DesignSense: A Visual Analytics Interface for Navigating Generated Design Spaces

by

Ahmed Mohammed Abu Zuraiq

B.Sc. in Computer Science, KFUPM, SA, 2017

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in the School of Interactive Arts and Technology Faculty of Communication, Art and Technology

© Ahmed Mohammed Abu Zuraiq 2020
SIMON FRASER UNIVERSITY
Fall 2020

Copyright in this work rests with the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.
Declaration of Committee

Name: Ahmed Mohammed Abu Zuraiq

Degree: Master of Science

Thesis title: DesignSense: A Visual Analytics Interface for Navigating Generated Design Spaces

Committee: Chair: Cheryl Geisler
Professor
School of Interactive Arts and Technology

Halil Erhan
Supervisor
Associate Professor
School of Interactive Arts and Technology

John Dill
Committee Member
Professor Emeritus
School of Interactive Arts and Technology

Robert Woodbury
Examiner
Professor
School of Interactive Arts and Technology
The author, whose name appears on the title page of this work, has obtained, for the research described in this work, either:

a. human research ethics approval from the Simon Fraser University Office of Research Ethics

or

b. advance approval of the animal care protocol from the University Animal Care Committee of Simon Fraser University

or has conducted the research

c. as a co-investigator, collaborator, or research assistant in a research project approved in advance.

A copy of the approval letter has been filed with the Theses Office of the University Library at the time of submission of this thesis or project.

The original application for approval and letter of approval are filed with the relevant offices. Inquiries may be directed to those authorities.

Simon Fraser University Library
Burnaby, British Columbia, Canada

Update Spring 2016
Abstract

Generative Design (GD) produces many design alternatives and promises novel and performant solutions to architectural design problems. The success of GD rests on the ability to navigate the generated alternatives in a way that is unhindered by their number and in a manner that reflects design judgment, with its quantitative and qualitative dimensions. I address this challenge by critically analyzing the literature on design space navigation (DSN) tools through a set of iteratively developed lenses. The lenses are informed by domain experts’ feedback and behavioural studies on design navigation under choice-overload conditions. The lessons from the analysis shaped DesignSense, which is a DSN tool that relies on visual analytics techniques for selecting, inspecting, clustering and grouping alternatives. Furthermore, I present case studies of navigating realistic GD datasets from architecture and game design. Finally, I conduct a formative focus group evaluation with design professionals that shows the tool’s potential and highlights future directions.

Keywords: Visual Analytics, Computational Design, Design Alternatives, Generative Design, Design Space Exploration, Design Space Navigation
Dedication

To my mom for teaching me ambition and my dad for inspiring me to care.
Acknowledgements

Alhamdulillah!

In the past two years, I have allowed myself to be consumed by intellectual pursuits like I never did before. This would not have been possible without the guidance and freedom to explore that my supervisor Dr. Halil Erhan provided me. Thanks Halil for being a mentor (and a father) throughout this journey. I would also like to extend my gratitude to my supervisors, Dr. Rob Woodbury and Dr. John Dill for challenging me to elucidate my ideas.

To my colourful siblings: Noha, Al-Maha, Muhammed, Anas and Osamah, for your continued love and support. To my parents Intisar and Muhammed, who supported us to pursue our dreams and who faced many hardships so we are never in need. I love you all!

To the members of the Computational Design lab: Maryam Zarei, Naghmi Shireen, Arefin Mohiudeen, Niloofar Kazemi, and Vahid Zahednejad. In whom I found a second family in these strange lands and times.

To Osama Alsalman, for the good times we spent as roommates, classmates, and lab-mates. To Abdulrahman Altamimi for our shared passion for science and the betterment of souls ever since teenagehood. To Angus Hon, for his support and the tonnage of bubble teas we so happily devoured.

I would also like to acknowledge the funding I received from Mitacs Canada, The Boeing Company Canada, and Stantec Vancouver, which gave me the peace of mind that allowed me to focus on research and growth. I also thank Alyssa Haas from Stantec for her supervision (and uplifting spirit!) throughout my Mitacs internship.

Finally, and sincerely, to anyone that ever taught me something and the giants of humanity on whose shoulders I continue to stand.
Table of Contents

Declaration of Committee ii
Ethics Statement iii
Abstract iv
Dedication v
Acknowledgements vi
Table of Contents vii
List of Tables xi
List of Figures xii

1 Introduction 1
   1.1 Challenges Facing Generative Design 2
   1.2 Points of Intervention 2
   1.3 Purpose Statement 3
   1.4 Contributions 4
   1.5 Thesis Structure 5

2 Background and Related Work 6
   2.1 Parametric Modeling and Design Spaces 6
   2.2 Generative Design 7
      2.2.1 Why Generative Design 7
      2.2.2 Modeling and Analysis 7
      2.2.3 Generation 7
      2.2.4 Before Navigation 8
      2.2.5 Navigation 8
   2.3 Visual Analytics 8
      2.3.1 Why Visualization 9
      2.3.2 Why Analyze 9
6.5 Reflecting on DesignSense ............................................. 98

7 Evaluation 100
7.1 Evaluated Features ...................................................... 100
7.2 Setting .......................................................................... 101
7.3 Results ........................................................................... 101

8 Discussion 105

9 Conclusion 109

10 Future Work 111

Bibliography 113
List of Tables

Table 5.1  Major missing features from the closest DSN systems. . . . . . . . . .  68
List of Figures

Figure 1.1 The generative design process. ........................................ 3

Figure 2.1 The visual analytics process (Based on the model by Kiem et al. [56]). Automated analysis methods (e.g., prediction, clustering, and recommendation) are tightly coupled with interactive visualizations. The process results in insights that inform future iterations of the process. 10

Figure 2.2 An example of a scatterplot. ........................................... 12

Figure 2.3 An example of a parallel coordinates plot (from Design Explorer [7]). 13

Figure 2.4 An illustration for how correlation patterns appear on scatterplots (top) and PCPs (bottom). Image retrieved from [58] ................. 13

Figure 2.5 An example of brushing and linking between two views: a scatterplot and a PCP. ................................................................. 14

Figure 2.6 An illustration of incremental selection. Retrieved from Wills [103]. 15

Figure 2.7 From left to right: clustering rectangles by their height, width and both. When clustering by one aspect (e.g., height) rectangles in the same cluster can still vary with respect to other aspects (e.g., width). 16

Figure 2.8 The red dots illustrate the alternatives on the Pareto frontier with two maximization objectives. These alternatives dominate all the other uncolored alternatives with respect to these objectives. [retrieved unmodified from Wikimedia Commons - by Njr00 - under https://creativecommons.org/licenses/by-sa/3.0/] ........... 18

Figure 2.9 An illustration of modeling a design space as a network. The rectangles at the bottom-left are alternatives in the explicit design space, which are already explored. The rest of the rectangles belong to the implicit design space, but they are accessible by one or more changes on already explored alternatives. .................. 19

Figure 2.10 Left to right: different generative models that are increasing in complexity. The first model only creates rectangles of a fixed width; the second allows variable width and height; the third allows rotating the rectangles. Each generative model has its own design space, and designers can either change the generative model or navigate its design space. .................. 20
Figure 2.11 A model capturing the interplay between the components of the generative design process and the visual analytics (VA) interfaces used for design space navigation. The insights gained through navigating with VA interfaces can be used to refine the generative model and the generation/evaluation of alternatives. 21

Figure 2.12 Retrieved from Daskilewicz and German [34]. 22

Figure 2.13 Design Explorer interface. The image is from the open-source (MIT License) web interface at https://tt-acm.github.io/DesignExplorer. 24

Figure 2.14 The interface of Dream Lens (left). An illustration of the form-driven selection (right). The images are from the publication by Matejka et al. [63]. 24

Figure 2.15 The interface of Cupid. Retrieved from https://vimeo.com/102600567 subject to a permitting licence (CC BY-NC-ND 3.0) 25

Figure 2.16 The Design Space Exploration Framework by Fuchkina et al. [42]. Interacting with any component in the user-created dataflow graph (right) shows a different view. To the left is a Self-Organizing Map (SOM) view. Both images are retrieved from a video subject to a permitting licence (CC BY-NC-ND 3.0) at https://vimeo.com/243872149 26

Figure 2.17 In the thesis by van Kastel [100], users navigate design alternatives by transforming and flying in a 3D digital environment. An accompanying video (at https://www.youtube.com/watch?v=APOsQQ2GMe4) was used for retrieving this image. 26

Figure 2.18 The interface of D.Star. A sample of two coordinated panels representing the same collection. 27

Figure 2.19 Retrieved from Doraiswamy et al. [37], as well as those captions: (a) Summary panel. (b) Building cart. (c) View panel. (d) 3D map. (e) Design exploration chart. (f) Design selection controls. 28

Figure 3.1 The research process comprises of iterative cycles of elicitation of the domain problem characteristics, consolidating the learned lessons, and developing prototypes that are guided by these lessons. Evaluation is a continuous activity combining the three major process components. 30

Figure 3.2 An example highlighting the need for coupling form and data. (A) The same parametric model, similar view and height but the top alternative form is malformed. (B) The same parametric model, similar view despite very different heights. (C) Different parametric models, similar view and height despite differences in the building’s shells. 32
Figure 3.3  A conceptual model of the coupling of design data and forms in a visual analytics context. The Form and Data similarity is derived by computing the distances between design alternatives. Examples of views for each are shown in the same coloured block. [Retrieved from a previous publication of mine [11]] 33

Figure 4.1  A: Dendrogram tree (Form or Data Similarity view), B: Scatterplot (Data view), C: Parallel Coordinates Plot (Data view), D: a grid of 3D views (Form view). [Retrieved from a previous publication of mine [11]] 37

Figure 4.2  A screenshot of Grasshopper [15], a parametric design tool. Varying the values for the sliders on the left results in different geometries. 38

Figure 4.3  A sample of the tabular data being visualized. 39

Figure 4.4  A scatterplot. 40

Figure 4.5  A parallel coordinates plot. 41

Figure 4.6  A grid of 3D views of design forms. 41

Figure 4.7  A dendrogram tree with alternatives at the leaves. 42

Figure 4.8  Interactions on the dendrogram tree using an artificial dataset. Top: clicking on a cluster node selects its members. The blue nodes (e.g., optimization) show collapsed branches. Middle: by dragging a node, users can move leaves or complete branches in the tree to different parents. Red circles around each target node show the range to which dragged nodes should be moved to belong to that target. A red link shows the target node that the dragged node will belong to once dragging is completed. Here the leaf (BetweennessCentrality) is moved to a new parent (cluster). Bottom: the tree after dragging is completed. 44

Figure 4.9  The brushing interactions between the components of the interface. 45

Figure 4.10  Details of the interaction in Figure 4.9. Using the same colors. 45

Figure 5.1  When clustering is integrated with navigation then a cycle can take place between the two. For example, designers can start by selecting a cluster of alternatives (left), then upon looking at their performance (right) they may pick the highest performing "green" designs and cluster them. 52

Figure 5.2  Form views are integrated with data views via visual links in the Design Galleries by Marks et al. [62] 54
Figure 5.3 Form views are integrated into a scatterplot matrix in Cupid [19]. The three rectangles are different scatterplots. Each colored dot is an alternative and the enlarged thumbnails are chosen to be from different clusters. ......................................................... 55

Figure 5.4 An example of different 2D views for a single office layout alternative from Project Discover [71]. Each view visualizes a different performance metric. From the left to right: adjacency preference, work style preference, buzz, productivity, daylight, and view to outside. ................................................................. 56

Figure 5.5 Each selection rectangle (brush) expresses a different criterion. Colored dots form the union of the two criteria. No alternatives meet both criteria. ................................................................. 60

Figure 5.6 Only a single rectangular brush is allowed in Design Explorer [7]. This makes it impossible to select different nonadjacent values for a categorical variable (e.g., blue and grey colors) or non-contiguous ranges on a quantitative one. ................................................................. 61

Figure 5.7 Focusing on one range of the input parameter misses out on sub-optimal, but potentially interesting, portions of the design space. ................................................................. 61

Figure 5.8 Variations on collections are represented in Spotfire [9] as bookmarks (save the current selection state), hierarchical lists, and tags. Note the minimal representation used. ................................................................. 64

Figure 5.9 A juxtaposition of alternatives in Asterisk [6] showing both their forms and data. ................................................................. 66

Figure 5.10 The interface composition of DesignSense. ................................................................. 69

Figure 6.1 Top: The operations top bar. On the left-most is a button for toggling inspection on hovering. In the middle are the operations related to selections such as creating sets, resetting selection, and inverting the selection. The right part contains information-seeking operations such as undo, zoom, filter, and returning to the initially loaded dataset. Bottom: An on-demand context menu that includes the same operations above. The icons are colour-coded to distinguish their category (selection or information-seeking). Some infrequent operations are left out to retain compactness. ................................................................. 75

Figure 6.2 Left: Rectangular brushing on the PCP view. Tooltips show the precise brush range. Right: The lines of an inspected alternative are thicker, while the lines of currently selected and unselected alternatives are given a lower visual priority. Finally, inspection is enhanced with tooltips that show that exact values for all dimensions. ................................................................. 77

Figure 6.3 The scatterplot view. The trophy button finds the Pareto frontier. ................................................................. 78
Figure 6.4: The Augmented Gallery: (A) Control panel. The icons at the top from left to right are: sorting dimension and order [arrow], the number of currently selected alternatives, filtering to selection [eye], and thumbnails' sizes [slider] (B) Detailed Inspection panel, shows the currently inspected alternative (or the last-inspected when no inspection is taking place). At the top of it, the icons from left to right are: closing the panel [X], switching to 3D view [3D], and changing to the next/previous image thumbnail [arrows] (C) Gallery panel. The rectangular brush at the bottom right shows selection on the grid. The thumbnail with the thick outline (top-right) denotes the currently inspected alternative.

Figure 6.5: Top: Brushing is denoted with dashed lines when the Augmented Gallery is only showing the current selection. Bottom: Direct comparison is currently enabled through a vertical panel below each thumbnail that lists the design data in key-value pairs. This feature becomes available when the size of the icons is set to the largest.

Figure 6.6: Left: Design Sets view is a list of cards, each representing a set. Each set card has a name and an associated comment. Users can delete a set or retrieve it (which integrates its members into the current selection). Right: The Clusters view. The thumbnails show the clusters' representatives. Users can select the number of clusters and the size of the thumbnails at the top. Through the drop-down at the bottom, they can also choose the quantitative dimensions they would like to cluster based on.

Figure 6.7: A store manager keeps a shared state consistent between all the system's components.

Figure 6.8: The store includes the dataset being explored in addition to user-generated data such as design sets, the currently selected or inspected alternatives.

Figure 6.9: A massing alternative.

Figure 6.10: A massing alternative under inspection.

Figure 6.11: The scatterplot on the left shows a strong negative correlation between SVF and Site Coverage. This is also clear from the crossing lines between the axes of these metrics on the PCP. The left PCP shows alternatives with high SVF already selected, and a tooltip shows the range on A/V ratio that will be selected once the user completes this operation. Since the current selection operation is Intersection, the result on PCP on the right shows alternatives meeting both criteria.
Figure 6.12 Users can alternate between (left) showing the selected alternatives in the context of those unselected, or (right) show only the selected alternatives. .......................................................... 91

Figure 6.13 By recurring inspecting cluster representatives, their performance and forms can be compared, and consequentially the clusters they represent. .......................................................... 92

Figure 6.14 Left: picking the dimensions to cluster on. Middle: selecting the cluster with a high A/V ratio and low site coverage (which is reinforced by looking at the scatterplot). Right: the dataset after excluding the cluster with low site coverage. .......................................................... 93

Figure 6.15 (1) zooming down with data queries. (2) alternating between data and form selection. (3) resulting alternatives are saved as a set, then the zooming is reversed and the set is viewed in light of the larger dataset. .......................................................... 94

Figure 6.16 A dataset of used cars. With some caveats, shopping for cars shares some similarities with design exploration, such as the need for coupling form and data. .......................................................... 95

Figure 6.17 The main view is a gallery of cars that can be sorted by their price or recency. A small text shows the price of the car with few user-chosen words. On the left is a set of components for filtering cars by different attributes, both quantitative and categorical. [A screenshot of https://vancouver.craigslist.org/search/cta] .......................... 96

Figure 6.18 DesignSense used to explore generated game levels. ................. 98

Figure 7.1 The interface of DesignSense at the time of evaluation. ............... 100
Chapter 1

Introduction

Music, paintings, puzzles, buildings, cars... this is a short list of things of artifacts whose designs can be generate today with computers. The computational power at our command is ever-increasing, and every day we are pushing the reach of what can be computationally generated.

In the architectural design domain, designers are increasingly solving problems with the aid of computational methods. Generative Design (GD) is one such example. GD combines the generation of design alternatives with advanced performance analysis and produces alternatives in unprecedented quantities. As such, GD can widen designers’ exploration and help architects meet their design goals and constraints [26], potentially in ways they did not foresee [28]. For example, GD allows architects to consider various ways to layout the floors of an office building while, among other goals, providing fair access to outside views and sunlight to the occupants of the offices [71].

How well an alternative satisfies the design goals is partially estimated with performance simulations and metrics in addition to any other qualitative criteria that are challenging to capture in such metrics. This is especially true in domains where qualities such as aesthetics and form composition are culturally important, as in architecture, industrial design, automotive design, and urban planning.

Navigating alternatives—particularly when large in number—calls for new computational design support tools. I contend that interactive data visualization and analysis should be at the core of such tools with features for examining, comparing, selecting, and sorting alternatives while simultaneously coupling the quantitative (e.g., performance) and the qualitative (e.g., form and aesthetics) aspects of alternatives.

This thesis focuses on exploring what the salient design navigation tasks could be and how they can be supported by new interfaces as an integral part of the design workflow. Especially, I am concerned with how well these interfaces augment the design exploration process and the degree to which they can address the challenges posed by Generative Design.
1.1 Challenges Facing Generative Design

Generative Design has a long story in the architectural design domain spanning at least 40 years [26, 65, 96]. Despite its promise, there are crucial criticisms for its practicality. A recent article by Daniel Davis [35] argued that the way they are employed in practice suffers from “choice overload, imprecise metrics, and a lack of design integration”. These challenges have been voiced in the research literature as well. For example, Brown et al. [24] and Erhan et al. [39, 40] describe the cognitive and choice overload that arise when choosing among many design alternatives. Adding to this overload is the observation that generated alternatives tend to have small perceptual differences between each other [12], making it harder to compare them [73].

The issue of "imprecise metrics", voiced by Davis, points at the shortsightedness of using quantitative metrics alone in navigating design alternatives. This shortsightedness is a result of attempting to reduce design quality, which is a complex phenomenon, to a few numbers. It is also due to the importance of qualitative criteria in evaluating design alternatives. Many calls were made for design navigation and analysis tools that support and couple both the qualitative and quantitative assessment of alternatives [16, 29, 41, 42, 63, 74, 100].

A study by Bradner et al. [22] showed that designers used the results of generative techniques as stepping stones upon which they continued their design process. Generative Design then does not happen in a vacuum and is part of a wider and ongoing design process. Integrating generative techniques with a design process can mean incurring the least imposition on it, or considering closely how these techniques fit in and inform the process. For example, Woodbury and Mohiuddin [106] criticize the mode of interacting with generators as oracles without inquiring into how they work. Instead, they call for a "dialogue with the design situation". The challenges put forth by Davis (and many others as I have noted) can be addressed at different points in the Generative Design process. Before describing the approach I take in this thesis, I will briefly illustrate the GD process.

1.2 Points of Intervention

A typical generative design process starts by creating the alternatives’ generator. The goal is for the generator to be able to produce valid and diverse solutions to the design problem. This step can be described as modelling the design space to be explored (Modeling stage in Figure 1.1). Within the architectural design domain, generators are commonly based on parametric models [43]. An automated search algorithm then proceeds with sampling alternatives from this design space while potentially favouring those that rank the highest with respect to chosen objectives (Generate-and-Evaluate cycle in Figure 1.1). The objectives can be chosen to reflect the performance metrics of the highest priority, e.g., as dictated by the stakeholders and designers in the project. Finally, designers are tasked with exploring the generated alternatives with the aim of refining the design space (i.e., by going back
Unlocking the full potential of Generative Design, as well as successfully addressing Davis' critique, hinges on many factors that span the whole generative design process as shown in Figure 1.1. These include the expressivity and validity of the underlying generative model, and the ability to properly sample the design space that the model endows for a set of diverse and optimal alternatives. Generative Design also relies on the ability to explore this generated set in a way that is unhindered by the size of this set, and in a manner that reflects design judgment, with its quantitative and qualitative dimensions. Understanding and responding to these factors is essential to reaching the full potential of generative techniques.

In this thesis, I take the route of designing interfaces that augment the generative design process. In particular, I focus on the interfaces used in exploring the generated design alternatives, also known as design space navigation (DSN) interfaces. My research goals are to develop DSN interfaces that:

**Goal 1** Reflect and support the way designers explore and make decisions.

**Goal 2** Address some of the challenges that are implied by generative design and which hinders its potential. Namely, I focus on the challenges of "choice overload", and "imprecise metrics".

### 1.3 Purpose Statement

Architectural design and similar domains are distinguished by the wickedness of their problems and the important roles that tacit knowledge and reflection play in how designers solve problems. This distinction motivates my first goal. My second goal stems from the challenges that generative design faces. I address the goals above as follows:
1. Critically analyzing the literature on DSN interfaces through multiple lenses that I developed throughout my research process. These lenses inquire about interface features that I deem necessary for meeting the research goals.

2. Building my lenses on a behavioural study by Shireen et al. [87] that showed how designers explore generated alternatives under cognitively-overloading conditions.

3. Iteratively developing a visual analytics solution that instantiates the suggestions resulting from analyzing the literature by these lenses.

4. Relying on interactions with design professionals throughout the process and ending with a focus-group evaluation with architectural design professionals who use generative design in their firm.

