A Non-linear and Divergent Digital Learning Resource for Design Computation

Aut, Serdar

Publication date
2024

Document Version
Final published version

Published in
Data-Driven Intelligence

A Non-linear and Divergent Digital Learning Resource for Design Computation

Serdar Aşut
Delft University of Technology, The Netherlands
{s.asut@tudelft.nl

This paper presents a novel approach to developing Digital Learning Resources (DLR) for Design Computation (DC). Learning DC requires the students to develop cognitive skills in algorithmic thinking and practical skills in using specific software. Few learning resources integrate cognitive and practical skills, often prioritizing the skills related to tool use with a focus on software functionalities. They typically follow linear narratives in audio-visual or text-based tutorials, which do not align well with the essence of computational thinking. The DLR presented in this paper is a self-paced learning resource that integrates a web of interconnected concepts, methods, tools, and instructions on a non-linear interface, and it aligns better with the divergent qualities of computational design thinking. It is developed for an MSc-level course that introduces computational design. This paper presents its design and implementation, evaluation of its pilot use, and directions for future improvements.

Keywords: *Design Computation, Architecture Education, Digital Learning Resources, Design Pedagogy, Computational Design Education*

INTRODUCTION

This paper presents a novel approach to developing Digital Learning Resources (DLR) in Design Computation (DC). Within the scope of this paper, DC refers to computation-based design practices (including parametric, generative, and algorithmic design) in which distinct computer software is used. Following the taxonomy presented by Caetano et al. (2020) the scope includes practices that integrate digital design, which is considered the use of computer tools in the design process, and computational design, which entails using computation to develop designs. The most commonly used computer software for DC in architecture includes Grasshopper (GH), Dynamo, and GenerativeComponents.

One of the common aspects of this type of software is that it offers a visual programming interface on which the user can develop parametric design definitions in the form of data flow diagrams. The output of the definitions is represented as design variations within the Computer Aided Architectural Design (CAAD) package in use. Celani and Vaz (2012) argue that visual languages can be very useful for making architecture students understand general programming concepts, and they emphasize the importance of the ability to shift between different representation methods, from more concrete to more abstract. Thus, DC by visual programming became a common practice among professionals and students in architecture.

Learning DC requires the students to develop cognitive skills in algorithmic thinking and practical skills in using specific software. One of the most common learning practices is browsing the online DLRs. Their qualities differ significantly, as theoretically anyone can (and do) produce and share them through the Internet. Few DLRs integrate cognitive and practical skills, often prioritizing practical skills with a focus on software. They typically follow linear narratives in audio-visual or text-based tutorials, which do not align well with the essence of computational thinking. The most widespread methods for teaching parametric design rely on following step-by-step tutorials or using ready-made scripts (Vazquez, 2023). Moreover, they neglect the varying entry levels of students. Despite the question of quality, this broad range of learning materials has become one of the most common resources for students.

This paper aims to explore a novel pedagogy that problematizes the design of DLRs on DC. It proposes a novel DLR design with better affordances, which aligns well with the very nature of computational thinking and is adaptable to students' diverse needs.

BACKGROUND

DC has gained significant value in architecture, and its integration into curricula is advancing considerably, requiring new pedagogies that align well with the nature of this domain and the changing educational landscapes. Already in the 1990s, Akin (1990) stated that the computer had the potential to radically change three fundamental ingredients in the classroom: student, instruction, and instructor and that changes of this kind spell out a commensurate change in design pedagogy. Similarly, Oxman (2008) stated that design computation and digital design had an influence on the development of theoretical, computational, and cognitive approaches by various researchers as a foundation for design education and pedagogy.

Vazquez (2023) presents a literature review claiming limited literature on pedagogical strategies on this subject. It includes some works that utilize analog methods to teach computation without computers (Alalouch, 2018; Howe, 2011), material-based computational frameworks (Abdelmohsen and Massoud, 2021), and explorations of incorporating digital tools in the design studio (Oxman, 2008; Schnabel, 2012; Agkathidis, 2015).

Research on DC education usually excludes teaching the use of the tool per se. Moreover, research that problematizes explicitly the learning materials' design is very limited. Even though existing research is valuable for developing effective pedagogies, this situation indicates a gap in this field. This paper aims to address this gap by demonstrating a DLR design approach. The DLR is developed and tested in the Introduction to Computational Design course taught at the Building Technology MSc. program.

