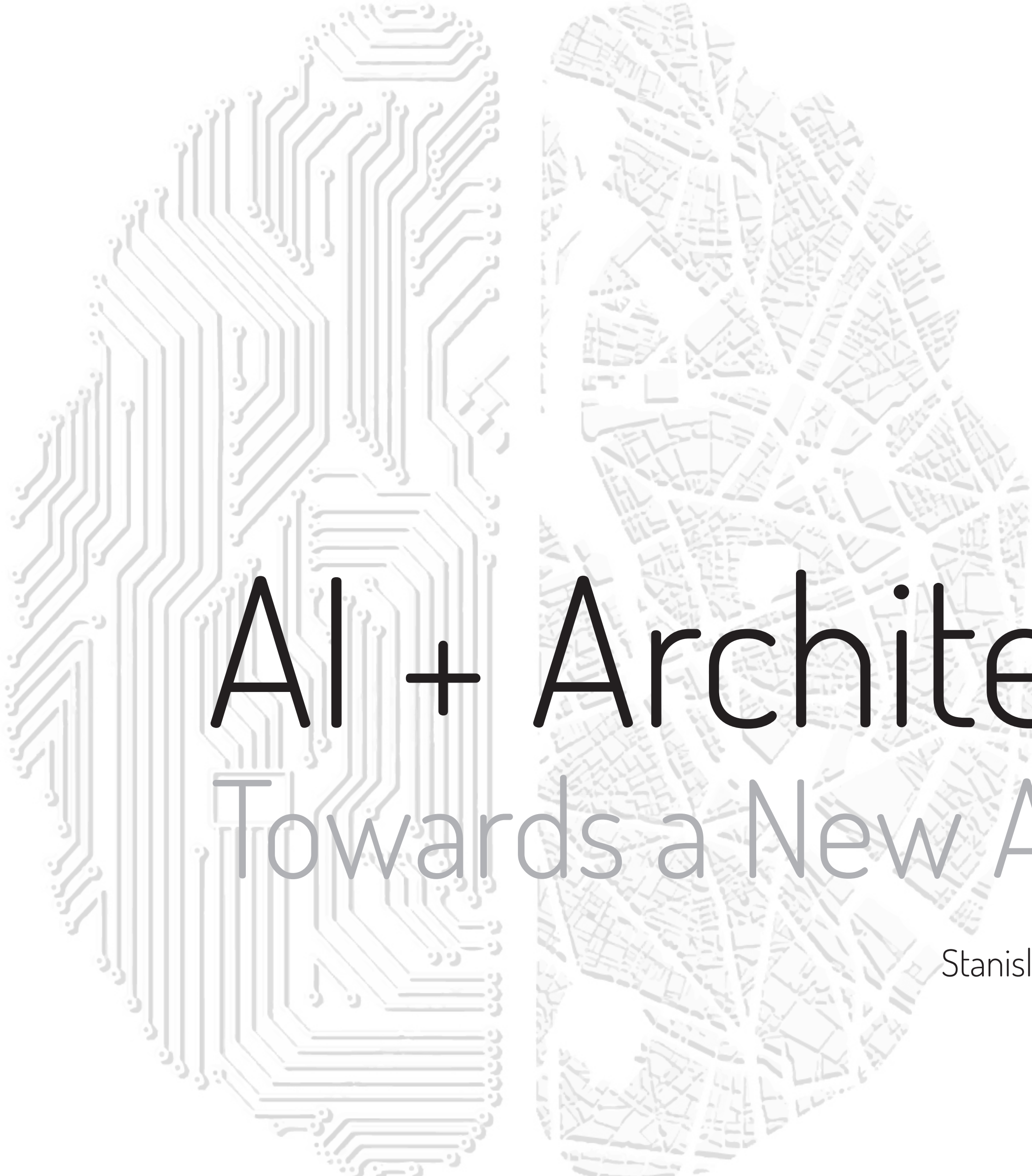
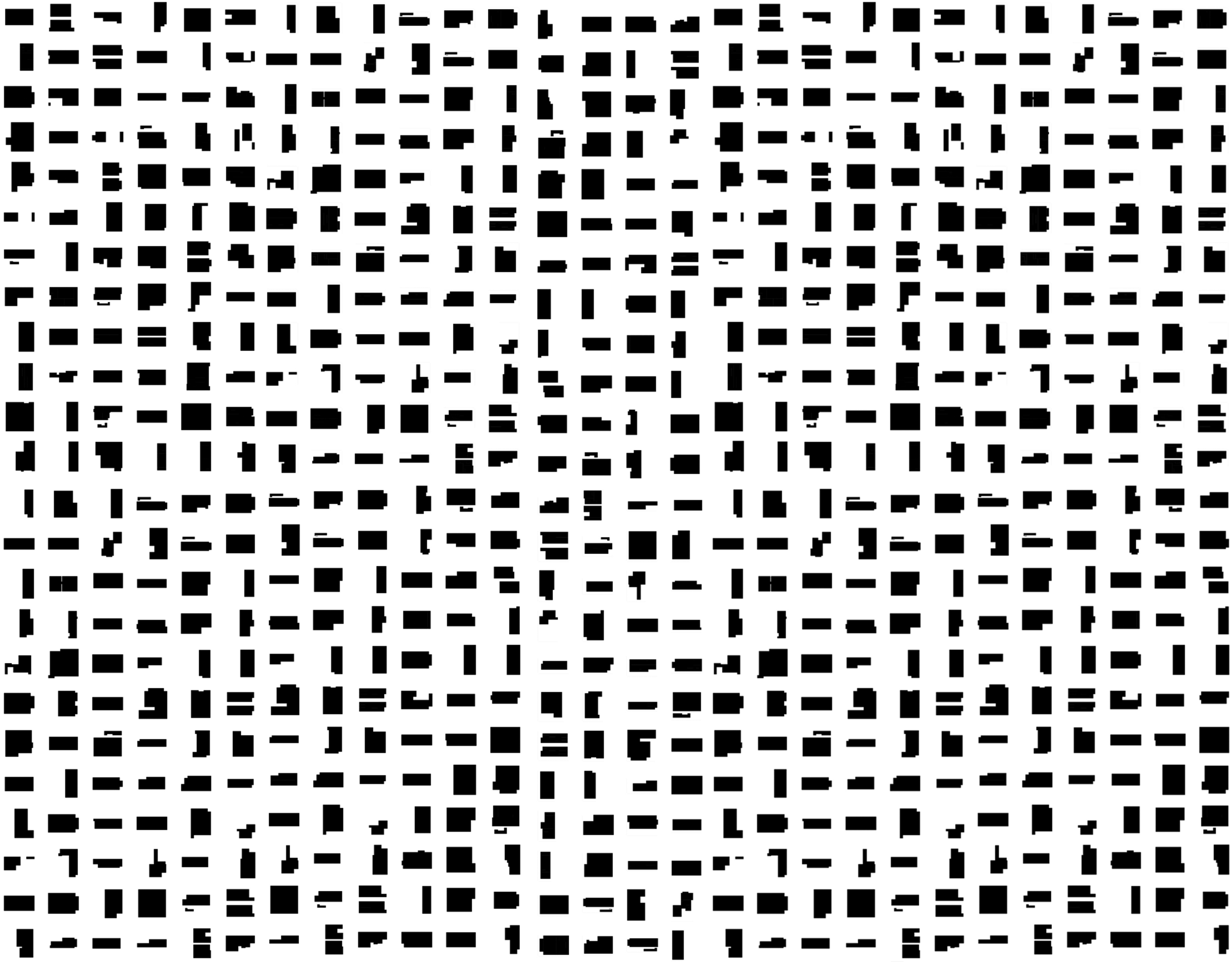


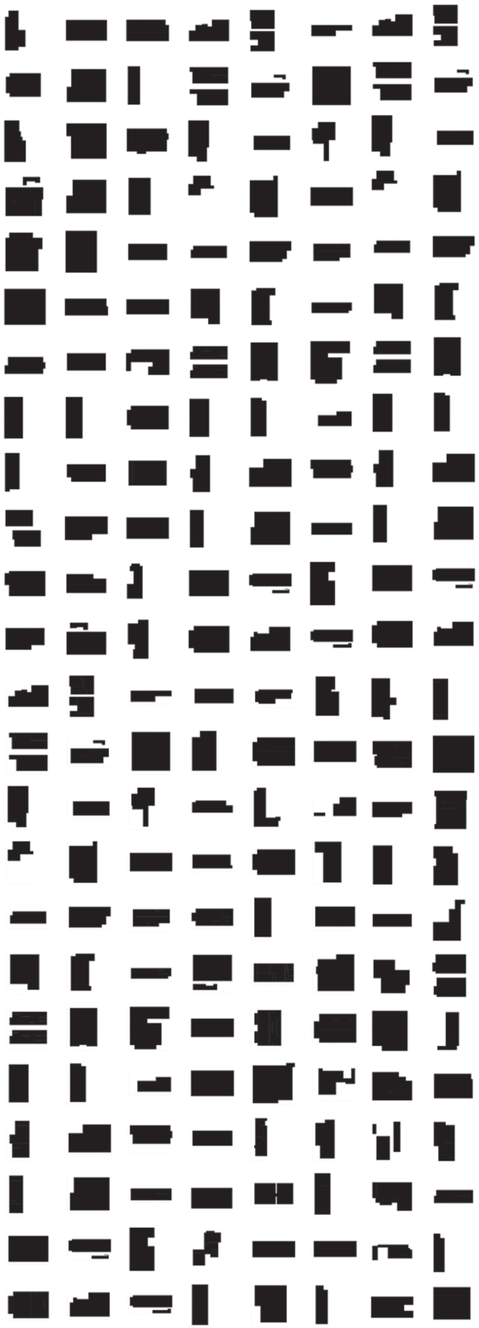
AI + Architecture

Towards a New Approach

Stanislas Chaillou | Harvard GSD | 2019

Harvard University
Graduate School of Design





Expliquer.

"Explain."

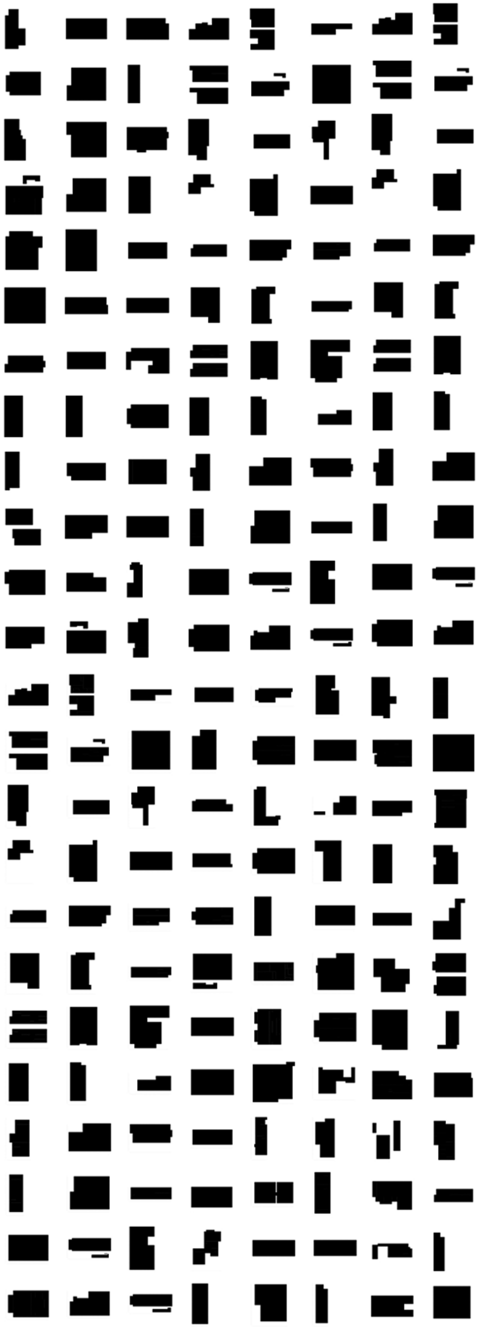
In 2014, the Ecole Normale Supérieur, one of the most competitive university in France, gave to its candidates for the entrance exam the following subject: "Explain". No subtitle, no annex. An 8-hour exam sat by more than 15.000 candidates, came down to this single word.

Far from the public outcry that immediately followed this event, we see this injunction as the single most valuable responsibility of our time: explaining. If one thing is to have convictions, another one is to find the right words to teach others.

The age of machines will not abolish this truth. To the contrary, creating the right taxonomy will more than ever be a crucial skill to configure the new forms of intelligence expected to leverage Architects' capabilities. But it is not a top-down exercise anymore: the machine would not just meekly execute the instruction transmitted by the "right word", he may in its turn suggest its own "optional words". In fact, the advent of Artificial Intelligence, or AI, is creating the conditions of a reflexive empowerment: the machine could become a trustworthy "assistant" provided professionals educate it, or properly "explain" the job.

Stanislas Chaillou, Harvard GSD, May 2019
© 2019 Stanislas Chaillou, All Rights Reserved.