The lenses in (1) ask about the degree of support for the quantitative and qualitative assessment of alternatives in these interfaces, and how tightly they are coupled. Providing multiple means for tight coupling (e.g., coordinated selection and inspection of alternatives) is my answer to alleviating the impact of 'imprecise metrics' (from Goal 2) and it also mirrors how designers assess alternatives (from Goal 1). The lenses also inquire into the means of simplifying the design space and putting order into it, which is my response to the "choice overload" problem (Goal 2) and this response is motivated by the behavioural study of design exploration in action by Shireen et al. (Goal 1).

1.4 Contributions

The plan I outlined above resulted in a functioning prototype that builds on the developed lenses. I call this prototype DesignSense and it is a visual analytics system for navigating design alternatives. Its interface consists of coordinated and linked views, representing and coupling both the geometric design forms (qualitative aspects) and their performance data (quantitative aspects). Aiding navigation are mechanisms for selecting from, inspecting, and grouping design alternatives, where groups can be both manually and automatically created. To summarize, the contributions of this thesis include:

1. Identifying evaluation lenses that can be used to assess design space navigation tools. I hypothesize that these lenses can act as proxies for how well a DSN tool meets the goals I stated earlier, namely reflecting the way designers explore and tackling some of the challenges posed by Generative Design.

2. Developing DesignSense, a visual analytics tool for exploring generated alternatives. The features of the tool are based on the lenses above. DesignSense integrates several features that are either novel or appeared separately in the surveyed literature, but not in the combination I present here, and not to the same degree of integration.
Furthermore, I strived to be transparent about the design decisions behind the tool, with the hope that they provide future researchers with solid ground to critique and improve on what I present.

3. Presenting multiple case studies demonstrating the utility of DesignSense in exploring alternatives in architectural design, game design as well as shopping.

4. Conducting a formative focus-group evaluation of DesignSense with design professionals who received it well and helped in highlighting promising future directions.

1.5 Thesis Structure

The rest of the thesis is structured as follows: I start by presenting the basic concepts and methods that I rely on like design space exploration and visual analytics in Chapter 2. Chapter 3 states my research process and briefly describes the iterations that this research project went through. I proceed in Chapter 4 to describing the first iteration in that process in which I created a prototype to probe the problem and get early user’s feedback. Then in Chapter 5, I describe and analyze related work on DSN tools through multiple lenses that I developed throughout my research process. The result of this analysis motivates the second prototype (DesignSense) in the process, which I present in Chapter 6.

In Chapter 6, I proceed by describing the prototype developed in the second iteration (DesignSense), including the decisions I made (Section 6.2), the implementation of the prototype (Section 6.3) and scenarios of using it for design navigation (Section 6.4). I follow that with a formative focus-group evaluation of DesignSense with professional architects and computational designers in Chapter 7. Finally, I conclude with a discussion of the results of the evaluation along with a more general reflection on the contributions of this thesis in light of my initial goals and the lenses I developed (Chapter 8).
Chapter 2

Background and Related Work

In this chapter, I present some of the concepts and methods that I use throughout this thesis. I start with parametric modelling and a break down of the generative design process. I use many techniques from the field of visual analytics (VA) in implementing DesignSense. So, the section that follows introduces VA with its two parts: interactive visualizations and data analysis. Following that, I discuss design space exploration (DSE) and how it applies to generative design. I conclude with a brief survey of using interactive visualizations in navigating design alternatives.

2.1 Parametric Modeling and Design Spaces

Requirements continuously change throughout architectural design projects, and responding to these changes can be costly, especially in the latter stages of a project. Parametric models were introduced to architecture as a flexible approach that requires less effort for making changes than other modelling approaches [36]. Today, parametric modelling tools such as Grasshopperr [15] and Dynamo [1] are growing in popularity.

A parametric model is a dataflow program encoding a set of related rules and operations that produce geometric objects. The outputs of the model depend on its input parameters, in that each new set of input values result in different outputs after being propagated through the dataflow program [105]. A designer can then explore a design problem and respond to the changing design requirements by iteratively refining both the underlying dataflow program and its input parameters [12].

The set of all possible input values and their corresponding outputs form a vast design space. Each point in that space is a design alternative. Yet, not all this space is useful or valid; hence the designer’s task is to navigate it, searching for ‘interesting’ design alternatives.
2.2 Generative Design

Generative Design is a process by which designers can explore vast design spaces. In particular, design alternatives are generated from that space with automated search algorithms that are steered by performance analysis. The examples of performance analysis in architecture vary from basic cost estimates and spatial proportions to advanced energy and daylight simulations [83, 99].

2.2.1 Why Generative Design

The reasons for using Generative Design vary. A commonly cited reason is achieving design performance [26]. Besides that, the "non-traceable" nature of the generation process allows surprising results to emerge [26]. The study by Bradner et al. [22] showed that generative design played a larger role in exploration among designers than identifying optimal solutions. Generative design can also be used for bringing together multidisciplinary teams in exploring performance trade-offs at early design stages [43].

The process of Generative Design can be broken into modelling, analysis, generation (or sampling) and navigation, as shown in Figure 1.1. The following is a brief description of the activities that can be involved in each.

2.2.2 Modeling and Analysis

Within architecture, it is currently common to base the modelling component of the generative design process in the context of parametric modelling tools. Furthermore, analyzing the performance of design alternatives is increasingly integrated with the modelling environment through external plugins and addons such as LadyBug [81] and EnergyPlus [32].

2.2.3 Generation

The generation (or sampling) process can take many shapes. First, it can be an automated search process, such as meta-heuristic optimization algorithms (the survey by Wortmann and Nannicini covers those [109]). It can alternatively be performed by uniformly or randomly sampling the parametric space of chosen input parameters. Designers may also alternate between these sampling strategies. For example, they can start with a uniform (but coarse) division of the parametric space they explore to identify the regions where the highest performing alternatives lie [72]. Upon finding those regions, an optimization process can find the optimal alternatives within the regions’ confines.

1Generative Design is also known under different terms such as Optieereing [43] and Performance-based design [83]
2.2.4 Before Navigation

Parametrically-generated design alternatives can be visually similar [12], which hinders their evaluation and comparison [73]. The works by Yousif et al. [111] and Erhan et al. [40] demonstrate how design alternatives can be reduced to a smaller but more visually distinct set before navigation commences.

2.2.5 Navigation

The last stage in the generative design process consists of navigating the generated alternatives in search of alternatives that meet the designer’s criteria. A significant challenge to navigation is the abundance of these alternatives and the small perceptual differences between them. The exploration can be supported with data visualizations that allow users to quickly filter alternatives by their performance or scan their visual appearance at the same time. I delay the details of the tasks involved in this stage to Section 6.1.2. Later in this chapter (Section 2.5.5), I describe more of the literature on assisting design navigation with data visualization and analysis.

2.3 Visual Analytics

Data is being increasingly relied on to inform decision-making in many fields, and architectural design is not an exception. Data can inform design in many ways; it can be part of understanding and formulating problems as well as guiding the creation and evaluation of their solutions.

The generative design process can be informed by users’ preferences or contextual data, such as the geographic and atmospheric conditions of the design project’s site. But this process produces data as well. Each generated alternative (of which there can be hundreds) can bundle multiple performance metrics, geometric forms, and image visuals. This data can also include attributes describing the process, such as the names of the designers involved, the version/iteration of the parametric model used to generate the design alternatives, and the specific hyper-parameters and settings for the automated search procedure. Finally, when designers navigate the generated design alternatives, they contribute their share of data to the above in the shape of annotations such as comments, ratings, tags.

The generated data is then abundant and multifaceted, creating a need for tools that support designers when making sense of it. This thesis focuses on developing visual analytics tools for navigating generated design alternatives. This work extends a line of similar tools in the research and practice of engineering and design domains. The common insight underpinning these tools is that visualizing the performance of design alternatives allows designers to scan and filter large sets of alternatives rapidly [98].
2.3.1 Why Visualization

The premise behind data visualization is that data can be better understood when represented in a visual format that capitalizes on the human’s visual system and amplifies their cognition with external representations [70]. Several visualization techniques are present in the field of Information Visualization (InfoVis), where each technique is aimed at different types of data and serves specific user tasks. For example, scatterplot charts visualize two-dimensional quantitative data and allow users to quickly identify correlations between variables. These visualizations can become more powerful with interactivity. For example, interacting with visualizations can allow users to shift from passively looking up information to exploring relations and asking what-if questions.

2.3.2 Why Analyze

In addition to being represented visually, performance data is also amenable to automated analysis techniques. These include calculating the correlation between variables, finding clusters of similar designs in a dataset, identifying the Pareto frontier\(^2\) for a set of design alternatives and building prediction models. These techniques can provide a systematic basis for understanding the design space at hand in addition to supporting decision making. The work by Brown and Mueller covers a number of them [25].

Relying on automated techniques alone is rarely sufficient for understanding design data. This is partly due to the role that domain knowledge plays in design and which is challenging to replicate or reflect through data-driven analysis. On the other hand, relying solely on interactive visualizations misses the potential that automated analysis can provide, especially when navigating large and complex design spaces. This is indeed the case with the outputs of generative design.

2.3.3 Visual + Analytics

The field of Visual Analytics (VA) combines both interactive visualizations and automated analysis [55]. This interplay between visualization and analysis is illustrated in Figure 2.1. The survey by Ramanujan et al. [79] demonstrates that visual analytics tools were used in design\(^3\) to: (1) navigate multi-dimensional design spaces, (2) understand the trade-offs between the parameters involved, or (3) generate insights to support making decisions.

\(^2\) More on that in Section 2.4.3

\(^3\) The survey tackles sustainable lifecycle design in general but I argue that the statements here hold for most data-informed design processes.
Automated analysis methods (e.g., prediction, clustering, and recommendation) are tightly coupled with interactive visualizations. The process results in insights that inform future iterations of the process.

### 2.3.4 Sensemaking

When the analysis tasks at hand move beyond simple information retrieval to exploring, understanding and structuring information, sensemaking starts. The sensemaking model, proposed by Pirolli and Card [76], consists of two main loops: foraging and sensemaking. In the first, users seek information, they search and filter, and they construct schemas (or structures) of their findings. In the second loop, users rely on their developed schemas as they formulate and test hypotheses. Users can start with hypotheses and seek evidence for it in the data (top-down), or they may let the data speak for itself, allowing schemas to emerge through an open-ended exploration that can then generate hypotheses (bottom-up). Pirolli and Card report that intelligence analysts employed both processes opportunistically.

A hypothesis in a design navigate task can take the shape of a set of design alternatives that satisfy the designer’s criteria. For example, it can be a statement like: "the buildings with circular forms save energy the most and are most visually pleasing.". This statement could act as a top-down hypothesis if designers experienced this quality of circular buildings before. The same hypothesis could be derived in a bottom-up fashion. For example, if after seeking the most energy-saving and pleasant-looking alternatives, the result mostly consisted of circular forms.

When designers navigate, they engage in seeking alternatives (foraging loop) in addition to formulating hypotheses concerning these alternatives and their relation to the design problem (sensemaking loop). It follows from the above description that design space navigation can indeed be characterized by the sensemaking model.
2.3.5 Visualization Tasks Typologies

Design exploration involves a combination of subtasks, like selecting, comparing, and grouping alternatives. The community of InfoVis contributed several typologies of visualization techniques. These typologies help visualization designers in matching user tasks with the techniques most effective at performing them. Section 6.1 uses the typology by Brehmer and Munzner [23] to describe the user tasks supported by data visualizations during design space navigation.

The typology by Brehmer and Munzner [23] was created by surveying over two dozen other theoretical frameworks and classification systems. The survey covered the literature of data visualization and human-computer interaction, among others. The authors argue that the surveyed classifications focused on either low-level tasks (which describe how a goal is accomplished) or high-level goals but did not cover tying them together. Instead, their typology does that by analyzing visualization tasks based on three questions: Why, How and What.

The first question Why: asks about the goal of using visualizations, in other words, the tasks they are intended for. What: asks about the type of data to be visualized. How: concerns the visual representations and interaction techniques employed on the data to achieve the the user tasks (Why).

We can design better visualization systems by matching the users’ tasks to the visualizations most suited for these tasks, and typologies can help us in describing both. Moreover, a typology can highlight gaps in the tasks supported by a system. Finally, abstractly describing systems’ tasks allow researchers who develop visualization solutions to compare their contributions to earlier ones.

2.4 Techniques

In this thesis, I present a Visual Analytics system for navigating design spaces. In building this system, I have relied on various visualization and interaction techniques from the InforVis literature.

2.4.1 Visualization Techniques

Visualizing data dimensions requires mapping them to concrete representations using various graphical attributes such as position, size, shape, or colour. Visual representations can be referred to simply as views. For example, a scatterplot is a view representing two quantitative dimensions by mapping them into different positions.

Scatterplots

A scatterplot view represents two quantitative dimensions by mapping them into two orthogonal axes (Figure 2.2). Alternatives are shown as dots with varying positions along
the axes, depending on their values. Scatterplots are effective at identifying correlations between dimensions [70]. They can also be used in finding clusters, outliers, minimums, and maximums.

![Figure 2.2: An example of a scatterplot.](image)

**Parallel coordinates plots (PCP)**

Parallel coordinates plots (PCP), first introduced by Inselberg [50], are used to visualize multidimensional data where the dimensions are represented by a set of parallel vertical axes, one for each dimension. Each alternative is represented as a series of line segments crossing the vertical axes at points that correspond to their values. Figure 2.3 below illustrates that. In practice, PCPs are used to provide an overview of the data dimensions. They are also commonly used for filtering and identifying outliers [70]. Since the order of axes in the PCP have an impact on whether a correlation is discernible, it is advised to enable reordering the axes.

Furthermore, since design data often involve performance metrics, the vertical ordering of the values along each axis should be reverse-able. This is because we may be seeking to maximize or minimize these performance metrics, then to ensure consistency, we can opt to change the vertical order of the axes such that the values at the top are always better (or vise versa). Finally, I have been assuming that the PCP will be oriented horizontally (vertical axes ordered from left to right), but that does not need to be the case.
In practice, PCPs are used to provide an overview of the data dimensions. They are also commonly used for filtering and identifying outliers [70]. Like the scatterplot view, correlations can also be discernible on the PCP, although less effective. Figure 2.4 below contrasts how correlations appear on scatterplot and PCP views.

Some standard interactions on the PCP include:

1. Correlations between dimensions are only visible if their axes are adjacent. Hence it is advisable to allow users to change the order of the axes.

2. Design data often involve performance metrics, which we might seek to maximize or minimize. Hence, the desired values might be at the bottom or the top of an axis. To ensure consistency (e.g., up is always better), users can be allowed to change the vertical order of the values along each axis.
2.4.2 Interaction Techniques

Brushing is the act of selecting elements on a view via user interaction. Brushing is accompanied by highlighting the selected elements, often by changing their size or colour. Upon brushing, the unselected elements can be filtered out of the view or de-emphasized; for example, by dimming their colours or reducing their size.

When elements are selected in one view, we can subsequently highlight or filter their corresponding representations in other views. This is an example of linking or coordination between views. Multiple Coordinated Views (MCVs) [80] refers to views with different representations on which operations are coordinated. MCVs are useful when no single representation enables all the tasks we demand or when the data is multifaceted and requires relating disparate aspects to each other. Figure 2.4 shows an example of MCVs.

![Figure 2.4: An example of brushing and linking between two views: a scatterplot and a PCP.](image)

There are different variants for coordination between views depending on whether they share the same/different representations and whether each view visualizes the whole dataset or a subset of it [70]. Examples of coordination variants include linked small multiples and Overview+details views [80]. In the first, views use the same representation but differ only in the data dimensions (e.g., a matrix of scatterplots). In the second, one view provides an
overview of the data, possibly in an abstracted or aggregated form, while the other presents
details on select parts of the overview.

**Information Seeking**

When exploring complex datasets, users can start from many places, but an overview can
be a starting point for exploring, after which designers can drill down into details. This
is an example of the Information Seeking Mantra [90]: "Overview first, zoom and filter,
then details-on-demand." The mantra tackles the strategy users may go through as they are
seeking information on these views, in other words, the first loop (the foraging loop) of the
sensemaking model in Section 2.3.4.

**Visual Queries and Incremental Selection**

To be able to zoom on or filter information, users need to select it first. Standard selections
methods in visualization systems include dynamic queries widgets [89] and brushing on
linked visualizations [70]. Both methods aim at providing immediate feedback to users
about the consequences of their selections. Widgets are common in commercial shopping
interfaces, while brushing is analogous to selecting icons in GUI operating systems [103].
When compared within a system for visualizing geographic information, widgets were more
effective at selecting value ranges while brushing (which takes place on visualizations) was
found suitable for evaluating trends and relations between variables [78]. Brushing allows
users to select items while cued by the patterns (e.g., trends, extremes, outliers, clusters)
appearing on linked visualizations. Dynamic query widgets have no parallel to that excepted
when augmented with visual scents [102].

To improve the utility of brushing, users can tune their selections by incrementally re-
fining them through different logical operations [103]. These operations dictate how newly
selected items relate to previously selected ones. Figure 2.6 illustrates some of these opera-
tions. Providing a "powerful, intuitive and forgiving" [103] selection system is key to a good
visualization system.

![Figure 2.6: An illustration of incremental selection. Retrieved from Wills [103].](image-url)
2.4.3 Data Analysis Techniques

This section introduces key data analysis techniques to help in exploring design spaces. These are:

- Design space simplification methods, primarily clustering and dimensionality reduction: These produce aggregated views of design spaces, which reduce the cognitive load of exploring large design spaces.
- Pareto frontier analysis: A method to reduce the volume of a design space by focusing on the highest-performing alternatives.

Clustering

Clustering is the process of automatically grouping similar items. To decide whether two items are similar, we need to compare them. Doing so requires choosing: (1) the aspects to be compared, and (2) a way to measure the difference (or distance) between these aspects. For example, Figure 2.7 illustrates clustering a group of rectangles by their height, width or both (aspects). The distance is measured using the Euclidean distance function, as shown in Equation 2.1.

\[ d(rect_1, rect_2) = \sqrt{(w_2 - w_1)^2 + (h_2 - h_1)^2} \]  

(2.1)

Common clustering algorithms include K-means, DBSCAN, and hierarchical clustering. As they are beyond the scope of this thesis, I refer the reader to Berkhin [20] for a survey of clustering methods.

Clustering can be used to find families of similarly performing alternatives or alternatives that resemble each other visually. Successfully doing so allows us to simplify the design space and more easily understand the underlying patterns in it. For a longer survey on clustering
for design navigation and a discussion of its benefits, I refer the reader to Abuzuraiq and Erhan [11].

**Dimensionality Reduction**

Dimensionality Reduction (DR) is a set of techniques that help analyze multi-dimensional data by minimizing the dimensions under consideration or mapping the data into a new, smaller space. DR techniques can simplify or denoise the data while retaining any intrinsic patterns found in the multi-dimensional data.

For example, in the context of parametric design models, a large number of parameters usually control the generated geometric variations leaving designers with the challenge of discerning the range of geometries that a parametric model can generate. Harding [45] proposed applying a DR technique to map alternatives, based on the similarity between their input parameters, into a two-dimensional grid. As a result, the geometric variations are more readily pronounced as we move along any of the two newly synthesized dimensions.

The technique used by Harding is called Self-Organizing Maps (SOM), first introduced by Kohonen [57], and it aims at creating a spatial map whereas inputs that are similar in their original multi-dimensional space are close to each other on the resulting map. In other words, SOM attempts to preserve the topology of the multi-dimensional space. Other standard DR techniques include Principal Components Analysis (PCA) [53] and Multi-dimensional Scaling [21], which map multi-dimensional data into fewer dimensions, albeit without being bound to a grid.

**Pareto Frontier Analysis**

Satisfying multiple objectives simultaneously is challenging. For example, increasing the mass of a car gives it more stability, but requires more energy to move it. The objectives of maximizing stability and minimizing energy consumption are at odds with each other. Situations such as these are common in design, and in the presence of such conflicting objectives, we can *satisfice* by picking the design alternatives lying on the Pareto frontier. The Pareto frontier consists of the set of designs that cannot be modified with respect to one objective without compromising on the others. Figure 2.8 below illustrates that.

Through considering the alternatives on the frontier, we can reduce the volume of a design space by concentrating on its well-performing alternatives. Furthermore, alternatives on the frontier can vary with respect to other properties, which allows designers to consider these when arbitrating between the members of the frontier.
Design Space Exploration (DSE) is the systematic process of exploring design alternatives. When considering its concrete digital application, the term has many senses. Sometimes it refers to the formal specification and automated exploration of a design space assisted with a search algorithm. This is the sense more common in the engineering disciplines where often the challenge is to explore a large combinatorial solution space of a well-defined problem. DSE can also refer to using tools that allow a higher degree of user involvement in investigating and evaluating design alternatives. Conceptually, Woodbury and Burrow [108] model design space exploration as movements in a network that connects the previously examined alternatives (explicit design space) to the yet unexplored ones (implicit design space). This network can be conceptual or concrete, depending on the existence of an underlying representation or formalism. To illustrate this model of design exploration, we can consider the actions of a designer exploring rectangular shapes in a 2D drawing software, as seen in the Figure 2.9. For a longer discussion about the different definitions of design space exploration, I refer the interested reader to Section 2.3 in the thesis by Ye Wang [101]. Meanwhile, to avoid the confusion resulting from the overloaded definitions of the term, I use Design Space Navigation (DSN) to refer to the particular set of design tasks that I tackle in this thesis.
2.5.1 DSE applied to Generative Design

When considering generative design, the sampling process seems to follow no clear structure as the one described by Woodbury and Burrow [108]. A degree of randomness is even essential to many generative techniques. For example, if we look closely at an evolutionary search algorithm, one may see in the mutations made to the populations of generated alternatives a parallel to the designer’s actions. But this analogy fails because the movements, in this case, are often happening in a predetermined solution space. In contrast, designers in more traditional processes move in both problems and solutions spaces [60].

A possible remedy to this is to include generative models in the picture. A generative model can be understood as the schema or representation that gives rise to the solution space from which we sample. In architectural design, parametric models map precisely to that.

