Interfaces for Design Space Exploration

Sara Garcia[1], António Leitão[2]
[1,2]INESC-ID [2]Instituto Superior Técnico, Universidade de Lisboa
[1]saraflgarcia@gmail.com [2]antonio.menezes.leitao@tecnico.ulisboa.pt

A Generative Design System (GDS) allows for the generation and exploration of a wide number of design alternatives and for the automation of analysis and optimization processes. Algorithmic Design (AD) tools effectively support the development of GDSs, but they tend to be less appealing for the usage of such systems, as they rely on complex algorithmic descriptions of the design that quickly become overwhelming for less experienced programmers. The usage of GDSs is facilitated by Design Space Exploration Interfaces (DSEIs), which allows users to iteratively explore, visualize, and select design alternatives among the multidimensional design space defined by the GDS. DSEIs promote the collaboration between designers, clients, and end-users, making GDSs more interactive and more accessible. DSEIs rely on graphical user interfaces that relieve users from the burden of dealing with AD programs. The creation of such interfaces can be automated, so that they can be quickly created and modified. Although AD-based DSEIs exist for at least three decades, they have been more intensively researched and commercialized over the past eight years. In this article, existing AD-based DSEIs are reviewed, characterized, and compared according to several criteria, such as: navigation, visualization, search, ranking, grouping, filtering, and selection techniques. From this comparative study, a set of guidelines for the development of DSEIs is proposed.

Keywords: *Design Space Exploration, Algorithmic Design, Graphical User Interface, Data Visualization.*

INTRODUCTION

Generative Design Systems (GDSs) allow for the generation of a large set of designs alternatives – called the design space – and for the exploration of such space, in the search for one alternative that best suits users' needs. The *development* and *usage* of a GDS may be regarded as two independent activities, as they can be performed by different persons in different places and moments in time.

The *development* of GDSs can be supported by different tools. Some of the most common are Algorithmic Design (AD) tools, which permit the development of GDSs through algorithmic descriptions based on visual or textual programming languages. However, independently of the programming paradigm, AD is difficult to grasp for those unfamiliar with programming, since AD descriptions rely on a spaghetti of boxes and wires (in visual languages) or complex written descriptions (in text languages). Moreover, using AD tools, design alternatives are typically generated and visualized one at a time, making it difficult for the user to simultaneously compare multiple alternatives.

These problems are mitigated with Design Space Exploration Interfaces (DSEIs). AD-based DSEIs support the *usage* of GDSs, allowing the exploration, visualization, comparison, and selection of design alternatives within a design space without exposing the user to the complexity of AD.

DSEIs rely on User Interfaces (UIs) that (1) conceal the complexity of AD descriptions and only display the options that users are potentially interested in (Ambrosius, 2015); (2) promote remote collaborative design between designers, end-users, clients, and other stakeholders, letting end-users co-design and configure products and enabling mass customization (Schiftner, Höbinger and Huard, 2015), and (3) protect the authorship of AD descriptions, since only the UI is disclosed. In short, DSEIs make GDSs more accessible and more interactive. Nevertheless, DSEIs face two main challenges: first, the search for feasible design alternatives can be time-consuming and inefficient, since design spaces are frequently very large and filled with incomplete or meaningless solutions; second, the navigation and visualization of multidimensional design spaces is challenging (Mohiuddin and Woodbury, 2022).

The workflow of the *development* and *usage* of an AD-based GDS is illustrated in Figure 1: (1) in a first moment, the developer (e.g., a designer with programming skills) uses an AD tool to develop a GDS (which produces a design space); (2) in a second moment, the user (e.g., a designer, end-user, client, or other stakeholder) uses a DSEI derived from the AD tool to search for design alternatives within a design space (which can be narrower than the one defined in the AD tool), until selecting a design.

A back-and-forth process is expected to occur between the *development* and *usage* of such a system, as a result of the dialogue between the system developer and user. As such, it is beneficial to automate the creation of DSEIs, so that, when changes in project's requirements are demanded, the developer can edit the AD program and quickly regenerate the DSEI. This allows for the rapid production of UIs customizable to the specificities of a given company, project, or user.