Harvard University
Graduate School of Design



Acknowledgments

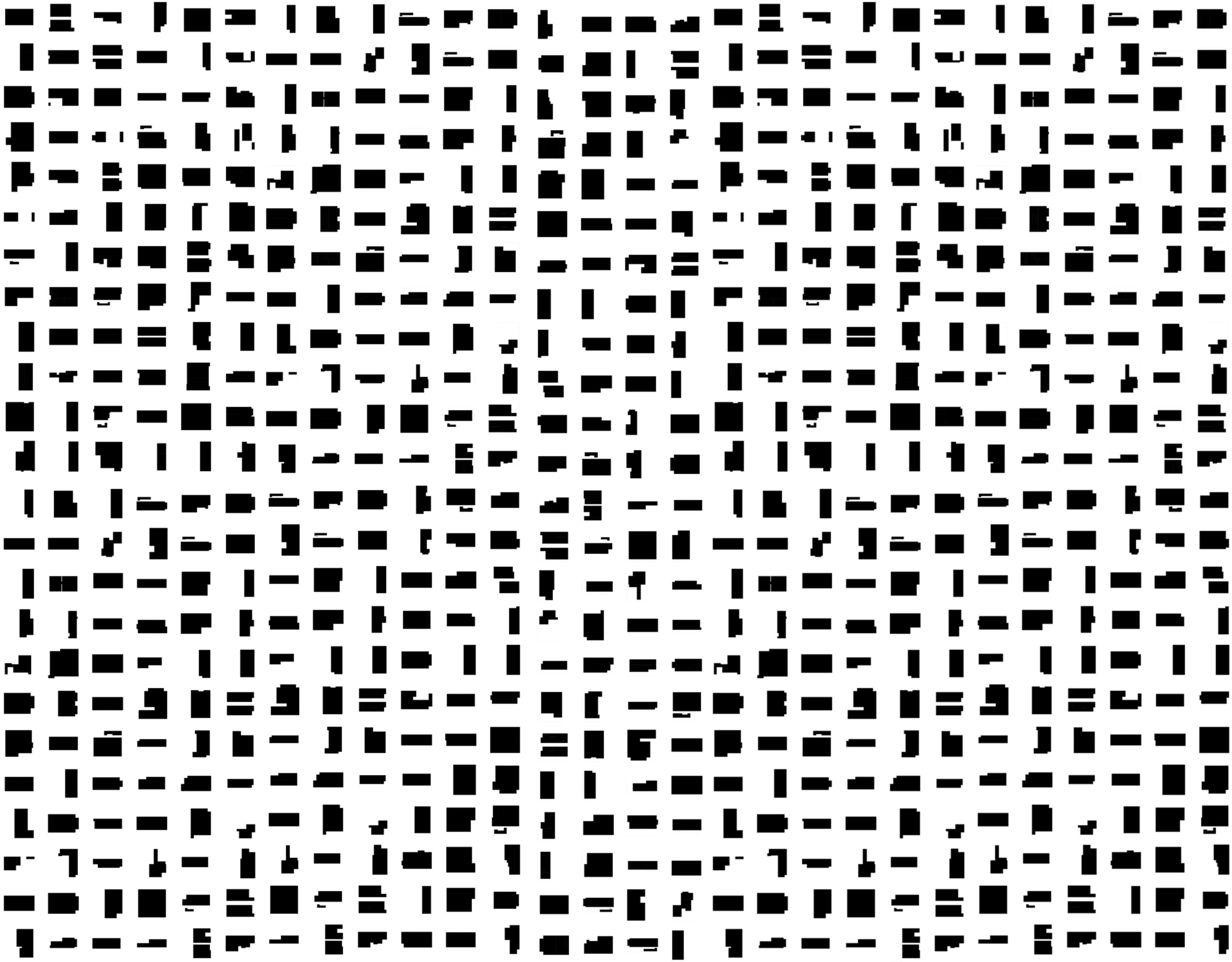
This thesis is the result of an intense academic journey, halfway between Switzerland and the US. It is certainly the result of ideas and influences found in each system; a blend between Swiss rigor and American optimism, with a dash of French Cartesian thinking. Along the way, it is thanks to inspiring and open-minded people that I have found the resources to carry on.

I am thankful to Andrew Witt, who advised me all along this thesis; his positivity and intellectual curiosity have been the ideal springboard. Through his work, he has inspired at the GSD an entire generation of new practionners, I am just one of the many.

None of this would have ever happened without the unfailing support of my Mum, Dad and sisters: *leur apport a été constant et parfait, tant sur un plan émotionnel qu'intellectuel. Il aurait été impensable d'avoir manqué à mes études, tant leur présence a rendu ces sept dernières années la source d'un épanouissement profond.*

To the Flux.io Team that catalyzed my intuitions, and trusted me early on, I am also grateful: Thomas, Anthony, Nicolas. Their advices have shaped the orientation I chose at Harvard, and this book would not be here today without them.

Finally, I would like to acknowledge a broader idea, my home country: France. If I was not fortunate enough to study back home, I am aware of the intellectual structure, and cultural heritage that this country has given me. From the singularity of its history, to the breadth of its culture, it is an endless well of knowledge. Too often taken for granted, and rarely acknowledged for its richness, I want to end this page by reaffirming how France has offered me the intellectual anchor & roots that have shaped my academic journey.



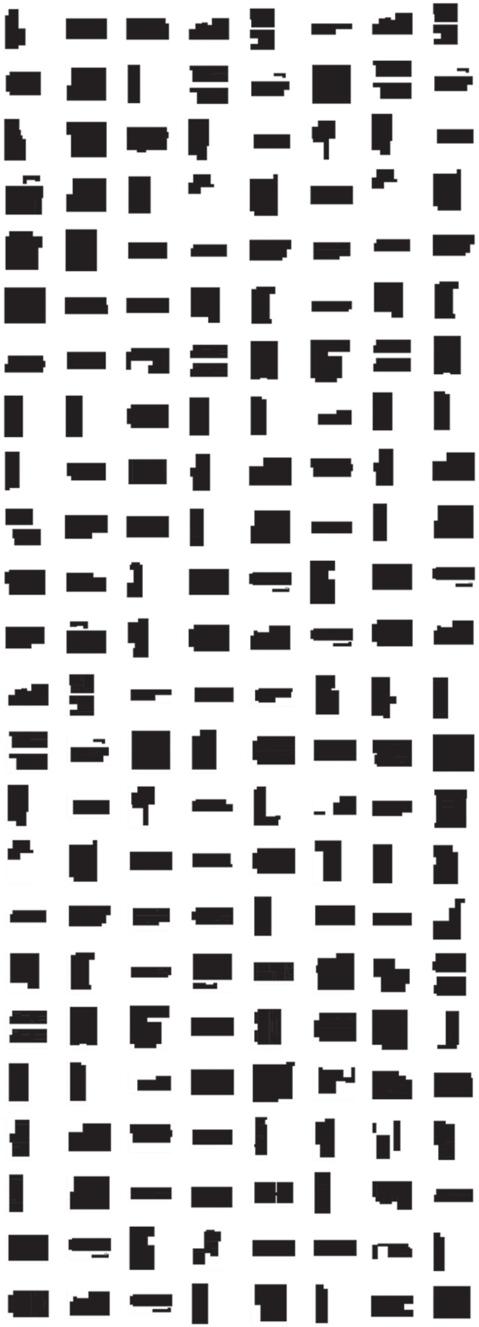
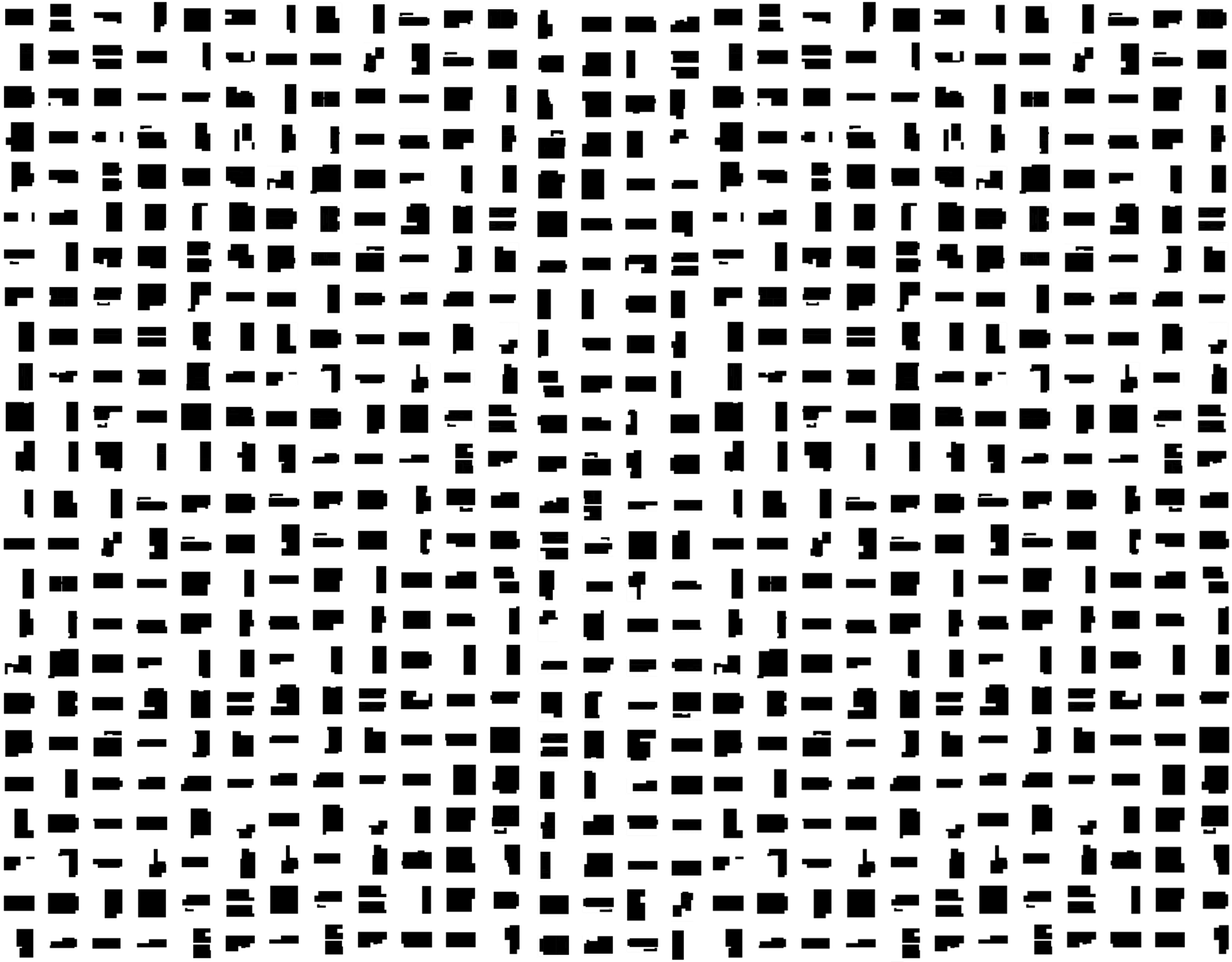


Table of Content



Introduction

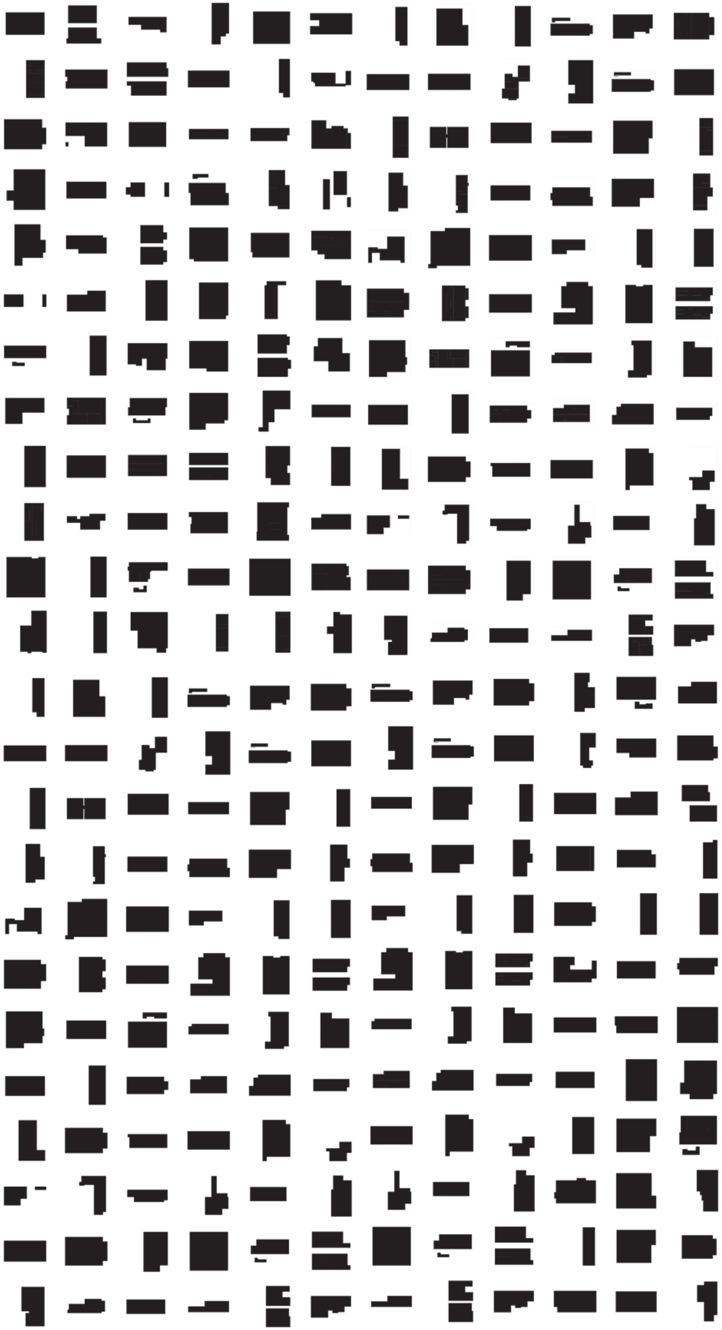
Artificial Intelligence, as a discipline, has already been permeating countless fields, bringing means and methods to previously unresolved challenges, across industries. **The advent of AI in Architecture is still in its early days but offers promising results.** More than a mere opportunity, such potential represents for us a major step ahead, about to reshape the architectural discipline.