COURSE SETUP

The DLR presented in this paper is developed primarily for the Introduction to Computational Design course, offered at the Building Technology MSc program of Delft University of Technology. It is a 5-EC compulsory course followed by, on average, 60 students in the second quarter of the program's first year. This is the first course that introduces DC in the program. The learning objectives are defined as:

- Understanding the uses of digital technologies in the design and production of the built environment.
- Understanding spatial data acquisition techniques.
- Using database and information models for design development and analysis.
- Formulating algorithms to generate solutions to architectural design and building engineering problems.
- Creating visual programs for design development, analyses, and fabrication.

The entry level of the students differs significantly, from complete beginner to experienced, as the group is diverse and international. The course starts with a survey to understand the entry level of the students in terms of their skills in using the software and their understanding of DC in general. Based on the survey results, the students are grouped as teams of 2 to 3 to let students with similar entry levels work together and to adjust the expectations from each team accordingly. The grouping is done solely based on the survey answers given by the students, and no further evaluation is performed. It is observed that this method works well for the beginner and expert teams. However, the differences within the average teams may need to be represented more precisely. This method can be improved for better team formation.

Learning activities include lectures introducing concepts and theories and workshops where students work on design exercises. The students receive formative feedback from the tutors during these exercises and individually submit their work after each workshop. These exercises are designed to guide the students towards their final design, which is a team project. During self-study, they integrate the workshop topics into their team project, and they receive formative feedback on it during the workshops. The team project is assessed twice, through the mid-term and final submissions.

The team project and individual workshops focus on data-driven design, performance analysis and optimization, fabrication, and presentation of a small-scale architectural design. The assignment of the last two years was designing a bus stop by:

- Using Geographic Information System (GIS) databases to find a bus stop location and to determine the needed size of the bus stop,
- Generating geometric design variations using population data retrieved from the GIS database,
- Implementing environmental analysis on different design variations,
- Optimizing the geometry towards fabrication,
- Producing scaled models of the design using digital fabrication.

BLENDED LEARNING

Over the last few years, we have been experimenting with blended learning (BL) approaches to improve this course by allocating more time for in-depth design discussions during the course hours and answering the changing needs of students in a more flexible and adaptable way. Garrison and Kanuka (2004) define BL as the thoughtful integration of classroom face-to-face learning experiences with online learning experiences, and they emphasize the appeal of integrating the strengths of synchronous (face-to-face) and asynchronous (text-based Internet) learning activities, as well as the considerable complexity in its implementation with the challenge of virtually limitless design possibilities and applicability to so many contexts. According to Norberg et al. (2011), online and blended learning is the new normal, and they increase access for students to education, responding to their lifestyles through flexible learning opportunities, while they also generate controversy about their quality compared to face-to-face classes.

BL requires combining different learning modes such as synchronous and asynchronous, guided and self-paced, distance and on-campus, or individual and collaborative. Our approach to BL is to make software training self-paced and asynchronous and use the classroom hours for higher-level discussions on design and computational thinking. Software instructions during the classes are ineffective as adjusting the pace in such a large group of students with different entry levels is challenging. The nature of software instruction involves time-consuming and repetitive tasks, which results in the loss of contact hours, which are already limited. Also, it is not a particularly inspiring task for the teaching staff, who prefer using the time with students for more advanced and design-related subjects. BL can make it possible to teach software use more effectively.

One of the most critical challenges for sustaining the quality of education in such a BL approach is the design of the learning materials. Our understanding of learning materials needs to adapt to this new learning mode, exploring novel pedagogies that align well with the changing landscapes of education.

DESIGN OF THE DIGITAL LEARNING RESOURCES

The DLR presented in this paper is a self-paced learning resource that integrates a web of interconnected elements on a non-linear interface. It enhances students' cognitive skills by providing a learning environment that aligns well with the divergent qualities of DC. It accommodates diverse starting points and does not rely on a fixed learning sequence. The learning experience is personalized, allowing students to adjust the pace according to their needs and create unique paths within the web. It is designed as an interactive interface, similar to a data flow diagram (Figure 1).