In the following section, we present a review of existing AD-based DSEIs. This review's methodology is based on literature review and systematic analysis of the selected tools and their documentation.

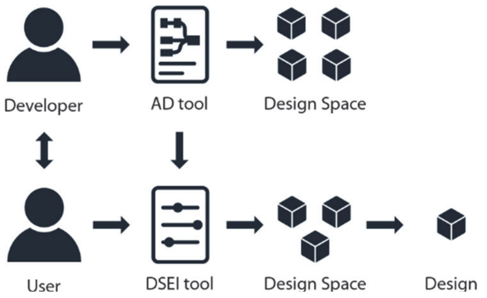


Figure 1. Workflow for the development and usage of an AD-based GDS.

DESIGN SPACE EXPLORATION INTERFACES

DSEIs allow users to explore, visualize, compare, and select designs and parameters through simple and intuitive interfaces, which include multiple types of representations that are usually synchronized. This diversity of representations is beneficial, since they provide different kinds of interactions and interpretations that complement each other.

In this section, we analyze nine AD-based DSEIs: Autodesk's Dialog Control Language (DCL), (Ambrosius, 2015); Design Explorer (Thornton Tomasetti and CORE studio, 2015); Design Gallery (Mohiuddin and Woodbury, 2022); Autodesk's Dream Lens (Matejka *et al*., 2018); Human UI (Heumann, 2016); Navigator (Garcia and Leitão, 2022); Autodesk's Refinery – an evolution of Project Fractal (Autodesk, 2016); ShapeDiver (Schiftner, Höbinger and Huard, 2015); and MakerBot's Customizer (Williams, 2014). Apart from DCL, which was launched in 1990, the remaining tools have been developed in the past eight years. The selected tools present a wide diversity of features that properly illustrate the topic addressed in this paper.

Table 1 shows a comparison between these nine DSEIs along several dimensions, which can be grouped into four categories: general features, designs, parameters, and management of alternatives. The following subsections discuss each feature, illustrated with a simple design of a colored box defined by four input parameters – length (l), width (w), height (h), and color (c), and two output parameters – volume and surface area.

Table 1. Comparison of design space exploration interfaces.

	DCL	Design Explorer	Design Gallery	Dream Lens	Human UI	Navigator	Refinery	Shape Diver	Customizer
Purpose	Create UI	Explore	Explore	Explore	Create UI	Explore	Explore	Configure	Configure
Version	Stable	Stable	Prototype	Prototype	Stable	Prototype	Beta	Stable	Stable
Platform	Desktop	Web	Web	Desktop	Desktop	Both	Desktop	Web	Web
AD tool	Text	Various	Visual	Various	Visual	Text	Visual	Visual	Text
Relation	Indirect	None	Indirect	None	Indirect	Direct	Direct	Hybrid	Direct
3D viewer		●	●		●	●		●	●
Modeling tool	●		●		●	●	●		
Thumbnail gallery	●	●	●	●		●	●		
3D viewer gallery		●		●	●		●		
Juxtaposed viewer				●					
Editable 3D viewer									
Sliders	●			●	●	●		●	●
2D sliders		Plots	Plots	Plots	●	Plots	Plots		
3D sliders						Plots			
Parallel sliders		Plots	●			Plots	Plots		
3D Parallel sliders									
Range sliders			●	●	●	●			
Categories input	●				●	●		●	●
Text input	●				●			●	●
Graphic input					●			●	
Output parameters	●	●	●	●	●	●	●	●	
Other plots					●				
Group Parameters	●				●				●
Export/import	●	●	●	●	●	●	●	●	
Disable/hide	●				●				●
Search		●	●	●		●	●		
Rank		●		●		●	●		
Group			●						
Filter		●	●	●		●	●		
Select			●	●		●			

General Features
General features characterize the DSEI tool (regarding its purpose, version, and platform), the AD tool, and the relation between them.