Our work proposes to evidence this promise when applied to the built environment. Specifically, we offer to apply AI to floor plans analysis and generation. Our ultimate goal is three-fold: **(1) to generate floor plans** i.e. optimize the generation of a large and highly diverse quantity of floor plan designs, **(2) to qualify floor plans** i.e. offer a proper classification methodology **(3) to allow users to "browse"** through generated design options.

Our methodology follows two main intuitions (1) the creation of building plans is a non-trivial technical challenge, although encompassing standard optimization technics, and (2) the design of space is a sequential process, requiring successive design steps across different scales (urban scale, building scale, unit scale). Then, in order to harness these two realities, we have chosen nested Generative Adversarial Neural Networks or GANs. Such models enable us to capture more complexity across encountered floor plans and to break down the complexity by tackling problems through successive steps. Each step corresponding to a given model, specifically trained for this particular task, the process can eventually evidence the possible back and forth between humans and machines.

Plans are indeed a high-dimensional problem, at the crossroad of quantifiable technics, and more qualitative properties. The study of architectural precedent remains too often a hazardous process, that negates the richness of the number of existing resources while lacking in analytical rigor. Our methodology, inspired by current Data Science methodologies, aims at qualifying floor plans. Through the creation of 6 metrics, we propose a framework that captures architecturally relevant parameters of floor plans. On one hand, **Footprint Shape, Orientation, Thickness & Texture** are three metrics capturing the essence of a given floor plan's style. On the other hand, **Program, Connectivity, and Circulation** are meant to depict the essence of any floor plan organization.

In a nutshell, the machine, once the extension of our pencil, can today be leveraged to map architectural knowledge, and trained to assist us in creating viable design options.



I
The Advent of Architectural AI

The practice of Architecture, its methods, traditions, and know-how are today at the center of passionate debates. Challenged by outsiders, arriving with new practices, and questioned from within, as practitioners doubt of its current state, Architecture is undergoing a truly profound (r)evolution.

Among the factors that will leave a lasting impact on our discipline, technology certainly is one of the main vectors at play. The inception of technological solutions at every step of the value chain has already significantly transformed Architecture. The conception of buildings has in fact already started a slow transformation: first by leveraging new construction technics, then by developing adequate software, and eventually today by introducing statistical computing capabilities (including Data Science & AI). Rather than a disruption, we want to see here a continuity that led Architecture through successive evolutions until today. Modularity, Computational Design, Parametricism and finally Artificial Intelligence are to us the four intricated steps of a slow-paced transition. Beyond the historical background, we posit that this evolution is the wireframe of a radical improvement in architectural conception.

A. A Four Period Sequence

Modularity, Computational Design, Parametricism and finally Artificial Intelligence are not air-tight steps, independent of one another: each period interpenetrates and borrows from the precedents. It is why, when looking backward at history it is critical to distinguish two levels of creation: inventions & innovations. Inventions stem from academic research, while innovations are induced by inventions. In architecture, Innovations actually shape a continuously moving practice. A practice which has been playing on the back and forth between periods, inventions & innovations. From there, our chronology aims at demonstrating the deeply interwoven evolutions of the computational and the architectural fields before introducing the age of architectural-AI, as a culminating point. It is why rebuilding the context and the highlights of the recent history of our discipline is a prerequisite to our work.

Modular Systems

Modularity could be set as the starting point of systematic architectural design. Initiated in the early 30's, the advent of modular construction brought to the conception phase both a language and an architectural grammar, contributing to simplify and rationalize building design.

Theorized for the Bauhaus by Walter Gropius, as early as 1920, the modular grid was carrying the hope of technical simplicity and affordability. Coming from different directions, modularity arose at first as a topic of investigation for academics and practitioners. Gropius initially introduced the idea of "Baukasten", a typical module to be then aggregated through strict assembly rules. This systematicity will be echoed one year later with Le Corbusier's "Modulor". By applying the modular rigor down to the human scale, Le Corbusier, as of 1946, offered a holistic implementation of the modular principles. The built environment dimensions would be aligned on key metrics and ratios derived from the human body. And indeed, from "La Tourette" to the "Unité d'habitation" in Marseille, Le Corbusier

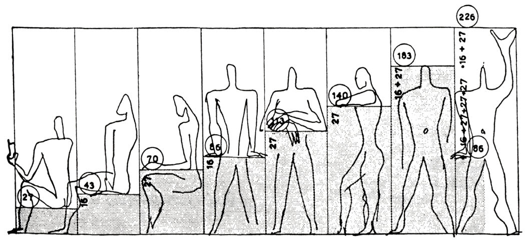


Figure 1: Corbusier's Modulor

systematized the dimensions and spans to match the prescription of the "Modulor". With Buckminster Fueller, however, Modularity rapidly evolved towards a more integrated vision embedding building systems within the module as exemplified by the Dymaxion House. This attempt pushed to its extreme the possibility of modular housing, setting a vibrant precedent, and proof-of-concept for the industry.

Thereafter, following these early theorists, architects were invited to bend their design ethos to the imperative of the matrix, and by the same token, to transfer part of the technicality of building design to the logic of the module. Less hassle, less costs, more predictability. Modularity would then swiftly extend to the industry as a whole: the Winslow Ames House, built by Professor Robert W. McLaughlin in 1933, and first large-scale modular project in the world, was perceived as a major breakthrough, as much as the very

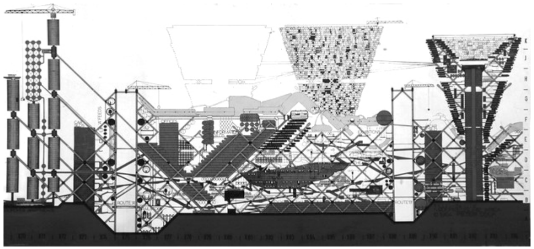


Figure 2: Plugin City by Archigram

expressive Habitat 67 from Moshe Safdie. City planning even got influenced at the turn of the 60's, when projects like the "Plugin City" of Archigram developed the possibility of modular cities. Through the constant assemblage and dismantlement of modules, fitted on a three-dimensional structural matrix, cities could find a renewed logic, addressing both the possibility of growth and -as always- the imperative of feasibility.

However, connecting the grid, the modules, and the assembly systems through mechanistic rules eventually led to a quasi-gamification of a LEGO-like conception of Architecture. But practice cannot be just a "put together" board game aggregating a set of basic assembly rules and processes. The monotony of the resulting designs rapidly trivialized the theory, and the constructive weakness of its assembly systems finally discouraged architects. Nevertheless, through its system of rules, Modularity is remaining today an underlying constructive principle still vivid throughout the practice.

Computational design

At the turn the 80's, as the complexity of modular systems was soaring, the advent of computation brought back feasibility and scalability to modular design. Beyond the resurrection of the module, the systematicity of rule-based design was somehow rehabilitated.

Coming from different directions, a high-level reflection about the potential of computational design started as early as the mid-50's, within an adjacent discipline: Engineering. In 1959, Professor Patrick Hanratty released PRONTO, the first prototype of CAD (Computer Assisted Drawing) software, geared towards engineered parts design. The possibility offered by such software, coupled to the potential computational power fast-paced evolution, jumped start a discussion within the architectural field. Soon after, Christopher Alexander, architect and then professor at U.C. Berkeley started the discussion by laying down the key principles for Computational Design. In his "Notes on the Synthesis of Form" (1964) and later, in "a Pattern Language" (1968), Alexander theorized why and how computers should be used to address the question of shape design. His early understanding of software potential for design was deeply contrasting with the hardware-centric focus at the time. The founding principles he defined in his book are still today the bedrock of software programming: concepts like recursions, object-oriented programming as well as their application to design have represented a radical move forward. Following this momentum, an entire generation of computer scientists and architects will create a new field of research: Computational Design. The Architecture Machine Group (AMG) at MIT, led by Professor Nicholas Negroponte is probably its most

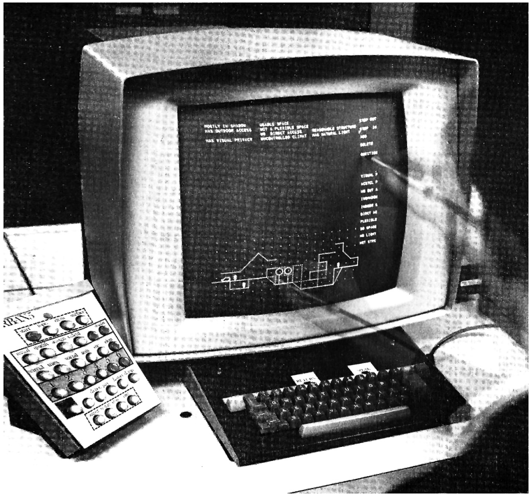


Figure 3: URBAN 5, AMG MIT

exemplary embodiment. Negroponte's book "The Architecture Machine" (1970) encapsulates the essence of the AMG's mission: investigating how machines can enhance the creative process, and more specifically, the architectural production as a whole. Culminating with the release of projects URBAN II and later URBAN V, this group will then demonstrate, even before industry would engage in any effort, the potential of CAD applied to space design.

Following such conclusive research, architects and the industry at large actively pushed these inventions to the state of innovations. Frank Gehry certainly was the most vibrant advocate of the cause. For him, the application of computation could drastically relax the boundary of assembly systems and allow for new shapes & building geometries. Gehry Technologies, founded by Gehry and Jim Glymph in the 80's typically used early CAD-CAM software—such as CATIA—from Dassault Systems to tackle complex geometric problems. Setting here the precedent for 30 years of Computational Design, Gehry Technology would demonstrate the value of computation to architects, provoking a landslide in the profession. Over the next 15 years, the irresistible growth of computational power & data storage capacities, combined with increasingly affordable and more user-friendly machines, massively facilitated the adoption of 3D-design software. Architects rapidly endorsed the new system on the base of a clear rationale: Computational Design (1) allows a rigorous control of geometry, boosting design's reliability, feasibility and cost, (2) facilitates and eases collaboration among designers, (3) and finally enables more design iterations than traditional hand-sketching could afford. More tests & more options for better resulting designs.

However, along the way, as designers were engaging with Computational Design, a couple of shortcomings eventually arose. In particular, the repetitiveness of certain tasks, and the lack of control over complex geometric shapes became serious impediments. Those paved the way to a brand-new movement which was emerging within Computation Design: Parametricism.

Parametricism

In the world of parameters, both repetitive tasks and complex shapes could possibly be tackled, when rationalizable to simple sets of rules. The rules could be encoded in the program, to automate the time-consuming process of manually implementing them. This paradigm drove the advent of Parametricism. In few words, if a task can be explained as a set of commands given to the computer, then the designer's task would be to communicate them to the software while isolating the key parameters impacting the result. Once encoded, the architect would be able to vary the parameters and generate different possible scenarios: different potential shapes, yielding multiple design outputs at once.

In the early 1960s, the advent of parametrized architecture was announced by Professor Luigi Moretti. His project "Stadium N", although theoretical initially, is the first clear expression of Parametricism. By defining 19 driving parameters—among which the spectators' field of view and sun exposure of the tribunes -, Moretti derived the shape of the stadium directly from the variation of these parameters. The resulting shape, although surprising and quite organic, offers the first example of this new parametric aesthetic: organic in aspect, while strictly rational as a conception process. Bringing such principle to the world of computation will be the contribution of Ivan Sutherland, three years later. Sutherland is the creator of SketchPad, one of the first truly user-friendly CAD software. Embedded at the heart of the software, the notion of "Atomic Constraint" is Sutherland's translation of Moretti's idea of parameter. In a typical SketchPad drawing, each geometry was in fact translated on the machine side into a set of atomic constraints (parameters). This very notion is the first formulation of parametric design in computer's terms. Samuel Geisberg, founder of the Parametric Technology Corporation (PTC), would later, in 1988, roll out Pro/ENGINEER, first software giving full access to geometric parameters to its users. As the software is released, Geisberg perfectly summed up perfectly the parametric ideal:

"The goal is to create a system that would be flexible enough to encourage the engineer to easily consider a variety of designs. And the cost of making design changes ought to be as close to zero as possible. "

Now that the bridge between design and computation was built thanks to Sutherland and Geisberg, a new generation of "parameter-conscious" architects could thrive. As architects were becoming more and more capable of manipulating their design using the proxy of parameters, the discipline "slowly converged" to Parametricism, as explained by P. Schumacher. In his book, "Parametricism, a New Global Style for Architecture & Urban Design" Schumacher explicitly demonstrated how Parametricism was the result of a growing awareness of the notion of parameters within the architectural discipline.

