, 23(4):603–616, 2019.
- [168] D. Whitley, S. Rana, J. Dzubera, and K. E. Mathias. Evaluating evolutionary algorithms. *Artificial Intelligence*, 85(1-2):245 – 276, 1996.
- [169] A. Gaier, A. Asteroth, and J.-B. Mouret. Data-efficient design exploration through surrogate-assisted illumination. *Evolutionary Computation*, 26:381–410, 2018.
- [170] A. Cully. Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 81–89, 2019.
- [171] A. Nguyen, J. Yosinski, and J. Clune. Understanding innovation engines: Automated creativity and improved stochastic optimization via deep learning. *Evolutionary Computation*, 24(3):545–572, 2016.
- [172] V. Vassiliades, K. Chatzilygeroudis, and J.-B. Mouret. Using centroidal Voronoi tessellations to scale up the multi-dimensional archive of phenotypic elites algorithm. *IEEE Transactions on Evolutionary Computation*, 2017.
- [173] Antoine Cully and Yannis Demiris. 2018. Quality and Diversity Optimization: A Unifying Modular Framework. *IEEE Transactions on Evolutionary Computation* 22, 2 (2018), 245–259.
- [174] A. Alvarez, J. Font, and J. Togelius. 2020. Towards Designer Modeling through Design Style Clustering. *ArXiv, abs/2004.01697*.
- [175] G. N. Yannakakis, P. Spronck, D. Loiacono, and E. Andre, “Player modeling,” *Dagstuhl Seminar on Game Artificial and Computational Intelligence*, 2013.
- [176] J. Fürnkranz and E. Hüllermeier, *Preference Learning*. Springer-Verlag New York, Inc., 2010.
- [177] A. Liapis, H. P. Martinez, J. Togelius, and G. N. Yannakakis, “Adaptive game level creation through rank-based interactive evolution,” in *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, 2013.
- [178] B. A. Goodman and D. J. Litman, “On the interaction between plan recognition and intelligent interfaces,” *User Modeling and User-Adapted Interaction*, vol. 2, pp. 83–115, 1992.
- [179] H.-C. Wang, “Modeling idea generation sequences using Hidden Markov Models,” in *Proceedings of the 30th Annual Meeting of the Cognitive Science Society*, 2008.



- [180] H. A. Kautz, "A formal theory of plan recognition," Ph.D. dissertation, Bell Laboratories, 1987.
- [181] D. Benyon and D. Murray, 1993. Applying user modeling to human computer interaction design. *Artificial Intelligence Review* 7(3-4):199-225.
- [182] A. Alvarez and J. Font, 2020. Learning the designer's preferences to drive evolution. In *Proceedings of the International Conference on the Applications of Evolutionary Computation*.
- [183] Kwon, N., Song, K., Park, M., Jang, Y., Yoon, I., & Ahn, Y. (2019). Preliminary service life estimation model for MEP components using case-based reasoning and genetic algorithm. *Sustainability*, 11(11), 3074.
- [184] Palomera-Arias, R., & Liu, R. (2015, June). Mechanical, Electrical, and Plumbing Systems in Construction Management: A Literature Review of Existing MEP Textbooks. In *2015 ASCE Annual Conference & Exposition* (pp. 26-1143).
- [185] Hassanain, M. A., Aljuhani, M., Sanni-Anibire, M. O., & Abdallah, A. (2019). Interdisciplinary design checklists for mechanical, electrical and plumbing coordination in building projects. *Built Environment Project and Asset Management*.
- [186] Wang, L., & Leite, F. (2016). Formalized knowledge representation for spatial conflict coordination of mechanical, electrical and plumbing (MEP) systems in new building projects. *Automation in construction*, 64, 20-26.
- [187] Li, Nan, Cheung, Sherman C. P, Li, Xiaodong, and Tu, Jiyuan. "Multi-objective Optimization of HVAC System Using NSPSO and Kriging Algorithms—A Case Study." *Building Simulation* 10.5 (2017): 769-81. Web.
- [188] Hulya Durur. "HVAC Optimization Based on Fuzzy Logic in Official Buildings." *International Journal of Computer Science and Software Engineering* 7.1 (2018): 1-5. Web.
- [189] Franco, Alessandro, Bartoli, Carlo, Conti, Paolo, Miserochi, Lorenzo, and Testi, Daniele. "Multi-Objective Optimization of HVAC Operation for Balancing Energy Use and Occupant Comfort in Educational Buildings." *Energies (Basel)* 14.10 (2021): 2847. Web.
- [190] Yuan, Xiaolei, Pan, Yiqun, Yang, Jianrong, Wang, Weitong, and Huang, Zhizhong. "Study on the Application of Reinforcement Learning in the Operation Optimization of HVAC System." *Building Simulation* 14.1 (2021): 75-87. Web.
- [191] Kotevska, Olivera, Johnston, Travis, Zandi, Helia, Kurte, Kuldeep, McKee, Evan, Munk, Jeffrey, and Perumalla, Kalyan. "RL-HEMS: Reinforcement Learning Based Home Energy Management System for HVAC Energy Optimization." *ASHRAE Transactions* 126.1 (2020): 421. Web.
- [192] Manuel, Mark Christian E, Lin, Po Ting, and Chang, Ming. "Optimal Duct Layout for HVAC Using Topology Optimization." *Science & Technology for the Built Environment* 24.3 (2018): 212-19. Web.
- [193] Korman, T. M., Fischer, M. A., & Tatum, C. B. (2003). Knowledge and reasoning for MEP coordination. *Journal of Construction Engineering and Management*, 129(6), 627-634.
- [194] Tatum, C. B., & Korman, T. (2000). Coordinating building systems: process and knowledge. *Journal of Architectural Engineering*, 6(4), 116-121.