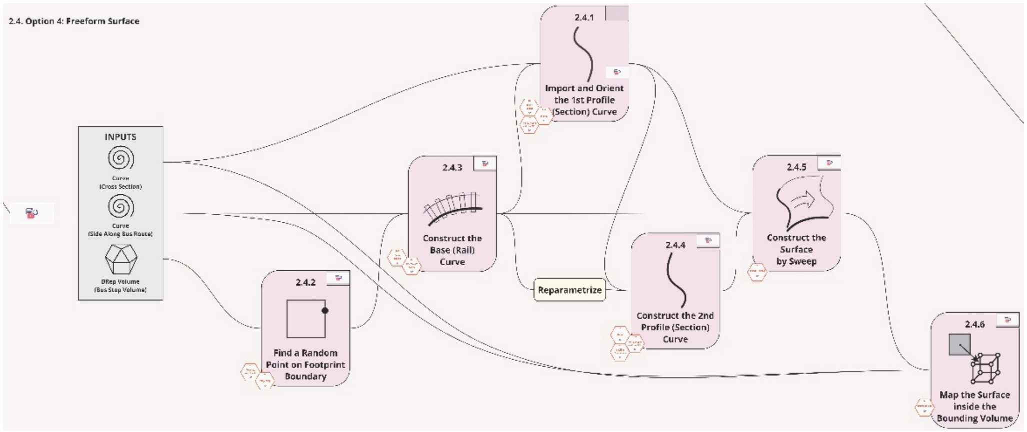


Figure 1
A partial view from the interface

Aish and Hanna (2017), when evaluating the learning curves within DC systems, point out that most systems offer multiple representations including a visual graph-based data flow representation, geometric representation, and sometimes a text-based program representation. The DLR presented in this paper is designed similarly to a graph-based data flow representation. Thus, the interface already invites students to think through data flow diagrams. It is visually and functionally similar to the common visual programming interfaces. Therefore, it can reduce the cognitive load for the students to relate the information on the interface to the actual software used to implement it.

The diagram is structured comparable to the algorithm that the students need to develop for the design assignment. In this course, the students use GH for programming. However, the principles behind the algorithm are universal, and they can be used with other visual programming interfaces or programming languages as well. To emphasize this, the diagram is software-neutral, prioritizing the cognitive skills instead of the software used. The same (or slightly modified) interface can be used to teach DC using other software in the future.

The nodes in the diagram contain content elements such as video tutorials, images, or quiz questions (Figure 2). They are represented with a custom icon that communicates their content. Each element is concise, making the use of the interface more active and engaging. Also, it makes it easier to maintain the interface by quickly upgrading and replacing the elements when needed. While the students browse the interface, they click on the nodes to study the elements organized within thematic groups.

One of the elements that are placed on the nodes is called Glossary Items (GI). These are videos that include short and practical explanations of

Figure 2
The nodes containing the elements

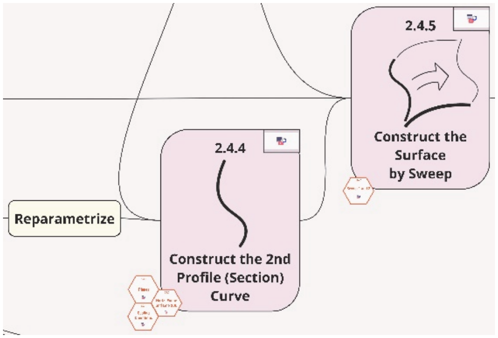


Figure 3
The nodes containing the elements

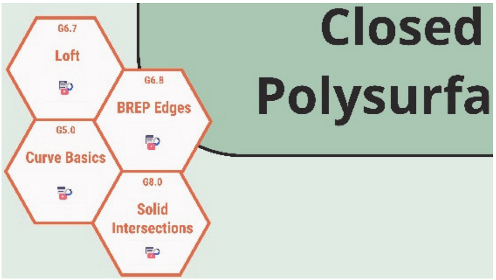
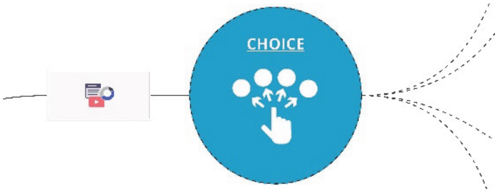


Figure 4
A node indicating diverging paths in the flow



fundamental concepts and methods such as mathematical operators, Boolean, surface normal, or planes (Figure 3). These explanations are generic, and they can be re-used in other assignments. GIs are placed next to every element in which they are used, making it easy to find them whenever needed.

Theoretically, the students can browse the interface freely and study the elements in any order, even in a non-linear way. They can start from any node they want, skip the ones they do not need to study, revisit the ones they need more time to understand, and imagine new connections between them, which aligns with the principles of divergent thinking.