But parametric models in architecture existed before they were used as a backbone for generative design. Aish and Woodbury [12] describe a multi-level interaction with parametric models where both their input parameters and the underlying schema are subject to change as designers explore different alternatives. This leads us to the conclusion that the multi-level interaction applicable to parametric models also applies to generative design.

Designers then explore at two different levels in the context of generative design. The first consists of the changes they make to the generative models and the second consists of navigating the relatively disconnected alternatives generated from these models. To illustrate that, Figure 2.10 uses the example of a generative model producing rectangles.
2.5.2 Supporting the Refinement of Design Spaces

I just described a way to map the activities in generative design to the conceptual account of design exploration by Woodbury and Burrow [108]. However, movements in the parametric model’s space are hindered by the technical programming knowledge they require, which is not often part of the training of architectural designers. Furthermore, these models tend to be inflexible as they grow in complexity [47].

This calls for better support for movements in the models’ space. For example, these movements can be mediated by sampling and visualizing alternatives from a model in order to assess its scope and complexity [72] or its validity and reliability [38].

These movements can also be automated, as seen in the work by Harding on metaparametric design [47]. Furthermore, they can also be supported by visualizing the history of changes to parametric models, as done by GHShot [33]. Finally, the joint exploration of both levels (models and alternatives) has been demonstrated by Zaman et al. in MACE [113].

2.5.3 Design Space Navigation: Concepts

In this thesis, I acknowledge the need for considering the movements in both spaces, that of models and alternatives. However, I focus on the navigation of generated alternatives aided with interactive data visualization and analysis, which I will continue to refer to as design space navigation (DSN). To recap, I see design space exploration when applied to generative design and assisted by data visualizations as consisting of two major tasks. The first task is the assessment, and subsequently, the refinement of generative models (as described in Section 2.5.2). The second task involves navigating alternatives generated
from these models; this navigation has the aim of finding the set of alternatives that best match the designer’s criteria. Two statements further qualify navigation. The first is that the designer’s criteria are in continuous flux, and they get refined by studying and comparing alternatives. The second relates to the observation by Bradner et al. [22] that designers use the found set of alternatives resulting from navigation as a starting point in an ongoing design story rather than as final solutions.

When we combine the Generative Design model in Figure 1.1 and the visual analytics model in Figure 2.1, we can create a model that captures the relation between the navigation of design spaces and their refinement. This is shown in Figure 2.11 below.

Figure 2.11: A model capturing the interplay between the components of the generative design process and the visual analytics (VA) interfaces used for design space navigation. The insights gained through navigating with VA interfaces can be used to refine the generative model and the generation/evaluation of alternatives.

The Impact of Modeling and Sampling on Navigation

The arguments and methods I describe here apply to the navigation of alternatives generated using any generative technique (be it parametric models, shape grammars, or constraints solvers) as long as these alternatives are evaluated by quantitative metrics that aid in filtering and comparing them. The exact sampling process does not matter from a visualization perspective, but it impacts the kind of accrued insights. For example, optimization-based sampling can provide a better glimpse of the trade-offs in the design space (e.g., through examining Pareto frontiers). On the other hand, uniform sampling can make it easier to systematically narrow down the design space to the most fertile input ranges (e.g., see the case study by Nagy et al [72]).
In the next section, I will describe the literature on interactive data visualization tools for the task of design space navigation. For brevity, I will refer to these tools as design space navigation tools or DSN tools.

2.5.4 Design Space Navigation: Form and Data

Using information visualizations for navigating design spaces has many precedents in different domains. Several systems have been proposed in engineering domains addressing this task, such as the ARL Trade Space Visualizer [97], modeFrontier [31], RAVE [34], and LIVE [110]. A primary goal of these systems is to integrate design optimization, visualization and decision-making. This thesis focuses on navigating and selecting among already-generated alternatives instead of interactive optimization or design steering, which many of these systems aim for. When considering the parts of these systems aimed at navigation, we find feature-rich support for various abstract visualizations such as parallel coordinates plots, scatterplot matrices, 3D scatterplots, histograms and more. The interface of RAVE is shown in Figure 2.12 as an example. Users of these systems can shop for alternatives [17,98] with the aid of interactive visualizations of their performance data.

![Figure 2.12: Retrieved from Daskilewicz and German [34].](image)

Yet within the domain of architectural design, it is not possible to select the best designs judging solely on their visualized performance for three reasons:

1. Not all criteria can be easily quantified; some are tacitly known to designers, such as the form composition and aesthetics.
2. The reported performance metrics usually reflect only a partial set of all the performance criteria that designers may care about. This set may contain essential metrics, but it is far from complete.

3. The exact performance metrics to calculate can be hard to select at the onset of a design project unless designers rely on precedent cases. Often the performance requirements and preferences unfold and change throughout the process.

For the above reasons, I focus on design space navigation (DSN) systems that involve visual displays of alternatives in the exploration and decision making. These visuals can represent the geometric forms of the alternatives in 2D or 3D views. Involving the design visuals makes it possible to assess the qualitative aspects of the design alternatives (first point above). It can also be used by designers to intuit the (yet) uncaptured performance criteria due to the second and third points.

An early example from the visualization community, although not titled as a DSN tool is EZChooser [104] but its support for visuals is very limited. I turn instead to an extensive list of DSN systems that come from the research or practice of architectural design or other related domains.

These systems include: Dream Lens [63], Refinery [3] and Fusion 360 [2] by Autodesk, Design Explorer [7], Thread [8] and Asterix [6] by Thornton Tomasetti CORE Studio, the Design Space Exploration Framework by Fuchkina et al. [42], Cupid by Beham et al. [19], ParaGen by Chaszar et al. [29], D.Star by Mohiuddin et al. [66], EC-CO-GEN by Marin et al [61], Biomorpher by Harding and Brandt-Olsen [46], and the systems by van Kastel [100], Goguelin et al. [44], and Doraiswamy et al. [37]. Finally, I will also be referring to unnamed systems and publications that tackle using data visualization and analysis in navigating alternatives.

I describe a number of these DSN systems next. Then in Chapter 5, I will analyze these systems and contrast them with DesignSense, which is the system I am presenting in this thesis. The analysis will be conducted in light of multiple lenses such as simplification (clustering and dimensionality reduction), the coupling of design forms and data, expressive selection, collections, information seeking, mode-less navigation. The lenses have been developed throughout the research process I describe in Chapter 3 and they tie back to the goals I outlined in the Introduction (Section 1.2).

2.5.5 Design Space Navigation: Examples of Systems

Design Explorer by Thornton Tomasetti [7] is an open-source DSN interface. It consists of a parallel coordinates plot (PCP), a scrollable and sortable gallery of design visuals, and an inspection window where one alternative can be studied in detail and ranked. Alternatives...
can be filtered on the PCP by drawing a rectangular brush along the desired input or performance ranges. Finally, Design Explorer allows users to zoom into their current selection or exclude it temporally from the dataset (Figure 2.13).

Dream Lens by Matejka et al. [63] uses a scatterplot visualization and query widgets to filtering alternatives based on their data values. The interface presents two ways of presenting and interacting with design visuals. The first is a gallery of sortable juxtaposed visuals that can be filtered by data queries (Figure 2.14 - Left). The second consists of a superimposition of the 3D geometries of alternatives. Users can directly filter/include alternatives that share common undesired/desired geometric parts (Figure 2.14 - Right).

Cupid is a visualization interface by Beham et al. [19] that aims to relate generated 3D geometries with their associated input parameters. The goal is to recognize the parameters
with the largest impact on the geometry, to find the parameter values producing similar geometries, and to identify invalid geometry and the input values that produce them. Cupid clusters alternatives based on their geometric similarity and presents the resulting clusters in an interactive radial tree (Figure 2.15). An augmented PCP is coordinated with the radial tree and supports selecting and inspecting alternatives.

Figure 2.15: The interface of Cupid. Retrieved from https://vimeo.com/102600567 subject to a permitting licence (CC BY-NC-ND 3.0)

The Design Space Explorer by Fuchkina et al. [42] is a DSN system that enables navigating alternatives on three different levels. At the strategy level, users can create a dataflow graph that connects multiple filtering, analysis and visualizations components (Figure 2.16 - Right). The next level involves visualizing the design data. The visualizations supported include a parallel coordinates plot and an interactive self-organizing map (SOM) (Figure 2.16 - Left). Both visualizations allow users to select sets of alternatives and send them as outputs to other components down the dataflow graph. The SOM visualization also enables users to semantically zoom into the dots on the SOM grid, which reveals the visuals of the alternatives they represent. Finally, the third level of the system allows users to inspect the details of a single alternative.
Figure 2.16: The Design Space Exploration Framework by Fuchkina et al. [42]. Interacting with any component in the user-created dataflow graph (right) shows a different view. To the left is a Self-Organizing Map (SOM) view. Both images are retrieved from a video subject to a permitting licence (CC BY-NC-ND 3.0) at https://vimeo.com/243872149

The thesis by van Kastel [100] introduces a visual analytics DSN system based in a 3D digital environment. Design data is encoded geographically, e.g., by using soil layers, terrain variations, and water levels. The alternatives are placed on a hexagonal grid, where each cell is 3D stacked bar whose colours and height represent the alternative’s performance (Figure 2.17). The terrain environment can be transformed in different ways (e.g., raising water levels or creating canyons) in response to user’s filtering. Other navigation-support techniques are also present, like filtering with decision trees, exploring individual alternatives in-person, and bookmarking alternatives.

Figure 2.17: In the thesis by van Kastel [100], users navigate design alternatives by transforming and flying in a 3D digital environment. An accompanying video (at https://www.youtube.com/watch?v=APOsQQ2GMe4) was used for retrieving this image.

D.Star [67] (formerly Design Galleries [14,66,107]) is a gallery-based interface for interactive design space exploration (Figure 2.18). The interface consists of resizable and movable
panels, each containing a collection of alternatives. Each collection can be viewed in one of several ways, such as a parallel coordinates plot or juxtaposed image thumbnails. D.Star is tightly connected to parametric modelling tools (e.g., Grasshopper [15]) that allow users to create new alternatives or modify existing ones on demand without leaving the interface. New collections can be created, and existing ones can be copied or discarded. Alternatives can be copied or moved between collections. Finally, visual cues are used to highlight the representation of alternatives that belong to different panels.

Figure 2.18: The interface of D.Star. A sample of two coordinated panels representing the same collection.

The interface by Doraiswamy et al. [37] allows users to explore a catalogue of building massing alternatives based on the quality of their views (Figure 2.19). Unlike the interfaces I described earlier, this interface shows building alternatives within the physical context where they are expected to be built. To facilitate exploring and filtering alternatives, the interface supports a PCP view and a vertical gallery of forms. Users can also add alternatives into a cart to save their progress.
2.5.6 How Designers Explore Large Design Spaces

Studies on consumer behaviour show that people face a cognitive overload when presented with many choices and can subsequently be less satisfied with their decisions [51]. Generative design techniques can produce hundreds or thousands of alternatives. So how do designers, who both create and choose alternatives, act under these conditions and what supportive mechanisms can be provided to alleviate the overload of choice? This is what the study by Shireen et al. [87] had set out to discover.

The experimental setup was aimed at creating the conditions for the cognitive overload of choice. To achieve that, a group of designers were each asked to explore a large number of generated residential building’s designs, which were depicted graphically on physical cards. The alternatives were first reduced in number by computing the pair-wise similarity between alternatives and keeping the most dissimilar 1000 among them. Finally, designers were placed in an open-ended simulated design scenario and were asked to "go through the designs and set goals for the next design phase" [87].

The subsequent analysis by Shireen et al. [85] highlighted actions on the design alternatives (grouping, comparing, discarding, marking...) and patterns of how designers utilized their physical workspace. Finally, this analysis was used to design a mock-up DSE interface [84] and a list of "spatial metaphors" that can aid the design of digital gallery-like design exploration tools [86]. The metaphors are broken down into three categories: Divide, Place and Mark:
1. **Divide:** Designers divide alternatives to collections such that each collection expresses a criterion. As the designer’s criteria evolve, so do the collections which get split, merged and discarded. Designers also divide their workspace into zones giving each a different function, much like a cook does in their station.

2. **Place:** Designers place alternatives spatially in ways that enable them to sort, scan and compare alternatives.

3. **Mark:** Designers mark their alternatives with bookmarks, tags, notes and ratings. These marks allow designers to add a layer of information to the design space and express their interest in portions of it.

The above spatial metaphors allow us to map the patterns observed in the study, which occurred in a physical setting, to the digital medium of interfaces. The study did not involve any quantitative data like performance metrics. The inclusion of performance data, however, is essential to the ability to filter, aggregate and quickly compare alternatives. This is especially true when exploring a large collection of alternatives. Nevertheless, I rely heavily on the results of this study for a simple reason: to the best of my knowledge, it’s the only current account of designers’ behaviour when navigating a large number of design alternative. Thus it acts, along with interactions with designers, as part of the domain problem characterization [69].
Chapter 3
Research Methodology and Process

This research project followed an iterative process reflecting that of a design study methodology [82]. A design study is a project in which a real-world problem, specific to some domain, is analyzed by visualization researchers to design a visualization system that supports solving the problem. A design study also involves validating and reflecting on the developed system, thus contributing to the knowledge surrounding the domain problem in question.

Figure 3.1: The research process comprises of iterative cycles of elicitation of the domain problem characteristics, consolidating the learned lessons, and developing prototypes that are guided by these lessons. Evaluation is a continuous activity combining the three major process components.
The process followed in this thesis is shown in Figure 3.1. The first part of the process aimed at characterizing the domain problem of Design Space Navigation (DSN). This was accomplished through a combination of:

- A literature review of systems and theory on both Visual Analytics and DSN.
- Eliciting expert’s feedback with an early DSN prototype (1st iteration)
- Referring to previous characterizations of the domain (Shireen et al. study on design exploration [87]).

The characterization lead to two components. First is a task abstraction describing the tasks supported by visualizations in a DSN interface (Section 6.1). The second is an in-depth analysis of current DSN interfaces, which lead to identifying opportunities for improvements on these interfaces (Chapter 5).

The third component of the process was ongoing from the start, and it consisted of developing prototypes that instantiates what is learned about the domain problem. These prototypes were informally evaluated by domain experts ending with a focus group evaluation (Chapter 7). The descriptions in the following two sections are mostly part of the domain problem characterization while the latter consists mainly of the design and development of a solution.

### 3.1 Design Analytics Case Study

This project fits in a larger research program investigating design analytics which is broadly studying the integration of interactive data analytics with design exploration. The beginning of this research program started in SmartGeometry (2018) *Inside the Black Box* cluster [95]. The cluster involved a group of 19 participants who were designers, programmers, and visualization developers. The group worked on the design of a mixed-use high-rise building located in a city context with a range of real-world form and performance concerns, such as land-use, functional-spatial distribution, heat loss and gain, solar exposure, view quality, etc. The five designers in the group developed six different parametric models for a mixed-use high-rise tower design and generated alternatives from them. The generated alternatives were then used as the basis of a case study on the application of visual analytics to design data coming from a realistic and collaborative setup.

The case study started with using off-the-shelf commercial visual analytics tools like Tableau [5] and PowerBI [4] to explore the data generated from the case study. The exploration lead to identifying some scenarios that demonstrate the utility of interactive data visualization in assessing the validity and reliability of parametric design models. For more

---

1To give some context, there were 250 alternatives each evaluated by 13 different performance metrics.
details on the case study and its findings, I refer the reader to the published article by Erhan et al. [38].

Of particular interest to us here, the case study highlighted the need for incorporating design forms into visual analytics tools. For example, performance data was sometimes misleading and imprecise, such as a scenario where design alternatives held the same value for one performance metric but had drastically different design forms (illustrated in Figure 3.2). This finding motivated me to focus my attention on design space navigation (DSN) systems that involve the design visuals (or design forms) in navigation and decision making.

![Figure 3.2: An example highlighting the need for coupling form and data. (A) The same parametric model, similar view and height but the top alternative form is malformed. (B) The same parametric model, similar view despite very different heights. (C) Different parametric models, similar view and height despite differences in the building’s shells.](image)

### 3.2 First Iteration

Motivated by this need, I went back to the literature on DSN systems from the architectural design domain. The systems I have surveyed were characterized by presenting both design data and forms. Furthermore, some of these systems acknowledged the choice overload problem in generative design. These included [7, 19, 25, 30, 40, 42, 45, 63, 66, 74, 100]. I have then analyzed these systems through different lenses like:

1. Identifying the ways in which design forms and data are represented, and the interactions supported on their representations.

2. Recognizing degrees of coupling between the representations of the alternatives’ data and those of their forms.

3. Identifying the role that simplification techniques, i.e., clustering and dimensionality reduction, played in service of design space navigation.
Some of the lenses above were motivated by a priori hypotheses about gaps in current DSN interfaces, and some evolved in an open-ended fashion. The detailed results of this analysis are published in a separate article [11], but the main results are:

1. A better understanding of the role that simplification techniques played or could play in design space navigation.

2. Recognizing the secondary treatment of design forms in the surveyed systems. This is in terms of the lack of interactions that can be initialized from the representations of design forms. Examples of such interactions, which I collectively referred to as form-first interactions, include: querying designs with similar geometric features or selecting alternatives based on their visual appearance.

3. A conceptual model that can be used to analyze DSN systems using the same lenses I outline above (The model is shown in Figure 3.3).

4. A prototype of a DSN dashboard that exemplifies some of what I learned through the analysis. I leave the details of this prototype to Chapter 4.

Figure 3.3: A conceptual model of the coupling of design data and forms in a visual analytics context. The Form and Data similarity is derived by computing the distances between design alternatives. Examples of views for each are shown in the same coloured block. [Retrieved from a previous publication of mine [11]]

I use the term "design space exploration" in that paper but I have favoured "design space navigation" here due to the disparate senses of the term exploration.
This first prototype incorporated representations of both design data and forms and demonstrated an integration of design space simplification (hierarchical clustering in this case) into a DSN system. To solicit feedback on the prototype, the dataset of generated alternatives from the case study in the last section was visualized using the prototype. It was then presented to two groups of professional designers and computational design leaders from two different architectural design firms. The prototype was also presented to some of the members of the Computational Design research lab at Simon Fraser University.

The formative feedback was generally positive, but it highlighted two crucial mismatches. The first is that a higher degree of coupling was needed between design form and data than present in the prototype. The second is the call for a more intuitive inclusion of simplification in DSN, possibly with design form visuals augmenting it.

3.3 Second Iteration

Based on the feedback I received, I continued exploring the question of providing a "tighter coupling between form and data". In parallel, I have continued exploring other visual analytics systems like Spotfire [9] and Gephi [18] in addition to widening my exposure to other DSN systems that met the same criteria of the first round of analysis. Furthermore, I started to widen my design criteria for the next iteration of a DSN system through the analysis done by Shireen et al. [87] on the behavioural patterns observed by designers when navigating large design spaces.

I could now list four high-level design requirements for the second iteration of this project:

1. Addressing the choice overload problem that arises when exploring a large number of design alternatives (also voiced by [24, 39, 40]).

2. Providing a tighter coupling between the visual forms of design alternatives with their performance data (similar calls by [16, 29, 41, 42, 63, 74, 100])

3. Guiding the design of DSN tools with techniques from the field of Visual Analytics.

4. Supporting the full range of the actions performed by the designers in the study by Shireen et al. [87].

I have analyzed several DSN interfaces (the full list is in Section 2.5.4) through lenses shaped by these requirements. These lenses extend the one I used in the first iteration. The result of the analysis (Chapter 5) gave me ground for designing the second prototype, which I called DesignSense. Finally, a formative focus group evaluation was conducted on this prototype with a group of professional designers and computational design experts.
I present the results of the second iteration in Chapter 6, a focus group evaluation was conducted on this second iteration and I describe it in Chapter 7 and discuss its results in Chapter 8.
Chapter 4

First Iteration

In this chapter, I describe the first prototype that was developed based on the initial analysis I did on the literature of DSN in architectural design [11]. This chapter is intended to be read as a self-contained report on that prototype.

I have chosen four different visualizations in a way that reflects what I thought was interesting to explore individually and in concert with each other. In particular, I have a combination of data views (scatterplot, parallel coordinates plot), form views (grid of 3D views of forms) and a simplification-based view (dendrogram representation of hierarchical clustering). Some of these visualizations were part of the DSN systems I surveyed when creating the prototype [7, 19, 40, 42, 63, 100], but not in the same combination presented here. This chapter will present these visualizations and argue for the design decisions I made and suggest potential directions for the future. Finally, the complete prototype can be seen in Figure 4.1 below.
Finally, I am using the Why-What-How framework from Brehmer and Munzner’s typology [23] to structure this chapter. *Why* is a question about the purpose of the visualization, *What* is about the data to be visualized and *How* is about the visual encoding and interaction techniques employed on the data to achieve the *Why*.

### 4.1 What

Parametric design modellers have become widely used in the practice of architectural design. A designer working with these tools (e.g., Grasshopper [15] and Dynamo [1]) describes an algorithm that generates geometrical objects (known as forms in the architectural domain) and specifies the constraints that these forms are subjected to. These algorithms can produce a variety of forms when presented with different input parameters. For example, an architect may generate a tower building by mapping input parameters, such as the height of the building and the dimensions of its base, into a collection of shapes and curves that would define the tower’s geometry.
In addition to the input parameters and geometry of each design alternative, we can also compute its performance metrics. For example, we can calculate the total surface area of a building or the number of floors it can accommodate.

To evaluate this prototype, I used a dataset of tower buildings that were generated by six designers during a Smartgeometry workshop [95]. This is the same dataset resulting from the Design Analytics case study in Section 3.1.

In the context of architectural designs, these geometric forms are used for exploring the possible ways that a building could look at a low resolution. The dataset is generated through Grasshopper, a scripting framework that works with the 3D modeller Rhinoceros. A program in Grasshopper looks like that in Figure 4.2. Each program can be called a model or a generator.