- [195] Korman, T. M. (2009). Rules and guidelines for improving the mechanical, electrical, and plumbing coordination process for buildings. In *Construction Research Congress 2009: Building a Sustainable Future* (pp. 999-1008).
- [196] Korman, T. M. (2001). Integrating multiple products over their life-cycles: An investigation of mechanical, electrical, and plumbing coordination. Stanford University.
- [197] Yarmohammadi, S., & Ashuri, B. (2015). Exploring the approaches in the implementation of BIM-based MEP coordination in the USA. *Journal of Information Technology in Construction (ITcon)*, 20(22), 347-363.
- [198] Riley, D., & Horman, M. (2001, August). Effects of design coordination on project uncertainty. In *Proceedings of the 9th Annual Conference of the International Group for Lean Construction (IGLC-9)*, Singapore (pp. 1-8).
- [199] Khanzode, A., Fischer, M., & Reed, D. (2008). Benefits and lessons learned of implementing building virtual design and construction (VDC) technologies for coordination of mechanical, electrical, and plumbing (MEP) systems on a large healthcare project. *J. Inf. Technol. Constr.*, 13, 324-342.
- [200] Khanzode, A. (2010). An integrated, virtual design and construction and lean (IVL) method for coordination of MEP. Unpublished Technical Report, 187.



## ▪ APPENDIX A: QUESTIONNAIRE AND RESPONSES

### ○ A.1 Original Questionnaire provided to partners

#### Questionnaire - Quality Diversity in PrismArch

##### Introduction:

This questionnaire has been prepared by the UM team, as a follow-up to the first workshop on Quality-Diversity generative AI (which took place on [...]). We are using this document as a shared space for the exchange of ideas, in order to identify the best ways for exploiting our available technologies in the context of PrismArch.

[Additional details for deadlines and links to presentations made for the purposes of this questionnaire are omitted from the public deliverable]

##### ▪ Questions:

<b>Question 1:</b>	<b>Which (combined) characteristics of a design / solution do you find more valuable to explore, especially during the initial stages of design? How do you measure / assess those characteristics?</b>
Q1 - Answers:	Please state your name, discipline, affiliation and <u>provide your answers below!</u>  ...

<b>Question 2:</b>	<b>If the characteristics that you use to explore the solution space are difficult to quantify, could you attempt to describe them in qualitative terms? Are there any quantifiable characteristics that they relate to, even indirectly?</b>
Q2 - Answers:	Please state your name, discipline, affiliation and <u>provide your answers below!</u>  ...

<b>Question 3:</b>	<b>What are the most critical constraints that are imposed on your discipline by other disciplines?</b>
Q3 - Answers:	Please state your name, discipline, affiliation and <u>provide your answers below!</u>  ...

<b>Question 4:</b>	<b>What are the most critical constraints from your discipline that you feel other disciplines should respect?</b>
Q4 - Answers:	Please state your name, discipline, affiliation and <u>provide your answers below!</u>  ...

<b>Question 5:</b>	<b>What are the most important features of a design / solution that your discipline attempts to optimize?</b>  <b>How can you measure / assess those features?</b>
Q5 - Answers:	Please state your name, discipline, affiliation and <u>provide your answer below!</u>  ...

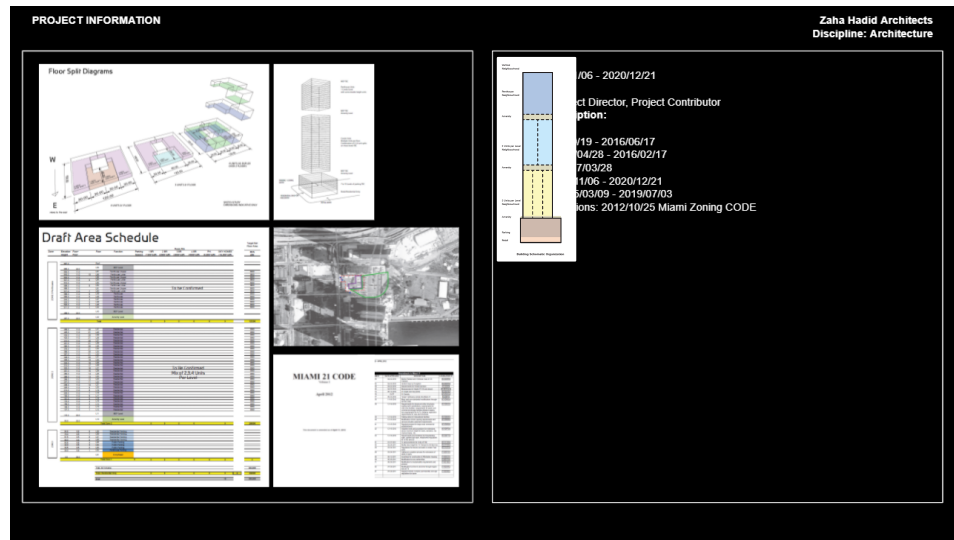
<b>Question 6:</b>	<b>If the optimization process occurs through an abstract / subjective process of refinement, could you attempt to describe it? Are there any quantifiable characteristics that it relates to, even indirectly?</b>
Q6 - Answers:	Please state your name, discipline, affiliation and <u>provide your answer below!</u>  ...

