How to write a thesis?

A Graduate Studio Guidebook for Generative Design Researchers

For graduation projects in the field of Generative Design and Augmented Urban Planning, supervised by Dr. Pirouz Nourian (p.nourian[-at-]utwente.nl), Assistant Professor of Digital Twinning @ the Department of Planning and Geoinformation Management (PGM), at ITC Faculty of Geoinformation Science, University of Twente, last time edited on maandag, 01 januari 2024

1. Purpose

This document is written to guide graduate students in the field of computational design and planning, especially those working on Generative Design and Generative Sciences subjects and supervised by Pirouz Nourian and his colleagues from 4TU[1] in their graduation thesis project (MSc and PhD), from a methodological point of view. For a basic introduction to generative design research methodology, watch this lecture (slides). For more information, read our position paper on Generative Design [1]. For a more elaborate introduction read our book chapter on generative design, a framework and a literature review about Augmented Urban Planning (spatial planning augmented with Planning Support Systems embedded in Digital Twins), and a forthcoming book chapter on Augmented Computational Design. The target audience of this guidebook might be the students of the following programmes:

A) MSc Geo-Information Science (ITC-UT)
B) MSc Architecture, Building and Planning (TU/e)
C) MSc Geo-Information Science (WUR)
D) MSc Spatial Engineering (ITC-UT)
E) MSc GIMA
F) MSc Industrial Ecology (Leiden-Delft)
G) MSc Civil Engineering (TU Delft)
H) MSc AUBS (TU Delft)
I) MSc Geomatics (TU Delft)
J) MSc Sustainable Energy Technology, track Solar Energy (TU Delft)

NB: Insofar as the procedural details are concerned, please refer to the specific programme manuals.

2. Overview

The main point of this document is to explain how to structure and write a graduate thesis on Generative Design, and in particular, how to structure the thesis proposal or the first introductory chapter with the following sub-sections: {Abstract, Objectives & Questions, [Disciplinary] Approach, Framework {Informatic Work-Breakdown Structure}, Problem Statement(s) {Given: Knowns/Inputs, Desired: Unknowns/Outputs}, Methodology {Type of Research, Fields and Methods}, Scope: what it could be but it does not aim to be}.

[1] The 4TU federation of the four universities of technology in the Netherlands (TU Delft, TU/e, UT, and WUR)

- Chapter 0/Proposal:
    - Abstract
    - Objectives, Questions, Deliverables
    - Disciplinary Approach (general methodology)
    - Framework & Problem Statement(s)
        - Informatic Work-Breakdown-Structure: how the problem is broken into smaller problems with inter-related inputs and outputs.
        - Given: Knowns/Inputs
        - Desired: Unknowns/Outputs
    - Methodology (specific methodology)
    - Scope: what it could be but it does not aim to be
- Chapters addressing sub-problems or phases of explorations

3. Introduction

A graduation thesis project at the master's level is arguably the last and thus the most important assignment a student does in partial fulfilment of the requirements for getting the highest possible educational degree, the Master of Science. Note that a PhD degree is a research degree and thus it is not categorized as an educational title. A thesis project at TU Delft will provide the student a period of roughly nine months, in which the student is free to explore a direction they see fit to their career plans, and thus it cannot be seen as merely a chance to prove yourself for the next/first job. A logical career plan is based on a vision/ambition for career development and a consistent strategy whose first move is to pick the right topic. A pragmatic way to look at the aptitude of a topic is to consider a thesis project as a challenge for skill development. What kind of challenges does the topic provide? What skills will you master after having conquered these challenges?

4. A Graduation Topic

While there exist a number of topics proposed by the school and the department for graduation projects, in principle you may also consider coming up with your own topic. View our list of proposed topics here: https://genesis-lab.dev/graduation/#topics

5. Graduating within a Company

Picking a graduation topic just because a company is paying for would be a strategic mistake. It could however be that a company happens to be interested exactly in what you are interested. If there are such matches, I would not hesitate to help you establish contact and get the sponsorship relevant for your project. In the past, I have even negotiated with companies on behalf of students for decent stipends. Please note that there may be serious considerations of intellectual property rights when working with companies. It should not be the case that an idea with valorisation potential is handed for free to a commercial body in exchange for a few months of internship stipends. It should also not be the case that the company regards the thesis project as an opportunity for hiring cheap labour force. However, keep in mind that getting a paid internship/traineeship position would imply engaging in day-to-day activities of the company, and occasionally doing work that is not necessarily part of your thesis. Everything considered, I would only recommend such joint gradation projects if they make sense from all these angles. By signing this document, among other things, you accept to not negotiate such matters with companies without having discussed with me first.

6. A Thesis or a Journal Article

In principle, instead of or in addition to a traditional thesis, you may consider publishing a journal article. This may have a number of key benefits, the most important of which with be structured clarity, brevity, and ultimate recognition of your work by the public.

7. Thesis Structure

There should be a first chapter in a thesis, which could be aptly be called **Introduction**, describing the structure of the thesis consisting of the following sections:

7.1. Abstract

An ideal abstract would be a 150-word-long paragraph with no jargon and no references, summarizing the problem and the proposed solution and thus the contributions of the thesis.