Purpose. The objective addressed by the DSEI: (1) create UIs from AD descriptions (e.g., Human UI), (2) publish product configurators for end-users (e.g., ShapeDiver), and (3) provide exploration tools for designers (e.g., Design Explorer) – such tools are called *Design Space Explorers*, which provide search mechanisms to guide the user throughout the design space (Woodbury, Datta and Burrow, 2000). The target user plays a major role in the development of a DSEI, as it will determine its complexity and the number of available features. DSEIs can also contemplate multiple users – such as the system developer, the designer, and the end-user, who typically have decreasing degrees of freedom as the design space becomes increasingly restricted (Castro e Costa *et al.*, 2020).

Version. The DSEI' development stage: prototype (e.g., Design Gallery), beta (Refinery), and stable (e.g., ShapeDiver). Some commercial tools are free open source (Design Explorer), while others are freemium (ShapeDiver). The development stage of a DSEI influences its availability and reliability.

Platform. DSEIs can be based on a web UI (e.g., Design Explorer), a desktop UI (e.g., Refinery), or both (Navigator). Web UIs facilitate remote collaborative design and are suitable for different devices – from desktop to mobile – and different interaction modes – from mouse/keyboard to touch (Schiftner, Höbinger and Huard, 2015). However, they tend to be slower than desktop UIs (unless the dataset is pre-generated, as in Design Explorer). Out of this paper's scope are natural interfaces – such as tangible and kinetic – and virtual reality interfaces.

AD tool. AD tools can be classified according to the algorithmic paradigm used, as visual-based AD tools, which rely on graphic descriptions, and text-based AD tools, which rely on written descriptions. When compared to the latter, the former are more intuitive and easier to learn (since graphical representations are more familiar to designers), support direct manipulation interfaces, and provide real-time feedback; however, they scale poorly with design complexity, i.e., programs' performance and readability decrease as complexity increases (Aish, 2013).

DSEIs can be produced from a given GDS description that can be defined on: (1) a visual-based AD tool, such as Grasshopper (Human UI, ShapeDiver, and Design Gallery) and Dynamo (Refinery); (2) a text-based AD tool, such as AutoLISP (DCL), OpenSCAD (Customizer), and Khepri (Navigator); (3) or in various AD and non-AD tools, such as parametric CAD tools (e.g., CATIA – exemplified with Design Explorer) and generative design tools (e.g., Dreamcatcher – exemplified with Dream Lens).

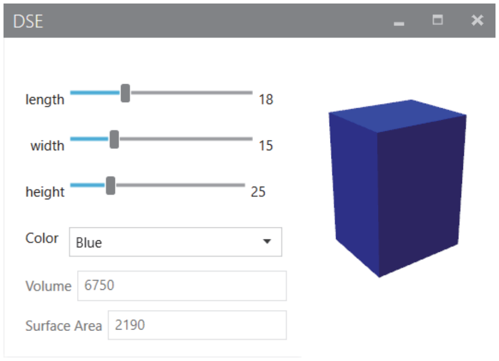
Relation. The relation between AD program elements and UI elements can be characterized by three strategies: (1) direct, where UI elements are directly related to AD parametric elements (Customizer, Navigator, and Refinery); (2) indirect, where program elements and UI elements have separate definitions (DCL, Design Gallery, and Human UI); and (3) hybrid, which combines both direct and indirect strategies (ShapeDiver). The more direct the relation, the lower the UI development cost but the less customizable the UI is.

There is also the case where the UI does not have any dependency from the AD tool (Design Explorer and Dream Lens); in this case, a design dataset must be pre-generated.

As an example, observe the interface depicted in Figure 2, which can be generated by one of the four pseudocodes observable in Figure 3. The pseudocodes illustrate the direct and indirect strategies (on the left and on the right, respectively), relying on text-based and visual-based AD tools (on the top and on the bottom, respectively). In the direct strategy, UI elements (sliders and dropdown lists) are inferred from input parameters, which can be identified by their value (numerical and categorical, respectively) or their range of variations (interval and comma-separated lists, respectively).