From the invention of parameters to their translation into innovations throughout the industry, we see a handful of key individuals, who have shaped the advent of Parametricism. This parametrization of architecture is best exemplified at first by Zaha Hadid Architects' work. Mrs. Hadid, an Iraqi architect trained in the UK, with a math background would found her practice, with the intent to marry math and architecture through the medium of parametric design. Her designs would typically be the result of rules, encoded in the program, allowing for unprecedented levels of control over the buildings' geometry. Each architectural move would be translated into a given tuning of parameters, resulting in a specific building shape. Hadid's designs are the perfect examples to this day of the possible quantification of architectural design, into arrays of parameters. Her work, however, would have not been possible without Grasshopper, software developed by David Rutten in the year 2000's. Designed as a visual programming interface, Grasshopper allows architects to easily isolate the driving parameters of their design while allowing them to tune them iteratively. The

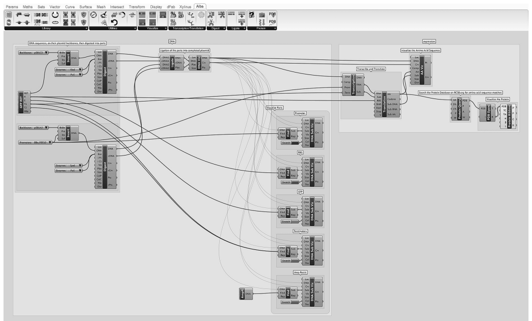


Figure 4: Grasshopper by David Rutten

simplicity of its interface coupled with the intelligence of the built-in features continues today to power most buildings' design across the world and has inspired an entire generation of "parametric" designers. Finally, beyond the short-term benefits of Grasshopper for building design, a more profound revolution, driven by parametrization and started in the early 2000s, is still underway today: BIM (Building Information Modeling). Spearheaded by Philip Bernstein, then Vice President of Autodesk, the birth and refinement of BIM has brought rationality and feasibility to a brand-new level within the construction industry. The underlying idea of the BIM is that every element in a building 3D model is a function of parameters ("properties") that drives each object's shape and document them. From Autodesk Revit -the major BIM software today- to Sutherland's SketchPad, we see a single common thread: the explicit utilization of parameters as the driving force of design.

However, the parametrization of design has proven over the past 10 years to have reached a plateau, both technically and conceptually. Parametric modeling failed to account for (1) the compounded effect of multiple variables at once, (2) the imperative of space organization and style over strict efficiency, (3) the variability of scenarios, and finally (4) the computational cost of simulations. Independently from its technical shortcomings, parametric design is flawed by its theoretical premise: Architecture could be the result of a fixed number of parameters, that the architect could simply encode, as an abstraction, away from its context, its environment, and its history. In fact, Parametricism, when applied 'by the book', proved to neglect the immense complexity of space planning: countless parameters and profound cultural & societal factors actually participate in the urban equilibrium. This deep reality, combining adjacent disciplines in a systemic way, can today finally be addressed, as our profession encounters Artificial Intelligence.

Artificial Intelligence: a Statisitical Approach to Architecure

Artificial Intelligence is fundamentally a statistical approach to architecture. The premise of AI, that blends statistical principles with computation is a new approach that can improve over the drawbacks of parametric architecture.

"Learning", as understood by machines, corresponds to the ability of a computer, when faced with a complicated issue, first to grasp the complexity of the options shown to him and second to build an "intuition" to solve the problem at stake. In fact, when coining down the concept of AI, John McCarthy, back in 1956, has defined it as "using the human brain as a model for machine logic". Instead of designing a deterministic model, built for a set number of variables and rules, AI lets the computer create intermediary parameters, from information either collected from the data or transmitted by the user. Once the "learning phase" achieved, the machine can generate solutions, not simply answering a set of predefined parameters, but creating results emulating the statistical distribution of the information shown to him during the learning phase. This concept is at the core of the paradigm shift brought by AI. The partial independence of the machine to build its own understanding of the problem, coupled with its ability to digest the complexity of a set of examples, turns upside down the premise of Parametricism. Since not all rules & parameters are declared upfront explicitly by the user, the machine can unexpectedly unveil underlying phenomena and even try to emulate them. It is a quantum leap from the world of heuristics (rule-based decision making) to the world of statistics (stochastic-based decision making).

The penetration of Artificial Intelligence in the architectural field was already forecasted early on by a few theorists, who, before us, saw AI's potential for architectural design. Far from crafting intelligent algorithms, these precursors designed and speculated on the potential of such systems. As URBAN II was released by Negroponte and his group, the idea of a "machine assistant" was already well underway. URBAN V, a later version, would assist the designer, by adapting rooms layout– defined as blocks—to optimize adjacencies and light condition as the user draws onto a modular grid. In fact, URBAN V distinguished two layers of information: implicit and explicit. The implicit dimension is the one handled and deduced by the machine, while the explicit one is the dimension set by the user. This duality of information in URBAN V is the direct translation of the machine-human complementarity wished by Negroponte. And it is within the set of implicit parameters, that the "intelligence"—in other words, the AI- built within the machine would find its expression. Corrections proposed by the computer, by tuning the implicit parameters, would be surfaced to the users as

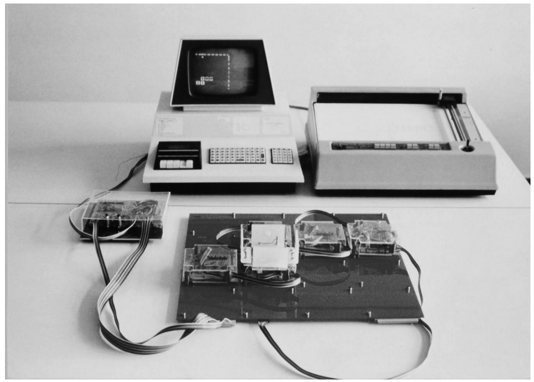


Figure 5: GENERATOR, Cedric Price

suggestions. To an ill-placed set of rooms, URBAN V would notify the user: "TED, MANY CONFLICTS ARE OCCURRING". A few years later, Cedric Price, then Professor at the chair of Architecture at Cambridge University, invented the GENERATOR (1976). Acknowledging Negroponte's work, Price used the AMG's work on AI and pushed it further, investigating the idea of autonomous ever-changing building, that would "intelligently" respond and adapt to users' behaviors. For Price, under the term "intelligent" lies the idea of encoding a behavior, that the GENERATOR would follow. However, bellow Negroponte's work, or Price's prototypes, lied an unresolved issue: the actual intelligence of the algorithm. Although the interface and protocols were in place, the actual procedural complexity of the core algorithms was still quite weak, based on simple heuristics relationships.

The design of intelligent algorithms, also called AI, actually found a renewed interest at the beginning of the '80s. The sudden increase in computational power and the steep increase of funding's brought back the question of intelligence at the center of AI's investigation. Key to this period are two main revolutions: expert systems and inference engines. The former corresponds to machines able to reason based on a set of rules, using conditional statements. An actual breakthrough at the time. The later, best exemplified by the Cyc Project, developed by Douglas Lenat, were involving machines geared towards inference reasoning. Using a knowledge base (a set of truth statements), an inference machine would be able to deduce the truthfulness of a new statement as compared to its knowledge base. It is not until the early 90's, and the mathematization of AI that the field would bring truly promising results. The advent of a new type of models would definitely reveal AI's potential: networks and machine learning. Through the utilization of a layered pipeline, also called network, a machine is now able to grasp higher complexities then previously developed models. Such models can be "trained", or in other words, tuned for specific tasks. More interesting even, is the idea embarked in one specific type of such models: Generative Adversarial Neural Networks (GANs). Theorized at first by Ian Goodfellow, researcher at Google Brain, in 2014, this model offers to use networks to generate images, while ensuring accuracy through a self-correcting feedback loop. Goodfellow's research turns upside down the definition of AI, from an analytical tool to a generative agent. By the same token, he brings AI one step closer to architectural concerns: drawing and image production. All in all, from simple networks to GANs, a new generation of tools coupled with increasingly cheaper and accessible computational power is today positioning AI as an affordable and powerful medium. If Negroponte's or Price's work were almost empty of true machine intelligence, nowadays architectural software can finally leverage such possibility.

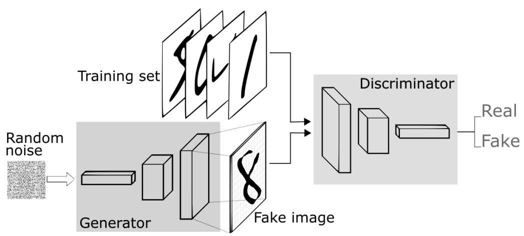
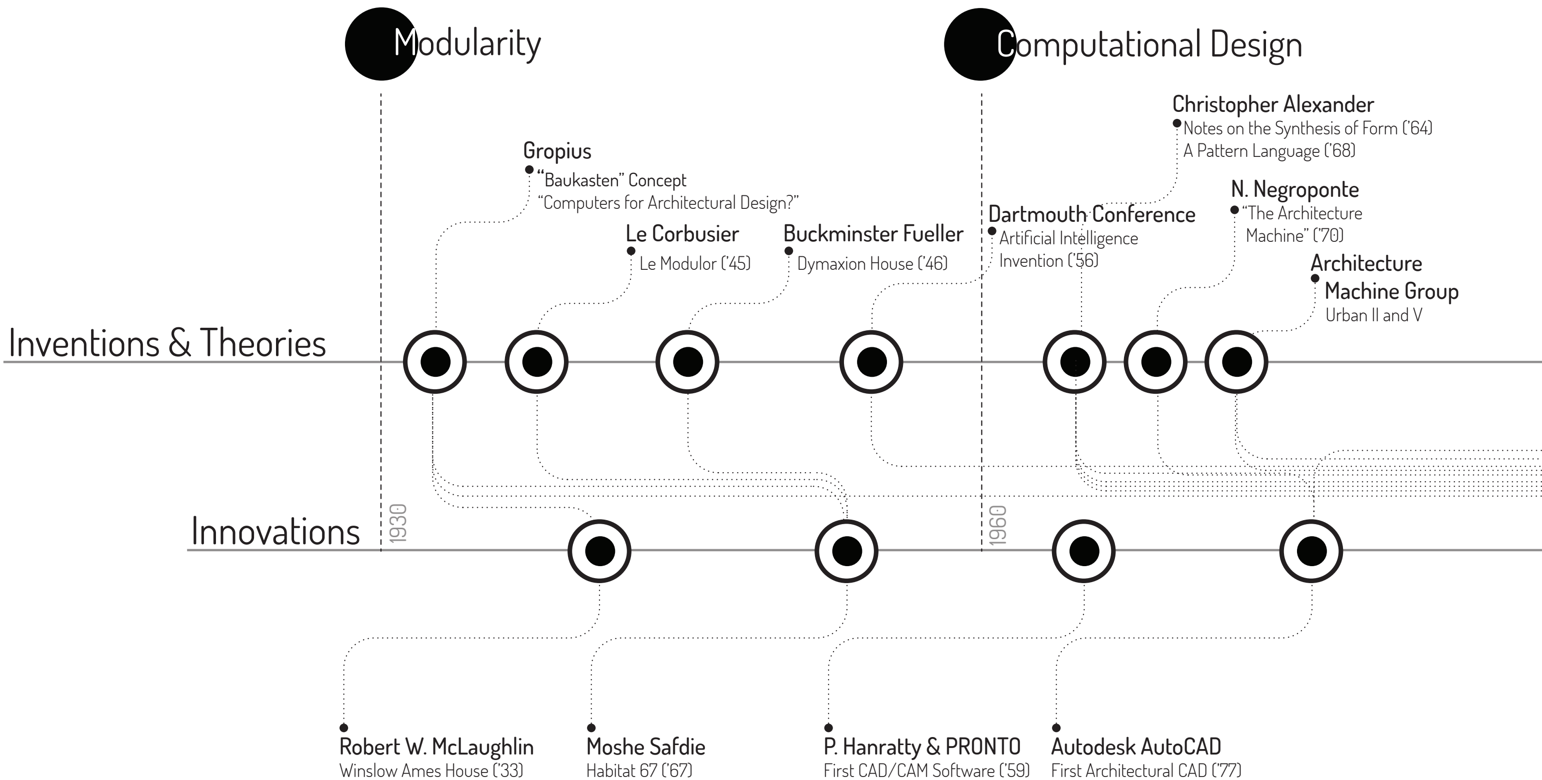
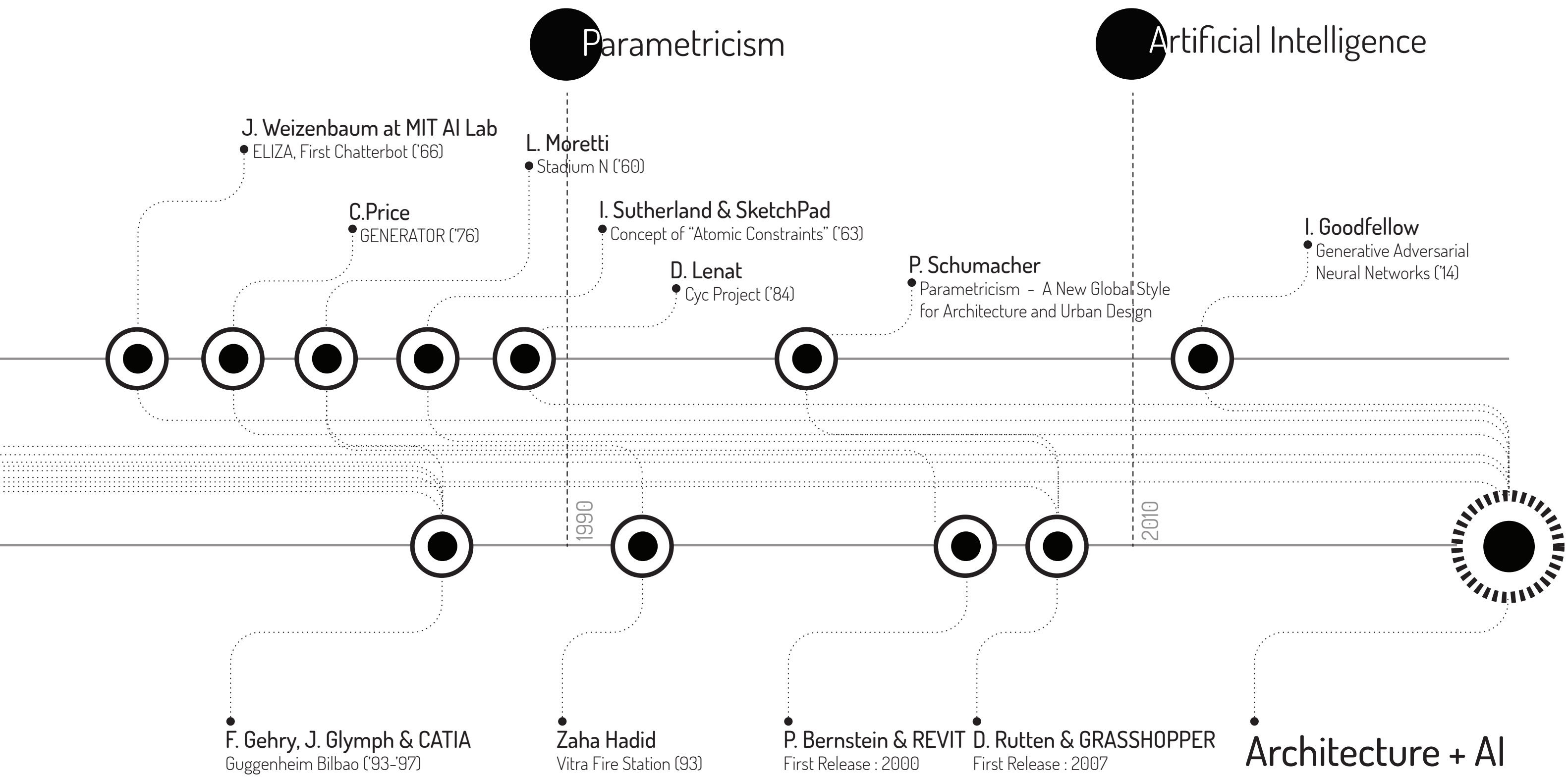
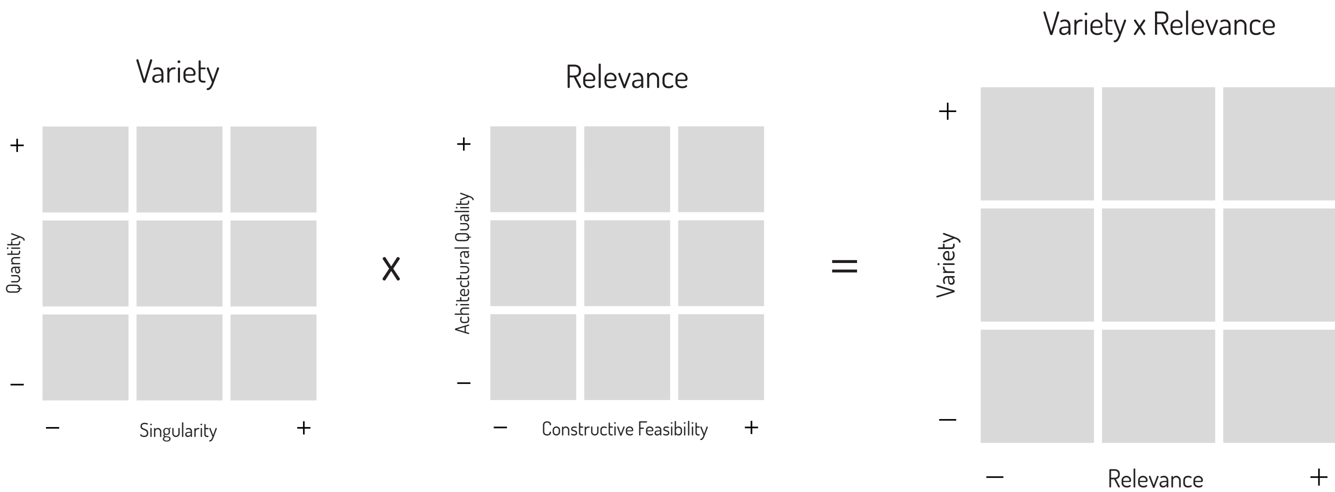