7.2. Context/Background/Motivation

What is the problem in lay/intuitive terms?; Why is it important?; Why are you interested (motivation)? You need to have two paragraphs explaining and highlighting explicitly the following matters:

- **societal relevance**: how is the result of this project potentially useful for the society
- **scientific relevance**: what new knowledge or know-how is this project going to add to the existing bodies of knowledge or what knowledge/know-how gaps are going to be bridged by this work.

7.3. Objectives, Deliverables & Question

the objective of research in computer science is almost always more important that the question. The objective is typically to devise a methodology to solve a problem or to formulate a problem to be solved using a well-known problem solving method. Thus, there needs to be a concrete description of the deliverables. The question is often times a rephrased version of the objective and thus it will be a ‘how to’ question. Consistent with the steps/work-packages mentioned in the “proposed methodology” there will need to be subordinate questions and smaller deliverable products. It is ideal to have a maximum consistency between these objectives, deliverables, and questions.

7.4. Scope

This section describes the nature of your thesis: what it is and what it is not. You will clearly state whether there is or there is not an architectural design component in your graduation project. If yes, the design component should be utilized as a context for formulating, prototyping, and testing the computational processes, i.e. a test-case (a.k.a. a case study). Then you give an address where in the thesis the design task, its objective, its requirements, and the design in response to those requirements is presented. Afterwards, you mention the names of disciplines[2] that relate to or constitute your multi-disciplinary field of research.

[2] A discipline is a branch of science or scholarly field of study that disciplines the way that branch of science is taught, communicated and practiced by having established academic textbooks, academic curricula consisting of courses with universal syllabi, methodologies, respected academic journals, scientific societies, and conventions. Physics, Mathematics, Chemistry, Electrical Engineering, Computational Biology, Computational Mechanics, and alike are examples of well-established disciplines.

Computational Design is not yet a well-established discipline but an emerging field of practice. There are not yet reliable text-books and not many specific-enough journals dedicated to computational design methods in architecture. Therefore, it is essential to describe the multi-disciplinary state of your subject and its methodology carefully and preferably in terms of an Euler Diagram or a Venn Diagram.
Afterwards, i.e. following a characterization of the disciplinary status of your research project you can add a sentence like the following: "While the thesis relates to the following subjects in some ways, they fall out of the scope the project:"; followed by a list of topics that may come to mind as related to the project and yet fall out of the scope because they are irrelevant from a theoretical point of view or simply because they would unnecessarily sophisticate and/or lengthen the process beyond the time-scope.

7.5. Framework and Problem Statement(s)

The main problem must almost always be broken down into smaller problems before deciding on ways to solve it. In doing so, we only pay attention to the informatics (i.e. the information content of) the inputs and outputs of each sub-problem without bothering with the way to solve each and every individual sub-problem. Describing the informatic inter-relations of such subproblems is what we colloquially know as the framework of the problem. Once this framework is set one can think about alternative methodologies to solve the [sub]-problems and compare them in terms of their efficacy or efficiency. The problem must be first introduced and explained in intuitive terms (in the context/background/motivation section). Such an explanation would provide a context and a motivation for a more specific description called the problem statement. A problem statement needs to be like a mathematical problem description, dry, clear, specific, knowns and unknowns. Ideally, the objectives, the questions, and the problem statement are all consistent with each other. What distinguishes a problem statement from a political propaganda talk is the informatic/mathematical specification of the nature of inputs and outputs, as well as the constraints and objectives/quality criteria to be satisfied/achieved.
Having a proper framework on the main problem will ensure that the methodological choices can be explained and justified properly.

7.6. Research Methodology

In this line of work, i.e. Generative Design, pure research is not relevant. Our projects are always Research and Innovation or Research & Development inherently, because our work is in the area of the "Sciences of the Artificial" [2]. For an introductory explanation to the questions and methodological approaches in 'generative sciences' see [3]. For an introduction to 'generative systems' watch this lecture on Generative Design Research Methodology based on [4].

Table 1: what are the sciences of the artificial, as opposed to the science of the natural?

The Sciences of the Natural	**The Sciences of the Artificial**
Physical Sciences and Life Sciences	Design Sciences and Optimization: Engineering Methods & Operations Research (Planning, Scheduling, Management)
concerned with the description, prediction, and understanding of natural phenomena, based on empirical evidence from observation and experimentation	concerned with reasoning, envisioning, and devising artificial systems based on given design requirements or optimization objectives
Questions	Problems
Theories as proven Hypotheses	Methods as approved Propositions
Explanation and Justification	Transformation and Reasoning
Deterministic Models, Stochastic Models	Configured Systems, Reconfigurable Systems
Mathematical and/or Computational Modelling, Analysis, & Simulation	Mathematical and/or Computational Problem Formulation & Problem Solving
How do things change?	How to change/make things?
MATHEMATICS	

To clarify what 'research' is all about, we should first locate research approaches relevant to generative design:

Table 2: types of scientific methods as to their epistemological approach, based on Philosophy of Science – Science Network TV

▲ Empiricism

← Atomism

Quantitative: [Spatial] Statistics	Qualitative: Interviews, Surveys, Workshops
Mathematical: Graph Theory & Game Theory	Philosophical: Logical Reasoning

Holism →

Rationalism ▼

Obviously, our line of work necessitates to follow a rationalistic-atomistic approach rather than empirical or holistic approaches.