UI elements labels, values, and ranges correspond to the parameters' names, values, and ranges, respectively. AD elements can have (or not) special tags that identify them as UI components: in this case, the elements in square brackets set the range of choices, and the hidden tag determines a parameter that will not be revealed in the DSEI.

In an indirect strategy, UI elements must be specified in the program. In a hybrid strategy, UI elements can both be created either by AD parametric elements or UI-related AD elements.



length 18
width 15
height 25
Color Blue
Volume 6750
Surface Area 2190

Figure 2. A DSEI showcasing a GDS that produces colored boxes (created with Human UI)

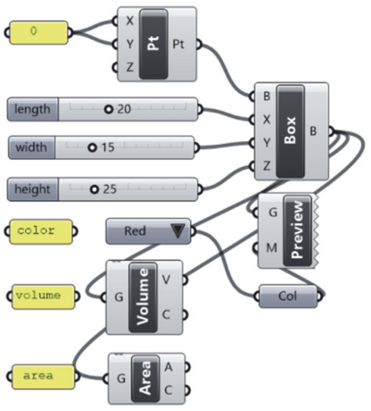
Figure 3. Four AD programs that can generate the DSEI depicted in Figure 2.

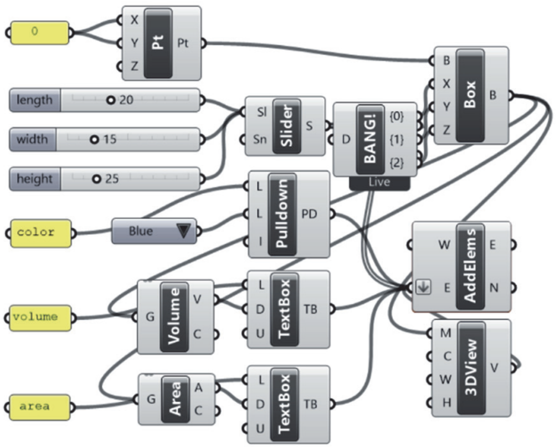
Direct

```
p = xy(0, 0) // [hidden]
l = 20 // [5:50]
w = 15 // [5:50]
h = 25 // [5:100]
c = blue // [red, green, blue]

colored_box(l, w, h, c) =
  with layer(c) do
    box(p, l, w, h)
  end

box_properties(l, w, h, c) =
  [:volume => box_volume(l, w, h),
   :area => box_area(l, w, h)]

window(colored_box, box_properties,
desktop, 3d_viewer)
```

Indirect

```
colored_box(l, w, h, c) =
  with layer(c) do
    box(p, l, w, h)
  end

box_properties(l, w, h, c) =
  [:volume => box_volume(l, w, h),
   :area => box_area(l, w, h)]

s1 = slider(5:50, 20, "l")
s2 = slider(5:50, 15, "w")
s3 = slider(5:100, 25, "h")
d = dropdown([red, green, blue], blue, "c")

t1 = textbox(volume)
t2 = textbox(facade_area)

window([s1, s2, s3, d, t1, t2], desktop,
3d_viewer)
```





Designs

Designs can be visualized in DSEIs in different ways, as described below and illustrated in Figure 4.

3D viewer. A 3D model of the design with editable camera properties (pan, zoom, and orbit), embedded in the DSEI. The model is updated whenever parameters change.

Modeling tool. The design can be visualized in a modeling tool, like a CAD tool such as AutoCAD (DCL and Navigator) and Rhinoceros 3D (Human UI and

Navigator), or a BIM tool such as Revit (Design Gallery, Navigator, and Refinery). This feature is more common in desktop DSEIs. In some cases, DSEIs can be portable between several modeling tools – e.g., Navigator, which borrowed this capability from the Khepri AD tool (Leitão, Castelo-Branco and Santos, 2019). Also, some DSEIs allow users to export the design to a CAD file format (Customizer and ShapeDiver). Comparing modeling tools and 3D viewers, the former better suit designers, since they allow to further develop the result of the exploration process in specialized design tools, while the latter better suit end-users, since designs are more responsive to parametric changes.