The dataset contains a table, trees and a collection of 3D models. The table, as in Figure 4.3, contains unique IDs (categorical attribute, an alphanumeric string where the first letter comes from the name of the model that the alternative was generated from), input parameters (quantitative attributes, these are model-specific), performance metrics (15 quantitative attributes, shared between all the alternatives), and name of the generator/model (categorical attribute). Some of the performance metrics had null values, and these were replaced with zeros.
The 3D models are OBJ files, where each corresponds to a single design alternative. Samples of those are found in Figure 4.6. The trees are dendrogram trees that are created through a hierarchical clustering algorithm. The clusters are formed based on pair-wise distances between the alternatives. The distances are, in turn, found based on the input parameters, performance metrics, and the geometry of the alternatives. In the first two cases, the Euclidean distance is used. In the latter case, I used the Hausdorff distance between the vertices of the compared geometries. This process results in three trees (dendrogram trees). The argument for deriving these trees is made in Section 4.3.4.

In total, the dataset I used in evaluating this prototype contained 250 design alternatives from 6 different parametric design models, and each alternative was evaluated against 12 different quantitative performance metrics.

4.2 Why

When designers work with parametric modellers, they generate many designs by varying the input parameters. In doing so, they are exploring a design space of potentially thousands of alternatives. This system is a design experiment for exploring the potential for using Visual Analytics to make sense of this design space. The literature on how designers explore large generated design spaces [87] and the previous systems addressing a similar problem [7, 19, 42, 63, 100] informed the initial goals of this system. Broadly speaking, the designer’s task is to obtain a limited set of alternatives given many of them. The selection process is expected to require an overview/summary of the data and detailed identification of and comparison between the alternatives. I will revisit the detailed purpose of each component in the system as I introduce them. I will also borrow Brehmer and Munzner [23] abstract tasks typology of summarizing, identifying and comparing to support that.

\[1\]

This gave an estimate of the overlap between the buildings' geometry.
4.3 How

In this section, I will briefly introduce each of the components in the system. All the dataset is shared between them, and they are linked using unique identifiers. For each component, I will motivate its use and justify its visual encoding.

4.3.1 Scatterplots

Scatterplots allow designers to identify correlation patterns and outliers in the data. An outlier with an interesting performance or form can take the design process in new directions. We can also use the scatterplot to identify the alternatives lying on the Pareto frontier. Once a selection is made through brushing, the axes can be changed while preserving the selection allowing us to examine the selection across dimensions. As Figure 4.4 below shows, each dot is an alternative, and the colour of a dot is based on that alternative’s generator. Since each alternative is generated from a single generator, these colours are mutually exclusive categories. The size of the dots is fixed to reduce the visual clutter resulting from competing visual encodings. Humans are also not as sensitive to changes in area/size channels as they are to position or few distinct colours [70].

![Figure 4.4: A scatterplot.](image)

4.3.2 Parallel Coordinates Plot (PCP)

The parallel coordinates plot is effective as an overview of multidimensional data [70]. It enables the designers to compare alternatives across many dimensions and filter them to specified ranges. Each vertical axis is a performance dimension, and each line is a single alternative. The colour of each line is based on the generator/model of the alternative. The axes can be reordered to address a problem in PCPs where correlations between dimensions become visible only when the axes are juxtaposed. Filtering can be performed on each axis
by drawing a rectangle (as in Figure 4.5). Upon filtering, the lines that are not selected are
greyed out to bring the selected ones to focus.

Figure 4.5: A parallel coordinates plot.

4.3.3 3D Views Grid

Not all aspects of a design can be quantified, for example, its form composition or what
makes it visually appealing. It is then important for designers to have visual access to the
design forms to identify these qualitative aspects and compare the designs forms with each
other. To support that, a grid of 3D views is filled with the currently selected alternatives.
The designers can zoom and rotate the camera in these views simultaneously. Finally, these
views can be highlighted with distinguishable outlines (Figure 4.6).

Figure 4.6: A grid of 3D views of design forms.
4.3.4 Dendrogram Tree

Generative design is an approach for automatically generating design alternatives. The goal is to find alternatives that satisfy the designers’ requirements by varying the relevant design parameters. This, however, can often result in a large number of alternatives with diverse forms and performance portfolios. When designers attempt to make sense of this space of alternatives, they can suffer from a cognitive overload of choices [24, 39]. Previous studies have shown that when exploring a large design space, designers create collections of alternatives to express their design criteria [87]. The creation of collections allows designers to gradually make sense of the space of alternatives by putting order into it.

The dendrogram tree in this prototype represents the results of the hierarchical clustering of the dataset (Figure 4.7). The leaves in the tree represent alternatives, and the texts are their IDs. These leaves are connected to nodes that represent the clusters they belong to. These nodes, in turn, are also connected to fewer nodes, which act as superclusters (a cluster containing smaller clusters). This process is repeated until the root is reached. The numbers on the non-leaf nodes represent their depth or distance from the root. Selection can be made on both leaves and non-leaves and upon selection, the color of the nodes is changed. Users can change the current clustering criterion which changes the presented tree to one of the three pre-computed ones.

Figure 4.7: A dendrogram tree with alternatives at the leaves.
The dendrogram\textsuperscript{2} can be used to select or compare sets of similar alternatives. For example, by clicking on one of the cluster nodes, users can examine a set of similarly performing alternatives (if performance was the cluster criterion) or similarly looking alternatives (if the criterion was based on geometric features). By examining the members of the cluster in the grid view, designers can decide if they need to drill down into a subcluster, for example, upon observing much variance or incoherence in the currently examined one. Designers can sequentially examine clusters at the same depth in the tree (i.e., sibling nodes) while looking at the grid view or any data view. By doing so, they gain an overview of the different families or categories in the design space (again depends on the clustering criterion).

4.3.5 Interaction

When designing the interaction in this prototype, the main intent was to facilitate communication links between the four views as they represent different aspects of the data. When alternatives in one view are selected (via brushing) on a view, then their representations in the other views are highlighted. I have favoured highlighting, rather than filtering, to maintain the context of the highlighted alternatives with respect to the unhighlighted. The exception is that I filter the alternatives on the scatterplot based on brushing on the PCP. This is intended to reduce the quantity of the alternatives. It also assumes that there will be hard constraints on the performance metrics that justifies that (e.g., Energy Efficiency must be above 80\%). This filtering feature is also intended to force a discarding mindset when something does not work, potentially reducing the choice-overload. Finally, even the low performing alternatives can be a source of insight for a designer. This made me reluctant about using filtering instead of highlighting. But I would argue that when the system supports the manual creation of collections, then the hard constraints can be removed once a line of inquiry has been saved as a collection.

In addition to the brushing interactions, the Tree view also supports collapsing/expanding branches of the tree to reduce its complexity and reduce the visual clutter. It also supports dragging nodes (leaves or non-leaves) into different parent nodes, allowing the designers to rearrange the hierarchy if deficiencies in the hierarchy were found. As a result, the tree can also be used to form manual collections. Figure 4.8 below illustrates these interactions.

\textsuperscript{2}The original implementation of this interactive dendrogram tree is by Robert Schmuecker at https://gist.github.com/robschmuecker/7880033
Figure 4.8: Interactions on the dendrogram tree using an artificial dataset. Top: clicking on a cluster node selects its members. The blue nodes (e.g., optimization) show collapsed branches. Middle: by dragging a node, users can move leaves or complete branches in the tree to different parents. Red circles around each target node show the range to which dragged nodes should be moved to belong to that target. A red link shows the target node that the dragged node will belong to once dragging is completed. Here the leaf (BetweennessCentrality) is moved to a new parent (cluster). Bottom: the tree after dragging is completed.

A design decision was made to start the grid with no form views and fill up the grid upon selection instead of filtering views down to the size of the selection. The latter approach was taken in Dream Lens [63]. This decision was made initially for technical optimization purposes, but it also enables an interaction that couples the performance and forms of alternatives. In particular, designers can move a rectangular brush on the scatterplot view (as in Figure 4.1, view B) gradually along one of the axes while updating the grid with the currently brushed alternatives only. This gradual movement demonstrates how the geometric forms are changing in relation to changes in performance metrics. Figure 4.9 shows the current set of interactions between views. All views except for the grid support brushing on them (the outcome of brushing are described Figure 4.10). When the 3D view of an individual alternative is hovered over, then its representation is highlighted in the other views.
Figure 4.9: The brushing interactions between the components of the interface.

<table>
<thead>
<tr>
<th>When brushing on</th>
<th>Scatter Plot</th>
<th>Tree</th>
<th>Grid</th>
<th>Parallel Coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scatter Plot</td>
<td>A rectangle shows the brushed area. Dots outside the rectangle are made transparent</td>
<td>The nodes of the selected alternatives (which will be leaves) are colored</td>
<td>A 3D view is created for each selected alternative</td>
<td>The lines of the selected alternatives are made thicker, and the rest are made transparent</td>
</tr>
<tr>
<td>Tree</td>
<td>The dots of the selected alternatives are left opaque; the rest becomes transparent</td>
<td>When a node is selected, all its children and the node itself will be colored</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grid (only hovering)</td>
<td>The dot of the hovered-over alternative is given a distinctive outline</td>
<td>The node of the hovered-over alternative is given a distinctive outline</td>
<td>The thumbnail of the hovered-over alternative is given a thicker outline</td>
<td>The line of the hovered-over alternative is made thicker</td>
</tr>
<tr>
<td>Parallel Coordinates</td>
<td>The unselected alternatives are filtered out, and the axes of the Scatterplot are updated accordingly</td>
<td>The nodes of the selected alternatives (which will be leaves) are coloured</td>
<td>Nothing</td>
<td>The unselected are made transparent</td>
</tr>
</tbody>
</table>

Figure 4.10: Details of the interaction in Figure 4.9. Using the same colors.

4.4 Why (revisited)

Here, I will list potential concrete scenarios of use given the above visualizations/interactions:
• **Identify** the set of alternatives that has a height in the range between 150 and 350 meters. (Filtering on PCP or the scatterplot))

• **Identify** alternatives with abnormal performance metrics (zero or null values) then **identify** the combination of the generator and the input parameters that lead to them. (First through the scatterplot, then through the PCP’ Model axis). This use case suggests that a table view showing the raw input values can be useful.

• **Compare** the different generators through their performance on Energy Use (select a generator category on the PCP then compare them via the scatterplot or PCP views)

• **Compare** the different generators through the appearance of their respective alternatives (Select generator category on the PCP then compare through the grid view)

• **Summarize** the diversity of a generator with respect to the performance or the forms of their alternatives. (Judging from how scattered the leaves of the alternatives are across the dendrogram trees)

• **Identify** a set of similarly performing alternatives (using the tree built from performance distances) but which are more diverse in their appearance (using the tree built from forms distances).

• Starting from a set of 3D views, **identify** the most promising alternatives based on qualitative criterion (using the selection on the grid view) then **summarize** their respective performances (using scatterplot or PCP views).

• **Identify** the alternatives on the Pareto frontier of Heat Loss vs. Surface Area (using the scatterplot) then **compare** their appearances (on the grid view) or how they compare with respect to other performance criteria.

• **Identify** outlying alternatives in terms of their Energy Use performance and **summarize** their input values for the purpose of generating more or less of these alternatives (using the scatterplot with an Energy Use axis to identify outlier then changing axes to different inputs parameters)

• After **identifying** a deficiency in the hierarchical clusters formed based on the similarity in the form (using a selection on the Tree then examining the grid view for dissimilar alternatives that were put in the same cluster), rearrange the tree branches to better reflect the similarities in the forms.

### 4.5 Further Work

The following are suggestions for the future that are made by reflecting on features supported in the past DSN systems in the literature, and through feedback from colleagues
of mine. The next section present a formative evaluation of the prototype with domain experts.

1. Grid View

(a) Linking the grid view to the rest of the views by
   i. Providing a way to show which 3D views in the grid view belongs to which dot in the Scatterplot or line in the PCP.
   ii. Allowing selection on the grid view that is reflected on the other views.
(b) Optimizing the construction of the 3D views so they could be built quickly on slow machines.

2. PCP View

(a) Allowing more than one rectangle when defining filtering ranges on an axis. This is especially necessary for categorical data like the generator/model.
(b) By default, the lines in the PCP do not provide precise numeric data and the values can be approximated using the nearest tick mark. Provide a way to get precise numeric details when demanded (e.g., when hovering over a line).

3. Scatterplot View

(a) Distinguish the Input and the Performance axes in the dropdown menu.
(b) Provide a faster way to change axes than the dropdown menu.
(c) Multiple rectangles selection
(d) Lasso selection for a more fine-tuned selection.
(e) [Requires derived data] Have a special visual encoding for alternatives on the Pareto frontier when demanded.
(f) For datasets with multiple generators, the scatterplot currently does not support input parameters axes because each generator might have different input parameters. When the user selects a single generator/model in the PCP, provide its respective inputs. This is a detail-on-demand pattern.

4. Tree View

(a) Coloring the nodes is not a sufficient highlight when the tree is large and zoomed out. The colored nodes will become too small in that case that they are not distinguishable. Explore an alternative way of highlighting.
(b) When branches are collapsed, information about whether it included highlighted nodes is lost, explore a way to preserve it.
(c) Allow for an accumulative selection on the tree branches.
(d) [Requires large displays] Juxtapose multiple tree views instead of alternating between them using a dropdown menu.

(e) More generally, the tree is currently very raw and can support much more interactions and visual encodings, refer to sources like https://treevis.net/ for inspiration.

5. General Concepts

(a) Allow a way to manually create collections in a way that is informed by selections on any of the views. These collections can intersect or be orthogonal in nature to the clusters in the Tree view.

(b) Building on the previous point, a potential goal but which was not explored fully in this system are the tasks of summarizing, comparing and identifying collections (instead of alternatives).

(c) When a selection set X is made in a view it will be highlighted in the other views (as indicated in Figure 4.3) but when moving on to making a selection Y in the other views then Y will be a subset of all the data instead of building upon the previous selection and being a subset of X.

(d) Explore how the different views would scale up to a large number of alternatives, in terms of visual clutter and performance.

4.6 Formative Evaluation

To solicit feedback about the prototype, the dataset of generated alternatives from the previously mentioned case study (Section 4.1) was visualized using the prototype. The prototype was presented at different times to groups of participants consisting of computational designers and design professionals (from two different firms) and some of the Computational Design research lab members at Simon Fraser University.

The presentation took the shape of a tour over the interface features, followed by giving example scenarios of using the different components of the interface. The scenarios included brushing on the PCP or Scatterplot views and showing the impact of that on the Grid; interactions on the dendrogram tree (brushing, dragging, and collapsing); verifying that selecting a cluster in the performance-similarity dendrogram shows closely bundled lines on the PCP and that the visual appearance of the alternatives on the Grid is similar if the form-similarity tree was used.

The prototype provided multiple trees, each created based on a different similarity criterion. Participants appreciated the ability to query designs having a similar performance or form by selecting subclusters in the tree and then seeing those highlighted in the other views (scatterplot, PCP or 3D grid). Some of the participants were familiar with brushing on the PCP (by using the Design Explorer [7]) and saw the scatterplot as a nice addition.
The formative feedback highlighted two crucial mismatches. The first mismatch is that a higher degree of coupling was needed between design form and data than present in the prototype. For example, participants could not identify which dot in the scatterplot corresponded to which 3D view (e.g., through a common visual encoding). Furthermore, the 3D views grid were only presenting design forms but did not support highlighting an individual alternative\(^3\) or selecting a set of alternatives. Sorting alternatives based on performance metrics was also not possible.

The second mismatch of the prototype comes from the hierarchical clustering representation. The concept of using clustering was exciting to many, but they faced a challenge in interpreting the tree structure, such as the meaning of a child/parent relation in it. Some suggested augmenting the tree with visuals of design forms to help them navigate the tree and associate the visual forms of the alternatives with the similarity criterion of the tree (e.g., verifying that members of the same class look similar).

Participants asked if clustering could be performed based on chosen performance metrics rather than all of them. The hierarchical clustering results in this prototype were computed in an external script after a few pre-processing steps. This made it hard to customize the clustering process on-demand. In summary, clustering was sought to be more flexible and to be more tied with design forms.

As mentioned earlier, the evaluation started with presenting the interface features, followed by a few potential usage scenarios. Due to being a feature-based evaluation, no conclusive statements can be made about how exactly the dendrogram tree (or other components for that matter) could be utilized by users when performing design space navigation tasks. However, the goal of this formative evaluation was to guide my design process and provoke some discussion with the participants. The feedback I received on the prototype’s features did that.

\(^3\)However this features is present at the time of writing, as Figure 4.10 indicates.
Chapter 5
Design Space Navigation: Lenses

In this chapter, I present a critical analysis of existing DSN system features to guide the design of new DSN systems. As complex systems, DSN interfaces can be described and analyzed from multiple perspectives. I developed and used a set of lenses based on the research problem I aim to address in this thesis. I developed these lenses considering the research (c.f. Chapter 3) and design goals emerged through an iterative process. The two of my most salient goals are to reflect and support the way designers explore and make decisions, and to address the critical challenges that hinder Generative Design. The lenses, therefore, bridge the research goals and the prototyping of DesignSense. The seven lenses I used in the DSN systems analysis are: simplification, coupling form and data, expressive selection, collections, information seeking, and mode-less navigation.

The analysis of DSN systems is interwoven with the results of the experiment conducted by Shireen et al. [87] on the navigation of large numbers of generated designs. The outcome of the experiment includes a full range of actions observed when designers work with a large number of computer-generated design alternatives. A subsequent analysis of the study [86] also reveals that there is a mismatch between the tasks designers perform when navigating generated alternatives and the system features in existing DSN systems. The authors also suggested multiple spatial metaphors to map these tasks to DSN interface features. However, there is a gap between the results of the study (which were based on exploring design forms only) and DSN interfaces (which represent both design forms and performance data), so to mend this gap I rely in the analysis on concepts from Visual Analytics.

In this chapter, I give a brief explanation for each lens and analyze example systems’ features found in the existing DSN systems under each lens. I also identify a list of system requirements based on that analysis. Finally, I foreshadow how these requirements shaped DesignSense.

I briefly describe some of these in Section 2.5.5 with images.
5.1 The Lens of Simplification

Both Erhan et al. [40], and Brown and Mueller [24] argue for simplifying design spaces to alleviate the choice overload fatigue created by working with a large number of alternatives. Simplification can have the effect of reducing the design space or creating order in it. In the literature of DSN, simplifying the design space have been achieved through various types of clustering [19, 25, 30, 40, 46, 75] or dimensionality reduction techniques such as Multidimensional Scaling [62], Self-Organizing Maps (SOM) [42, 45, 74, 100], or Principal Component Analysis (PCA) [59]. Brown and Mueller [24] categorize Pareto analysis (Section 2.4.3) as a simplification technique as well, in that it reduces the design space to a set of well-performing designs.

In a separate article [11], I survey multiple DSN tools and identify the roles that simplification played in them; these include:

1. Alleviating the choice overload.
2. Creating visual overviews of the design space.
3. Opening up new venues for interrelating design forms and data (similarity-based coupling, see Section 5.2.7)
4. Introducing new ways of querying alternatives through form-driven criteria (e.g., selecting all the alternatives sharing similar geometric features).

The first prototype that I developed involved simplification in the form of hierarchical clustering (see the dendrogram tree in Section 4.3.4). The feedback on the prototype highlighted the need for flexible and integrated simplification when it is used for DSN. Flexibility is necessary because the designer’s needs can change during navigation. There are multiple ways to achieve flexibility:

1. Choose the subset of the design space to simplify.
2. Choosing the aspects to simplify based on (e.g., by inputs, performance, or geometry).
3. [in case of simplifying by inputs or performance metrics] Choosing the data dimensions to simplify by. For example, choosing to cluster rectangles by their height, width or both as illustrated in Figure 2.7.
4. Changing the simplification technique as needed.
5. Changing how the results of simplification are represented.

The second component for simplification in the service of DSN is integration. Designers’ criteria are continuously changing as they navigate, examine and compare alternatives. The choice for when and how to simplify should be made during navigating and not before
it. Take this (common) scenario as an example: A designer wants to use clustering or dimensionality reduction for any of the advantages they offer. They use a machine learning or data analysis software library to perform either, they save the results and then import it into a DSN interface where they can then visualize and interact with the results.

What is missing from this picture is that simplification algorithms often have multiple hyper-parameters that require tuning. The only way to tune them or to customize them to the designer’s needs is to repeat the process of setting the parameters of the algorithms, running them, and importing their results. There is a gap between recognizing a specific need for simplification and the ability to actualize this need. In other words, the gulf of execution is wide [49]. By bridging this gulf through integrating simplification into DSN, designers can make full use of the flexibility examples in the list above. Figure 5.1 illustrates a possible interplay between navigation and simplification.

Figure 5.1: When clustering is integrated with navigation then a cycle can take place between the two. For example, designers can start by selecting a cluster of alternatives (left), then upon looking at their performance (right) they may pick the highest performing “green” designs and cluster them.

A major challenge for both flexibility and integration is the computational cost of simplification. A high cost can hinder the navigation process. This challenge can be addressed by preferring simpler but sufficiently effective simplification techniques. I argue that the fluidity of simplification should be favoured over accuracy in situations such as these where the design criteria are in flux. Finally, it might be preferable to offload heavy computations to different processing threads or to cloud servers, although this is not always viable.
5.2 The Lens of Form and Data

The literature on design exploration and analysis argues for coupling the qualitative and quantitative characteristics of design alternatives during their evaluation, comparison, filtering, etc. [16, 29, 41, 42, 63, 74, 100]. My analysis of the DSN interfaces distinguishes the parts of these interfaces that represent design forms (form views) and those representing design data like performance metrics and input parameters (data views). I present examples of each separately in the next sections then discuss their combination.

5.2.1 Data Views

DSN systems use abstract data visualizations for selecting alternatives based on their performance. For example, parallel coordinates plots [6, 7, 19, 37, 42, 66] or two-dimensional scatterplots [2, 54, 63] are used separately or together for supporting multidimensional data visualization [3, 8, 29, 44]. Linking multiple visualizations have the advantage of opportunistically using either for the tasks at which they are more effective [80].