To avoid further confusions regarding the difference between the process and the products of computational design research, I distinguish the meanings and differences of a few terms in the context of computational design research:

- Methodology: a structured collection of methods[3]
- Method: "a particular procedure for accomplishing or approaching something, especially a systematic or established one" (Oxford Dictionary)
- Technology: a structured collection of techniques[4]
- Technique: "a way of carrying out a particular task, especially the execution or performance of an artistic work or a scientific procedure"(ibid)
- Model: a mathematical/computational replica of a system, process, or construct

I reserve the terms models and methods for mathematical or algorithmic constructs and use the term technique for referring to matters pertaining to programming languages, testbed environments, execution dependencies and alike.
To give an overview of how you research has been structured, you depict a workflow in a flowchart, focusing on processes and the flow and the transformation of data throughout the process. Wherever the data is processes manually, you use shapes and arrows different from shapes and arrows conventionally used in drawing flowcharts. From the conventional shapes used in drawing flowcharts, I would recommend using rectangles as process indicators and parallelograms as data containers. Wires must be clearly directed. On the data containers you write the data type and data structure as if you are writing code in a strict programming language, e.g. '$A[i,j]$ as a matrix of double'.
I would strongly recommend you to make a graphical abstract, preferably referring to a toy problem, explaining what the methodology is about, what happens in every step to the inputs.
Note that your main product in a thesis project on Generative Design is going to be a methodology; that is distinct from your research methodology. As for describing your research and development methodology, i.e. the way you get from the objective to the results, you can refer to the a Computational Design Science Methodology [5] (chapter 1, chapter 2), A Design Science Research Methodology for designing Decision Support Systems [6], and A Methodology for Designing Information Systems [7].

[3] "A system of methods used in a particular area of study or activity" (Oxford Dictionary)

[4] A technology, literally meaning a science of craft, can be eventually operated by individuals that do not necessarily know deeply how to act following the underlying scientific methods. In that sense the important role of technology in computational design research is to bring scientific know-how into practice.

These methodologies agree on prescribing the following steps in the process:
According to [5] *&* [6]*:*

- Problem Identification and Motivation,
- Definition of Objectives for a Solution,
- Design and Development,
- Demonstration (which entails implementing the proposed solution),
- Evaluation (verification[5] and validation[6]), and
- Communication (scientific publication for peer-review)

Or similarly, according to [5] *&* [7]*:*

- Build (develop a solution by means of programming and mathematical modelling),
- Evaluate (verify and validate),
- Theorize (generalize, abstract, formulate, mathematically describe and explain), and
- Justify (mathematically prove that the results are correct, and/or your algorithm should converge)

7.7. Literature:

This section is not to be mistaken by [scientific/scholarly] literature review. Not every piece of text with a bunch of academic references is a scientific publication, but a scientific publication, apart from being scientific in terms of its information-content, has to have references quoted according to one of multiple prevailing protocols such as the IEEE or APA styling conventions. You will refer to relevant pieces of literature at the beginning of your methodology chapter, specifically once per each section in which you cite the most relevant papers and books from which you have learned, quoted something, adopted a method or used some construct. In such a section you mention the main lines of research literature that you have identified as pertinent to your topic, the keywords you use to search the body of literature, and the platforms on which you search[7], how you formulate advanced queries, and the criteria you consider in critically assessing the aptitude or relevance of what you find in the literature. Having a reference/citation manager such as Zotero or Mendeley would be strongly recommended. For a comparison, see this page.

[5] Ascertaining that we are doing things right. In computational research, verification is about testing algorithms and mathematical methods in terms of their response to well-known problems. It is often useful to test methods against symmetrical problems, test them against extreme cases, and test them against different data structures to ensure that they do not have bugs.

[6] Ascertaining that we are doing the right thing. Is the problem being solved the problem that really occurs in practice? Are our simplifications affecting the generality of the problem or can the problem be generalized in a more elegant way? Has anyone ever solved this problem in any other context?

[7] Look at one alternative research search engines: https://www.dimensions.ai/

7.8. Proposed [Design] Methodology:

Note that your work is expected to result in applicable methods, structured logically in relation to a specified type of design problems. In this section, you describe your main test case (a.k.a. case study) as a generic design problem in a particular context, e.g. a factory layout with some typical requirements etc. In addition, you mention a couple of so-called toy problems to be used in the course of research and development. A toy problem is a problem that is small and simple and yet is representative of many bigger or more sophisticated problems or the same genre, and thus it is interesting to utilize as a context for developing and testing our methods. Some very well-known toy problems are referred to as benchmark problems, i.e. problems for which there exist solutions that can be improved in terms of either effectiveness (better solutions according to an objective function) or efficiency (faster methods in terms of computational time complexity or more memory-efficient processes in terms of space complexity). In a preliminary assessment, you are expected to show preliminary results with respect to such toy problems.

7.9. Work Breakdown Structure & Time Planning

Abstract the process of conducting research and break it down into work packages based on the logical breaks between them. Then map the work-packages and their interrelation in at least two types of diagrams: a Gantt chart and a PERT chart.