<p><b>Question 7:</b></p>	<p><b>Based on the example applications of Quality Diversity (QD) that you saw and discussed in the last workshop, how would you envision the use of QD in your discipline and across disciplines?</b></p> <p><b>Some example features are the following:</b></p> <ul style="list-style-type: none"> <li>- <b>The system is used in order to analyze the design space and provide the designer with a broad overview of the possible solutions, across a number of behavioral dimensions.</b></li> <li>- <b>The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process.</b></li> <li>- <b>The system keeps track of the user's interaction and learns to make suggestions that are more suitable to the user's preferences.</b></li> </ul>
<p><b>Q7 - Answers:</b></p>	<p>Please state your name, discipline, affiliation and <u>provide your answers below!</u></p> <p>...</p>

○ **A.2: Responses from ZHVR**

<p><b>Question 1:</b></p>	<p><b>Which (combined) characteristics of a design / solution do you find more valuable to explore, especially during the initial stages of design? How do you measure / assess those characteristics?</b></p>
<p><b>ZHVR:</b></p>	<p><b>Design/ solution as an emergent property</b></p> <p>The architectural design/solution space is defined through ongoing interactions with the client, and the other professional teams. It is important to note that when we begin a project, we have no way of knowing the end result. The result is an emergent property of our interactions and design activity. There are no readymade solutions. Design is a process - a path of experimentation.</p>

<b>Question 2:</b>	<b>If the characteristics that you use to explore the solution space are difficult to quantify, could you attempt to describe them in qualitative terms? Are there any quantifiable characteristics that they relate to, even indirectly?</b>
<b>ZHVR:</b>	<p><b><u>Defining the bounds of the solution space</u></b></p> <p>The characteristics that define a solution space are more akin to boundary conditions. These are defined by the client brief; the zoning and local regulations; site characteristics, such as orientation, adjacencies and contextual site conditions (e.g. a highway to the south, high-rise residential developments to the north); and other contributing factors such as quality of soil, existing infrastructure below grade, or any planting that needs to be preserved, etc. In addition to the characteristics listed, the budget plays a critical role in setting out what we can aim to achieve in a project.</p> <p>A combination of the client brief and the site conditions will create the boundary constraints for the future building volume. In any project, there will be a mixture of the parameters and limitations; there will be hard and soft limitations, and everything in-between. The building regulations and some other persistent limitations can formulate clear qualitative and quantitative guidelines for the approximation of the maximum build volume, which can be visualized in initial massing studies as a first step toward defining the possibility space of the project.</p> <p>Typically, the massing studies will show the maximum building volume envelope, as a step toward establishing the parameter space for the ensuing design exercise. This step itself can be related to the “workmanship of certainty” defined by Davie Pye. There are few risks involved. Ultimately, this is not an imposed limitation to the future design exercise, but a suggestion. It is important that designers receive the sources of information that led to this envelope limitation, to allow the industry to maintain the Golden Thread principle. By understanding how we arrived at that envelope, the designers can correct any wrongful assumptions of AI. Therefore, it is also the capacity of the designer to interact with the suggestions and the design parameters is critical to the successful implementation of the tool.</p> <p>It is important to note here that we do not know/ do not have access to all the limitations of the site from the very onset. A lot of these constraints and limiting values are revealed through iteration of proposals for the site. Often with building regulations there is some leeway in how they can be interpreted, and room for negotiation with the governing bodies. The client often engages with the local authority and is able to leverage the permissions based on leveraging other projects in the jurisdiction, etc.</p>

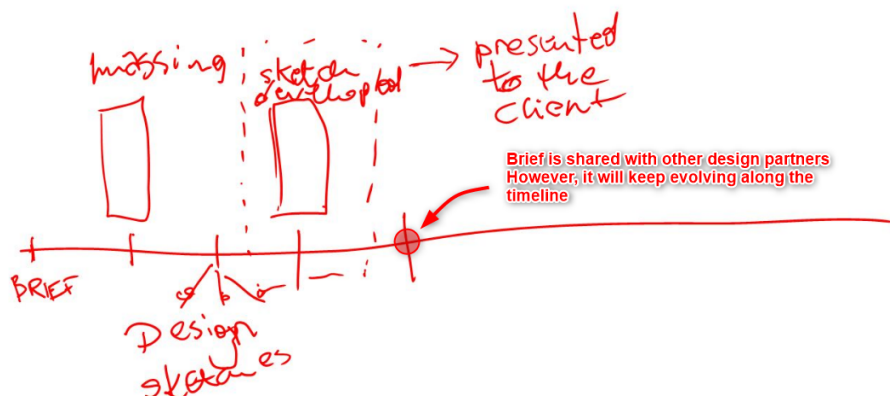


WP1 Case Study 02

<https://docs.google.com/presentation/d/1xKbqKpYfjkRwSSuY6SszBODX6sYcCpglAmlqU9WUNak/edit?usp=sharing>

As set out in D1.1, design result/solution is an iterative, emergent process:

An initial brief (statement of need) is presented by the client, and then refined throughout the feasibility study stage together with the architects. The resulting final brief is shared with the other design partners.