5.2.2 Form Views

Galleries of design forms can be seen in multiple DSN interfaces. These include interactive (e.g., scrollable, sortable) grids of thumbnail images [7, 29] or 3D geometries [63]. More advanced galleries, like Pivot by Microsoft Live Labs 2 (later inspiring commercial systems like Zegami 3), allow users to structure visuals by categorical variables, filter them with dynamic data query widgets, and semantically zoom between the full gallery and the individual visuals. Shireen et al. [87] (c.f. Section 2.5.6) has shown the role of grouping alternatives based on form expressions as a means for expressing design criteria. The mock-up interface proposed following this study, ParaXplore, is another example of a gallery-like tool for exploring design forms [84]. ParaXplore demonstrates a possible translation of the actions frequently performed by designers to concrete interface features [84].

Exploring a large collection of images is akin to form exploration in design and has been studied in other domains (c.f. [77] for a comprehensive review). As noted in this review, techniques like clustering and dimensionality reduction can be used as a basis for visualizing and exploring images. This was my observation as well when analyzing the literature on DSN tools.

Visualizing design forms can start by applying simplification techniques on the navigated dataset. This is followed by embedding 2D or 3D visuals of the forms in the simplification results. For example, form visuals can be embedded in the spatial grid resulting from Self-Organizing Maps (e.g., Figure 2.16 from Fuchkina et al. [42]), or the node-link radial tree

2https://www.microsoft.com/silverlight/pivotviewer
3https://zegami.com/
resulting from hierarchical clustering (e.g., Figure 2.15 from Cupid [19]). Alternatively, we can cluster design alternatives and obtain the representative of each cluster, e.g., by revealing the closest alternative to the cluster’s centroid. The design forms of the representatives can then be visualized in thumbnail galleries [59] or a rectangular grid [46]. Finally, design forms can also be integrated into data views through visual links [62] (Figure 5.2) or by replacing the abstract graphical elements (e.g., dots in a scatterplot) with the visuals themselves (see the Composite Scatterplot Matrix by Cupid [19] in Figure 5.3).

Figure 5.2: Form views are integrated with data views via visual links in the Design Galleries by Marks et al. [62]
Form views are integrated into a scatterplot matrix in Cupid [19]. The three rectangles are different scatterplots. Each colored dot is an alternative and the enlarged thumbnails are chosen to be from different clusters.

5.2.3 Coupling Form and Data

Providing a logical and visual coupling between form and data is a major goal of this thesis. I propose three analytical viewpoints concerning the coupling of form and data in DSN interfaces: structural relation between their representations, the scope of the coupling (namely: whether it operates over a set of alternatives or a single alternative), and the nature of the criteria that can be expressed via their coupling.

5.2.4 Structural Relation

Multiple patterns exist for combining form and data views [52], for example juxtaposition (e.g., Dream Lens [63]), superimposition (e.g., colouring office floor plans in Project Discover [71] based on performance metrics in Figure 5.4), nesting (e.g., visual thumbnails inside SOM grids [42]), overloading (e.g., forms’ details window inside PCPs in Cupid [19]), or integration (e.g., Design Galleries [62]). Javed and Elmqvist [52] discuss the advantages and disadvantages of each pattern from a visualization design perspective.

5.2.5 Coupling Scope

The third viewpoint when analyzing form and data coupling is the scope of their connection—namely, individual alternatives or sets of alternatives. The examples in the literature utilize one or more of the following strategies: (1) coordinated inspection of a single alterna-
tive; (2) detailed inspection of a single alternative; (3) data-driven selection of alternatives; and (4) form-driven selection of alternatives.

In coordinated inspection, the representations of an individual alternative in both data and form views are visually linked, often by giving them a distinctive and common visual encoding [3,37,63,66]. This allows users to cognitively relate the quantitative and qualitative aspects of each alternative. The second strategy (detailed inspection) uses an enlarged 2D image or/and an interactive 3D view to graphically represent a design form for a single alternative [7,8,42,61,63]. These are often accompanied by presenting data in a textual format (e.g., for cognitive coupling). Figure 2.13 shows coordinated inspection in D.Star and Figure 2.13 illustrates detailed inspection in Design Explorer.

I also found examples of supporting multiple 2D visuals per each alternative [7,42]. For example, each 2D visual captures the 3D geometry from different angles or different visualizations of performance metrics on 2D form views as in Figure 5.4 by Nagy et al. [71]. Finally, the DSN system by van Kastel [100] allows users to inspect an alternative’s form in a 3D environment inspired by digital video game design.

Figure 5.4: An example of different 2D views for a single office layout alternative from Project Discover [71]. Each view visualizes a different performance metric. From the left to right: adjacency preference, work style preference, buzz, productivity, daylight, and view to outside.

In data-driven selection, as the third strategy, users can select alternatives on data views. The selections can be reflected in form views. A common variant of this strategy is to filter down a gallery of design forms based on selections on a scatterplot [2,54,63] or a parallel coordinates plot [7,29,37].

The fourth strategy, form-driven selection, is not covered in the literature as extensively as others. Notable exceptions include the chisel tool in Dream Lens [63], which enables
filtering alternatives by directly interacting with geometric features of superimposed forms of a set of alternatives (shown in Figure 2.14).

Another example is the radial tree in Cupid [19], where the nodes in the tree represent subclusters in the design space. The users interact with nodes and branches, and that results in brushing or highlighting subclusters. I consider the radial tree as an example of a form-driven strategy as well, because the tree is built based on the clustering of alternatives considering their geometric similarity. In other words, interaction with the tree results in exploring families of design alternatives that share similar visual or geometric features.

The chisel tool is largely problem-specific and can generalize only for the problems that share similar characteristics to the case study demonstrated by [63]. The radial tree of Cupid relies on computing the similarity between geometric forms, which can be computationally demanding and problem-specific. Nevertheless, the examples of Dream Lens and Cupid are promising and illustrate the potential of form-driven selection for enabling designers to express their form-driven criteria, which are hard to express otherwise.

Finally, Pan et al. [75] argue for methods that support exploring design spaces not only through their performance metrics but also their geometric typology. Similar to others, the authors propose clustering design alternatives by topology, which is consistent with clustering using form-driven criteria seen in Cupid.

In summary, we can look at these strategies according to how they propose the coupling of form and data: by studying each alternative individually (strategies 1 and 2) and by interrelating sets of design alternatives (strategies 3 and 4). However, none of the surveyed systems except Dream Lens [63] supported all of them. When both data-driven and form-driven selections are supported, users can opportunistically alternate between them. I argue that doing so allows a conversation between the qualitative and the quantitative criteria to take place. But this conversation can take place in the user’s mind (cognitive or implicit) or can have a logical basis in the DSN system. Next, I elaborate on the possible nature of this conversation.

5.2.6 Nature of Criteria Expression

The third viewpoint on form and data coupling concerns the question: What criteria can be expressed by interacting with their coupled views? Brushing on a scatterplot is often based on a data-based criterion, and selecting image thumbnails in a gallery of design forms is expressing a form-based criterion. But how about brushing on a scatterplot where the dots are connected with visual links to images of design forms, for example as in the Design Galleries [62]? or filtering away alternatives that share similar components in their form,

\footnote{For example, the best way to compare cups may not be the same for comparing building designs.}

\footnote{In other words, the criteria can be explicitly represented and manipulated in the system.}
like the chisel tool in Dream Lens [63]? and how about selecting individual thumbnails in a gallery that are first sorted according to a performance metric [29]? and what about selecting alternatives based on their appearance if those alternatives were embedded in a self-organizing map that was created based on performance similarity [42]? The first shows a data-driven query on the logical level, but it is guided by forms cognitively, by virtue of their visual appearance. The second (the chisel tool) illustrates a pure form-driven query since data plays no role in the creation of the superimposed representations in Dream Lens, nor does it cognitively guide the filtering. In the third example, sorting the thumbnails by performance allows the expression of data-based criteria, and the ability to select alternatives by directly interacting with their visual thumbnails enables form-driven criteria as well. Both the data and form criteria here are expressed on a logical level. In contrast, the Design Galleries did not support interacting with the image thumbnails directly and hence only supported a cognitive coupling of form and data. Finally, the fourth example shows data-driven criteria in that the map is created based on performance similarity, but it also supports logical form-driven criteria by enabling the selection of alternatives by interacting with their form representations, e.g., thumbnails.

My analysis demonstrates that DSN tools can employ both cognitive or logical couplings of design form and performance data. Although cognitive coupling is easier to implement as it doesn’t require a direct (logical) link between the views, it falls short from the potential of explicitly expressing form-driven criteria. However, cognitive coupling imposes fewer requirements for precision and explicitness when expressing design criteria, and a degree of fuzziness can be advantageous when design criteria are in their formative stages. An alternative solution may then be to provide precise mechanisms of expressing form-driven criteria but design these mechanisms to be visually-guided, fault-tolerant, and encouraging of what-if questions.

5.2.7 How Simplification and Coupling Relate

Simplifying a design space through clustering or dimensionality reduction is followed by representing the outcomes. For example, the clusters can be viewed on node-link trees or treemaps, and the results of dimensionality reduction on hexagonal grids and 2D scatterplots. Form and data coupling can be implemented by embedding form views in these representations. The advantage of this approach is that coupling is achieved as a by-product. Users can look at the design visuals that are close to each other on a self-organizing map view and study how they are similar in terms of a specific set of performance metrics. Interacting with these representations enable designers—depending on the similarity criterion—to either query the performance of alternatives with similar forms or the appearance of similarly performing alternatives. I call this similarity-based Coupling as discussed in [11].
5.3 The Lens of Expressive Selection

The designers in the study by Shireen et al. [87] repeatedly worked with collections of alternatives, where each collection may represent a different criterion. The subsequent analysis of the designers' behaviour showed that by manipulating these collections, designers were also refining their selection criteria. For example, they created new criteria, removed unneeded ones, merged some and split others into subcategories. Since each alternative had only one physical representation (printed on index cards) in this study, an alternative could only belong to one collection at any given time. When moving to a digital medium, alternatives can be cloned or copied where the operations like splitting can produce new collections while retaining the originals.

5.3.1 Disjunctive Queries

If we assume that a user selects a set of alternatives based on an implicit or explicit criterion, then we must ask whether the means of selection in DSN interfaces can facilitate queries that can incorporate such criteria. Earlier, I have described form-driven and data-driven selection as two different types of selection techniques. These two techniques can express both qualitative and quantitative criteria. Next, I focus on the operations that can be applied to selections or sets of selections. The majority of the DSN interfaces that I surveyed only support data-based selections, which are potentially coordinated and shared between all the views in the interface. Most of these interfaces only supported conjunctive data queries that are expressed by incrementally brushing on scatterplots or parallel coordinates plots. A conjunctive query combines criteria with logical \textit{AND} operations only. An example of this query is to select alternative designs for a mixed-use tower project with a total "commercial floor area bigger than 10K m$^2$" and with total "residential floor area smaller than 12K m$^2$". In other words, a conjunctive query finds the intersection between all the sets and narrows the design space with each additional criterion.

As opposed to conjunctive queries, disjunctive queries expand the design space by applying logical \textit{OR} operations. If we had a criterion such as "total commercial floor space smaller than 10K m$^2$" or "total residential floor space bigger than 12K m$^2$" then a disjunctive query allows us to select alternatives that meet either criterion (i.e., finds their union). This can be effective when the criteria in question are conflicting (e.g., the commercial and residential area trade-offs). Finding the intersection between two criteria that are conflicting can result in few alternatives or no alternatives at all (e.g., see Figure 5.5). Designers can keep their options open by using disjunctive queries, which in turn may lead to preventing pre-mature narrowing of the design space.
Figure 5.5: Each selection rectangle (brush) expresses a different criterion. Colored dots form the union of the two criteria. No alternatives meet both criteria.

Only a few DSN interfaces allowed users to express disjunctive queries. For example, ParaGen [29] allows writing text-based SQL queries that support a variety of query types. The data-flow interface by Fuchkina et al. [42] allows users to union selections by connecting different flows together.

5.3.2 Selecting Multiple Ranges

Another selection-related concern is whether a DSN interface supports selecting multiple values ranges for a data dimension. Most of the surveyed interfaces allow a single-range brushing per dimension. For example, Figure 5.6 shows single-range brushing on the parallel coordinate plot in Design Explorer [7].
Figure 5.6: Only a single rectangular brush is allowed in Design Explorer [7]. This makes it impossible to select different nonadjacent values for a categorical variable (e.g., blue and grey colors) or non-contiguous ranges on a quantitative one.

There are multiple motivations for selecting multiple ranges. First, when filtering ranges on a quantitative input parameter, we cannot assume that all the desired alternatives are going to appear within the same range. Depending on the relation between inputs and performance, high performing alternatives can appear throughout the design space. Hence, we should be able to combine multiples ranges for the same input parameter (i.e., an OR operation). For the example in Figure 5.7, a multi-range selection would look like this: (20<=Input<=30) OR (60<=Input<=70). Similarly, nominal categorical variables have no inherent order (e.g., colours choice in Figure 5.6) and so selecting multiple categories requires supporting multiple ranges.

Figure 5.7: Focusing on one range of the input parameter misses out on sub-optimal, but potentially interesting, portions of the design space.
Figure 5.6 shows an example of a persistent brush (i.e., stays visible after selection is over), but other systems feature transient brushes (e.g., lasso selection in Dream Lens [63] and all selections in Spotfire [9]). Multi-range selection can be implemented by combining multiple persistent brushes (e.g., Figure 5.5) or by finding the union between subsequent transient brushes (i.e., a disjunctive query). The implementation of the first strategy requires supporting multiple persistent brushes for every type of visualization supported in the DSN interfaces. On the other hand, the second strategy retains a single transient brush but allows users to select multiple ranges by incrementally combining new selections with the current. The latter strategy is called incremental selection (see Figure 2.6) and is implemented in commercial systems such as Spotfire [9].

Disjunctive and incremental selection give designers more flexibility and expressiveness. Such qualities are needed because designers’ criteria cannot be anticipated, as they are shaped during exploration. In the face of this uncertainty, adopting a wider selection arsenal mitigates the risk of forcing designers to explore in a certain way.

5.4 The Lens of Collections

Designers make sense of a large number of design alternatives by organizing them into collections (or sets) that embody and dynamically change with their design criteria [87]. The search tasks involve divergent and convergent decision-making based on design criteria applied to design alternatives and resulting in collections of alternatives. This process is not linear and involves the development of multiple collections. The role of collections goes beyond selection: collections can be created to mark a set of solutions as, e.g., potentially useful, to be ignored, to be revisited, etc.

5.4.1 Non-Linear Exploration

The design alternatives are split into collections opportunistically considering the criteria identified or emerging criteria in the exploration. For example, designers divide a given set into an accepted set and a rejected set based on given criteria, only to repeat the same process with the accepted set. This is similar to the filtering pattern created by conjunctive queries. Shireen et al. argue that the order of applying filters determines the content of collections; a change in the order can create different collections with different content.

Filtering is appealing when faced with a large number of alternatives, but as the analysis by Shireen et al. shows, designers’ actions during navigation go beyond that. For example, designers created more than one accepted or rejected sets or branched an accepted set into smaller accepted sets. These examples illustrate a non-linear, opportunistic exploration pattern. Each set is a different exploration path, so to speak, in contrast with a single accepted set that is incrementally narrowed as supported by most of the DSN interfaces.
Furthermore, collections can be seen as a means for putting order into the design space. They can potentially prevent fixation on portions of the design space with the assurance they bring: that any part of the design space can be visited and revisited. Furthermore, their most basic function might be to save progress, much like a shopping cart. The use of user-created collection(s) is scarce in DSN systems. We can find examples of DSN systems supporting a single collection under the name of a "cart" [37], or bookmarks [100] or the "privileged" [61] collection. A single collection may be useful for accumulating the progress of exploration, but it does not give rise to the full potential that multiple collections provide, namely initiating the sense-making loop.

5.4.2 Supporting Multiple Collections

An important goal of DSN is to select alternatives that possibly meet the preset or emerging criteria. There are a few examples of working with multiple collections in the literature of DSN [42,107]. Therefore, I expand my exploration of supporting multiple collections to general-purpose visual analytics systems, such as Spotfire [9], Tableau [5] and VisFlow [112]. This choice is justifiable as the creation of collections in DSN is akin to using collections for generating and testing hypotheses in the sense-making model described by Pirolli and Card [76]. In addition, supporting multiple collections is consistent with supporting multiple hypotheses in visual analytics tools, which are essential to the sense-making loop.

The Design Exploration system by Fuchkina et al. [42] relies on a data-flow model to link different components together, which include views like self-organizing maps and PCPs. The system also supports operations of clustering and filtering by logical queries. Alternatives are selected manually (e.g., via brushing) or algorithmically. The selections (e.g., clusters) can be saved in a view and propagated downstream to subsequent views. This allows users to explore and refine sets of alternatives serially through various means. Collections or selections here are what flows and connects the components of the system. A similar approach, though not involving design visuals, can be found in VisFlow [112], which enables the flow of selection sets called "subset flow" [112] from one view to another downstream.

A different view on collections can be seen in D.Star [67,107]. Collections in the gallery are treated as first-class objects. For example, all alternatives must belong to at least a single collection. Collections have physical representations as panels that are visually transparent about their members. This is a parallel to the concept of "visually-accessible collections" by Shireen et al. [85]. However, this representation is different from some visual analytics systems such as Spotfire and Tableau, where collections are represented abstractly and minimally (e.g., see Figure 5.8). The members of the collections in D.Star can be refined, and each collection can represent its members through one of many views like a PCP, a table, or a gallery of thumbnails. Collections can also be copied, discarded, or become the basis of generative operations like the Cartesian Product.
Yet another perspective on collections comes from TIBCO Spotfire [9], which stores selections as 'markings'. Users can create multiple markings, and they can be created from any of the views in the system. At any time, a single marking, as well as the whole dataset, are shared globally by all of the coordinated and linked views. In contrast to the data-flow approaches, Spotfire does not commit to a one-way flow of data and selections between views. This allows two-way communication and cross-examination of alternatives across views; users can alternate back and forth between views as is appropriate for the task. Markings affect the visual encoding of the views in the interface, but unlike D.Star, collections do not have their own spatial containers.

In contrast, coordinated views are achieved in the D.Star by cloning a collection first and then choosing a different view per collection. Alternatives in the new collection maintain a reference to their representations in the original collection. As a result, interactions (selection and inspection) on one view are coordinated with the others. To summarize, the main difference between D.Star and Spotfire is whether views are a reflection of collections or if collections are reflected in views.

5.5 The Lens of Information Seeking

To continue the analysis on collections in the DSN interfaces, I studied how a collection can be manipulated or shaped as users search alternatives, i.e., information seeking. Data-flow approaches support brushing on visualization views. The selected alternatives can then be passed downstream to other views or be subjected to logical operation (union, intersection). The process is repeated until plausible alternatives are identified.

One way of seeking information in the D.Star is to select alternatives through any of the supported views. This action can create a new collection of the selected alternatives. Alternatives can also be copied between collections or discarded. Spotfire, however, compiles a selection through an incremental application of logical operations as described by Wills [103]. The operations are performed on the views and revealed by brushing on other
views. This process is supported by saving and retrieving the current selection as a markings set, and logical operations can be performed on these markings.

All the three approaches follow the Information Seeking Mantra of: "Overview first, zoom and filter, then details-on-demands" [90] when navigating a design space. But both data-flow systems and D.Star attempt to physicalize the navigation of the design space. Data-flow components and collections are constructed on-demand and in space. On the other hand, Spotfire is more economical in its use of space, as it relies on changing the visual encoding of the views already present in the interface. Information seeking happens, in time, through the operations of filtering or zooming on the current selection.

5.6 The Lens of Mode-less Navigation

"They focused on designs individually and collectively to identify their features and compared designs to justify their selection or elimination patterns." [85]

Designers require mechanisms for gaining an overview of the design space they are navigating in addition to means for comparing and inspecting individual alternatives. The switching between these tasks can be made explicitly by moving between different tabs in the interface as in Refinery [3], or staged views each dedicated for a specific purpose as in Goguelin et al. [44] and Fuchkina et al. [42]). A mode-less DSN interface allows users to switch between tasks during navigation without hiding away or disabling the parts of the interface outside their locus of attention. Mode-less navigation allows users to easily switch between tasks on the same interface while maintaining a cognitive and visual link between them. This can often be seen in coordinated views or dashboards like ParaGen [29], Spotfire [9], and Design Explorer [7]. It can also be seen in systems combining both abstract and spatial visualizations, such as described by Doraiswamy et al. [37] and van Kastel [100].

5.7 Other Lenses

Shireen et al. report other design space navigation patterns, such as rapid and slow scanning on juxtaposed multiple views, externalizing decisions by tagging, marking, commenting, and continuous reconfiguration of the workspace. I discuss the following lenses in relation to these patterns.

5.7.1 The Lens of Comparative Analytics

Comparing alternatives is integral to design exploration and decision making [108]. Comparison can be facilitated by data visualizations such as PCPs and scatterplots, although these representations are used mainly for identifying correlations, distributions and outliers. Other representations such as bar charts or tables can be more suited for directly comparing a few alternatives. Multiple DSN interfaces allow design alternatives to be di-
rectly compared in juxtaposed views, where each view represents both the design form and performance data [2, 6, 8, 29, 44, 68, 100]. Figure 5.9 shows an example of direct comparison in Asterisk [6].

Figure 5.9: A juxtaposition of alternatives in Asterisk [6] showing both their forms and data.

5.7.2 The Lens of Knowledge Externalization

Knowledge externalization is an essential component of the sensemaking process [92]. Some DSN interfaces allow users to annotate alternatives or sets of alternatives with tags, comments or ratings [2, 7, 8, 44, 63].