Thumbnail gallery. A collection of small 2D images of design alternatives, allowing a side-by-side comparison of such alternatives (e.g., Design Explorer). Those images can be automatically created when exporting a design's parameters.

3D viewer gallery. A collection of small 3D viewers. The camera properties of the models can be synchronized, so that when one viewer changes, the others change accordingly (e.g., Dream Lens).

Juxtaposed 3D viewer. A set of 3D models with different opacities on top of each other – allowing a more direct comparison between the designs (Dream Lens).

Editable 3D viewer. A 3D model editable by direct manipulation (e.g., by dragging points). This mode is less explored than the remaining ones but promises to be more intuitive for designers, since it avoids shifting their focus between design representation and parameter representation (Mohiuddin and Woodbury, 2022).

Parameters

Parameters can be classified by the type of their values, such as: *numerical parameters* (comprising a set of numbers), *categorical parameters* (comprising a set of elements), *text parameters* (comprising a description), and *graphic parameters* (comprising an image) (Wilke, 2019). These categories can be further decomposed: for instance, *binary parameters* can be regarded as categorical parameters composed of two elements. Parameters can also be classified into *input parameters* (independent and manipulable) and *output parameters* (dependent and unmanipulable). The latter are typically related to design's properties (e.g., area, weight, cost, and maximum displacement).

Different parameter types require different interactive components (or widgets), such as sliders or numerical input fields (in the case of numerical parameters), list or dropdown boxes (in the case of categorical parameters), and toggles or checkboxes (in the case of binary parameters).

As visible in Table 1 in slider elements, users can visualize and manipulate those parameters or only visualize them (plots). Moreover, users can visualize/manipulate one or multiple design alternatives and/or change one or more parameters simultaneously. Parameter's visualization types are described below (whereas numerical parameters are illustrated in Figure 5).

Sliders. 1D numeric representation, where each parameter is manipulated independently and one alternative is displayed at a time. Sliders can have diverse shapes, such as parallel or radial, and can be combined with numerical input fields. Despite being frequently used, sliders' manipulation in high-dimensional design spaces can be time-consuming and frustrating (Lopes, Anjos and Jorge, 2018).

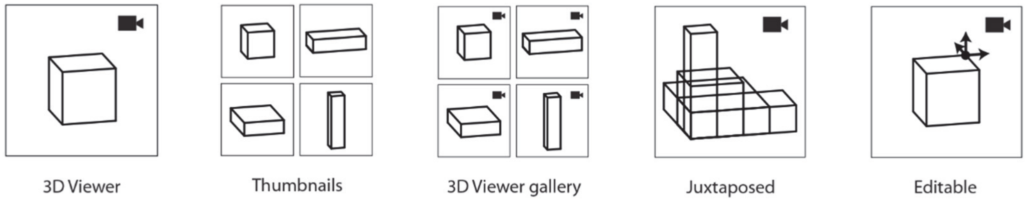


Figure 4. Designs' visualization and manipulation in DSEIs.

2D sliders. 2D numeric representation in the format of 2D sliders, which allow to visualize and manipulate two parameters simultaneously. Scatter plots can be regarded as 2D sliders that only allow visualization, commonly displaying multiple alternatives (represented by dots). 2D sliders can display more than two dimensions, e.g., given by the X-axis, Y-axis, dots color, and dots size (Refinery). Multiple 2D sliders can be presented side-by-side (Design Gallery) or combined in an axonometric projection (Lopes, Anjos and Jorge, 2018). A paradigmatic example of a 2D slider is a 2D color picker.

3D sliders. 3D numeric representation in the format of 3D sliders, allowing the manipulation of three parameters. Note that none of the analyzed DSEIs possess this feature. On the other hand, scatter plots allow the visualization of three parameters simultaneously. This representation type brings some issues: the manipulation of a 3D object in a 2D medium is not straightforward and visualization is hindered by occlusion, an issue that can be overcome with virtual reality UIs. These problems are aggravated when the number of dimensions increases.

Parallel sliders. 1D numeric representation in the format of parallel coordinate plots, displaying multiple alternatives represented by polylines connecting slider values. Polylines can be static (e.g., Navigator) or can be sketched and edited by dragging points, segments, or the whole polyline (Design Gallery).