Inside PrismArch, all disciplines will be branching off design options, and evaluating them based on different qualities / parameters.

One of the design goals of the massing is a cross-check with the program and the tally of the floor areas, as well as a check of adjacencies, daylight dependencies, access, etc. However, it is important to note that some of the most iconic and

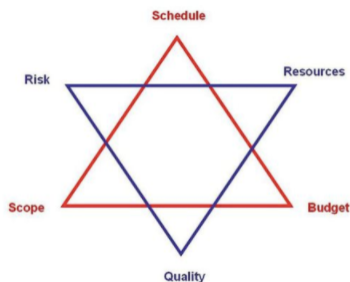
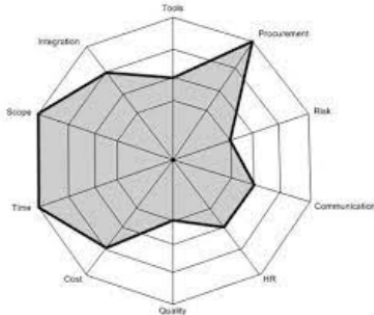


	<p>memorable architectural designs challenge these received rules and wisdoms. The Centre Pompidou in Paris is a textbook example of an inversion of the typical formula.</p>
--	---

<p><b>Question 3:</b></p>	<p><b>What are the most critical constraints that are imposed on your discipline by other disciplines?</b></p>
<p><b>ZHVR:</b></p>	<p><b>Critical constraints:</b></p> <ul style="list-style-type: none"> <li>- <b>Efficiency and responsiveness of the design process</b> (ability to turn around a design revision in a short time). Nimble, professional output is needed for client satisfaction.</li> <li>- What's crucial throughout the project is the <b>other disciplines' ability to highlight any potential risks and propose possible solutions early on in the project.</b></li> </ul> <p><b>Other constraints:</b></p> <ul style="list-style-type: none"> <li>- Design impacts of structure and MEP</li> <li>- Cost <span style="float: right;">impacts*</span> *A certain threshold of information is required before we can begin to address the budget.</li> <li>- Cost impacts* of facade / glazing openness (seen as an impact to MEP)</li> <li>- Cost impacts* of specialized, coordinated, combined solutions, or custom technological solutions (anything non-standard)</li> <li>- Both the structural and MEP disciplines require an architectural proposal to respond to and evaluate. Architects embed our early understanding of the economy of the project into the initial proposal.</li> <li>- Both the structural and MEP disciplines affect the usable space. If a building is imagined as a volume with strict boundary conditions (defined by the characteristics listed for Question 2), then the MEP solution will require ducting and (a) machine room(s) of certain dimensions. The structural solution will also take up a percentage of the volume - it may be a core and slab solution requiring columns, or cantilevers that increase the depth of each floorplate, but whatever structure is needed to support the building influences the inhabitable space.</li> </ul>

<b>Question 4:</b>	<b>What are the most critical constraints from your discipline that you feel other disciplines should respect?</b>
<b>ZHVR:</b>	<p><b>Coordination and design intent</b></p> <ul style="list-style-type: none"> <li>- The habitable spaces cannot be modified (additional structure, ductwork,etc) without coordination with architects. Architectural professionals design the interface between the human and the built environment; therefore the location of faucets, access panels, fire alarms, sprinklers, electric sockets, etc - all need to be coordinated with the architectural team. The experience of the end user must be taken into account from the architectural perspective.</li> <li>- The uniqueness of the project / design solution requires Structure and MEP to work within the envelopes that we propose.</li> <li>- Ideally, we move from the large gesture / big picture to the detail level, going from large to small in the narrative. It is tremendously difficult to redefine the big picture narrative late in the project development. The most hurtful interventions by other disciplines are ones that damage or break the golden thread, or break the design history.</li> </ul>

<b>Question 5:</b>	<p><b>What are the most important features of a design / solution that your discipline attempts to optimize?</b></p> <p><b>How can you measure / assess those features?</b></p>
<b>ZHVR:</b>	<p><b>Mission statement for architecture:</b></p> <p>(ref. RIBA Code of Professional Conduct )</p> <ul style="list-style-type: none"> <li>- duty to the wider public local community and society</li> <li>- future-proofing and sustainable solutions (forward thinking design approach, making the building adaptable to possible future uses)</li> <li>- consider the impact of a project on the natural environment</li> </ul> <p><b>Client satisfaction</b></p> <ul style="list-style-type: none"> <li>- usability of space &amp; fitness to the brief, design intent (broader design narrative that captures the client's imagination), cost, timeliness and accuracy of deliverables.</li> <li>- The satisfaction of the client is defined as the convergence of the proposal's cost / time / risk, overlaid with the client's available</li> </ul>

	<p>budget / procurement options / desired scope. The quality expectation is usually communicated as a set of performance-based characteristics (environmental, social, and project-type specific [i.e. acoustic] etc.)</p> <div style="display: flex; justify-content: space-around; align-items: center;">   </div>
--	---