5.7.3 The Lens of Flexible Workspaces

Most visual analytics systems, such as Tableau, Power BI and Spotfire, support a high level of flexibility in both the types of visualizations supported and their layout. Few DSN systems allow users to flexibly customize their workspace to a high degree [7, 8, 66] such as for creating custom dashboards or allowing users to resize and move the views and tools. The flexible and configurable workspaces are essential for DSN systems for several reasons. For example, design is a collaborative process involving multiple disciplines with different concerns. This may demand different visualizations, features, and interface layouts to better support the varying goals of the design stakeholders. Furthermore, the increasing affordability and availability of displays present an opportunity for designing interfaces capitalizing on larger screen spaces [107]. Hence, DSN interfaces can consider large or multiple displays in their design for presenting different aspects of design alternatives simultaneously.
5.8 Summary of Requirements

Analyzing the literature through the above lenses resulted in a list of (high-level) requirements suggested by each lens. These requirements can influence the feature development of DSN systems, and a detailed justification for each was made earlier. Furthermore, these requirements are not exhaustive by nature, as they "mostly" reflect the features of the surveyed DSN systems in addition to the actions taken by the designers in the study by Shireen et al. Other requirements can arise that were not observed in the study or which interpret the results of the study differently. Briefly, the requirements include:

- **Simplification**: Support some form of simplification, e.g., clustering, dimensionality reduction or Pareto analysis. Aim for integrated and flexible simplification.

- **Coupling Form and Data**: (a) support multiple and different data views; (b) employ more strategies for coupling form and data (coordinated and detailed inspection, and data-driven and form-driven selection); (c) support both logical and cognitive coupling.

- **Expressive Selection**: Provide multi-range, incremental and disjunctive selection.

- **Collections**: Supporting non-linear exploration and sensemaking through multiple collections.

- **Information Seeking**: Support zooming on and filtering of alternatives, preferably aided by expressive selection.

- **Feature a mode-less shift of tasks during navigation.**

- **Support directly comparing alternatives.**

- **Allow externalizing the knowledge gained during navigation through annotations (comments, tags, ratings, marking).**

- **Support flexible adaptation of workspaces and anticipate large displays.**

The next chapter will present DesignSense which is a DSN interface that attempts to implement these requirements and integrate them with each other.

5.9 DesignSense in Light of the Lenses: A Foreshadow

In the previous sections, I have analyzed the literature on DSN interfaces through multiple lenses. I continue by analyzing DesignSense under the same lenses. Please note that I describe the full details of DesignSense in Chapter 6; I will foreshadow the results here then reflect on the contributions of DesignSense afterwards (Section 6.5). As a summary, compared to other DSN interfaces in the literature, DesignSense integrates several features that
are either novel or appeared separately in the surveyed literature, but not in the combination I present here, and not to the same degree of integration. Section 5.9 below highlights the missing features from the DSN systems that are closest to DesignSense.

Table 5.1: Major missing features from the closest DSN systems.

<table>
<thead>
<tr>
<th>Closest Competitors</th>
<th>Missing Major Features (based on the lenses)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAVE [34], LIVE [110], modeFrontier [31]</td>
<td>form and data coupling</td>
</tr>
<tr>
<td>Dream Lens [63]</td>
<td>simplification, collections, expressive selection, comments, multiple data views</td>
</tr>
<tr>
<td>D.Star [67]</td>
<td>simplification, expressive selection, comments, form-driven selection</td>
</tr>
<tr>
<td>van Kastel [100]</td>
<td>simplification, multiple collections, expressive selection, comments, multiple coordinated data views</td>
</tr>
<tr>
<td>Fuchkina et al. [42]</td>
<td>visual and incremental selection, mode-less navigation</td>
</tr>
<tr>
<td>Thread [8]</td>
<td>simplification, collections, expressive selection, form-driven selection</td>
</tr>
</tbody>
</table>

A Synopsis of Features and How They Are Integrated. DesignSense is a visual analytics interface for design space navigation and working with a large number of design alternatives. It consists of multiple data views (scatterplots, parallel coordinates plots), and a form views (a sortable, scrollable, and selectable gallery of visuals). The full interface of DesignSense is shown in Figure 5.10. Data and form views are juxtaposed and tightly coupled, both logically and visually, on the interface. Users can select a set of alternatives or inspect an individual alternative starting from any of these views upon which these operations are coordinated with the other views. Alternatives can be inspected in detail in an enlarged 2D or 3D view. Selecting alternatives follows an incremental strategy where each new selection can replace, intersect with, or add to the current selection (i.e., conjunctive and disjunctive queries). DesignSense also enables discrete or continuous multi-range selections. Users can choose to focus on the current selection only (zoom), exclude it from the dataset (filter) or undo any of these operations.

At any time, users can save the current selection as a set (sets and selections are equivalents). Multiple sets are supported and are represented minimally as a list (as opposed to a more physically dominant representation). Users can comment on sets, and sets can be integrated into the current selection exactly as new selections. By systematically integrating sets into the selection, users indirectly operate on sets. DesignSense allows users to seamlessly switch between overviewing, comparing, and inspecting alternatives while retaining
cognitive and visual links between these tasks. The interface enables direct comparison of alternatives by their form and data. Supportive analysis techniques like clustering can be used at any point in the exploration process. The users are given feedback on selections visually that can be impacted-by or impacting the current selection. Finally, the components of the interface can be resized and moved to configure workspace (or the dashboard), which allows users to capitalize on large display spaces.

![Interface Composition of DesignSense](image)

**Interface Breakdown.** The DesignSense interface has the following components as noted on Figure 5.10: The components marked with 1 and 2 are data views includes a parallel coordinates plot and a scatterplot, respectively. The third component is the Augmented Gallery view containing a gallery of thumbnails [3.a] that can be sorted and resized. The alternatives represented by the thumbnails can be selected/unselected by either a rectangle brush or single clicks. The inspection panel [3.b] shows an enlarged 2D or 3D view of a single alternative. The views in components 1, 2 and 3 are coordinated such that selections on one are reflected on the others. The Design Sets view (component labelled with 4) contains a vertical list of set cards. Each card has a title and comments on it. The buttons on the set cards are for removing or retrieving the set. The Clusters view, indicated as the 5th component, shows the cluster representatives that result from clustering the currently visualized dataset by chosen data dimensions. The top bar marked with 6 has several operations, such as: zooming on or filtering the current selection, undoing any of these [6.a]. It also has tools for inverting the current selection or resetting selection and creating design sets. 'Replace'
on the right [6.b] shows the current selection operation that will be applied to new selections and set retrievals. Other operations include Adding, Intersecting, and Toggling. To the left of these tools [6.b] is a button enabling Editing Mode, which allows users to resize and move all the views on the interface to create a custom workspace configuration.
Chapter 6

Second Iteration: DesignSense

6.1 Data and Task Abstraction

This section presents the data and task abstraction of DSN in line with the design study methodology [82]. These abstractions are influenced by descriptions of design space exploration in Section 2.5, and the studies on how designers navigate large design spaces in Section 2.5.6. Finally, they are also built on an understanding of the tasks already supported by current DSN systems.

6.1.1 Data Abstraction

A typical dataset resulting from a generative design process has a tabular form, with each record being a different design alternative. The dimensions associated with each alternative include the input parameters used to generate it and the output performance metrics resulting from evaluating it. Each design alternative also has an associated design form that is visually represented with 2D images, 3D geometry or both. The 2D images can represent different viewpoints of the design form (e.g., top view or perspective view). Additionally, they can also be used to visualize the results of performance simulations, such as colouring the parts of a building’s shell based on their respective access to the solar energy (e.g., as in the image in Figure 6.4).

6.1.2 Task Abstraction

As I have mentioned earlier, design exploration in the context of generative design can be seen as consisting of three stages: modelling, generation-and-evaluation, and navigation (as in Figure 1.1) that can be (re)visited in an iterative manner.

The process I describe above assumes that navigation and generation happen at sequential and discrete stages. In particular, this mode of a posteriori interaction with generated design alternatives, especially when mediated with data visualizations, can be referred to as design-by-shopping [17,98]. When contrasted with traditional design processes, such as non-generative form-making and direct modeling, design-by-shopping offers the potential
for evaluating far more design alternatives in the same amount of time. Navigating alterna-
tives through data visualizations is a generic approach that can be applied to many design
and engineering problems. However, I focus here on developing visualization solutions that
are attuned to the demands of design domains like architecture.

Design space navigation (DSN) consists of two main subtasks, which are to Understand
and Narrow the design space being navigated. I mainly focus on supporting the second
task (Narrow) in this thesis but I discuss understanding as well since it is inseparable from
narrowing.

**Understand**

When a large collection of alternatives is navigated with interactive visualizations, designers
can learn more about the problem and solution spaces they are exploring. For example,
visualizations can:

1. Reveal relations between the generative model’s inputs/variables and its outputs,
   which leads to identifying the variables with the largest impact.
2. Display trade-offs between performance objectives.
3. Foster the elicitation of designer’s preferences among these objectives.
4. Provide insight into the relationship between the geometric forms of design alternatives
   and their performance.
5. Highlight families of alternatives sharing similar properties.

Understanding the design space, which includes learning about the relationships and
patterns in it, subsequently guides narrowing and refining the design space.

**Narrow**

To narrow a design space, designers identify the feasible or acceptable regions\(^1\) in the design
space. The goal is to search for alternatives that *satisfice* their implicit and explicit criteria.
This is qualified by the following: In design, a design alternative is an idea or a proposal.
Being so, sometimes even the poorly performing alternatives can grant access to better
ones [108]. Furthermore, the selected set of alternatives from this process are used as stepping
stones rather than finalized solutions [22], i.e., they are further refined (by manually
editing them) or used to guide a second round of exploration (e.g., as seeds or targets for
modeling and generation).

The following section casts the above description in the typology of abstract visualization
tasks by Brehmer and Munzner [23].

\(^1\)Or more generally, sets.
Visualization Typology

Using the Why part of the typology by Brehmer and Munzner [23], we can outline the analysis tasks involved in DSN as Consume and Produce. The first analysis task is Consume aimed at selecting a set of satisficing designs [93]. As the designer’s criteria are continuously changing and might be initially ill-defined [94], the search tasks include: explore and browse (target is unknown), and when the criteria are more stable and understood it involve lookup and locate. Navigation involves coupling both the performance and the form of design alternatives, and so these also constitute the targets of the search.

The best design alternatives might not be objectively identifiable (due to trade-offs); instead, designers can identify the alternatives on the Pareto frontier and arbitrate between those based on their preferences.

When considering multiple data attributes, the selection task may involve relating inputs to outputs and outputs to each other or design forms. This is done in search for dependencies or correlations. Once a set of alternatives is selected, designers will be interested in Identifying different aspects of them (e.g., their visual forms or data attributes) as well as Comparing them. Supporting that is sorting alternatives by their data attributes or arranging their visuals in a gallery-like view (c.f. Arrange metaphor [86]).

Summarizing the qualities of those designs is also important for deciding on whether they should be committed to memory. This brings us to the second analysis task, which is to Produce/Derive sets of designs and annotate them with comments, tags, and labels. Sets can also be combined, split and discarded. The above are consistent with the spatial metaphors of Divide Designs and Mark by Shireen et al. [86].

6.2 System Components

This chapter presents the second major prototype iteration in my research process. The prototype is called DesignSense, and it is a visual analytics system of coordinated and linked views, representing and coupling both the geometric design forms and their performance data. Aiding that are mechanisms for selecting from, navigating through, and grouping design alternatives, where groups can be both manually and automatically created. Figure 5.10 shows the full interface of DesignSense. The remaining of the chapter delves into the details of the system’s components, justifying my decisions where needed.

The dataset of alternatives to be explored is shared between all the views in the interface.

6.2.1 Multiple Coordinated Views

With the goal of coupling design forms and data, DesignSense dedicates views for each and coordinates the selections made on them (i.e., Juxtaposed composite views [52]). Doing so allows us to fine-tune each view to its purpose. This kind of Multiple Coordinated Views
(or Multiform Views) [80] enables users to explore their data from multiple perspectives and supports a broader repertoire of analytical tasks.

### 6.2.2 Selection via Brushing

Users can select alternatives starting from any of the provided views, and this selection will be coordinated with the other views through brushing and linking. A common visual encoding scheme is used across the interface to enable distinguishing alternatives in different views. The selection of alternatives happens through different types of brushes as suitable for each view. This selection is shared between the views and can be incrementally refined through the following selection operations. The default operation is Replace while the rest are bound to key modifiers:

1. **Replace (default):** Replaces the current selection with the new.

2. **Add (modifier 1):** Finds the union between the current and the new selections.

3. **Intersect (modifier 2):** Finds the intersection between the current and the new selections.

4. **Toggle (modifier 1 and 2):** For the alternatives in the new selection, if at least one was previously unselected, they will all be selected. Alternatively, if they are all already selected, they will be removed from the current selection.

This incremental selection system is inspired by that of Wills [103] and the system TIBCO Spotfire [9]. A good system should be "powerful, intuitive, and forgiving" [103] and enable the operations we expect the users to perform the most. Furthermore, DesignSense also supports inverting the current selection and clearing it (Figure 6.1).

DesignSense utilizes a memoryless selection strategy [103], whereas the exact queries used to arrive at the selection are not saved or even represented in the system. Instead, the system keeps track of the currently selected alternatives directly through their unique identifiers. A memory-able strategy is better suited for finding alternatives that meet a given criterion [103] since users can write an exact query to retrieve them. But the fact that design criteria are continuously changing during design exploration makes me favour a memoryless strategy. Another reason I opted for a memoryless strategy is that it makes it easier to coordinate views regardless of their underlying representation. Views only need to share the same data with common identifiers. This comes as an advantage when exploring new views in a multiform system as new views can be integrated with the rest by exchanging the identifiers of the currently selected alternatives and leaving the task of responding to that to the view itself. For example, I found that the selections made on a form view (e.g., a gallery of image thumbnails, as shown in Section 6.2.8) did not translate well to contiguous intervals on data dimensions (e.g., height between 10 and 50).
6.2.3 Information Seeking

The dataset being explored is shared between all the views, but sometimes we want to focus on parts of it. At any time, the shared dataset can be restricted to the currently selected design alternatives (zoom), or we may exclude the current selection from it (filter). We may also undo any of these operations or return to the initially loaded dataset. The implementation of these operations are guided by the Information Seeking mantra [90] of 'Overview first, zoom and filter, then details-on-demand'. The operations can be called through the top-bar menu or an on-demand context menu on demand (Figure 6.1).

![Figure 6.1: Top: The operations top bar. On the left-most is a button for toggling inspection on hovering. In the middle are the operations related to selections such as creating sets, resetting selection, and inverting the selection. The right part contains information-seeking operations such as undo, zoom, filter, and returning to the initially loaded dataset. Bottom: An on-demand context menu that includes the same operations above. The icons are colour-coded to distinguish their category (selection or information-seeking). Some infrequent operations are left out to retain compactness.](image_url)

6.2.4 Inspection

While exploring alternatives, users may wish to inspect a single alternative to learn more about it or to visually link its representation in different views. For example, users may desire to learn about the performance of a single design alternative whose visual form caught their attention (and vice versa). DesignSense supports the coordinated inspection of alternatives. Doing so changes the current visual encoding in all the views to emphasize the inspected alternative while relatively preserving the visual encoding for the currently
selected or unselected alternatives. Inspection marks the last component of the Information Seeking mantra ('details-on-demands'). To inspect an alternative, users can hover over the sought alternative’s representation in any of the provided views. This includes thumbnails in a gallery, lines in the parallel coordinates plot, or dots in the scatterplot.

By default, hovering needs to be accompanied by holding a key modifier. Alternatively, the "Inspection on Hovering" mode can be enabled in the system (Figure 6.1) upon which inspection only requires moving the mouse around. Because inspection temporally overrides the current visual encoding of the selected/unselected, this mode is not enabled by default. Enabling it by default can also cause frequent changes in the interface as the mouse is moved around, which can be cognitively demanding.

6.2.5 Parallel Coordinates Plot (PCP)

PCPs are aimed at visualizing multidimensional data [50]. They can be used to provide an overview of data dimensions, besides identifying outliers and filtering [70].

DesignSense provides a PCP view and facilitates its usage by enabling reordering the axes of the PCP as well as inverting their vertical orientation. To select alternatives on the PCP, users find the axis representing the data dimension they are interested in. They can then proceed to drag a rectangular brush over the interval of values they wish to select. Once the dragging is over, the alternatives whose values belong to that interval will be integrated into the current selection, depending on the current selection operation. To enhance the precision of selection on the PCP, DesignSense shows a tooltip with the exact range of the currently drawn brush (Figure 6.2).
6.2.6 Scatterplots

Scatterplots are often more effective at finding correlations between data dimensions than PCPs [70]. They can be used in identifying clusters, outliers, and extremes. DesignSense provides a scatterplot view and enables selecting alternatives on it through rectangular brushes. Similar selection and inspection features are supported on the scatterplot view as to those on the PCP view.
Simplification with Pareto Analysis. In the presence of conflicting performance objectives, designers can *satisfice* by picking the design alternatives lying on the Pareto frontier. The Pareto frontier consists of the set of designs that cannot be modified with respect to one objective without compromising on the others. Since DesignSense deals with already generated data, we may only select the frontier if it existed. If the objectives in questions are maximization objectives, then the frontier will consist of the most top and right alternatives (see Figure 6.3). When desired, users can choose to find the frontier for the dimensions currently selected on the scatterplot (i.e. only two at a time). Finally, the result can be integrated into the shared selection based on the current selection operation.

6.2.7 Visual Encoding Decisions

All the brushes in DesignSense are currently transient, meaning that they disappear once selection operations are complete. Keeping the brushes requires maintaining their intervals in response to selection operations on the other coordinated views. If all the views we are interested in were limited to rectangular brushes in the data space, then finding a mapping between them is trivial. But that is not the case if we adopt more complex operations like Lasso selection or selection mechanisms that are independent of data dimensions (e.g., form-driven selection). An example of the latter is when we select individual alternatives based on their appearance. In this case, the corresponding interval ranges on the data views will not be contiguous or easily maintainable. As another option, the singled-out alternatives might be given a special visual encoding.

In terms of the visual encoding, all the representations of design alternatives in DesignSense have three different states. These states are: selected, unselected, or inspected. Unless a specific dimension is mapped to the colour channel, only the blue and grey colours are used throughout the interface. The different levels of encoding do not use the colour channel so as to keep it free for other encodings. For example, in the PCP, a line’s colour is
grey if it is not selected, and its thickness will be small. Once a line is selected, it is coloured and its thickness is increased. When an alternative is inspected, the unselected alternatives go further back into the background (i.e., encoded with dimmer colours), and the selected take the same encoding as the unselected when no inspection is taking place. For the PCP, the lines representing the inspected alternative are made thicker and are augmented with tooltips that show the exact values for that alternative (see Figure 6.2).

The thumbnails in the Augmented Gallery, akin to the other views, have three levels of encodings. When their corresponding alternatives are not selected, they are greyed out. When they are selected, they retain their normal colours with an added coloured outline. Once an alternative is inspected, the outline of its thumbnail is made thicker.

Figure 6.4: The Augmented Gallery: (A) Control panel. The icons at the top from left to right are: sorting dimension and order [arrow], the number of currently selected alternatives, filtering to selection [eye], and thumbnails’ sizes [slider] (B) Detailed Inspection panel, shows the currently inspected alternative (or the last-inspected when no inspection is taking place). At the top of it, the icons from left to right are: closing the panel [X], switching to 3D view [3D], and changing to the next/previous image thumbnail [arrows] (C) Gallery panel. The rectangular brush at the bottom right shows selection on the grid. The thumbnail with the thick outline (top-right) denotes the currently inspected alternative.
6.2.8 Augmented Gallery

DesignSense provides a forms view to allow users to filter, compare, and inspect alternatives based on qualitative design criteria. This view is an interactive gallery of image thumbnails, referred to as the Augmented Gallery (AG). The Augmented Gallery\(^2\) comprises three parts, the gallery, the controls panel, and the detailed inspection panel(Figure 6.4). The gallery panel shows a grid of image thumbnails, each representing a different design alternative. Through the options on the controls panel, these images can be resized (to enhance the visibility of their content as desired) or sorted based on a data dimension. If the size of the images combined is larger than the allocated space, the grid also becomes scrollable. The following are some of the interactions that make the AG unique.

Selection

Form-driven selection is enabled by dragging rectangular brushes on the thumbnails. This is akin to how icons can be selected in GUI operating systems. Selection on the AG is most effective when combined with the appropriate sorting order and a thumbnail’s size large enough to discern the qualities of the design alternatives portrayed\(^3\). Additionally, it is also possible to select/unselect alternatives by simply clicking on their thumbnails.

Selections on the other data views do not necessarily correspond to adjacent thumbnails on the AG unless they are made on the same dimension that the thumbnails are sorted on. This results in scattering the selected thumbnails among those unselected. I retain this behaviour to enable viewing these thumbnails in the context created by the sorting-order. Furthermore, the thumbnails in the grid can be sorted based on the binary dimension of whether they are currently selected or not. Doing so brings the selected thumbnails to the top of the grid (or vice versa).

Filtering to Selection

Through an option on the control panel (eye icon), the thumbnails can be filtered to include only those currently selected. Users can continue refining the currently selected alternatives when this option is enabled. But since no greyed out (unselected) thumbnails will be shown here, a secondary encoding (dashed outlines) is introduced to visually cue the users on which alternatives they are currently brushing (Figure 6.5 - Top).

\(^2\)While the gallery is not novel per se, the interactions supported on it makes it powerful enough to warrant a name.

\(^3\)Thumbnails can be resized on the gallery for that purpose, but ultimately, being able to discern the thumbnails’ details depends on the images that the user provides. For example, images with fewer and sharper details are easier to examine and compare at a glance.
Figure 6.5: Top: Brushing is denoted with dashed lines when the Augmented Gallery is only showing the current selection. Bottom: Direct comparison is currently enabled through a vertical panel below each thumbnail that lists the design data in key-value pairs. This feature becomes available when the size of the icons is set to the largest.

Comparison
DesignSense facilitates an in-depth comparison of few alternatives through on-demand expandable panels containing a vertical list of data key-value pairs (Figure 6.5 - Bottom).