Figure 6: GAN Typical Architecture

Although the potential AI represents for Architecture is quite promising, it still remains contingent on designers' ability to communicate their intent to the machine. And as the machine has to be trained to become a reliable "assistant", architects are faced with two main challenges: (1) they have to pick up an adequate taxonomy i.e. the right set of adjectives that can translate into quantifiable metrics for the machine and (2) they must select, in the vast field of AI, the proper tools and train them. Those two preconditions will eventually determine the success or the failure of AI-enabled architecture.


Modularity
Computational Design
Christopher Alexander
Notes on the Synthesis of Form ('64)
A Pattern Language ('68)
Gropius
"Baukasten" Concept
"Computers for Architectural Design?"
N. Negroponte
"The Architecture
Machine" ('70)
Dartmouth Conference
Artificial Intelligence
Invention ('56)
Le Corbusier
Le Modulor ('45)
Buckminster Fueller
Dymaxion House ('46)
Architecture
Machine Group
Urban II and V
Inventions & Theories
1930
1960
Innovations
Robert W. McLaughlin
Winslow Ames House ('33)
Moshe Safdie
Habitat 67 ('67)
P. Hanratty & PRONTO
First CAD/CAM Software ('59)
Autodesk AutoCAD
First Architectural CAD ('77)


Parametricism
Artificial Intelligence
J. Weizenbaum at MIT AI Lab
ELIZA, First Chatterbot ('66)
L. Moretti
Stadium N ('60)
C.Price
GENERATOR ('76)
I. Sutherland & SketchPad
Concept of "Atomic Constraints" ('63)
D. Lenat
Cyc Project ('84)
P. Schumacher
Parametricism - A New Global Style for Architecture and Urban Design
I. Goodfellow
Generative Adversarial Neural Networks ('14)
1990
2010
F. Gehry, J. Glymph & CATIA
Guggenheim Bilbao ('93-'97)
Zaha Hadid
Vitra Fire Station (93)
P. Bernstein & REVIT
First Release : 2000
D. Rutten & GRASSHOPPER
First Release : 2007
Architecture + AI


Variety
+
Quantity
–
–
Singularity
+
X
Relevance
+
Achitectural Quality
–
–
Constructive Feasibility
+
=
Variety x Relevance
+
Variety
–
–
Relevance
+

B. A Continuous Progress

Modularity, Computational Design, Parametricism, and Artificial Intelligence: this four-period sequence reflects the chronology of the progress which, step-by-step, has been shaping and refining the architectural means & methods. We want to see in such momentum a form of "continuous progress" as the one experienced in the industry at large, rather than a series of unrelated disruptions. From there, an appropriate set of matrices has helped us mapping this dynamic.

First, to evidence our claim, we posit here that Architecture can be understood as a process of generating designs one can describe through two dimensions: on one side the diversity of the output produced or "Variety" and on the other side the applicability of the designs or "Relevance".

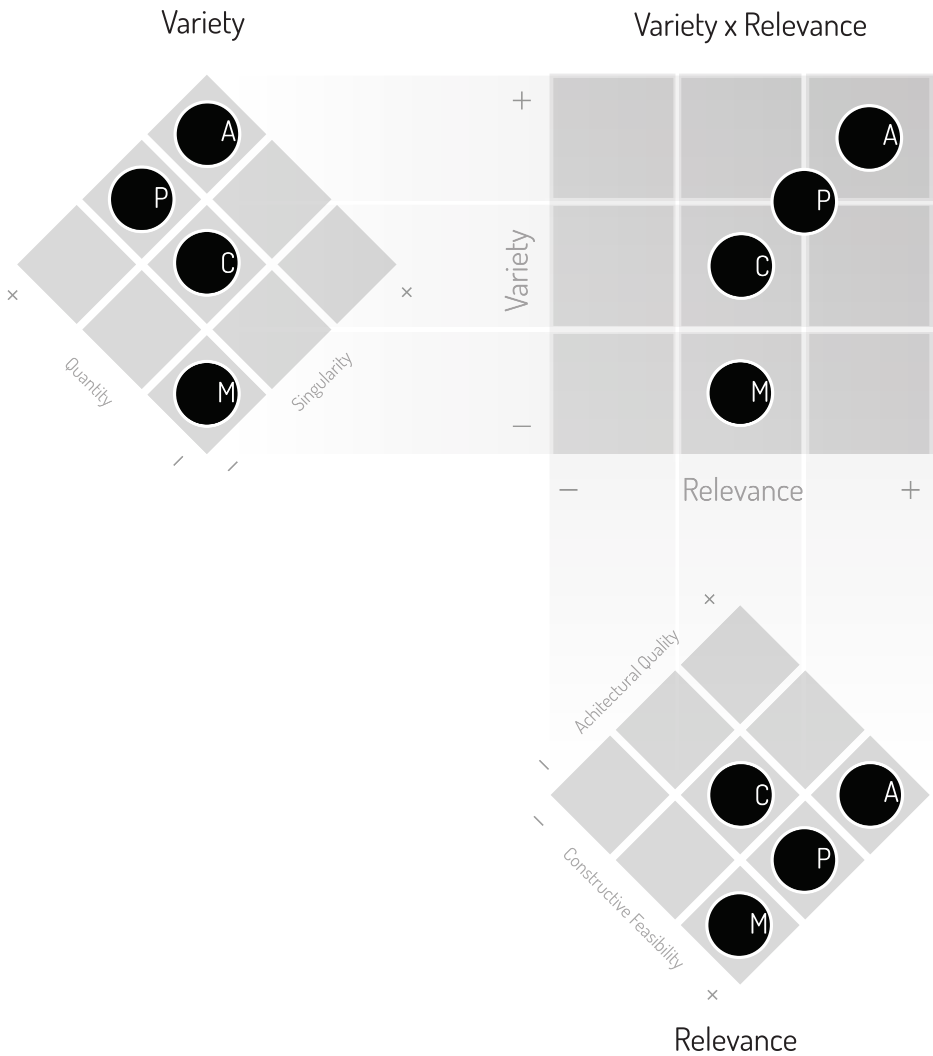
> "**Variety**" is contingent upon two underlying metrics: the "quantity of designs" sizing the volume of options created and the "singularity of designs" measuring their respective disparity.

> "**Relevance**" compounds the "constructive feasibility", i.e. the workability of the designs and their "architectural quality" including optimal program organization, space layout and contextual fit.

Ultimately, the combination of Variety (Quantity X Singularity) and Relevance (Constructive Feasibility X Architectural Quality) creates a framework which (1) maps out and contrasts the respective positioning of our four periods -Modularity, Computational Design, Parametricism, and Artificial Intelligence and (2) clearly evidences the culminating point of progress AI represents for our discipline.

- Modularity
- Computational Design
- Parametricism
- Artificial Intelligence

This display, although directional and qualitative, is a powerful grid to represent the concept which lies at the heart of our thesis. In summary, the dynamic of this continuous progress has been triggered by the limit of each movement at a certain point in time, exacerbated by the competition of the new one coming in.


Variety
Variety x Relevance
A
P
C
M
Quantity
Singularity
+
Variety
—
—
Relevance
+
Achitectural Quality
Constructive Feasibility
Relevance

Modularity

Since its inception, modular conception has proved to be a highly constraining system, yielding low variety of potential design options. Although such options were easily meeting construction feasibility criteria, their architectural quality got questioned early on.

Computational Design

With computation design, designer could finally afford to escape the rigidity of the grid, to design feasible buildings, with actual singularity. Conception software, allowing to solve complex shapes, would help drive down conception costs and generate more design iterations.

Parametricism

Parametricism brought even more control over organic shapes, increasing constructive feasibility. By systematizing their geometry, entire buildings could be discretized into buildable elements, and resolved assembly systems. The parametric style, however, got rapidly trivialized, generating generic and repetitive patterns spaces leading to poor architectural quality.