3D Parallel sliders. Parallel sliders displayed in 3D space, where alternatives are represented as polylines connecting dots with different heights – corresponding to the values of each parameter (Contreras, 2019). This type of sliders is not present in any of the analyzed DSEIs.

Range sliders. Sliders that present one or multiple subranges among which alternatives can be selected. They allow users to filter or narrow the design space. The concept of range sliders can be extended to 2D and 3D sliders, whereas the subrange is represented by a 2D region and a 3D volume, respectively.

Categories input. Categorical parameters representation, which can be in the format of list boxes, checkboxes, radio buttons, etc.

Text input. Text parameters representation. The UI element receives a free-form text as input, which can be composed of a set of alphanumeric characters.

Graphic input. Graphic parameters representation. Graphic inputs can comprise a bitmap image or a vector graphics image (including, for example, a given geometry). This type of parameters can add an extra degree of freedom to the AD process.

Output parameters. Output parameters representation. Since these parameters are calculated from input parameters, they should be updated whenever an input changes. Output parameters play a key role in the exploration process, as the more information the DSEI displays, the more informed the selection process becomes and the better solutions can be achieved.

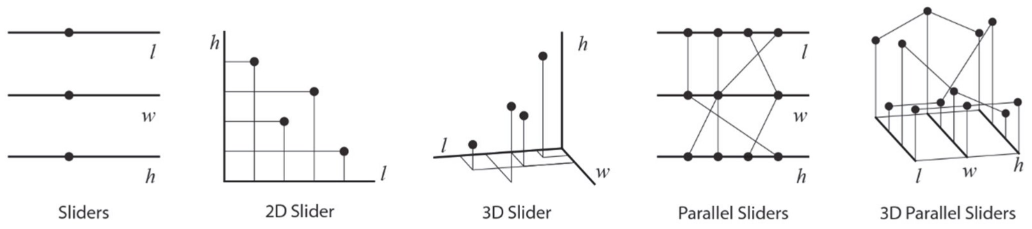


Figure 5. Numerical parameters' visualization and manipulation in DSEIs.

Other plots. Beyond scatter plots, parameters can also be represented by other types of plots or charts, such as bar and pie charts (Human UI). This feature is useful for quantitative analyses of input/output parameters.

Group parameters. Parameters can be grouped into labeled clusters, using tabs or child windows. This feature becomes particularly important when dealing with high-dimensional design spaces.

Export/import. Parameters can be stored in spreadsheet-like files (and be retrieved from such files), which is a useful feature when one wishes to save designs. Importing parameters is a mandatory feature for DSEIs that are independent of AD tools. Files can contain a single design or several designs arranged in tables (Human UI), while 2D images of designs can also be created.

Disable/hide. Parameters can be disabled (meaning that they are visible but not manipulable) and hidden (meaning that they are invisible and not manipulable), or enabled and shown likewise. This feature is useful when the availability of some parameters depends on the value of other parameters.

Management of alternatives

DSEIs can provide features to find and manage meaningful designs among a large number of alternatives, allowing users to search, rank, group, filter, and select alternatives. These features are described below and illustrated in Figure 6.

Search. DSEIs can support both divergent exploration and convergent optimization. The search for designs can be made by manual manipulation of widgets or by automated search modes, such as: (1) exhaustive search, which gives all possible designs from a given range (Design Explorer, Design Gallery, Dream Lens, Navigator, and Refinery), (2) random search (Navigator and Refinery), (3) similarity-based search, which gives designs similar to a given design or hybrid designs in between (Navigator and Refinery), and (4) optimization, which gives optimal designs considering given objectives (Navigator and Refinery). Some modes are faster and more explainable than others, making them more or less suitable for end-users (Garcia and Leitão, 2022).

Rank. Designs can be sorted by their names or by some input or output parameter value (Dream Lens, Navigator, and Refinery). Moreover, importance weights can be given to each parameter – which can be automatically calculated from the selection of favorite designs (Dream Lens).