Detailed Inspection
The thumbnails may be useful in facilitating selection and quick comparisons, but it is often necessary to examine an alternative more closely. The final component of the Augmented Gallery is the detailed inspection panel. This panel shows an enlarged image of the currently
inspected alternative. If the right 3D geometry data is available, users can also show a 3D view of the alternative. The 3D view supports panning, rotating, and zooming in the 3D context.

The images and the 3D models associated with each alternative are left completely up to users. In doing so, users can present the aspects of their designs that best suit their interests. Finally, an alternative can have more than one image. The first image is shown by default in all the thumbnails in the grid. Users can navigate between the images associated with the currently inspected alternative through two arrows at the top of the panel.

6.2.9 Design Sets

At any time, users can save the current selection into a saved set or retrieve it. The Replace, Add, Intersect, and Toggle operations apply when retrieving a set. In other words, retrieving a set is treated like making a new selection. By systematically retrieving sets, users indirectly operate on sets and compose new sets from existing ones. This process is visually informed since retrieving a set updates the current selection, and the current selection is visually distinguishable in all the views. The ability to visually represent selections and to incrementally refine and undo operations on them aligns with the principles of direct manipulation [88].

![Design Sets](image)

Figure 6.6: Left: Design Sets view is a list of cards, each representing a set. Each set card has a name and an associated comment. Users can delete a set or retrieve it (which integrates its members into the current selection). Right: The Clusters view. The thumbnails show the clusters’ representatives. Users can select the number of clusters and the size of the thumbnails at the top. Through the drop-down at the bottom, they can also choose the quantitative dimensions they would like to cluster based on.
Users can also externalize their knowledge about the design sets [92] by adding comments on each. Users can also delete any of the created sets. Finally, a sample of the design sets view is shown on the left of Figure 6.6.

6.2.10 Clustering

To integrate design space navigation and simplification, clustering is supported through the Clusters view. Alternatives are clustered based on chosen quantitative dimensions (the clustering criteria) using the K-means algorithm. The distance/similarity between alternatives is calculated using the Euclidean distance, and the values per dimension are first normalized to prevent their scales from impacting the distance calculations. The clustering algorithm is run every time the data being visualized is transformed (e.g., through filtering or zooming operations as in Section 6.2.3).

To visualize the results of clustering, I use cluster representatives. A cluster representative is defined as the alternative closest to the center of the cluster. Representatives are shown as image thumbnails in the Clusters view (Figure 6.6 - Right). The number of clusters in the K-means algorithm and the size of the thumbnails can be adjusted by users.

By clicking on any of the thumbnails, the members of the chosen cluster are integrated into the shared selection (again depending on the current selection operation). Through a combination of showing only the currently selected in the Augmented Gallery and clicking on a cluster’s thumbnail, the design forms in that cluster can be easily examined.

The purpose of clustering is often to identify any underlying groups existing in the data, but it also has the advantage of endowing a structure to the design space. In other words, the cluster representatives provide an overview of the design space (depending on the clustering criterion) upon which they can be examined in detail. Overviewing the design space can be achieved by inspecting the cluster representatives, which is enabled by hovering over their thumbnails in the Clusters view. Finally, at the moment, the cluster’s view is not affected by selection operations on the rest of the interface, unless they were followed by filtering out or zooming on that selection.

6.2.11 Flexible Layout and Supporting Multiple Displays

All of the views in DesignSense can be resized and moved. Upon detecting potential overlap between moving panels, the system responds by preventing that move. These features are implemented with the intention of supporting the utilization of the screen space made available by multiple displays setups. Some of the system operations like creating sets, zooming, or enabling the inspection on hovering can be accomplished by pressing on their corresponding buttons at the top right bar. In a multiple displays setup, users will have to travel a large distance frequently to perform these operations. To enhance their experience, DesignSense provides a radial context menu that can appear on demand by clicking anywhere on the right mouse button (Figure 6.1).
6.3 DesignSense: System Implementation

DesignSense is implemented as a front-end web interface with a small back-end contribution. The technology used (Vue.js) is a reactive and modular JavaScript framework that formed the backbone of tying the different pieces of the system to a common dataset and shared selection. The library D3.js was used to create the data visualizations in this system. Following in step with the commonly used tool Design Explorer [7], I enable loading the data (a CSV table, images, and 3D files) into the system by first hosting it on a cloud service and providing a URL pointing to it. The advantage of this technique is that it is easy to set up and does not require storing the data in a database first. The modularity of the system makes it possible to easily integrate any new view with the other coordinated view by implementing some necessary methods and responding to a set of system events.

![Figure 6.7: A store manager keeps a shared state consistent between all the system’s components.](image)

The overview in Figure 6.7 show the store, which manages the state that is shared between all the components in the system. Actions can be dispatched by any of the components (e.g., selection operations) which in turn is translated to mutations on the state. Actions can be put on hold pending a server-side processing to finish. This is useful when desiring to offload a computationally heavy process to a high performance server. All the system’s components have access to the shared state and can choose to listen to changes to specific parts of the state (Figure 6.8). As an example, brushing on the scatterplot can change the set of currently selected alternatives in the store. When the change takes place all the other components will be listening to it and will update their visual encoding accordingly.
Figure 6.8: The store includes the dataset being explored in addition to user-generated data such as design sets, the currently selected or inspected alternatives.

Whenever possible I operate mainly on the unique identifiers (IDs) of the alternatives and not the objects representing those alternatives. For example, selection operations (Section 6.2.2) are applied to sets of IDs. The alternatives’ objects can be retrieved at any time using a dictionary that maps between the IDs and the objects. So when a change happens to the current selection, each component listening to that change receives only a list of IDs for the newly selected alternatives. Then since each component has access to the full dataset, they can translate that list of IDs to the actual alternatives data to be visualized.

This approach is also applied in the implementation of the navigation operations (Section 6.2.3). These operations are applied on the dataset rather than the current selection. In particular, to ensure that these operations are reversible, I keep copies of the past states of the dataset in the store (e.g., before and after zooming) and this is accomplished by storing the IDs of the alternatives in each state. Finally, the advantage of the above approach is that it requires less memory and leads to faster processing times. The disadvantage is that it creates additional components of the state that need to be maintained, namely now we have lists of IDs, the dataset of alternatives and the dictionary that translates between the two. But this disadvantage is lightened by the fact that the reactive programming afforded by Vue allow us to easily change the new state components (IDs and dictionaries) based on the user’s actions.
6.4 DesignSense: Scenarios

Here I present a set of scenarios for using DesignSense for exploring realistic datasets. I start with a dataset of generated alternatives for an architectural design problem. I also briefly probe into the applicability of DesignSense to other problems such shopping for used cars, and exploring generated game puzzles.

6.4.1 Case Study 1: Generative Design in Architecture

One of the reasons for using generative design is widening exploration at the early design stages. Due to the open-ended nature of the generative process, the produced designs can be surprising to designers [26]. Generative processes produce a large number of alternatives, which if properly generated and explored, can lead to considering far more alternatives within the same duration needed to consider a few alternatives in a non-generative design process.

An architectural design firm collaborating in this research project gave us access to a dataset resulting from a generative design process that they used in a past project. The dataset tackles an architectural design problem, namely a massing study, i.e., finding the masses or forms of a group of buildings that will be placed on a single site such that these buildings jointly meet certain requirements and maximize some objectives. The goal of this dataset is to explore different form compositions and spatial layouts of these forms in the building site. This dataset includes about 100 generated alternatives each representing a different massing solution and evaluated against multiple performance objectives.

A Sample. Figure 6.9 shows a sample from the dataset. The massing consists of the coloured buildings. The colours on the surfaces represent the solar radiation that the surfaces received at a single moment of time (red means more radiation). The gray buildings are part of the site (i.e., already built). This alternative includes three distinct masses. The space between the two on the right is secluded from sunshine, which is not necessarily good in the cold weather of Vancouver where the project is taking place. The tall block in the mass at the bottom blocks the views of the other masses to the south and south-east. This, in turn, blocks the sought winter sun that comes from the south. The metrics alone may not be informative except when compared against those of the other alternatives.
Figure 6.9: A massing alternative.

**Quantitative Criteria.** The objectives involved in this study can be found in the list below. Some of those (like solar radiation) are dependent on the geographic location of the project (Vancouver, Canada) and some are based on the form composition and its relation to the site. A number of these metrics require some form of aggregation across points on the site or on the building’s surfaces. In these cases, the site/surfaces are discretized first and the average of calculating the metric for each discrete unit is reported.

- **FSR:** after dividing each building mass into floors, FSR is the ratio between the area of these floors and that of the site.

- **A/V ratio:** the ratio between the surface area of all the building masses to their volumes.

- **Site_Cvr%:** the percentage of the site that has buildings on it.

- **Roof_Solar, Horizontal_Solar:** the annual solar energy gain on building’s roofs and surfaces, respectively.

- **SVF:** the Sky View Factor is the ratio of the sky hemisphere visible from the ground.

- **Isovist:** the volume of space visible from a given point. In this case the point is taken to be a point on the ground.

- **Sunshine_hrs:** the annual hours of exposure to sunlight.

- **Max_H:** the height of the tallest building in the massing.
- Surface View: the percentage of access to views outside the site to a chosen radius.

In general, this is an example of a sustainable design project where multiple interdisciplinary objectives are sought. Considering these objectives at an early stage often yields the highest returns, since some major (or macro) decisions can be made at this stage that can impact the eventual environmental footprint of the buildings. Furthermore, changing design alternatives is relatively cheap at this stage.

**Qualitative Criteria.** The following are some qualitative aspects that can be assessed by examining alternatives in either 2D images or a 3D geometry view: The symmetry of the massing, the availability of setbacks, accessibility (e.g., ease of traversal between buildings), empty spaces (how much they are and whether they can be well utilized), how "broken" the massing is (fewer and more connected building blocks vs. more in number but disconnected blocks), and the general complexity of the massing and consequently the ease of its construction.

**Form and Data.** Coupling the qualitative and quantitative aspects of alternatives can be achieved in through coordinated selection or inspection.

**Coordinated Inspection.** To illustrate how inspection works we can look at the alternative I have described earlier. When inspecting an alternative, DesignSense highlights it in all the supported views. First, we see a large image view at the top of the gallery view. This view can also be switched to show a 3D model of the alternative, or different 2D images of the same alternative (using the icons at the top-left of the large image). Below that is a gallery of sorted thumbnail, and the thumbnail of the inspected alternative is highlighted with thick borders. Note that the thumbnails here show only the selected alternatives, the crossed eye icon at the top bar indicates that. DesignSense allows inspection by hovering over the thumbnails in the gallery, lines in the PCP or the dots in the scatterplot.

Since the alternatives are sorted by their Site Coverage, we learn that this alternative achieved this target well since sorting is in descending order (notice the downward arrow the top bar) and the inspected alternative appears at the top of the gallery. When looking at the parallel coordinates plot, we find the exact metrics’ values are shown in tooltips. We can compare this alternative’s performance to others by observing the placement of the line along any of the vertical axes. For example, the Site Coverage has an average value when compared to all the alternatives in the dataset but is high when compared against the selected alternatives only.

Coordinated inspection can be used to ensure that design forms and data are matching. For example, we can ensure whether the site coverage metric is reliable by inspecting a few alternatives visually. We can do that by starting from the alternative with the lowest site coverage to the highest while keeping an eye on the large image in the gallery. If the metric is reliable, then we should observe that the amount of empty space on the site decreases as the coverage metric increases. Finally, we can achieve this by inspecting alternatives along the site coverage axis on the PCP starting from the bottom.
Coordinated Selection. When navigating design alternatives, we may have well-defined criteria for what we are looking for, or we form these criteria as we compare and examine alternatives. These criteria can be quantitative, qualitative or both. When a set of alternatives are selected in DesignSense, then the selection is coordinated across the interface. This allows us to select alternatives based on their quantitative performance (e.g., on the scatterplot view) and follow that with refining the selection based on the visual appearance (qualitative) of the alternatives in the selection. Alternating between quantitative and qualitative criteria is accomplished by alternating between selecting on data and form views.
Figure 6.11: The scatterplot on the left shows a strong negative correlation between SVF and Site Coverage. This is also clear from the crossing lines between the axes of these metrics on the PCP. The left PCP shows alternatives with high SVF already selected, and a tooltip shows the range on A/V ratio that will be selected once the user completes this operation. Since the current selection operation is Intersection, the result on PCP on the right shows alternatives meeting both criteria.

An example of clear criteria might be to find a set of massing alternatives that have a high SVF, a medium A/V ratio, and a ‘simple’ massing. Since the quantitative criteria are clear, we can start by applying it. First, we select the alternatives with a high SVF by brushing on the PCP. We do not want to quickly narrow the design space, as that will prevent us from flexing around trade-offs if they existed. For example, the site coverage metric negatively correlates with SVF (as indicated by the crossing lines in the PCP). If we selected only the alternatives with the highest SVF then we are limited to alternatives with very low site coverage. We might decide that a slightly lower SVF is appropriate if we found other reasons to motivate that. Either way, we do not have to strongly impose our criterion at the start so as to allow us to see and then consider these trade-offs. The SVF metric spans between 0.37 and 0.60. To follow this guideline, we can select the alternatives with SVF higher than 0.53. We can precisely select this range on the PCP using the tooltips that appear while brushing.

The next criterion concerns the A/V ratio. Inspecting few alternatives along this metric’s axis on the PCP shows that the masses tend to be simpler with higher values for this metric (and vice versa). We can find the intersection between the currently selected alternatives (with high SVF) and those with middle to high A/V ratio (Figure 6.11).

Upon filtering by data metrics, we can examine the design forms. First we can enable the ‘Show only Selection’ mode on the gallery so that only the thumbnails of the selected alternatives be shown (Figure 6.12). We can then inspect these thumbnails one by one or
use their thumbnails in the gallery (if they were telling us enough) to further refine the selection by discarding the alternatives with complex masses.

Figure 6.12: Users can alternate between (left) showing the selected alternatives in the context of those unselected, or (right) show only the selected alternatives.

Showing only the selected alternatives in the gallery is also effective at understanding the relation between forms and data (just like coordinated inspection, as described above). One can systematically select portions of the design space based on data queries and see only the forms in that part. The process can be repeated as needed to gain an understanding for the impact of data-based queries on the changes in the forms.

Finally, the example above shows filtering by data and form when clear criteria exist. When that is not the case, then we need to build our understanding of the design space first. We can do that by observing the correlations between the data metrics using the PCP or the scatterplot views. We can also scan through the design forms to shape an impression on the variety that exists. More what-if questions can be involved here than when we had clear criteria. This can be done by experimenting with selections. For example, an initial selection can be made on any of the data metrics. We can observe how the values of the other metrics are impacted by that as well as to inspect the forms of the selected alternatives. Doing so allows us to understand the consequences of this selection. This process can be repeated, potentially by building more complex criteria using the intersection or union operations. Furthermore, our exploration might invoke any of the navigation strategies in DesignSense like zooming, clustering, or creating sets. The next sections will describe those in more detail.

**Inspecting Cluster Representatives and Navigating with Clusters.** Instead of examining the alternatives in the Augmented Gallery or the PCP views, users can choose to explore the dataset at a coarse level first. In other words, explore families of alternatives before examining individual alternatives. The cluster view can help in that. As a start, users may pick all or a few performance dimensions to cluster on. Then by inspecting the representatives, they gain an overview of the most distinct alternatives from a performance
perspective. Inspection highlights an alternative on all the views (gallery, PCP, scatterplot), which allows users to compare representatives on all these views by quickly alternating between them (Figure 6.13). Lastly, picking a small number of clusters (e.g., \( K = 4 \)) means that much of the variation is suppressed. Users can then systematically increase \( K \) such that representatives give a slightly finer and nuanced view of the design space but not too fine that it defeats the purpose of clustering.

Another way to use the clusters view is to couple it with zooming (Figure 6.14). For example, upon clustering by the Area/Volume Ratio (A/V\%) and the Site Coverage (SC\%) into 4 clusters we get clusters of [high A/V\%, low SC\%], [low A/V\%, high SC\%], [mid A/V\%, slightly high SC\%], and [slightly high A/V\%, mid SC\%]. A user may decide that a low site coverage (first cluster) means under-utilizing the land available, which they do not wish to have. By clicking on each of the other clusters (while pressing the Addition selection modifier), they can collect the clusters they are more interested in then zoom into the current selection. Alternatively, they can select the undesired cluster and exclude it.
Figure 6.14: Left: picking the dimensions to cluster on. Middle: selecting the cluster with a high A/V ratio and low site coverage (which is reinforced by looking at the scatterplot). Right: the dataset after excluding the cluster with low site coverage.

Following a zoom or an exclusion operation, the clusters are recalculated again. Users can choose to repeat either of the above strategies (inspecting representatives or navigation with clusters) on the new clustering or switch to other means of exploration. A final note to make is that the number of clusters or the choice of dimensions to cluster on are visually guided by the exploration process instead of being performed as a pre-processing step.

**Sets as Shopping Carts.** After a line on inquiry leading to a smaller set of alternatives, users may want to save their progress to revisit it later or combine the results of this inquiry with earlier results. Sets can help in that. Saving the progress is aided with the ability to associate comments with sets. Comments remind users of the content of the set or the criterion for creating it. Sets can also be combined using the selection operations of Intersection and Union. Combining sets allow us to connect criteria that may not be otherwise reconcilable. For example, a set may be created based solely on visual aesthetics and another based on the compactness of the buildings’ forms (e.g., using the Area/Volume ratio). Finding the intersection of these sets amounts to finding alternatives meeting both criteria. If the intersection resulted in an empty set, then that may suggest that some conflicting objectives are at play or that the sets were simply not large enough.

**Reflection with Sets.** As users navigate, they make decisions about what to focus on and what to ignore. These decisions may lead to fixating on portions of the design space. By saving their progress as a set, users can then view it in light of all the other alternatives in the design space, leading them to recognize the trade-offs and decisions they committed to in the course of navigation. Figure 6.15 illustrates that.
6.4.2 Case Study 2: Shopping for Used Cars

The problem of navigating alternatives while considering both qualitative and quantitative criteria is present in other areas. The second case study concerns using DesignSense to shop for used cars. The dataset in this case study was created from online public postings of used cars on a shopping website. Each car has a picture and a number of attributes such as its price, year of manufacturing, manufacturer, and the mileage it travelled, among others. Figure 6.16 shows the cars dataset in DesignSense.
Figure 6.16: A dataset of used cars. With some caveats, shopping for cars shares some similarities with design exploration, such as the need for coupling form and data.

**Remarks**

This kind of navigation of alternatives is commonly performed through commercial interfaces that use simple interfaces and queries. For example, Figure 6.17 shows a typical interface for exploring used cars. Commercial interfaces such as this often use dynamic queries that filters the alternatives they present, but they vary in their support for curating sets of alternatives, form-first selection, forming complex logical queries (dynamic queries often only allow narrowing with AND operations) or integrating exploration with analysis (e.g., clustering). While I do not argue that commercial interfaces should employ these techniques, I argue that the technical expertise of designers and the challenging nature of their tasks can motivate raising the complexity of the used techniques. Designers seek a deeper (and communicable) understanding of the trade-offs in the problem and the relations between the inputs to their models and their outputs (sensitivity analysis). The demand for understanding the design space calls for more sophisticated representations like scatterplots and parallel coordinates that do not only facilitate selection (like dynamic query widgets), but also help in identifying correlations, outliers or Pareto frontiers.
Another remark I can make is that users of shopping interfaces only select out of the presented alternatives. Designers, on the other hand, can manually create or procedurally generate new alternatives. Finally, navigation in design does not end in the same way as shopping, i.e., by buying the car or house deemed worthy. Instead, these alternatives continue in a longer story of multiple iterations and a long process that involves multiple stakeholders, collaborating designers and engineers.

6.4.3 Case Study 3: Game Puzzles

Concepts such as coupling form and data, or handling the overload of choice which DesignSense addresses are not limited to exploring architectural designs. Other design domains, such as game design, are facing similar challenges. The field of Procedural Content Generation (PCG) explores the techniques of automatically generating game content such as game levels, puzzles, stories and others [48]. Seeing the similarity between the challenges that architects and game designers face with exploring generative content, I was motivated to explore the applicability of DesignSense to a dataset of levels for a puzzles game.

The game levels were generated by myself using a generative technique (evolutionary search), and the levels were explored in a case study where the designer of the game was asked to explore the generated levels using DesignSense. The full details on this case study...
was published in a separate article [10]. Below is a synopsis of that. The goal of navigating generated levels is to select a set of them to ship in the game after potentially refining and play-testing them by game designers. This contrasts with most other design domains where it is understood that only a single alternative will be eventually built. Figure 6.18 shows the interface loaded with said game levels.

The best way to evaluate a game level is to play it. This is because the gameplay experience unfolds in time or depends on the interactions of the player(s). A game level can be evaluated against multiple factors that can be hard to quantify like whether it is 'fun', appropriately difficult, or have a good flow. But playing out all of the generated levels is not feasible. So to narrow down the generated levels, they were evaluated against a few heuristics first. The heuristics were structural or behavioural in nature. The first included information about how a level is set up, like the number of pieces of different types. The behavioural heuristics were derived by simulating the gameplay of the generated levels by finding their optimal solution and counting a few metrics as the solution is played out. For example, these included the number of steps in the solution and the frequency of using some game mechanics.

DesignSense supported multiple images per alternative at this stage, and so these images were used to show subsequent steps in the solution of each game puzzle. During the case study, the designer alternated between the gallery having the levels’ visuals and the PCP. The PCP was first restricted to a few metrics as the designer wanted to understand the intuition behind them before adding more metrics to the navigation. Few suggestions were made to make DesignSense more suitable for exploring game levels. Firstly, animated pictures (e.g., in GIF format) were requested as a quick way to understand the gameplay behaviour of a level. Secondly, tighter integration with game development tools was requested to enable play-testing levels in the full game environment and possibly with end-users (players). Thirdly, the designer asked for the ability to freely add to design sets, i.e., to use them as piles. At the time of the case study, this was only achievable by creating a new set every time that holds the new alternatives.

My interactions with architects resulted in feature suggestions that somewhat parallel those made by the game designer in the case study. For example, architects asked for integrating DesignSense with parametric modellers like Grasshopper (for both sending and receiving alternatives) and supporting 3D views of alternatives. One computational designer also asked me for dynamic (pile-like) sets as well.