7.10. Verification & Validation

In the context of computer science:

- verification:= "are we doing things right?"
- validation:= "are we doing the right thing?"

Verification in this sense is mostly concerned with the [mathematical] consistency of the model, whereas the validation step is concerned with the usefulness of the model.
In the context of statistical models, however, validation is mainly concerned with figuring out whether the predicted results of the model match observations. This is often combined with Calibration of the model parameters (if in the context of model-fitting of a parametric model) or a revision of the model formula/derivation process (if non-parametric).

8. Generative Design Literature

Generative Design in Architecture and Built Environment is an emerging field, for which there is hardly any comprehensive textbook. There is a lot of propaganda, delusion, delirium, and promotional material about parametric design and generative design in the wild that are often misleading. While there are many blurry cases and applications in which these approaches are all utilized in a single process, in order to clarify our definition of generative design, it is essential to distinguish the main two dominant paradigms in computational design and their differences:

Table 3: two dominant paradigms in computational design: Creative/Parametric Design vs. Generative/Combinatorial Design

Procedural Design	
Computational Design	
Creative Design	**Generative Design**
Parametric Design	**Combinatorial Design**
Continuous Geometry Variations	Discrete Geometry Variations
Inherently Analog	Inherently Digital
Feed-back Optimization	Feed-forward Optimization
Real Parameters ($\mathbb{R}^3$)	Integer Parameters ($\mathbb{Z}^3$)
Geometric Variation/Evolution	Topological Variation/Evolution
Formal Design Thinking	Functional Design Thinking
Additive Manufacturing	Modular Construction
Analogous to composing on the Violin	Analogous to composing on the Piano

You are advised to remedy your math knowledge by reading extra books e.g. [8] pp.137-181 [9] pp.29-51, [10], [11] pp. 9-26 & pp. 225-236, [12], and [13]. Most of the topics about computer geometry are covered in [14] pp. 1-5, 7-13, 21-24, 29-33 and [15] pp. 62-63. Highlighted pages are the least you are recommended to read from the following books. For computational/combinatorial design look at [16], [5], [17]. For getting started with programming, check [18] [19].
For getting a smooth introduction to the essential mathematics and computer science of generative design you can start from reading the preprints of a Work-in-Progress book by the author on Fundamentals of Spatial Computing and Generative Design [20] and [21]. For a relevant introduction to programming you can refer to [22], [23], and [24].

9. On Academic Writing

The most important point in academic writing is conciseness, which boils down to brevity and specificity. A well-written document that is only 10 pages long is 10 times more valuable than a mediocre 100-pages-long document. Reproducibility of whatever form of knowledge or know-how reported in the thesis must be the ultimate goal of academic writing. Most irrelevant words and paragraphs enter into academic papers due to several typical reasons:

- Avoiding the main subject, which should often be reporting a systematic workflow, its findings, and limitations.
- Showing nicety to the elders of science, referring to their work just out of courtesy.
- Exaggerating success
- Pretending to have invented something that is not really innovative.
- Competing with other authors & claiming that this work is right and they are all wrong.
- Shying off from admitting the limitations.
- Using poetic, literary, journalistic, or political language.
- Unnecessarily using adjectives, adverbs, and superlative forms.

Here are a few online resources on academic writing: first resource, second resource, and the types of academic writing.

10. On Computer Programming

Find the shortest possible introduction to programming in sections 1.4 and 1.5 of these lecture notes. I advocate for modular processes and vectorized coding [25]; and so, I would recommend that you study Linear Algebra more deeply. You can start from these lecture notes. Most of the projects in Genesis Lab can benefit from our home brewed software such as topoGenesis [26].

11. Educational Praxis

The following principles underpin the ethos and the pedagogy of our generative design education.

0) **Agility**: the frontline of digital technology is quickly advancing and disruptive technologies are steadily changing the nature of various sectors. This means that the scope and the required features of a project may change during its development [27]. Frequent changes in the project have also been identified as the co-evolution of problem-solution pairs (see [28] and [29]).
1) **Applicability**: Generative Systems and Sciences are applied primarily in the following areas.
    a. Digital Architecture, Engineering and Construction (AEC)
    b. Planning Support Systems and Spatial Decision Support Systems:
    c. Serious Gaming: Participatory Governance, Management, and Planning
    d. Industrial Engineering: Facilities Management, Facilities Planning, Operations Research
    e. Geo-Spatial Analysis, Geo Spatial Simulation, and Spatial Intelligence
2) **Reproducibility**: Open Science & Open Source guidelines are embraced as foundational principles
3) **Generalizability**: To differentiate the major and minor issues from each other, abstraction is necessary to clarify the structure of the problem. In the context of spatial design problems, in order to prioritize different aspects of design problems we must primarily address abstraction with respect to the mathematical representation media, i.e. graphs, topologies, and geometries (cf. a categorization of spatial abstractions in [30]). Such an approach brings about the possibility to generalize methods from one context to another. The nature of graphical and topological representations is discrete and combinatorial, and thus they perfectly fit into the paradigm of the digital. Conversely, the continuous nature of geometric problems entails that starting from geometric design necessities an analogue computing approach.
4) **Explainability**: Methodological learning of essential Mathematics & Computer Science is necessary for devising explainable systems and producing explainable insights.
5) **Modularity:** Whether in products or processes, modularity is a pre-requisite for reusability, reparability, and recyclability. Modularity of processes in particular is contingent upon shared terminologies, data-exchange standards, and representation protocols. Participation in open-source initiatives is necessary to ensure a critical mass required for adoption of modularization standards.
6) **Inclusivity:** Transparency and explication of processes, assumptions, and inputs are key to ensure that stakeholders can meaningfully participate in the act of decision-making rather than only choosing options (i.e. decision-taking).
7) **Community**: Long-term collaboration within open source communities and ecosystems is essential to empower innovative, unbiased, non-proprietary, and non-governmental bottom-up initiatives.
8) **Intelligibility**: Scientific standards in writing and presentation are fundamental for guaranteeing intelligibility. This primarily pertains to the use of adequate representation media for mathematical,