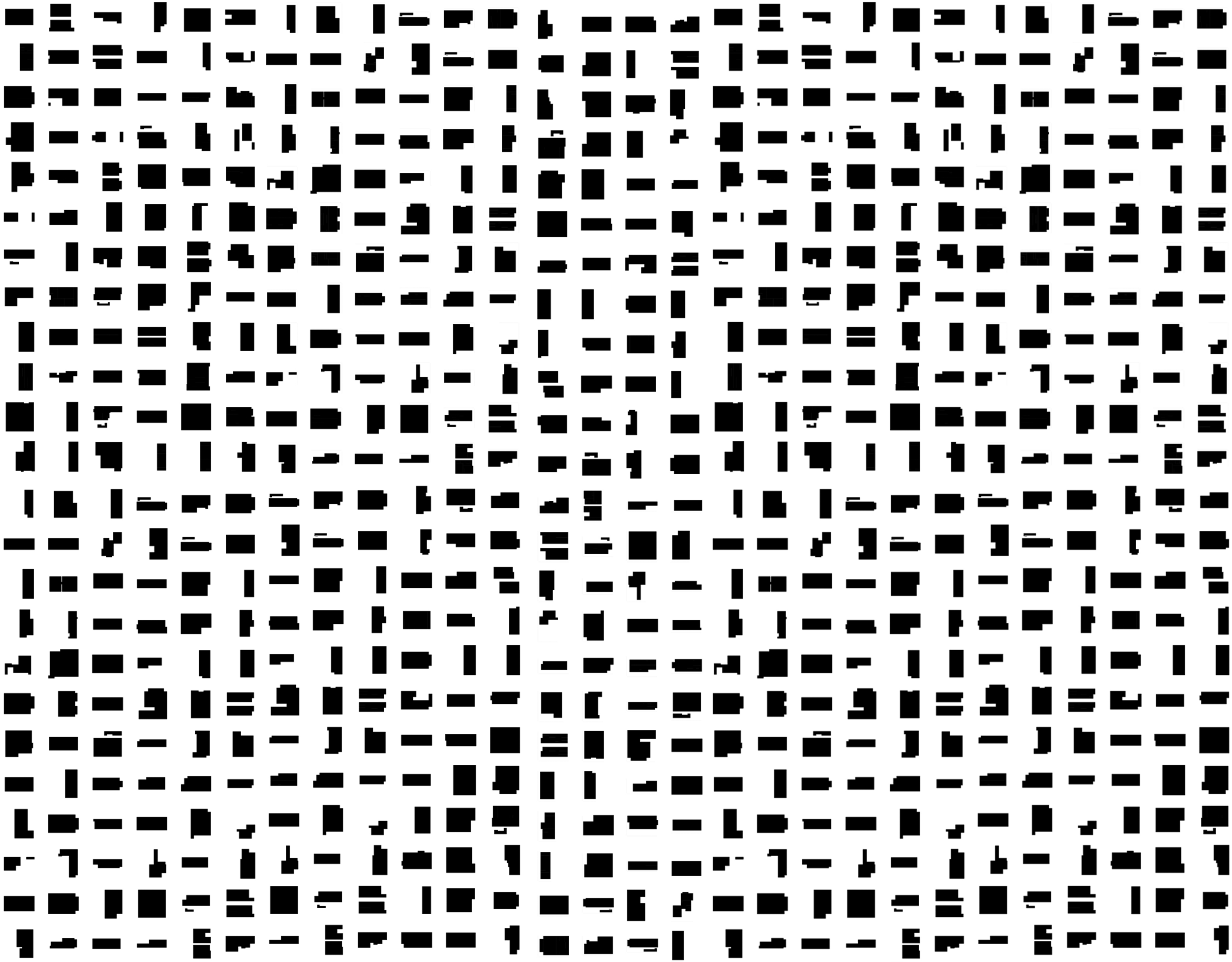
Artificial Intelligence

Artificial intelligence finally carries the promise of embarking all the benefits of previous generations, while escaping the legibility and generic style of Parametricism. It is the ultimate push towards greater architectural quality and, to us the springboard of a brand-new era.

C. Closing Remarks

We are today faced with a fantastic challenge: bringing AI to the world of architectural design. We know we will have to resist to preconceived ideas and natural fears. Let's make it clear: we do not believe that AI will ever automate the architect's intuition and will substitute to his/her sensitivity. We consider that, in the foreseen future, humans will continue using the machine as their tool; not the other way around. However, we are also convinced that the inception of an AI-powered "intelligent assistant" is a game changer within our reach. And, as previously documented, we consider it conceptually fits in a technological continuum, away from the simplistic extension of the tabula rasa theory. It is why we are quite confident that the architectural community while recognizing the radical breakthrough AI represents, will carefully study, test and experience its applications further.

Meanwhile, it is the attempt of our thesis, expressing our belief that AI can balance design efficiency and organicity in an unparalleled way. We want to simply demonstrate here that AI-powered space design can help optimally combine hard sciences, such as engineering, math & data science, and softer-sciences, including design, architecture & planning. We offer in the next chapters a framework with the intent to shape this challenging topic and bring early evidence of its materiality. Our hope is to jump-start the discussion and lay down the premise of machine-assisted architectural design.



D. Framework

Our work finds itself at the intersection of Architecture and Artificial Intelligence. The former is the topic, the latter the method. Both have been simplified into clear & actionable categories.

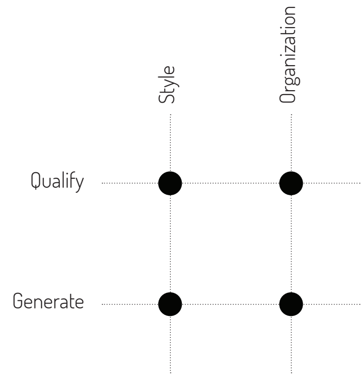
Architecture is here understood as the intersection between Style and Organization. On one hand, we consider buildings as vectors of a cultural significance, that express through their geometry, taxonomy, typology, and decoration a certain style. Baroque, Roman, Gothic, Modern, Contemporary: as many architectural styles that can be found through a careful study of floor plans. On the other hand, buildings are the product of engineering and science, answering to strict frameworks and rules -building codes, ergonomics, energetic efficiency, egress, program, etc—that can be found as we read a floor plan. This organizational imperative will complete our definition of Architecture and drive our investigation.

Artificial Intelligence will be employed, using two of its main fields of investigation—Analytics and Generative Adversarial Networks—as an investigative tool.

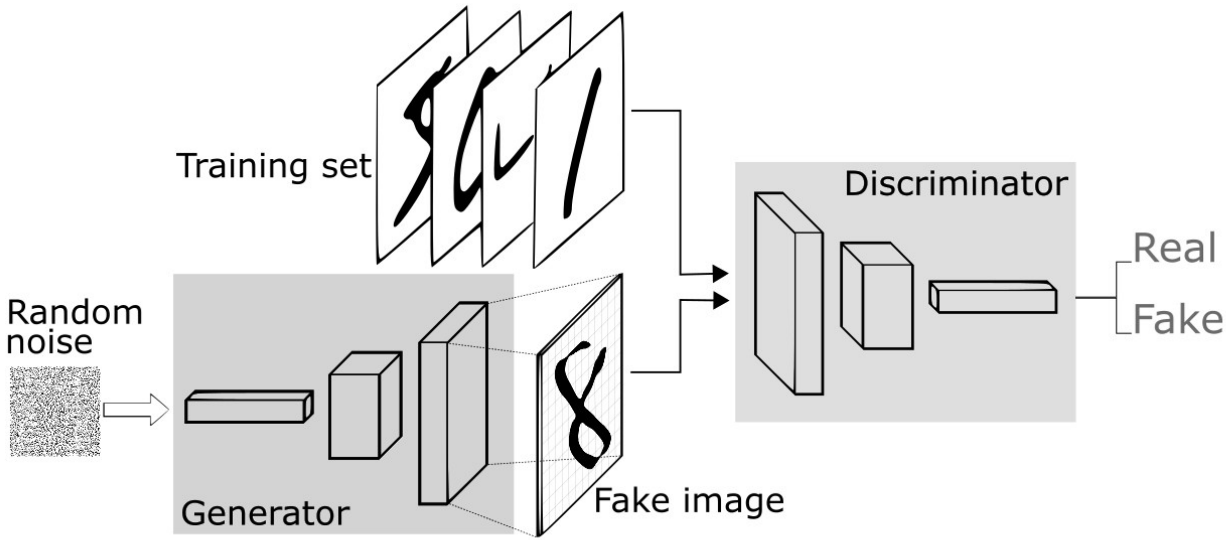
At first, we will dive into the topic of Generation. Using GANs, we offer to educate our own AI systems to architectural design. We postulate that its utilization can enhance the practice of the architectural discipline. This field is as recent as it is experimental and yields to this day surprising results. Our hope is to be able to train it to draw actual building floor plans.

Then, we will come up with a robust analytical framework to qualify and classify the generated floor plans. Ultimately, our goal is to organize the results of our GANs, to offer the possibility for the user to browse seamlessly through the variety of created design options. To that end, the quantity and ubiquity of tools offered by Data Science will prove to be valuable to our investigation.

Through this dual lens, at the crossroad of Style & Organization, Qualification & Generation, we lay down a framework that organizes the encounter of Architecture & AI.



Framework Matrix | Source: Author



Typical GAN Architecture

Generate II

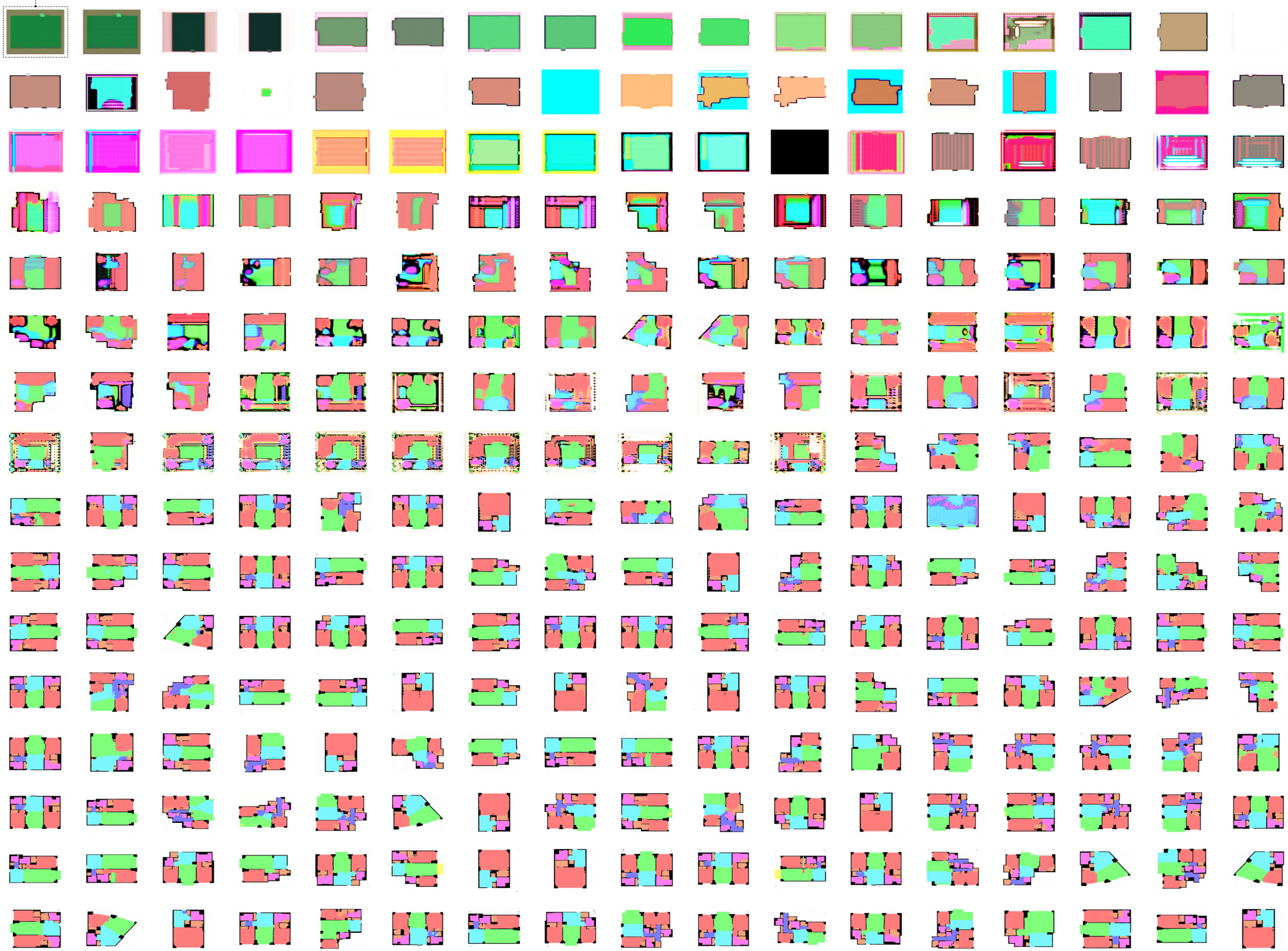
The design of architectural floor plans is at the core of the architecture practice. Its mastery stands as the gold standard of the discipline. It is an exercise that practitioners have overtime relentlessly tried to improve through technology. In this first part, we dive into the potential of AI applied to floor plan generation, as a mean to push the envelope even further.

Using our framework, to tackle floor plans' style and organization, we lay down in the following chapter the potential of AI-enabled space planning. Our objective is to offer a set of reliable and robust tools to both evidence the potential of our such an approach and to test our assumptions.

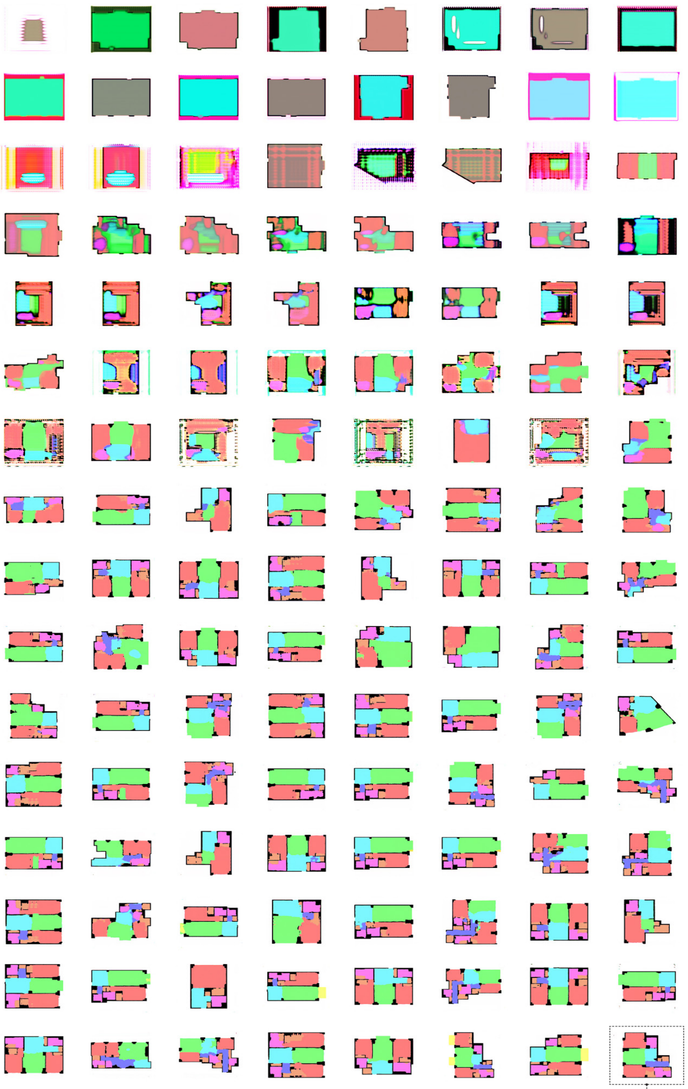
The challenge is here threefold: (1) choosing the right toolset, (2) isolating the right phenomena to be shown to the machine and (3) ensuring that the machine "learns" properly.