<p><b>Question 6:</b></p>	<p><b>If the optimization process occurs through an abstract / subjective process of refinement, could you attempt to describe it? Are there any quantifiable characteristics that it relates to, even indirectly?</b></p>
<p><b>ZHVR:</b></p>	<p>Optimization. n. = the action of making the best or most effective use of a situation (or resource).</p> <p><b>The designer is the resource that we must make the most effective use of.</b> Saving time and effort is key, as well as avoiding cognitive overload and finding the right information.</p> <p><b>Additionally, supporting the design narratives and understanding the information that is recorded, and tracking the reasons behind design choices is very important.</b></p> <ul style="list-style-type: none"> <li>- <b>Early concept</b> <ul style="list-style-type: none"> <li>- massing: multi-parameter solution space</li> <li>- clarification/ tracking of site conditions, structural, and other issues that will need to be considered from the onset of the design</li> </ul> </li> <li>- <b>Facilitating immersive design work**</b> <ul style="list-style-type: none"> <li>- meeting constellation(s): recently-contacted list &amp; key meeting spheres superimposed</li> <li>- content curation: suggesting best view mode, etc. for a specific meeting type;</li> </ul> </li> </ul>

	<ul style="list-style-type: none"> <li>- <b>**creation of curated meetings spheres (in detail)</b> <ul style="list-style-type: none"> <li>- review current status of the design</li> <li>- -search for people who are involved in a particular subject</li> <li>- latest-approved elements, people who are involved in the decision-making in this particular element</li> <li>- people who attended the last meeting, &amp; minutes of the last meeting on the subject</li> </ul> </li> <li>- <b>PrismArch-wide design process analytics</b> with AI inference <ul style="list-style-type: none"> <li>- What is the current status?</li> <li>- How far are we from the contractual submission target?</li> <li>- Keeping track of critical timelines with multiple disciplinary inputs</li> <li>- How has “Designer A” changed the model?</li> <li>- How many assets has “Designer A” updated?</li> <li>- Are there any points that the AI can identify that we need to address?</li> </ul> </li> <li>- <b>Access to information</b> depending on the user status and/or organisation, and IP protection.</li> <li>- <b>Smart Query Tool filtering system</b> for enormous amounts of information. <ul style="list-style-type: none"> <li>- AI-filtered search results</li> </ul> </li> <li>- <b>Tagging</b> <ul style="list-style-type: none"> <li>- assistance with the naming</li> </ul> </li> <li>- <b>In-Line optimization</b> <ul style="list-style-type: none"> <li>- a change in design parameter could trigger AI automation of revisions, based on the completed workflow (ex. trimming of 50 slabs)</li> </ul> </li> </ul>
--	--

Question 7:	<p><b>Based on the example applications of Quality Diversity (QD) that you saw and discussed in the last workshop, how would you envision the use of QD in your discipline and across disciplines?</b></p>
-------------	--

	<p><b>Some example features are the following:</b></p> <ul style="list-style-type: none"> <li>- <b>The system is used in order to analyze the design space and provide the designer with a broad overview of the possible solutions, across a number of behavioral dimensions.</b></li> <li>- <b>The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process.</b></li> <li>- <b>The system keeps track of the user’s interaction and learns to make suggestions that are more suitable to the user’s preferences.</b></li> </ul>
<p><b>ZHVR:</b></p>	<p>Architects are curators of information, aimed toward a certain purpose. From our perspective, the ideal focus of QD would be aimed at reducing and focusing information for the numerous workflows and scenarios illustrated in D6.1, which show how we process and present information.</p> <p>We must answer the questions set out in this questionnaire in collaboration with our technical partners, using the roadmap set out in D6.1 as a guide. To determine the most appropriate location of AI inside PrismArch, we need to know what the system records and how the libraries are structured.</p> <p>The creation of spheres inside PrismArch is the core of the data organization process. The ability to select, group, and sphere items, introducing a timestamp and other associated data, should be made as intuitive as possible. Representation and access to this evolving dataset is another field where we see the need for AI assistance.</p> <div data-bbox="437 1464 1394 1989" style="border: 1px solid black; padding: 10px;"> <p>VR Experience 1 - UI Reference</p> </div>

VR Experience 2 - UI Reference

Teleport buttons "attached to" the physical location within the 3D model

attached to the user

- Project specific - different lighting scenarios buttons

attached to the user

- Controller/language toggle button

Slide from AI Workshop ZHVR Presentation, showing basic functionality (PUI [personal user interface] and QUI [query user interface]) needed for intuitive interaction by the IH (immersed human) with the datasphere.

Additionally, we have identified that UI/UX for the datasphere is another essential element of PrismArch where AI assistance is needed:

- **customisation of personal workspace** (so IHs don't get information / sensory overload)
- **locating optimal representation** (first-person user experience) for each member of PrismArch
  - presentation of 2D, 3D, or multiple media
- **help with work basics:** where did I leave "my desk" yesterday and where do I want to start today?

The Key Parameters for UI/UX and sphering inside PA are:

- level of staff, team organisation
  - the process of information optimisation / curation can be done according to idiosyncratic disciplinary preferences.
  - looking at information flows in teams and the collective (multi-disciplinary) information flows
    - i.e. Are there overlaps in functional roles?