Group. Similar alternatives can be grouped into clusters, for instance, by classifying alternatives with text tags (Design Gallery).

Filter. The design space can be narrowed, e.g., by manual manipulation of range sliders, or by queries regarding parameters.

Select. One or more alternatives can be selected, e.g., by picking designs in a thumbnail gallery or by picking dots in 2D sliders. The selected alternatives can be synchronized in multiple representations (e.g., by highlighting both images and dots).

As we have seen, DSEIs provide several features for dynamic exploration, visualization, comparison, and selection of designs. In the next section, a discussion regarding the analyzed DSEIs will be conducted.

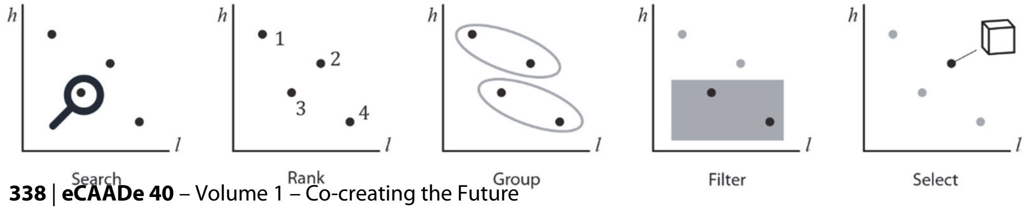


Figure 6. Management of alternatives in DSEIs.

DISCUSSION

In the previous section, we presented a description of 30 DSEI features, while discussing some of its advantages and disadvantages. We also conducted a qualitative comparative study where nine AD-based DSEIs were analyzed regarding the mentioned features. From this study, several considerations can be withdrawn:

- DSEIs targeting end-users tend to have less features when compared to DSEIs targeting designers, as the experience of the former regarding AD and GDSs is lower;
- Desktop-based DSEIs present more features than web-based ones;
- Regarding the relation between DSEIs and AD tools, we observe that the more indirect the relation, the more features it presents (as it provides more options to customize the interface);
- Most DSEIs allow the visualization of single and multiple designs; however, they lack juxtaposed viewers and direct manipulation of the model (which would be more intuitive for designers);
- The majority of DSEIs allow the manipulation of parameters through 1D sliders (which only permit the manipulation of one parameter and the visualization of one design at a time) but lack mechanisms to manipulate several parameters and visualize several alternatives at a time (e.g., 2D/3D sliders and 2D/3D parallel sliders);
- The analyzed DSEIs present few features to group and disable/hide parameters;
- Most DSEIs present a small variety of options to generate design alternatives (e.g., by random, similar, and optimization methods) and lack proper grouping mechanisms.

Based on the previous study, we propose a set of guidelines for the development of AD-based DSEIs:

- They should allow the manipulation of input parameters (represented by widgets) and the visualization of the resulting design and output parameters (which are updated whenever input parameters change);
- They should allow the manipulation of multiple parameters and design alternatives at a time;
- They should allow the comparison of designs by displaying galleries of 2D images or 3D models;
- They should allow the comparison of parameters, by displaying 2D/3D scatter plots and 2D/3D parallel coordinates plots;
- Their functionalities should support different types of parameters (numerical, categorical, text, and graphic) and consider both input and output parameters;
- They should provide mechanisms to search, rank, group, filter, and select designs;
- They should address GDSs with any design space dimension within different application domains (e.g., architecture and product design);
- Their target users should be carefully delineated to determine the number and characteristics of the displayed features;
- The relation between the DSEI and the AD tool should be considered, regarding the intended speed of creation of a DSEI from an AD tool and its desired level of customization.

For future DSEIs, we recommend the development of underexplored interaction and visualization modes, such as 3D parallel sliders, editable 3D viewers, 2D/3D sliders, and juxtaposed viewers.