AI & Generative Adversarial Neural Networks

Generative Adversarial Neural Networks—or GANs- are here our weapon of choice. Within the field of AI, Neural Networks stands as a key field of investigation. The creative ability of such models has been recently evidenced, through the advent of Generative Adversarial Neural Networks. As any machine-learning model, GANs learn statistically significant phenomena among data presented to them. Their structure, however, represents a breakthrough: made of two key models, the Generator and the Discriminator, GANs leverage a feedback loop between both models to refine their ability to generate relevant images. The Discriminator is trained to recognize images from a set of data. Properly trained, this model is able to distinguish between a real example, taken out of the dataset, from a "fake" image, foreign to the dataset. The Generator, however, is trained to create images resembling images from the same dataset. As the Generator creates images, the Discriminator provides him with some feedback about the quality of its output. In response, the Generator adapts, to produce even more realistic images. Through this feedback loop, a GAN slowly builds up its ability to create relevant synthetic images, factoring in phenomena found among observed data.



Training Sequence, Model II | Source: Author

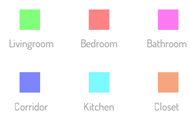


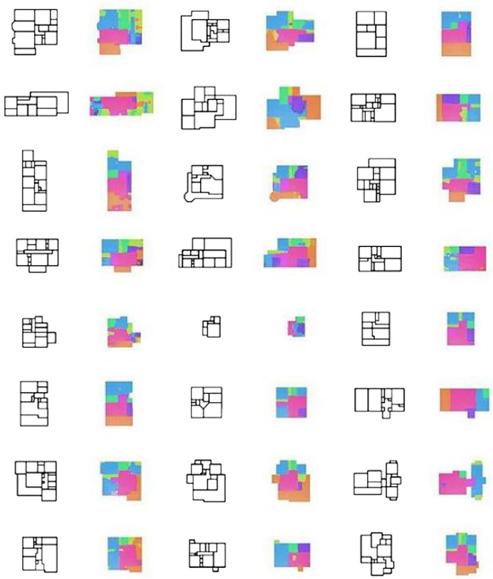
Representation & Learning

If GANs represent a tremendous opportunity for us, knowing what to show them is crucial. We have here the opportunity to let the model learn directly from floor plan images. By formatting images, we can control the type of information that the model will learn. As an example, just showing our model the shape of a parcel and associated building footprint will yield a model able to create typical building footprints given a parcel's shape. To ensure the quality of the outputs, we will use our own architectural "sense" to curate the content of our training sets: a model will only be as good as the data we give him, as architects.

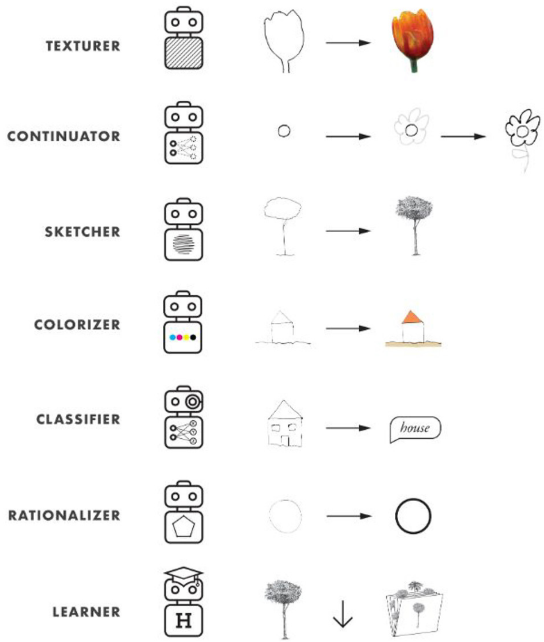
Here on the left, we illustrate a typical training sequence: this sequence, realized over the course of a day and half of training, displays how one of our GAN-models progressively learns how to layout rooms and fenestration for housing units.

Although the initial attempts are imprecise and confuse, after 250 iterations the machine builds for itself some form of intuition.



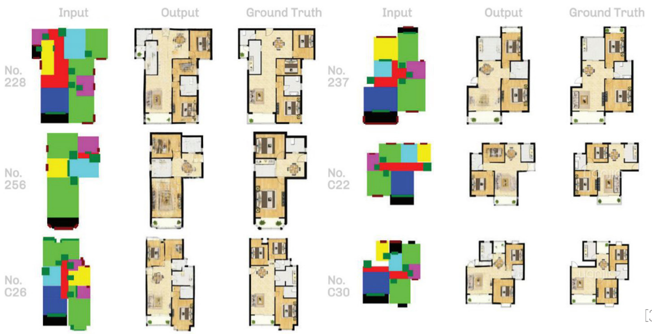


[5]



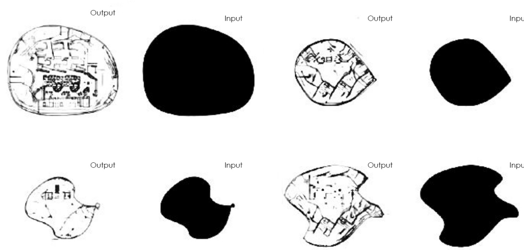
TEXTURER
CONTINUATOR
SKETCHER
COLORIZER
CLASSIFIER
house
RATIONALIZER
LEARNER

[6]



Input
Output
Ground Truth
Input
Output
Ground Truth
No. 228
No. 256
No. C26
No. 237
No. C22
No. C30

[3]



Output
Input
Output
Input
Output
Input
Output
Input

[3]

A. Precedents

Since the early work of Goodfellow et al. in 2014 **[1]**, the field of Generative Adversarial Nets (GAN) has expanded, from simple hand-drawn digits generation to more complex taxonomies generation. Isola et al. in November 2018 set a key precedent in this field **[2]**, by enabling image-to-image translation. In their model, Pix2Pix, pairs of images are fed to a GAN architecture, while the network learns the proper mapping from one image to the other. Part of their work investigates building façade generation and opens the door to architectural design using GAN.Professor Andrew Witt expands on this work in his exhibition QUILTING
By enlarging the final layers of Pix2Pix, Witt creates larger facade designs, showcased as one linear endless animation of an urban skyline.

Remaining within the realm of Architecture, floorplans design using GAN is first studied by Zheng and Huang in 2018 **[3]**. Using Pix2PixHD **[4]**, the authors propose to use GANs for floorplan recognition and generation. Floorplan images processed by their GAN architecture are translated into programmatic patches of colors.

Inversely, patches of colors in their work are turned into drawn floorplans. If the position of openings and rooms is specified by the user, the elements actually being laid out by the network are the furniture. The same year, Nathan Peters **[5]** in his thesis at the Harvard Graduate School of Design tackles the possibility of laying out rooms across a single-family home footprint. Peters' work turns an empty footprint, into programmatic patches of color, without specified fenestration.

Regarding GANs as design assistants, Nono Martinez' thesis **[6]** at the Harvard GSD in 2017 investigates the idea of a loop between the machine and the designer to refine the very notion of "design process". Martinez trains models for specific sketching tasks and offers an interface through which users can solicit each model at different moments of the drawing process.

Our work expands on these precedents and offers to nest 3 models (footprint, program, and furnishing) to create a full "generation stack" while improving results quality at each step. By automating multi-units processing, our work then scales to entire buildings generation, and masterplan layouts. We further offer an array of models dealing with style transfer. Finally, our contribution adds a rigorous framework to parse and classify resulting outputs, enabling users to "browse" consistently through generated options.

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. 2014. "Generative Adversarial Networks". arXiv:1406.2661.

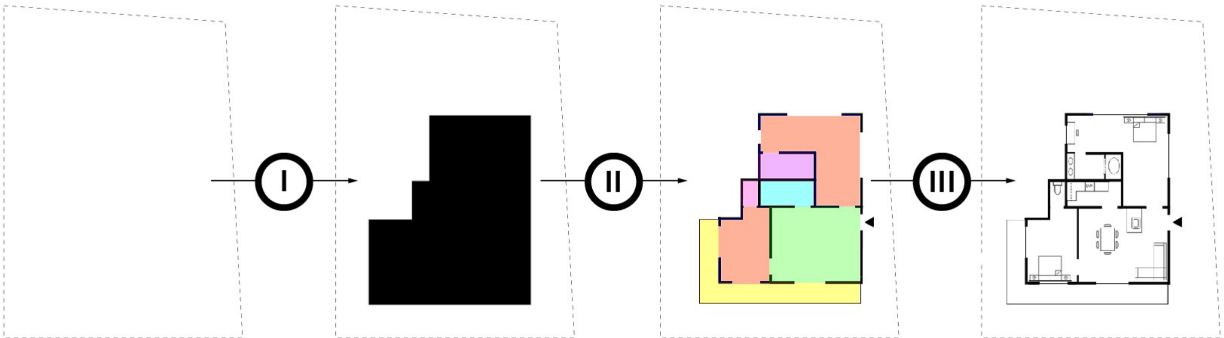
[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. 2017. "Image-to-Image Translation with Conditional Adversarial Networks". CVPR. arXiv:1611.07004.

[3] Hao Zheng, Weixin Huang. 2018. "Architectural Drawings Recognition and Generation through Machine Learning". Cambridge, MA, ACADIA.

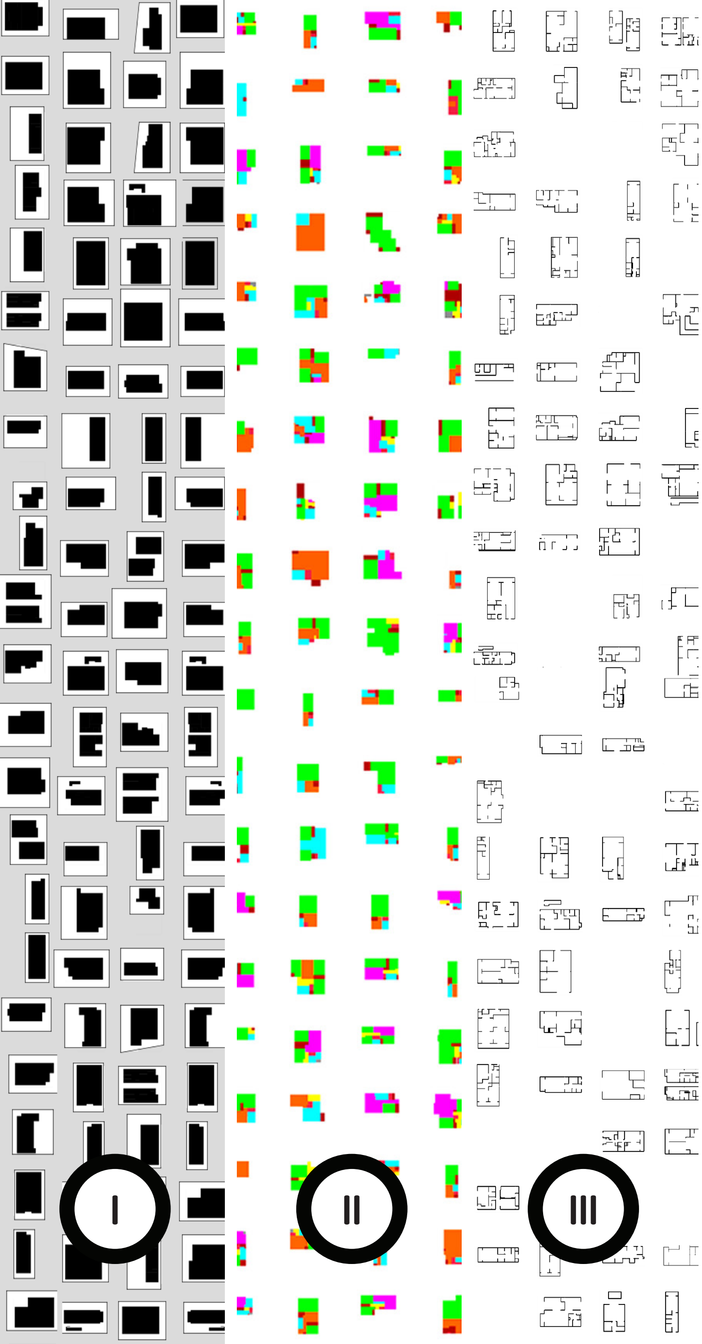
[4] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, Bryan Catanzaro. 2018. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. CVPR. arXiv: 1711.11585v2.