algorithmic, and spatial concepts, mechanisms, and cases. Furthermore, clear-cut problem-formulations are key to explaining computational approaches for solving problems.

TUlib: https://tulib.tudelft.nl/
TUlib: how to cite: https://tulib.tudelft.nl/writing-publishing/how-to-cite /
TUlib: APA Citation Examples: https://tulib.tudelft.nl/apa/
The Ten Commandments of Responsible Referencing

12. On Critical Thinking and the Scientific Method

We are living in an era characterized by mass propagation of fake news, political propaganda, commercial/promotional advertisements and viral political partisanships fuelled by mass clustering of communication channels. Being critical and analytical has always been important in science but in face of such an overwhelming amount of nonsense in the wild, it feels like it has never been so important! Logical Fallacies on the one hand and Academic Careerism on the other hand form the warps and wefts of the fabric of this dangerous landscape for early-stage researchers. The former is the distinctive mark of science from pseudo-science and the latter is the often neglected elephant in the room of academic integrity. A hallmark of shady attitudes in science is name-dropping and the use of 'ad-hominem' ("this is right because Mr. John Doe who is a big name says so"). The other highlight that must not go unnoticed is the famous academic pretensions, best understood in the light of sophism versus philosophy: i.e. the pursuit of science as a way of presenting wisdom and gaining wealth rather than a genuine interest in knowing or spread of knowledge. Salami publishing is the seal of such practices. All in all, one must be aware of such issues with many papers in the wild and find a mechanism to separate trivia from the genuine pieces of knowledge. Keeping a calculated distance from the so-called hot-topics and trends would also be an advisable safety mechanism to protect oneself from mumbo jumbo. The best separator of nonsense from sense appears to be the criteria of reproducibility and explainability. I have much more to preach about this topic and this is just an appetizer. Stay tuned for a longer sermon.

13. Bibliography

[1] S. Azadi and P. Nourian, "Collective Intelligence in Generative Design," *BouT RUMOER periodical for the Building Technologist, No. 76: Generative Design*, Delft, pp. 7–16, Apr-2021.
[2] H. A. Simon, *The Sciences of the Artificial Third edition*. MIT Press, 1996.
[3] P. Nourian, "Mathematical and Computational Modelling of Doubly Complex Systems," 2019. [Online]. Available: https://www.researchgate.net/publication/337922945_Mathematical_and_Computational_Modelling_of_Doubly_Complex_Systems_Theoretical_reflections_on_the_theory_and_the_practice_of_the_Scientific_Method_in_modelling_the_social_and_temporal_dynamics_of_soci.
[4] P. Nourian, "Generative Design Research Methods," Delft, 2020.
[5] P. Nourian, "Configraphics: Graph Theoretical Methods for Design and Analysis of Spatial Configurations," *Doi.Org*, vol. 6, no. 14. pp. 1–348, 2016.
[6] S. T. March and G. F. Smith, "Design and natural science research on information technology," *Decis. Support Syst.*, vol. 15, no. 4, pp. 251–266, 1995.
[7] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, 2007.
[8] R. Fenn, *Geometry*. Springer London, 2001.
[9] C. Tremblay, "Mathematics for game developers," 2004.
[10] M. Batty, *Essential Engineering Mathematics*. 2011.
[11] D. Cherney, T. Denton, R. Thomas, and A. Waldron, *Linear Algebra*. 2013.
[12] G. Farin and D. Hansford, *Practical Linear Algebra: A Geometry Toolbox*. Taylor & Francis Group, 2014.
[13] H. Schey, *H. M. Schey Div, Grad, Curl, and All That An Informal Text on Vector Calculus, 3ed 1997.pdf*. .