CONCLUSION

Generative Design Systems (GDSs) allow users to create and explore design spaces. The development of GDSs can be done with Algorithmic Design (AD) tools, through algorithmic descriptions. However, such descriptions quickly become overwhelming for GDSs users with little or no AD experience. A more intuitive way to use GDSs is given by Design Space Exploration Interfaces (DSEIs), which allow users to interactively explore, compare, and select design alternatives, and share such alternatives with different stakeholders. DSEIs relieve users from the burden of dealing with complex AD programs.

Moreover, they can be automatically generated from any GDS depicted in an AD tool.

Although AD-based DSEIs exist for at least three decades, they have been more intensively researched and commercialized over the past eight years. In this paper, we revised nine existing DSEIs along 30 features grouped in four categories, which characterize: (1) the DSEI and its relation with the AD tool; (2) the visualization and manipulation of designs; (3) the visualization and manipulation of parameters and properties; and (4) the mechanisms to search, rank, group, filter, and select designs. We also conducted a comparative study and proposed a set of guidelines for the development of DSEIs.

Ultimately, we expect DSEIs to contribute to approximate end-users and designers to AD and GDSs. For future work, we plan to evaluate DSEIs regarding usability (through user tests) and performance (through benchmark tests).

ACKNOWLEDGEMENTS

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with references UIDB/50021/2020 and PTDC/ART-DAQ/31061/2017.

REFERENCES

Aish, R. (2013) ‘DesignScript: Scalable Tools for Design Computation’, in *Proceedings of the 31st eCAADe Conference*, pp. 87–95.

Ambrosius, L. (2015) *AutoCAD Platform Customization: User Interface, AutoLISP, VBA, and Beyond*. Indianapolis, Indiana: Sybex, John Wiley & Sons.

Autodesk (2016) *Autodesk Project Refinery*. Available at: http://www.autodesk.com/campaigns/refinery-beta (Accessed: 10 February 2021).

Castro e Costa, E. *et al.* (2020) ‘Enabling parametric design space exploration by non-designers’, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 34(2), pp. 160–175.

Contreras, C.H. (2019) ‘Surfaces Plot: A data visualization system to support design space exploration’, in *Proceedings of the 37th eCAADe and 23rd SIGraDi Conference*, pp. 145–152.

Garcia, S. and Leitão, A. (2022) ‘Navigating Design Spaces: Finding Designs, Design Collections, and Design Subspaces’, *International Journal of Architectural Computing*, 0(0), pp. 1–20.

Heumann, A. (2016) *Human UI*. Available at: http://grasshopperdocs.com/addons/human-ui.html (Accessed: 10 February 2021).

Leitão, A., Castelo-Branco, R. and Santos, G. (2019) ‘Game of Renders: The Use of Game Engines for Architectural Visualization’, in *Proceedings of the 24th CAADRIA Conference*, pp. 655–664.

Lopes, D., Anjos, R. and Jorge, J. (2018) ‘Assessing the usability of tile-based interfaces to visually navigate 3-D parameter domains’, *International Journal of Human-Computer Studies*, 118, pp. 1–13.

Matejka, J. *et al*. (2018) ‘Dream Lens: Exploration and Visualization of Large-Scale Generative Design Datasets’, in *Proceedings of the 2018 CHI Conference*, pp. 1–12.

Mohiuddin, A. and Woodbury, R. (2022) ‘Interactive Visualization for Design Dialog’, in Gero, J.S. (ed.) *Design Computing and Cognition ’20*, pp. 491–508.

Schiftner, A., Höbinger, M. and Huard, M. (2015) *ShapeDiver*. Available at: http://www.shapediver.com/ (Accessed: 10 February 2021).

Thornton Tomasetti and CORE studio (2015) *Design Explorer*. Available at: http://tt-acm.github.io/DesignExplorer/ (Accessed: 10 February 2021).

Wilke, C.O. (2019) *Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures*. Sebastopol, CA, USA: O’Reilly Media.

Williams, A. (2014) *OpenSCAD for 3D Printing*. CreateSpace.

Woodbury, R., Datta, S. and Burrow, A. (2000) ‘Erasure in Design Space Exploration’, in Gero, J.S. (ed.) *Artificial Intelligence in Design ’00*. Dordrecht: Springer.