○ **A.3: Responses from AKT II**

i. **Responses from Jeg Dudley (AKT II)**

<b>Question 1:</b>	<b>Which (combined) characteristics of a design / solution do you find more valuable to explore, especially during the initial stages of design? How do you measure / assess those characteristics?</b>
<b>Answer by Jeg Dudley (AKT II)</b>	<p><b><u>Cost:</u></b></p> <p>For many of the smaller projects I've worked on, the cost of different design options is critical to determine whether the entire project is viable or not (i.e. these are not large structures in which there is more opportunity to, say, carry out 'value engineering' by making the structure a percentage smaller).</p> <p><b><u>Quantities:</u></b></p> <p>However cost is difficult for structural engineers to quantify, so instead we extract all of the factors that determine cost - i.e. area of material, material thickness/ volume, length of element external edges (if the structure is formed in metal, this characteristic directly determines another characteristic - structural weld length). If the material is being bolted instead of welded, we can start to roughly calculate the number of fixings needed.</p> <p>In early stage of projects we often measure these through tabulated data exports from quick digital models generated in Rhino, AutoCAD, or occasionally Revit (though this is a slower software to use, so is less frequent in early stages). If the model is in Rhino, it is highly likely that we use Grasshopper and computational workflows to extract these quantities.</p> <p>We might then compare the relative material quantities to generate very rough estimates for relative 'cost' of different options.</p> <p><b><u>Materiality:</u></b></p> <p>Obviously in all of these studies we will also be considering different material options - e.g. in Metal: weight of a steel vs aluminium option, in Timber: perhaps Glulam option vs Laminated Veneer Lumber option, etc, etc.</p> <p>We assess these via material tables describing the structural properties of these materials. Note that many materials have different grades/ sub-grades/ mixes/ etc which can subtly or significantly affect the material performance - e.g. different grades of concrete.</p> <p>On very rare occasions, the structural behaviour of a material is not fully understood when we join the project, and so we undertake physical testing of small material samples to determine its characteristics. The results of these tests then feed back into our digital structural analysis models.</p> <p><b><u>Structural Analysis:</u></b></p>

	<p>In all of our projects we are assessing the basic structural behaviour of design options: the deflection under different loading conditions:</p> <ul style="list-style-type: none"> <li>- self weight</li> <li>- weather and environmental loads: wind, snow, seismic, etc</li> <li>- Live loads from people, equipment, furnishings, etc</li> </ul> <p>Also the utilisation of the material - i.e. what percentage of the material is being used to carry the load, which tells us how efficient the sizing of elements is and also how close they are to capacity.</p> <p>There are many more structural factors considered - however I will let the structural engineers in our team comment on those. Please see questionnaire responses by Edoardo Tibuzzi and Joel Hilmersson.</p> <p><b><u>Fabrication:</u></b></p> <p>We also work on many 'non-standard' structures - e.g. those with unusual forms, irregular shapes, etc. Many of these structures are prefabricated by specialist fabricators, who are the only parties with either the expert fabrication knowledge and/ or fabrication equipment needed to produce these works.</p> <p>So it is critical to determine how these structures are transported to site: i.e. how the structure could be broken into separate pieces for transport and on-site erection (segmentation/ discretisation of the form and/or size on truckbeds and within shipping containers, etc).</p> <p>These factors also relate to the weight of the overall structure and/ or separate pieces, and thus whether they can be moved from factory to site via cranes, hiabs/ lorry loaders, forklifts, etc.</p> <p>The weight and unusual centres-of-gravity will also factor into how the pieces must be supported in the temporary condition - i.e. before the whole structure is assembled and it is self-supporting. In the temporary condition systems like scaffolding, timber props, cranes, etc can be used to support the forms. In general, as structural engineers we are not responsible for designing the temporary works - but this is a requirement on some projects.</p> <p><b><u>Carbon:</u></b></p> <p>We are increasingly focusing on the embodied carbon of design options as well, which encompasses a vast array of characteristics - the carbon generated in producing a raw material, forming it, transportation, maintaining the material through a structure's duration, and in reusing or recycling it at the end of a structure's lifespan.</p> <p>These must also be balanced against the material's inherent physical characteristics - for example, it's thermal transmittance - which might mean that it has high embodied carbon, but it results in less heating and cooling of a structure throughout its lifespan, and so less carbon is generated overall.</p>
--	--



<b>Question 2:</b>	<b>If the characteristics that you use to explore the solution space are difficult to quantify, could you attempt to describe them in qualitative terms? Are there any quantifiable characteristics that they relate to, even indirectly?</b>
<b>Answer by Jeg Dudley (AKT II)</b>	See Q1.

<b>Question 3:</b>	<b>What are the most critical constraints that are imposed on your discipline by other disciplines?</b>
<b>Answer by Jeg Dudley (AKT II)</b>	<p>We are often constrained by characteristics related to <b>maximum acceptable floor depths</b> (perhaps in office or residential projects), facade buildups, and generally sizing of structural elements/ volumes within a building.</p> <p>These factors are understandably driven by a set of <b>commercial imperatives</b>: balancing number of floors vs usable area per floor vs distance from the cores (i.e. circulation lifts and stairs) vs density of the structural grid and size of columns/ shear walls in that grid, etc, etc.</p> <p>In relation to architectural collaborators, we are often asked to ensure <b>uninterrupted spans in buildings</b> - i.e. avoiding columns or other vertical structures that would disrupt the movement of users, or block sightlines. To achieve these spans, we must often increase the structural depth of elements above, which in turn affects those floor depths mentioned above. So the entire collaboration is a compromise between these factors.</p>