[14] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon mesh processing*. 2010.
[15] H. Edelsbrunner and J. Harer, *Computational Topology, an Introduction*. 2010.
[16] Y. Friedman, *Toward a scientific architecture*. MIT Press, 1980.
[17] L. March and P. Steadman, “The geometry of environment: an introduction to spatial organization in design,” 1974.
[18] E. Matthes, *Python Cash Course*. 2016.
[19] R. Issa, “Essential Mathematics for computational design.” Mc Neel, 2020.
[20] P. Nourian, “Rudiments of Linear Algebra & Computer Graphics,” in *Fundamentals of Spatial Computing & Generative Design*, Preprint., no. September 2019, Delft, 2019, pp. 1–18.
[21] P. Nourian, “Rudiments of Geometry and Topology for Computational Design,” in *Fundamentals of Spatial Computing & Generative Design*, Preprint., no. September, Delft, 2020.
[22] A. B. Downey, B. • Cambridge, • Farnham, • Köln, • Sebastopol, and • Tokyo, “Think Complexity.”
[23] A. Downey, “Think Python How to Think Like a Computer Scientist 2nd Edition, Version 2.2.23,” 2015.
[24] T. E. Oliphant, “Python for scientific computing,” *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 10–20, 2007.
[25] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, “The NumPy array: A structure for efficient numerical computation,” *Comput. Sci. Eng.*, vol. 13, no. 2, pp. 22–30, Mar. 2011.
[26] S. Azadi and P. Nourian, “topoGenesis: an open-source python library providing topological structures and functions for Generative Systems and Sciences,” 2020.
[27] I. Mavuru, “Traditional vs. Agile Software Development Methodologies.” 2018.
[28] M. Lou Maher and J. Poon, “Modeling Design Exploration as Co-Evolution,” *Comput. Civ. Infrastruct. Eng.*, vol. 11, pp. 195–209, 1996.
[29] K. Dorst and N. Cross, “Creativity in the design process: Co-evolution of problem-solution,” *Des. Stud.*, vol. 22, no. 5, pp. 425–437, 2001.
[30] P. Nourian, S. Rezvani, I. S. Sariyildiz, and F. D. van der Hoeven, “Spectral Modelling for Spatial Network Analysis,” *Proc. Symp. Simul. Archit. Urban Des. (simAUD 2016)*, 2016.

14. New References

Nourian, Pirouz, Shervin Azadi, and Robin Oval. “Generative Design in Architecture: From Mathematical Optimization to Grammatical Customization.” In *Computational Design and Digital Manufacturing*, edited by Panagiotis Kyratsis, Athanasios Manavis, and J. Paulo Davim, 1–43. Management and Industrial Engineering. Cham: Springer International Publishing, 2023. https://doi.org/10.1007/978-3-031-21167-6_1.
Nourian, Pirouz, Shervin Azadi, Roy Uijtendaal, and Nan Bai. “Augmented Computational Design: Methodical Application of Artificial Intelligence in Generative Design.” In *Artificial Intelligence in Performance-Driven Design: Theories, Methods, and Tools Towards Sustainability*, edited by Narjes Abbasabadi and Mehdi Ashayeri. Wiley, 2023. https://doi.org/10.48550/arXiv.2310.09243.
Nourian, Pirouz. “Augmented Computational Design.” May 25, 2023. https://doi.org/10.13140/RG.2.2.32447.89766.
Azadi, Shervin, Dena Kasraian, Pirouz Nourian, and P.J.V. Wesemael. “Augmented Urban Planning: A Framework for Strategic Urban Planning.” In *Proceedings of CUPUM 2023 - The 18th International Conference on Computational Urban Planning and Urban Management*, 2023. https://www.researchgate.net/publication/371865347_Augmented_Urban_Planning_A_Framework_for_Strategic_Urban_Planning.
Nourian, Pirouz. “Digital Design Games Reflections on Digital Design Games for Participatory Design.” June 13, 2023. https://doi.org/10.13140/RG.2.2.13382.65602.
Nourian, Pirouz, and Shervin Azadi. “Voxel Graph Operators: Topological Voxelization, Graph Generation, and Derivation of Discrete Differential Operators from Voxel Complexes.” arXiv, September 27, 2023. https://doi.org/10.48550/arXiv.2309.15472.

15. Practical Suggestions

DO NOT TRY TO WRITE YOUR THESIS IN ANY GRAPHICAL DESIGN SOFTWARE! Do not ever spend time on making fancy graphical slideshow presentations with animations and decorative layout features. Focus on the content and write it in the most hassle-free way without making a graphical design project out of it. Good pictures and a succinct text are much more important than the layout of the document. Preferably, use Python for programming. Use Draw.IO for drawing your diagrams, and make sure to start from the right menu. They have many menus for different kinds of diagrams. Use InkScape for drawing vector graphics. Using LaTeX (e.g. in an online editor such as Overleaf) is preferable to MS Word, especially if you will need to write a lot of maths. You may find these two LaTeX tools handy for typesetting Mathematical Formulae and Tables. For writing maths in MS word, consult this document by Prof. Hugues Hoppe. Never insert pictures, tables and alike for decorative purposes. When you insert a picture in your text, give it a proper image caption and refer to it in your text.

16. Disclaimer

The document does not explicitly consider the formalities of the graduation project, but wherever the procedural matters are mentored, this document does not mean to override them but to add explanations. This document is only made for informative purposes and thus it is not meant to replace any existing regulation concerning graduation projects. Please note that I am not supposed to know all these details and that I am giving you these tips cannot be used against me because there are inconsistencies of any kind whatsoever. This document is only made for informative purposes. Mentions of commercial products are impartial, didactic, and completely unsponsored by commercial parties.