Design thinking in general, and DC in particular, resemble divergent thinking, especially in the ideation and conception phases. Marron and Faust (2018) describe divergent thinking as a process of retrieving existing knowledge and associating and combining unrelated knowledge in a novel and meaningful way. The protocol analysis of Lee and Ostwald (2022) presents clear evidence of the connection between divergent thinking and the production of ideas. Tran et al. (2022) state that divergent thinking is one of the most intensively studied component processes involved in creative thinking and that it involves the exploratory and flexible generation of multiple possible ideas, often in response to open-ended or ill-defined problems. According to Sun et al. (2020), while one's creativity can be influenced by both domain-specific factors (e.g., domain knowledge and expertise) and domain-general factors (e.g., creative potential and personality), training on cognitive skills especially for divergent thinking is an effective approach to improving creativity. The design of the DLR can stimulate divergent thinking and enhance this cognitive skill already by means of its interaction with the user (Figure 4). To this end, it is designed to encourage the students to explore new connections between pieces of information and discover new ideas.

IMPLEMENTATION

The first step of implementation included making a GH script that includes all of the concepts and methods the students need to learn in the course. It also served as a sample for the design brief. It was divided into phases, and the distinct steps in each phase are defined. A flowchart diagram is made in Miro to visually represent the script, resulting in a layout comparable to the visual program in GH. In the diagram, each step of the computation is defined as a node. These steps are explained in short videos that include abstract diagrams, images, or screen recordings from the software, and they are placed on the nodes. The explanations did not include each GH component used in that step, assuming this is known to the students. However, a separate GI section was developed to contain videos, each of which explains a specific GH component in a basic and generic way. Wherever these components are used, they are cross-referenced to the relevant GI within the node to let the students study them whenever needed.

The nodes are connected with curves, similar to the wires in GH. Some curves contain quiz questions as another type of element placed before the nodes. The quizzes are made in H5P and integrated into Miro. They help the students check if they have gained the knowledge needed in the next node. Students who cannot answer correctly in this quiz are referred to the relevant node or GI to (re)study it.

CONCLUSIONS AND DISCUSSIONS

The first version of the DLR presented in this paper was used in the course between November 2023 and February 2024. Our initial observations demonstrate that it improves the learning experience and they provide directions for improvement.

The most significant benefit is that it considerably reduces the need for guidance compared to previous years when traditional video tutorials were provided to the students weekly. This gave us more time to engage in design-related discussions in class and allocate more time to beginner students. Additionally, experienced students felt more confident and motivated to experiment with new techniques. These arguments require further investigation in future research by comparing them with a control group. Also, the effects of this learning experience on the design outcomes can be investigated.

During the workshops, we noticed minor errors in the elements a few times. It was very easy to fix these mistakes and replace the items with upgraded ones as they were very concise. Most of the time, it was possible to replace them already during the workshop.

We concluded that Miro is not the ideal application for developing this interface. The interaction with the video content was not smooth, and they required a few clicks to start watching it. Also, it does not allow the customization of the thumbnail images, which would make it much easier to recognize the content of the elements. Moreover, the feedback tools that can be integrated are limited. For the improved version, we are exploring other applications to build the interface on and considering building a custom one from scratch. Using an open-source platform instead of proprietary software is considered more ideal as it would allow customization and it would be free to use both by the students and developers.

In the next version, we aim to demonstrate a more complex algorithm and allow for more divergence. This may make it more difficult to navigate while browsing the interface. The principle is that while the students browse the interface and study an element, they should always be aware of the whole diagram and not get lost in it. We are exploring the options to develop a feedback tool to ensure this. Also, a feedback tool that informs the students on their completed elements would be useful.

Another useful improvement would be better integration of the entry-level survey into the DLR. We are exploring options to let students see which elements they need to study more and which elements they can skip based on the survey results, enhancing their self-regulation and improving the learning experience.

Acknowledgments

This work is supported by "TUDelft Blended Learning Support". The author acknowledges the contributions of F. Alsaggaf, F. Gouwetor, L. Flieger, and M. Koch for their work in implementing the DLR presented in this paper.