<b>Question 4:</b>	<b>What are the most critical constraints from your discipline that you feel other disciplines should respect?</b>
<b>Answer by Jeg Dudley (AKT II)</b>	<p>Because our collaborators from other disciplines understand that structural engineering is vital to ensure that buildings stay up, and do not harm anyone, in general they are very respectful of our constraints, just as they are the requirements from building regulations.</p> <p>All of our collaborators' questions or issues usually just relate to ensuring that the building is working as 'hard' as possible - i.e. that there is no potential to reduce material costs or increase spans anywhere, that might reduce carbon emissions, bring the cost down, or accelerate/ simplify construction.</p>

<b>Question 5:</b>	<b>Answer to Question 5, by Jeg Dudley (AKT II)</b>
<b>Answer by Jeg Dudley (AKT II)</b>	See Q1.

	<p><b><u>Fabrication Optimisation</u></b></p> <p>Once we are at the stage of optimisation, we might be looking to ‘regularise’ elements in some way - perhaps by making all of the connections in a structure conform to one of a limited set of different typologies (number of connection into the node/ angle of the connections/ material/ etc), or by making all of the member lengths identical/ one of a limited set of different sizes</p> <p>Because we work on many non-standard structures, we also encounter many issues related to ensuring planarity of elements. So this relates to simplifying forms and ensuring that wherever possible we are making forms from their planar sheet materials, and if needed cold forming them to slightly-curved forms, rather than utilising expensive hot forming solutions.</p> <p><b><u>Structural Optimisation</u></b></p> <p>We do many types of structural optimisation - some of the most basic being automated selected of section profiles from a pre-determined groups of options, assigned based on FEA analysis.</p>
--	---

<b>Question 7:</b>	<b>Answer to Question 7, by Jeg Dudley (AKT II)</b>
<b>Answer by Jeg Dudley (AKT II)</b>	<p>This is a difficult question, but the later two examples you state in the question intuitively feel more suitable for the types of (very highly dimensional) problems we encounter:</p> <ul style="list-style-type: none"> <li>- (1) <i>“The system keeps track of the user’s interaction and learns to make suggestions that are more suitable to the user’s preferences.”</i></li> <li>- I could imagine that for certain types of problems we encounter on projects, their ‘configuration’ for that project is so specific that potentially we cannot apply any tools ‘trained’ on the data/ configurations of previous projects. In this scenario, the ability to allow the user to follow their own design experience and knowledge in order to generate viable design options - and then use those ‘solutions’ to train the current system - could be very powerful.</li> </ul>

- (2) *“The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process.”*
- Even applying the ideas in (1) above, it seems likely that - even just in the first few iterations - the system will still not be generating ideal/ viable options, and thus giving the user the ability to at any stage step into the process and redirect the ongoing evolution seems very sensible.

### **Exploration of Non-Geometric Design Spaces**

As stated previously, some of the most significant challenges posed in projects are related to the logistics of cross-disciplinary collaboration: sharing and maintenance of communications - sketches, site photos, reports - as well as the expected digital models and drawings.

Any Tools that can highlight when these resources are not being used optimally would be hugely beneficial across the lifespan of a project.

*Examples:*

- Offer suggestions to tag and/ or attach documents/ emails/ sketches to an item that the system has flagged as related.
- Tag prior iterations of a design artifact to the current version.
- XXX
- XXX

### **Model Synchronisation and Diffing**

As a more general topic: Something that emerges naturally from the multi-disciplinary design process is the complexity of ensuring that the (necessarily) abstracted structural analysis model is an accurate reflection of the architectural model. Problems arise if one of these models is changed, and that does not carry through to the other. There might therefore be value in producing tools to highlight when elements have changed substantially enough in the architectural model to potentially need adjustment in the structural version too.

--	--

## ii. Responses from Edoardo Tibuzzi (AKT II)

<b>Question 1:</b>	<b>Which (combined) characteristics of a design / solution do you find more valuable to explore, especially during the initial stages of design? How do you measure / assess those characteristics?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	The main characteristic we are exploring in the initial stages of a design are Sustainable outputs, Cost and structural performance. We use simulation software combined with designer experience to evaluate these parameters both in isolation and then combine them to identify potential options to pursue in the latter stages.

<b>Question 2:</b>	<b>If the characteristics that you use to explore the solution space are difficult to quantify, could you attempt to describe them in qualitative terms? Are there any quantifiable characteristics that they relate to, even indirectly?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	The Characteristics above are all quantifiable. More precisely in respect to cost and sustainability, their values are fluctuating, but we still can use their outputs as a qualitative indicator to where the project is directed.

<b>Question 3:</b>	<b>What are the most critical constraints that are imposed on your discipline by other disciplines?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	<p>Critically, cost is the main driver in the process and structure being a primary fundamental part of the project is directly constrained by it.</p> <p>Architecture requirements are imposing geometric constraints to the structural model and often are locking non optimal solutions to the benefit of the architectural performance.</p> <p>MEP requirements often impact structural elements requiring openings and increasing the structural element size and cost.</p> <p>Site physical constraints are also a big influencer, ground condition, existing services etc are for example influencing structure element position or maximum loading allowed etc.</p>

<b>Question 4:</b>	<b>What are the most critical constraints from your discipline that you feel other disciplines should respect?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	Respect site constraints, fire requirements, seismic requirements, performance requirements.