17. On choosing your mentors

I would suggest to choose your mentors with eyes wide open. Your graduation project is the most serious project you are going to do in your whole curricular life. It is a long period of time that you can spend wisely on developing skills that you will need in 'your' future career. Not everything that looks cool today will be useful tomorrow. The relevance of your topic must be such that instead of relying on some niche academic interest, it will lead to learning some useful methodological and technical skills, while being a motivating/generally useful subject of course. The best cure or preventive measure for headaches to occur later at your desk in a company is a solid methodological background. Your chosen mentor must be able to help you in developing such a background, in your chosen area of work. Curiosity is not the same as knowledge. I may be interested in and curious about biology but I know close to nothing about biology; and I cannot learn it from LinkedIn and Twitter; and so I would only fool myself mentoring a project about say 'biological optimization' (a fictitious subject BTW). Read this article about the illusion of knowledge.
In the context of Architecture and Built Environment, Design is as important (or even more important) as Research in graduation projects. Depending on the focus of your project you are advised to consider having an experienced design mentor as well as a research mentor and take the design aspect of your project at least equally seriously as the research part. In the real world, many problems are hard to frame and formulate and that is exactly where you will benefit from a mentor with a design-oriented background rooted in practice.
Check the graduate work of your prospective mentor or their professional design portfolio. PhD theses are public by law. Regardless of formalities and social niceties, a design portfolio or a PhD thesis is the ultimate business-card of an academic. In particular, if you cannot find a PhD thesis of someone with a Dr (doctorate) title, you have every right to ask them to provide it to you. There are academic disciplines that cannot be practiced without going through some seriously structured graduate education. Mathematics and Computer Science are two good examples of those. If you want to choose a topic on these subjects, then your mentor must have had a formal training or equivalent hard-core experience with programming. In particular, it may be possible to practice computational design without some structured knowledge of mathematics, but it is certainly impossible to innovate in computational design without some advanced knowledge of mathematics and computer science. There are no shortcuts in life.
Please bear in mind that your second mentor has much less hours to commit to your supervision (24 hours for mentors from our faculty and 0 hours for mentors from other faculties). This means that a mentor from another faculty should really be interested personally in your topic to help you. I request you to make it as easy as possible for your second mentor to help you by being well organized, prepared, and focused for Q&A sessions.

18. On asking questions

According to the norms of TU Delft the faculty of architecture, As a first mentor, I have exactly a virtual budget of 44 hours to spend in total on your graduation project[8]. If I spend more time, which is often the case, the faculty regards it as 'my problem'. Thus, I cannot possibly promise to help you with your technical questions, unless they are either on meta levels or very detailed and clear-cut

[8] Based on the same rules, your second mentor has a budget of 24 hours in total to spend on supervision, attending your presentations, reading your reports, etcetera! Of these amounts, you must subtract the times spent on attending 4-5 presentations and all the trivial super-interesting things we need to do to make a time-plan without doodle and slack, just because we love bureaucracy, the equivalent of 8 hours.

questions in Python or C#. When you reach to such detailed questions, you can post on the online forums mentioned below and tag me with your question, making sure that I will see it.
You can send me your GH questions; but please make sure that your model is structured properly:

- You have cleaned your file streamlining input/output wires, sockets, and groups: inputs and outputs of all groups, regardless if they are shared with others, are separately grouped at the left and right sides of each group. In case an input is an output of another group, it must be copied for both. The same applies to outputs that are inputs of other processes.
- You do not use 3rd party plugins (Kangaroo is now part of GH so that is not an issue)
- You have evidently done your best searching the related professional for an answer already (e.g. SYNTACTIC group, GH forum, Mc Neel's Q&A, stack-overflow).
- You do not have a philosophical question like 'how do I use computation in architecture?' but a specific methodological question.
- You have made sure that you have abstracted your question and detached it completely from trivial and/or project specific details. For instance, you do not post a question online saying "graduating at TU Delft on generative design, I do not know what to do to make a beautiful project out of graph theory using a graph-traversal algorithm in python" Instead, you search for the phrase "graph-traversal, how does Dijkstra algorithm work? + Python example".

19. Selected previous theses:

For a complete and up-to-date overview of the completed graduation projects and suggested topics for future research visit this page on the wall of our lab: https://genesis-lab.dev/graduation/

	Degree	Name	Topic	Sponsor/ Partner
2023	MSc Student Geoinformation Sciences @ University of Twente	Iván Cárdenas	From Trash To Digital Treasure	Municipality of Pretoria, South Africa
2023	MSc Arch.	Ruben de Leeuw	Housing the Homo Ludens	Architect at Inbo
2023	MSc Build. Tech. & MSc Structural Engineering	Kaan Akbaba	Topology Optimization of Masonry Structures	
2023	PhD Design Informatics	Cemre Çubukçuoğlu	HOPCA: Hospital Layout Design Optimization using Computational Architecture	Industrial Engineering
2023	MSc B Build. Tech.	Tim Schumann	Encoding Building Products	Product Design
2023	MSc Arch. Engineering	Jacek Baczkowski	Play Well Housing	Architectural Engineering
2022	MSc Build. Tech.	Roy Uijtendaal	Bayesian Network Metamodeling NTA 8800 Energy Performance Design Space Exploration	Nieman, Building Physics, Temporary Embargo
2022	MSc Build. Tech.	Qinglu Chen	A Topology Optimization Process for Discrete Modular Design based on Discrete Element Modelling for generating reconfigurable funicular structures	Structural Design & Mechanics, Temporary Embargo
2022	MSc Build. Tech.	Baolian Liu	Topological Stereotomic Design of System of Interlocking Stackable Modular Blocks for Constructing Multi-Storey Masonry Buildings	Structural Design & Mechanics, Temporary Embargo
2022	MSc Build. Tech.	Solkyu Park	Houscaper: Developing a tool for easy access of architectural polygonization using the Boolean Marching Cubes Algorithm	Structural Design & Mechanics
2022	MSc Architecture	Leticija Petrova	Reconfigurable architecture for compression-only structures	Architectural Engineering
2022	MSc Architecture	Anna Kaletkina	ShellTerra	Architectural Engineering