REFERENCES

Abdelmohsen, S. and Massoud, P. (2021). 'A material-based computation framework for

parametric design education', *Open House International*, 46(3): pp. 459-475. Available at: https://doi.org/10.1108/OHI-02-2021-0043 (Accessed: 24 March 2024)
Agkathidis, A. (2015). 'Generative Design Methods - Implementing Computational Techniques in Undergraduate Architectural Education' in *Proceedings of the 33th International Conference on Education and Research in Computer Aided Architectural Design in Europe (eCAADe) [Volume 2]*, pp. 47-55. Available at: https://doi.org/10.52842/conf.ecaade.2015.2.047 (Accessed: 24 March 2024)
Aish, R. and Hanna, S. (2017). 'Comparative evaluation of parametric design systems for teaching design computation', *Design Studies*, 52: pp. 144-172. Available at: https://doi.org/10.1016/j.destud.2017.05.002 (Accessed: 24 March 2024)
Akin, O. (1990). 'Computational design instruction: toward a pedagogy' in Malcolm; Mitchell McCullough, William J (eds.), *The Electronic design studio: architectural knowledge and media in the computer era.* Cambridge: MIT Press, pp. 301-316.
Alalouch, C. (2018). 'A Pedagogical Approach to Integrate Parametric Thinking in Early Design Studios', *International Journal of Architectural Research: ArchNet-IJAR*, 12(2), p. 162.
Caetano, I., Santos, L. and Leitão, A. (2020). 'Computational design in architecture: Defining parametric, generative, and algorithmic design', *Frontiers of Architectural Research*, 9(2), pp. 287-300. Available at: https://doi.org/10.1016/j.foar.2019.12.008 (Accessed: 24 March 2024)
Celani, G. and Vaz, C.E.V. (2012). 'CAD scripting and visual programming languages for implementing computational design concepts: A comparison from a pedagogical point of view', *International Journal of Architectural Computing*, 10(1), pp. 121-137. Available at: https://doi.org/10.1260/1478-0771.10.1.1 (Accessed: 24 March 2024)
Garrison, D.R. and Kanuka, H. (2004). 'Blended learning: Uncovering its transformative potential in higher education', *The Internet and Higher Education*, 7(2), pp. 95-105. Available at: https://doi.org/10.1016/j.iheduc.2004.02.001 (Accessed: 24 March 2024)
Howe, N. (2011). 'Algorithmic Modeling: Teaching Architecture in Digital Age' in *Parametricism (SPC) ACADIA Regional 2011 Conference Proceedings*, pp. 31-37. Available at: https://doi.org/10.52842/conf.acadia.2011.x.a0p (Accessed: 24 March 2024)
Lee, J.H. and Ostwald, M.J. (2022). 'The relationship between divergent thinking and ideation in the conceptual design process', *Design Studies*, 79, p. 101089. Available at: https://doi.org/10.1016/j.destud.2022.101089 (Accessed: 24 March 2024)
Marron, T.R. and Fasut, M. (2018). 'Free Association, Divergent Thinking, and Creativity: Cognitive and Neural Perspectives' in, *The Cambridge Handbook of the Neuroscience of Creativity*, p. 261.
Norberg, A., Dziuban, C.D. and Moskal. P.D. (2011). 'A time-based blended learning model', *On the Horizon*, 19(3), pp. 207-216. Available at: https://doi.org/10.1108/10748121111163913 (Accessed: 24 March 2024)
Oxman, R. (2008). 'Digital architecture as a challenge for design pedagogy: theory, knowledge, models and medium', *Design Studies*, 29(2), pp. 99-120. Available at: https://doi.org/10.1016/j.destud.2007.12.003 (Accessed: 24 March 2024)
Schnabel, M.A. (2012). 'Learning Parametric Designing' in, *Computational Design Methods and Technologies: Applications in CAD, CAM and CAE Education*. IGI Global, pp. 56-70.
Sun, M., Wang, M. and Wegerif, R. (2020). 'Effects of divergent thinking training on students' scientific creativity: The impact of individual creative potential and domain knowledge', *Thinking Skills and Creativity*, 37, p. 100682. Available at:

https://doi.org/10.1016/j.tsc.2020.100682 (Accessed: 24 March 2024)
Tran, K.N., Kudrowitz, B. and Koutstaal, W. (2022). 'Fostering creative minds: what predicts and boosts design competence in the classroom?', *International Journal of Technology and Design Education*, 32, pp. 585-616. Available at: https://doi.org/10.1007/s10798-020-09598-7 (Accessed: 24 March 2024)
Vazquez, E. (2023). 'Teaching parametric design: fostering algorithmic thinking through incomplete recipes', *Open House International*, Vol. ahead-of-print. Available at: https://doi.org/10.1108/OHI-06-2023-0135 (Accessed: 24 March 2024)