Navigating alternatives in DesignSense can be a generic approach that is agnostic to the game genre for which we are generating the content and the type of that content (puzzles, maps,... etc.) or the generative technique used. However, the suggestions above hint at the benefits of customizing navigation interfaces, like DesignSense, to the particular demands of each domain.
6.5 Reflecting on DesignSense

I conclude in this section with a high-level reflection on the core contributions of DesignSense in light of my initial goals and the lenses I developed. This is not a formal evaluation of the interface, but a synopsis of how I see its core strengths as its designer.

At its core, DesignSense aims at navigating a large number of alternatives while enabling both qualitative and quantitative assessment of alternatives in a way that supports the sensemaking loop and builds on techniques from visual analytics. The combined qualitative and quantitative assessment is achieved via interactive and coordinated form and data views that support bidirectional selection and inspection of design alternatives. This two-way communication enables a conversation to arise between the qualitative and quantitative criteria. Data does not only drive forms but is also driven by it. The cycle of alternating among data views (scatterplot, PCP) and between data and form views can enable a continuous conversation without compromising either.

To further aid (and capitalize on) the coupling of form and data, operations like grouping (design sets, clustering), information-seeking (filtering, zooming, undoing), and annotating (comments) are supported. Creating groups allow designers to put order into the design space. Each group constitutes a selection of alternatives, and every selection captures a criterion (qualitative, quantitative, or both). Information seeking allows designers to focus on or exclude portions of the design space. Designers can zoom to one group upon which they can further divide it into smaller groups then possibly view the smaller groups in light of the
larger design space. Finally, annotations allow designers to add their subjective semantics into the groups and to remember and build on their previous progress in navigation.

Furthermore, complex selection queries can be made visually and incrementally using different logical operations to accommodate a variety of design criteria. The same operations can be applied to subtract, intersect, or add groups to each other. Supportive analysis techniques like clustering and Pareto analysis can be called upon at any point in the navigation process while being informed visually and impacted-by/impacting the current selection.

In addition to that, users can alternate between different levels of design analysis: overview (where they analyze sets of designs), comparison (of two or more alternatives in details) and inspection (of a single alternative across views and a single alternative in details like 3D and enlarged images) without switching modes. Finally, the interface components can be resized, moved, and customized to support the changing needs of design projects. This flexibility, in addition to on-demand context-menus, allows users to capitalize on large displays.
Chapter 7

Evaluation

After implementing the first prototype of DesignSense, I conducted a focus group review. The goal was to receive an initial formative and immediate assessment of the design decisions on DesignSense. For the characteristics of DesignSense tool and the stage of its implementation, this approach is cost-effective and appropriate [70,91], with the caveat that a more structured (task-based) evaluation of the tool is due for the next iteration.

7.1 Evaluated Features

Figure 7.1 shows the interface of DesignSense at the time of the evaluation.

![Figure 7.1: The interface of DesignSense at the time of evaluation.](image)

The features that were supported at the time of the evaluation include: The PCP, scatterplot, and the Augmented Gallery (without the inspection panel), and coordinated
brushing and linking on those views; information-seeking; design sets; clustering (by all the available quantitative dimensions); the ability to move and resize the panels on the interface.

7.2 Setting

The participants were gathered in an online video conferencing system (Skype Business) due to the social-distancing, and two participants were located in a different geographic location. The session lasted about two hours. The session was recorded on video, and meeting notes were taken during the session.

The participants were composed of a group of eight architectural designers and computational design researchers at the same architectural design firm. All eight participants were familiar with generative design and have used gallery interfaces. Please note that the review was not based on a comparison of DesignSense features to other tools but rather on receiving feedback for its own proposed features and how the practicing expert designers receive them. Among participants, four were expert computational designers, of which two were specialized in building performance.

Two researchers led the session. One researcher presented DesignSense, and the other researcher moderated the session for keeping the focus on the system features and goals. Since this was the first review of DesignSense by professional designers and using a real-world project, the aim was to discover what participants expect from the system in a free-flow discussion loosely structured around the system features. The participants were told that they would be asked how DesignSense can be used for exploring the alternatives generated for a project in their firm. They were also asked to comment on what additional features or improvements can make it an effective system. Finally, they were asked to focus on the features of DesignSense rather than the architectural project presented in it.

In the first half-hour, the features of DesignSense were demonstrated using a generative design project from the participant’s firm that was shared earlier. This was followed by an open-ended discussion and informal feedback about what the participants thought of the prototype’s features. During the discussion, the interface DesignSense was frequently revisited.

One researcher reviewed the video recording and the meeting notes to identify the salient issues that emerged during the session. Generally, the participants agreed that DesignSense features for supporting navigation tasks are either comparable to or enhanced from other navigation interfaces, such as Design Explorer [7] that is used in the firm’s architectural practice.

7.3 Results

Below is a summary of the key points made during the discussion categorized by themes:
Visualization and shifts in design concerns A salient theme emerged around distinguishing and prioritizing the data visualized based on the two concerns, namely: design form and building performance. Both involve using different suites of tools while working closely with each other on a design project. For example, the result of performance analysis is reported to designers to inform their design decision-making; changes in the forms imply changes in the performance measures.

The participants with interest in building performance expressed a potential integration of their usual analysis process into DesignSense. One such procedure was creating a value function over the performance objectives of interest. This involved a weighted sum of the normalized performance values where the weights denoted the preferences within a given project. The value function was used in the firm to rank design alternatives and to insert some form of needed subjectivity into decision making. A different type of procedure is sensitivity analysis, which aimed at finding the input parameters that impacted the value function the most. Other types of analysis that they expressed interest in included descriptive statistics, trend lines, data normalization, prediction through surrogate modelling, and ANOVA tests.

The participants who were more focused on form were more interested in working with both design forms and data closely. They asked for the ability to inspect a design’s geometry in more detail, e.g., they asked for "...bigger images or 3D views" and found that "image thumbnails alone to be too small". One participant proposed a balanced emphasis on views of form and performance data, "...links back to form and performance and how do you balance between them and how do you actually get form to be part of your decision making". In terms of screen space, the participants preferred that more of it was assigned to form views rather than data views such as the parallel coordinates.

Scatter plot Although a simple and very common visualization, the current tool used by the designers in the firm only featured a static scatterplot. The participants were interested in the ability to brush data between scatterplot and other visualizations, and highlighting on the scatterplot on selection was received positively. They asked for a demonstration of brushing on various parts of the interface multiple times with different "design questions".

Clustering The participants found the clustering feature very relevant. The combination of clustering, creating sets and brushing all contributed to filtering down design alternatives and was found to complement each other. A remark was also made about its potential for "...making the decision making" process 'easier'. The following quote demonstrates using clustering as filtering: "...I see,...two different perspectives... from the designer perspective, I truly believe that the clustering, grouping, and filtering... basically most of them belonged in my interpretation to the filtering family of operations.... The participant did not describe the second perspective; delved into other details about post-processing done in performance analysis.
Set-based Exploration The participants valued the idea of using sets to store selections of alternatives based on their design preferences or criteria. One participant stated that "design does not happen in a day" and added "...how do you save the clusters if you are done and I got some kind of curious about that but you just modify the CSV?". This is a planned and important feature, but missing in the current implementation. Other participants agreed on the utility of mechanisms for storing, retrieving, and recalling the content of the sets, "...this would enable designers to continue where they left off". One participant took that further and stated that with the existence of many alternatives in a generative design process and noted "we would like to evaluate a specific alternative only when needed". There was an agreement on using both automated clustering and manual sets creation, which does not exist in the current tool they use. The clustering of solutions in sets based on select parameters and using K-means analysis was found potentially useful for grouping families of designs.

Derived Data One participant suggested providing features that can generate or derive additional data from given performance data. At the least, they wished to see functions that can present descriptive statistics (e.g., min, max, average, std) for comparing families of solutions. They noted that "...organizing big data sets is one thing and [creating] results out of it another...for example how to normalize the results...", which followed by "...we try to avoid picking one winner [and instead] we try to find the characteristics of winning designs [by looking at data in general]". Such functions can be of great utility when integrated into design-decision workflows. One participant pointed out the difficulty in "subjective intervention" when working with derived data. Indeed, data visualizations may present such insights, e.g., parallel coordinate plots display some form of "normalized" data as all the dimensions are visualized with the same length. But of course, this is not sufficient alone to make inferences on overall design performance. As for the 'subjective intervention' or design preferences as suggested by the participant, they mentioned that assigning "weights to different concerns, such as energy [use], daylighting and view" can be a useful solution.

Diagonal Sorting Upon seeing the sorting capability of thumbnails in the Augmented Gallery view, the participants asked for additional sorting strategies. They showed how they use a particular sorting that they call "diagonal sorting" for laying out the alternatives manually. Diagonal sorting would make it possible to "scan the thumbnails from one corner of the grid to the other", knowing that they are transitioning in a somewhat continuous way. This was missing in the regular row-by-row, left-to-right, and top-to-bottom sorting which required going back and forth between the beginning and end of the rows. However, the

---

1 This refers to the preference weights assigned to each performance objective to compute the *value function* mentioned earlier. Doing so allows designers to arbitrate between different objectives –by assigning higher weights to objectives deemed more important- albeit at the cost of lost transparency and implicit bias.
example they showed included 100 solutions that can fit on one printed page. These layouts will require a focused investigation on how scaleable they can be.

**Comparison** The participants showed an interest in being able to compare a small number of design alternatives in detail. They pointed at a different interface for comparing designs side-by-side. This feature is now implemented in the prototype but was not part of the evaluated version.

**Multiple Ways of Visualising Design Forms** The participants requested supporting multiple images for each design alternative to be able to view different performance metrics. For example, they mentioned using data mapped on the views for showing *energy, daylight, solar radiation* metrics.

**Flexible Layout** The participants appreciated the ability to freely resize and move the different panels in the interface of DesignSense. One participant pointing out that *"...the way people do design reviews until today is they print things up so that they can get a broad view and they can make comparisons"* but that with current tools *"...you are stuck in a screen"*. 
Chapter 8

Discussion

Matches and Mismatches. The lenses in Chapter 5 highlighted some requirements that DSN tools could benefit from. These requirements then shaped the features of DesignSense. The evaluation shows that they seemed to resonate with the participants. To be more specific, three categories of features resonated the most with participants, judging from the positive feedback they received or the discussion they provoked.

The first category relates to the coupling of form and data, which is a requirement that I carried along since the early stages of this work. Coupling in the evaluated version consisted of coordinated selection that originates from both data and forms views. The second category is set-based exploration, which includes all the features that support the creation and manipulation of sets like coordinated selection, disjunctive queries, information seeking, design sets and clustering. The interest in these features can be seen as a response to the large design spaces of generative design, hence motivating systematic set-based navigation. The third category of features contains more specific features that were received positively, like supporting the scatterplot view and flexibly changing the interface layout.

Participants also proposed multiple features that highlighted some of the system’s mismatches at the time of evaluation. These include: assigning more screen space to form views, integrating more data analysis into navigation, and linking the representations of individual alternatives on data and form views.

Navigation + Analysis. The evaluation revealed the need for better supporting the process of design decision-making and the demand for integrating design navigation with data analysis (i.e. Design Analytics). This can be accomplished through visually-assisted support for techniques such as sensitivity analysis and value function elicitation, among others.

Emphasis on Selection. Throughout DesignSense, I placed selection at the core. Selection is important because it can be the starting point of other tasks such as comparison, summarizing and drilling down [87]. Most of the supported interactions such as brushing, clustering, creating sets, computing the Pareto frontier, zooming, or filtering all either stem from or contribute to the current selection. The different selection operations of ad-
dition, replacement, intersection or toggling are also consistently applied to most of these interactions. This was accomplished while retaining visual feedback guiding all these interactions. However, the participants in the evaluation were silent about disjunctive queries (Section 5.3.1) which extend the conjunctive (narrowing) queries prevalent in DSN tools. Future evaluations should address this.

The participants showed interest in both clustering and design sets and saw those as means for developing a navigation paradigm that relies on selections (set-based exploration). In particular, they expressed their interest in the ability to examine sets (families of designs) before studying individual alternatives. However, factors like the study setting and time limits did not permit drilling down into how exactly they saw clustering or sets to be contributing to set-based exploration. Here again, a detailed evaluation is needed.

**Designers and Scientists.** The evaluation clearly highlighted the different concerns that designers and building scientists had when using a DSN interface like DesignSense. In fact, Matejka et al. [63] made a similar distinction regarding the potential users of Dream Lens: "We have found that users of generative design solutions tend to fall into two broad categories: The 'engineer' types, who heavily focus on the simulated properties (such as weight, strain, etc.) of the produced designs, and pay little or no attention to the appearance of the artifact; and the 'artist' types, who are less concerned with the model properties, but care greatly about the aesthetics of the generated design" [63]. Matejka et al. also argue that accommodating both groups (and those in between) can encourage them to appreciate each other's concerns. The evaluation I have conducted contributes specific feature suggestions by each of the groups.

**Revisiting Shireen et al.** The studies on how designers made sense of design spaces [87] highlighted behavioural patterns like creating and organizing design collections, adapting workspaces to their tasks, and other actions such as sorting and scanning. These patterns were translated into interface features in DesignSense, such as clustering, collections, and the ability to edit the interface layout. The spatial metaphors of Divide, Place, and Mark [86], that are based on the study, were either:

1. Implemented at the time of the evaluation and received positively, like flexible workspace layouts (Divide Zones), design sets (Divide Designs), sorting by data in the gallery (Place to Arrange), or adding comments on sets (Mark/Notes).

2. Suggested by the participants. For example, comparing few alternatives (Place to Compare) was requested, the same goes for diagonal sorting (Place to Scan), and value function elicitation (Mark/Ranking).


This may act as a formative validation of these metaphors, owing to the percentage of features they propose that were either already implemented or suggested by participants.
Revisiting the Research Goals. Going back to the two goals I outlined in the Introduction:

**Goal 1** Reflect and support the way designers explore and make decisions.

**Goal 2** Address some of the challenges that are implied by generative design and which hinders its potential. Namely, I focus on the challenges of "choice overload", and "imprecise metrics".

Imprecise metrics create a challenge for reliably navigating alternatives and judging their merit. But I have replaced the goal of addressing their imprecision in favour of tightly coupling the qualitative and quantitative aspects of alternatives during navigation (form and data coupling). Providing such coupling can reduce the negative impact of imprecise quantitative metrics by allowing designers to rely more on qualitative assessment in their navigation, e.g., through visuals of design forms. After all, the problem of "imprecise metrics" is not a product of poor navigation tools but is attributed to the tacit and complex nature of many design qualities that defy precise quantification. This –admittedly difficult– problem starts before navigation. Still, I argue that navigation interfaces should be prepared to respond to it, especially given the unlikelihood of completely resolving this problem any time soon.

My research goals could be then restated as:

**Revised Goal 1** Reflect and support the way designers explore and make decisions.

**Revised Goal 2** Address some of the challenges that are implied by generative design and which hinders its potential. Namely, I focus on the challenges of choice overload, and tightly coupling form and data.

**Form and Data Coupling: a Bird-View.** Coupling the visual appearance of the phenomenon being visualized with the data that describes it is not an exclusive demand of design domains. I expect other domains to follow in suit and recognize this demand as well. For example, the case studies I presented in Section 6.4 suggest a similar need when navigating alternatives in other domains like game design and consumer shopping.

I generally do not find the same adoption in the engineering and scientific domains. While it may be the case that the characteristics of these domains\(^1\) did not require it, it is as likely that their current tools simply did not support it. For example, this was indeed the case with the astronomical data analysis tools, as reported by McCurdy and Meyer [64]. As the authors illustrate, astrophysicists were using a set of tools for visually exploring galaxies, and another set for data analysis and visualization. Bridging this gap was a major contribution of their work.

\(^1\)For example, data records in the scientific domains are often not alternatives to choose from but observations to understand and analyze. There may also be a lesser reliance on implicit criteria in these domains.
Navigating with Design Sense. Design space navigation interfaces, like the one described in this thesis, will not only augment navigation but can also influence, directly or indirectly, the whole generative design process leading to navigation. Whether in how problems are modelled and up to the methods through which design alternatives are sampled and evaluated. This is because I assume that designers inevitably limit their generative models (by baking-in their design knowledge in a way that makes them less expressive) because of the challenge they foresee in navigating a larger design space. Designers may also be tempted to reduce the number of alternatives generated for similar reasons.

The process of generative design and the challenges it faces are not unique to architectural design. Generative techniques are used in other design and engineering disciplines as well. I argue that different disciplines can be placed on a spectrum depending on the prevalence of data-driven versus form-driven design processes and culture. The increasing use of data at the various design stages is not likely to stop any time soon. Yet I argue that the over-reliance on data can equally misguide the navigation of alternatives as the lack it, albeit in different ways. When not provided with DSN tools that adequately facilitate form and data coupling (under choice overload conditions), designers may disproportionately rely on data analysis at the onset of generative projects. I then see this work as a small step towards reconciling the tacit and explicit or the intuitive and the analytical.

The challenge of navigating generated design alternatives remains regardless of the decisions made in the modelling and generation stages. I argue that the existence of "trustworthy" navigation tools, i.e. tools that ameliorate the above challenges, can have an impact that propagates back to how designers model their solutions and generate design alternatives. One can imagine a situation where designers no longer need to limit the expressivity of their generative models or the scale of how they sample them; that is if they trusted that they could properly make sense of what they will generate, without forgoing their design sense.
Chapter 9

Conclusion

The computational generation of design solutions increasingly applies across various disciplines such as engineering, industrial design, visual arts, and architecture. But the success of generative design rests on the ability to navigate through the generated alternatives. Information visualizations have been used to support this task, but my critical review reveals mismatches between characteristics of the (architectural) design domain and the proposed approaches.

Generally, the current practice uses a suite of mainstream visualization systems that do not involve design forms (e.g., in the shape of images or 3D models of the design alternatives) or design space navigation (DNS) tools that leave more to be desired in terms of tightly coupling design forms and data. Filtering down the generated alternatives based only on explicit criteria (e.g., performance data) can be shortsighted as it may ignore the implicit criteria that are challenging to quantify (e.g., aesthetics). Other mismatches include: (1) not directly addressing the overload of choice created by large design spaces; (2) limited support for expressive selection mechanism that accommodates the fluid nature of design criteria; (3) not adequately supporting the sensemaking process that designers engage in when navigating alternative, e.g., through supportive operations for grouping, annotating and information-seeking.

Identifying these mismatches relied on critically analyzing current DSN interfaces through a set of lenses. These lenses were developing through a combination of eliciting expert’s feedback on early prototypes and building on studies that describe designers’ behaviour when navigating large design spaces. Beyond identifying these mismatches, I have proposed applying techniques borrowed from Visual Analytics to develop a DSN tool (DesignSense) that addresses them. I also present multiple case studies that demonstrate using DesignSense in navigating alternatives in different domains.

Finally, this thesis presents a formative evaluation of DesignSense in a focus-group setting with professional architects and computational designers. The evaluation shed more light on the potential of DesignSense (e.g., set-based exploration, form and data coupling)
and some of its limitations at that time (e.g., lacking more integrated data analysis and demanding bigger roles for forms views).
Chapter 10

Future Work

The requirements generated from the literature analysis and the feedback received from the evaluation both motivated a set of new features that are now part of DesignSense (as described in Section 6.2). These include: Coordinated and detailed inspection; supporting 3D views and multiple images per alternative; custom dimensions for clustering; finding the Pareto frontier on the scatterplot (the trophy icon); signalling the current selection operation (a label at the top left); on-demand context menu; and the Inspection on Hovering mode; augmenting the PCP with tooltips for a more precise selection and data-reading; supporting alternatives side-by-side on the Augmented Gallery when the thumbnails are large enough.

More of the results of the evaluation remains to be implemented, the same applies for the navigational patterns which Shireen et al. [86,87] observed. Furthermore, this work needs a more structured (task-based) evaluation of the current tool features, possibly by comparing it to other DSN tools. The architectural generative design case study in Section 6.4 can be a basis for that.

I want to investigate the comparisons of design sets as a step towards Set-based exploration. This may involve generating descriptive statistics on these sets, as the participants in the evaluation suggested. More dedicated views may also be used to visual sets and the relations between them [13], alternatively the visual colour encoding on DesignSense (which I intentionally restricted to a single colour) could be utilized in comparing few sets. Sets may also be enhanced with some form of tagging or ratings and implementing supportive mechanisms for organizing and operating on these sets.

On the visualizations side, I am interested in extending the visualization techniques supported by DesignSense to support scatterplot matrices, histograms and tables. This will also necessitate considering how the interface layout could respond to adding or removing new views.

At the moment, DesignSense automatically enforces a preemptive no-overlap rule between moving panels on the interface. This rule takes control from the users of the system and contradicts the suggestions by Shireen et al. [87] for affording high flexibility for users.
over how they arrange their workspaces. I intend to consider more permissive strategies like semi-automated or optional forcing.

On the analysis side, I would like to explore alternate clustering techniques, form-driven (e.g. visual similarity) clustering criteria and consider some form of dimensionality reduction as long as it can be implemented in an integrated and flexible manner. The formative evaluation also suggested integrating techniques like sensitivity analysis and value function elicitation into the navigation of alternatives.

In this thesis, I have focused on navigating already generated alternatives. This is a decision that I made to focus on the interface features concerning navigation only, especially those that tackle the challenges of choice overload and form/data coupling. Earlier in Figure 1.1, I presented a model of generative design and argued that generative models could be refined based on the insight gained during navigation. However, navigation (with data visualizations) can be more tightly linked to generation, i.e. on a systemic rather than cognitive ground. Examples of that include design steering and interactive optimization [27,46], or Cartesian products and interactive alternative’s sketching [67]. Promising future work is then to explore the potential of combining these techniques with navigation in DesignSense.
Bibliography


[76] Peter Pirolli and Stuart Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. Proceedings of International Conference on Intelligence Analysis, pages 2–4, 2005.