2022	MSc Build. Tech.	Vasilka Espinosa	A Participatory Design Game for Social Housing Configuration in the Context of Manaus, in Brazil	Heritage & Values
2022	MSc Build. Tech.	Max Ketelaar	Generating building envelopes using multi-objective optimization techniques	Building Physics
2021	MSc Build. Tech.	Anastasia Florou	Generative Solar-Climatic Configuration	Building Physics
2021	MSc Build. Tech.	Aditya Pravin Soman	Generative Architectural Configuration	Architectural Engineering
2021	MSc Build. Tech.	Selina Bitting	Block Parti: generating dry-fit, stackable blocks for constructing vaulted ceilings	BRG
2021	MSc Build. Tech.	Bezawit Bekele	Participatory Design Game for Urban Slum Upgrading	Global Housing Group
2021	MSc SET	Karthick Subramanian	Opto-Geometric Modelling of Complex Urban Landscapes	PVMD
2020	MSc Arch.	Jakub Wycoski	Customized Collective Housing	Explore Lab
2020	MSc Build. Tech.	Rick van Dijk	Topology optimization as architectural form finding	3ME, Structural Optimization and Mechanics
2020	MSc Build. Tech.	David den Ouden	A Facility Layout Optimization Model for Using the Gradient Descent Approach	Dapp, TPM, Energy & Industry
2020	MSc Build. Tech.	Lincheng Jiang	A Computational Method for Generating Floor Plans for Nursing Homes	Building Physics
2020	MSc Build. Tech.	Tolga Ozdemir	A computational toolkit for early-stage cost assessment and optimisation of BIPV façades	Architectural Engineering
2020	MSc Build. Tech.	Konstantina Chouliara	Cochleas: a methodology to convert an existing layout into a residential one	Building Physics
2019	MSc Build. Tech.	Thomas Van Loon	Subdivision & Approximation of free-form steel structures	White Lioness, Structural Design and Mechanics
2019	MSc Build. Tech.	Okan Turkcan	Ventilation Design & Optimization	Building Physics
2019	MSc Build. Tech.	Ivan Avdic	Bio-Inspired Topological Design	3ME, Structural Optimization and Mechanics
2019	MSc Build. Tech.	Idil Gumruk	Bio-Inspired Topological Shell Design	Structural Design and Mechanics
2019	MSc Geomatics	Ioanna Tsakalakidou	Pedestrian Mobility Modelling	CiTG, Transport Planning
2018	MSc Geomatics	Kotryna Valeckaite	Design of a Spatial Decision Support System using an Agent-Based-Model	ARUP, TPM, Game Lab
2018	MSc Build. Tech.	Michael Cobb	Investigating Principal Stress Lines: Optimization of Grid-shell Structures	Structural Design and Mechanics
2018	MSc Architecture	Yannick Macken	Automated Plan Layout	Explore Lab
2017	MSc Geomatics	Fanny Bot	A graph-matching approach to indoor localization	SWECO, GIS Technology
2017	MSc Geomatics	Rafael Sulzer	Spectral Machine Learning for Shape Recognition	ARUP, Computer Vision Lab
2015	MSc Geomatics	Rusne Sileryte	Spatial Network Analysis for Recreational Use Assessment	VR Lab

20. About the author

Pirouz Nourian is an assistant professor of Digital Twinning at the University of Twente and scientific director of the laboratory of Generative Systems and Sciences (Genesis Lab) with expertise and interest in generative design, spatial computing, and planning support systems, specifically by means of mathematical modelling, geometrical, topological, or graph-theoretical computing (also known as computational geometry, computational topology, and computational graph-theory). He has developed several computational tool-suites, most well-known among which are Space Syntax and Cheetah for Grasshopper 3D. Pirouz teaches mathematics and programming (Algorithms, Data Models, C#, Python) within MSc Architecture, MSc Building Technology, and MSc Geomatics programs. He has a PhD in Design Informatics, an MSc in Architecture, and a BSc in Electrical Engineering with a major in Control Systems Engineering. He is an architect with three years professional experience, a carpenter, and a research software engineer.

21. Colophon

Published on FigShare.
Corresponding Author: Dr. Pirouz Nourian
Cite as:
Nourian, Pirouz, How to write a thesis? A Generative Design Graduate Studio Guidebook, 2019, Delft University of Technology, Faculty of Architecture and Built Environment, Department of Architectural Engineering and Technology. DOI: 10.6084/m9.figshare.11987328
Suggestions for improvement are always welcome (p.nourian [[at]] utwente.nl).