<b>Question 5:</b>	<b>What are the most important features of a design / solution that your discipline attempts to optimize?</b> <b>How can you measure / assess those features?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	One of the most important features is structure weight. By optimising weight generally cost and sustainability are optimised subsequently. We use parametric tools connecting geometry and structural simulation to do so. Often also we require mocking up physical testbeds, running performance testing on them and analysing data to push performance further.

<b>Question 6:</b>	<b>If the optimization process occurs through an abstract / subjective process of refinement, could you attempt to describe it? Are there any quantifiable characteristics that it relates to, even indirectly?</b>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	The process is often driven by cost appraisal, we use the limits constrained by client budget to assess the structural impact and then provide alternatives that combine different structural options, different grid opportunities (Spacing between structural elements) and when those constraints are matched with architectural and MEP ambitions we go into the finer refinements of structural material optimisation as described in the answer Q%

<b>Question 7:</b>	<b>Based on the example applications of Quality Diversity (QD) that you saw and discussed in the last workshop, how would you envision the use of QD in your discipline and across disciplines?</b> <b>Some example features are the following:</b> <ul style="list-style-type: none"><li>- The system is used in order to analyze the design space and provide the designer with a broad overview of the possible solutions, across a number of behavioral dimensions.</li></ul>
--------------------	---

	<ul style="list-style-type: none"> <li>- <b>The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process.</b></li> <li>- <b>The system keeps track of the user's interaction and learns to make suggestions that are more suitable to the user's preferences.</b></li> </ul>
<b>Answer by Edoardo Tibuzzi (AKT II)</b>	-

○ **A.4: Responses from Sweco**

<b>Question 1:</b>	<b>Which (combined) characteristics of a design / solution do you find more valuable to explore, especially during the initial stages of design? How do you measure / assess those characteristics?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	In Building Services can be several things. From finding best routes with less frictions, calculations of spaces and clearance areas, effective arrangement of services in corridors and/or risers to more simple solutions which may include user interface adjustment based on the user and effective search tool based on specific criteria.

<b>Question 2:</b>	<b>If the characteristics that you use to explore the solution space are difficult to quantify, could you attempt to describe them in qualitative terms? Are there any quantifiable characteristics that they relate to, even indirectly?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	Further discussion needed in quantification to better understand the question. We have a separate section in BIM which involves quantification so not sure what this question relates to.

<b>Question 3:</b>	<b>What are the most critical constraints that are imposed on your discipline by other disciplines?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	We are driven by decisions made by other consultants such as structural engineers and architects. We need constant collaboration to coordinate our services and save costs from errors happening in the construction site. Often,

	we need to request changes from the other two disciplines which are discussed and agreed. Tolerances are always under discussion and taken under consideration.
--	---

<b>Question 4:</b>	<b>What are the most critical constraints from your discipline that you feel other disciplines should respect?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	The need for space for installation and maintenance purposes. This may include plant areas, corridor ceiling voids, apartment ceiling voids or even roof space for installation and maintenance of units.

<b>Question 5:</b>	<b>What are the most important features of a design / solution that your discipline attempts to optimize?</b>  <b>How can you measure / assess those features?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	Sizing and balancing of systems aiming on the most cost effective and sustainable solution with use of the right materials. There are a number of ways in a simulating environment to assess and evaluate the solution based on software specialized to perform these simulations.

<b>Question 6:</b>	<b>If the optimization process occurs through an abstract / subjective process of refinement, could you attempt to describe it? Are there any quantifiable characteristics that it relates to, even indirectly?</b>
<b>Answer by Dinos Ipiotis (Sweco)</b>	Need more input on this to understand the question, however I would say that any abstract / subjective process of refinement is dealt within the digital environment performing necessary tasks to mitigate it and reevaluate our solution. Aim is to spend as much time as possible in a simulating environment to prevent errors while construction or operation of the services

<b>Question 7:</b>	<b>Based on the example applications of Quality Diversity (QD) that you saw and discussed in the last workshop, how would you envision the use of QD in your discipline and across disciplines?</b>  <b>Some example features are the following:</b>
--------------------	--

	<ul style="list-style-type: none"> <li>- <b>The system is used in order to analyze the design space and provide the designer with a broad overview of the possible solutions, across a number of behavioral dimensions.</b></li> <li>- <b>The user is able to interact with the system, by selecting their preferred solutions or modifying existing solutions, thus intervening in the evolutionary process.</b></li> <li>- <b>The system keeps track of the user's interaction and learns to make suggestions that are more suitable to the user's preferences.</b></li> </ul>
<p><b>Answer by Dinos Ipiotis (Sweco)</b></p>	<ol style="list-style-type: none"> <li>5. The system is used to provide effective UI to the logged in users</li> <li>6. The system can perform smart search based on the given keywords and present relevant results which will be driven from the user interactions</li> <li>7. The system can propose design solutions after given a set of parameters and restrictions</li> <li>8. The system can identify potential coordination errors and highlight to the user</li> <li>9. The system keeps track of interactions, learn from them and adapt the experience to better suit the user's needs. This may contain predefined scenarios for the user to choose while logging in (e.g. coordination scenario, design scenario etc)</li> </ol>