[5] Nathan Peters. 2017. Master Thesis: "Enabling Alternative Architectures: Collaborative Frameworks for Participatory Design".Harvard Graduate School of Design, Cambridge, MA.

[6] Nono Martinez. 2016. "Suggestive Drawing Among Human and Artificial Intelligences", Harvard Graduate School of Design, Cambridge, MA.



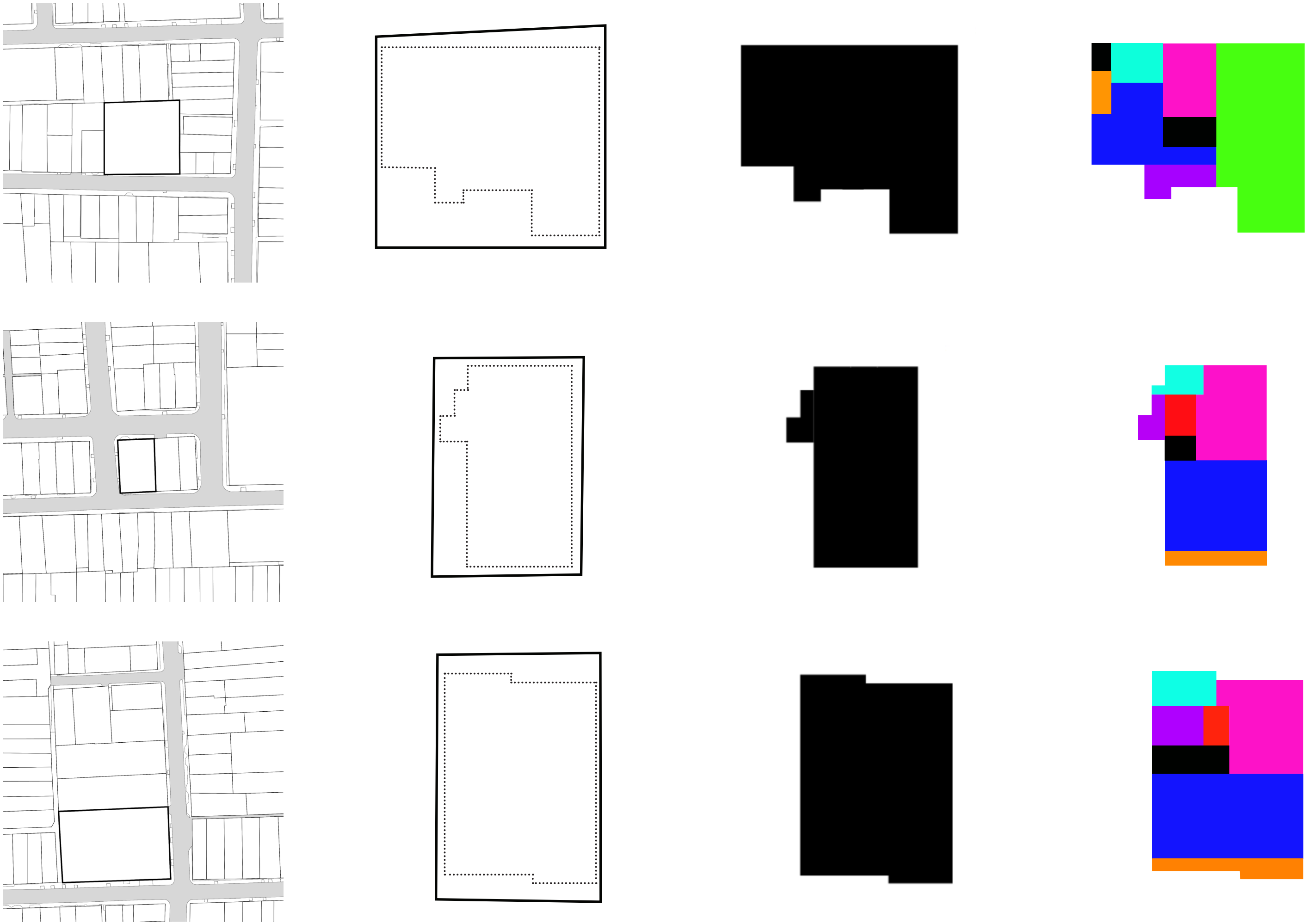
Generation Pipeline, Model I to III | Source: Author

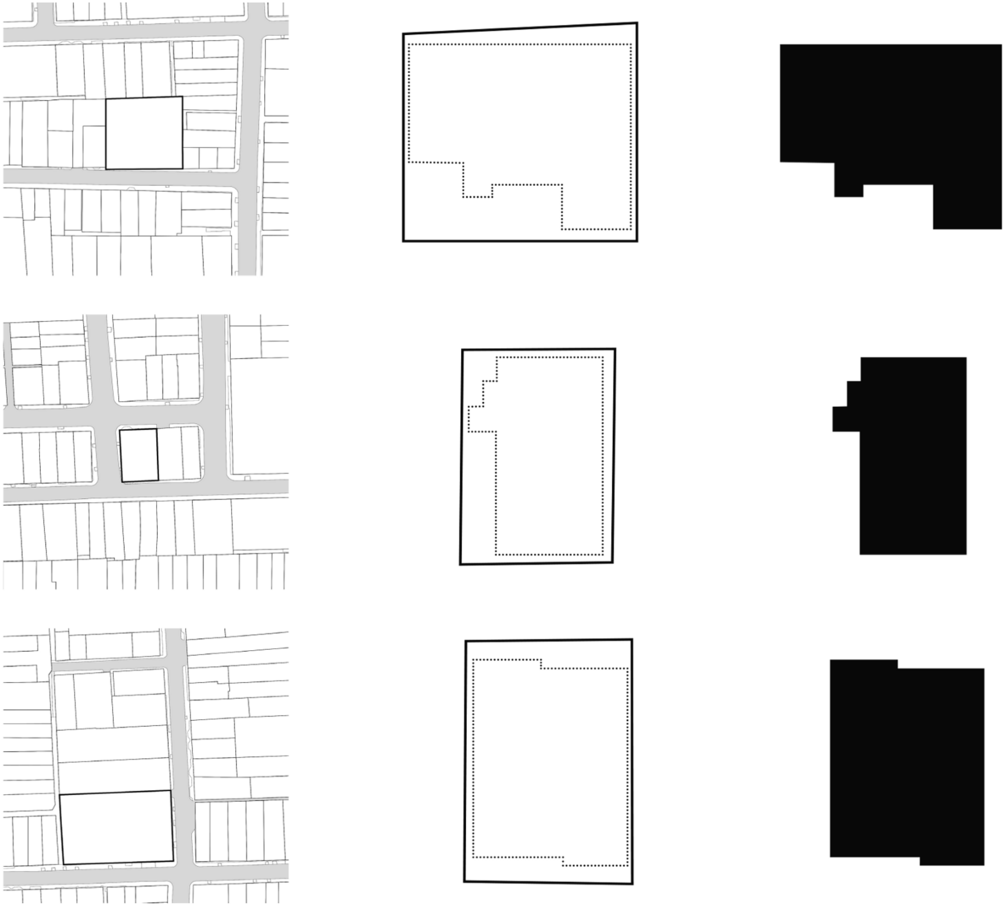


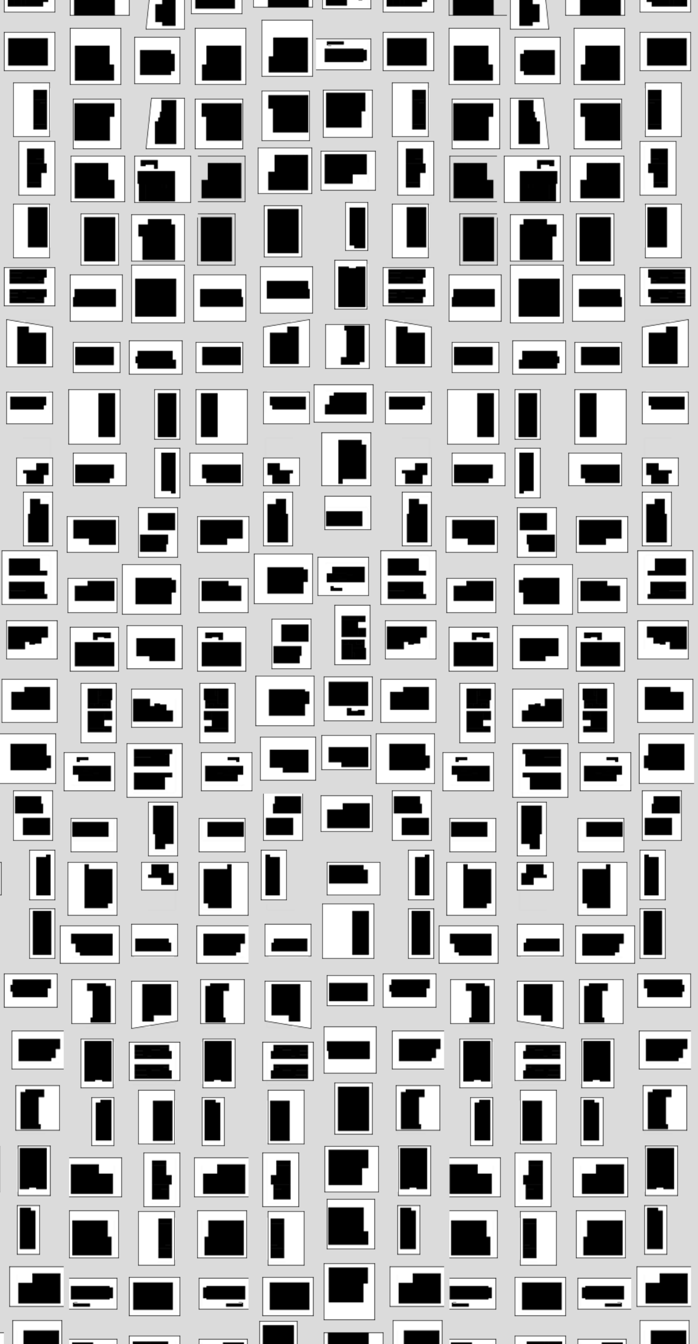
B. Organization

In this section, we offer a multi-step pipeline, integrating all the necessary steps to draw a floorplan. Jumping across scales, it emulates the process taken by an architect and tries to encapsulate each step into one specific model, trained to perform each given operation. From **(I)** the parcel to the building footprint, **(II)** from the footprint to a room split, **(III)** from a room split to a furnished one, each step has been carefully engineered, trained and tested.

At the same time, by dividing the pipeline into discrete steps, the system allows for the architect's intervention between each model. As each model generates multiple options at each step, the architect's ability to select the output of a model and edit it before transferring it to the next model keeps him/her in control of the design process. Its input shapes the decisions made by the model, therefore achieving the human-machine interaction expected.



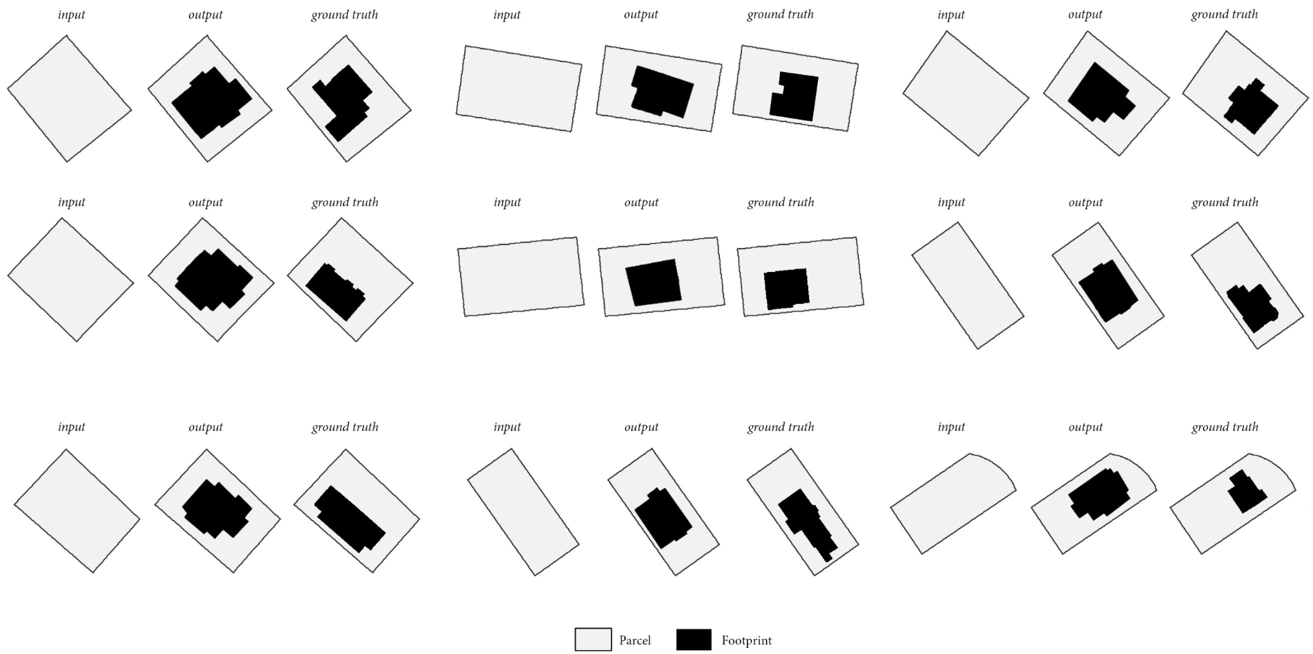




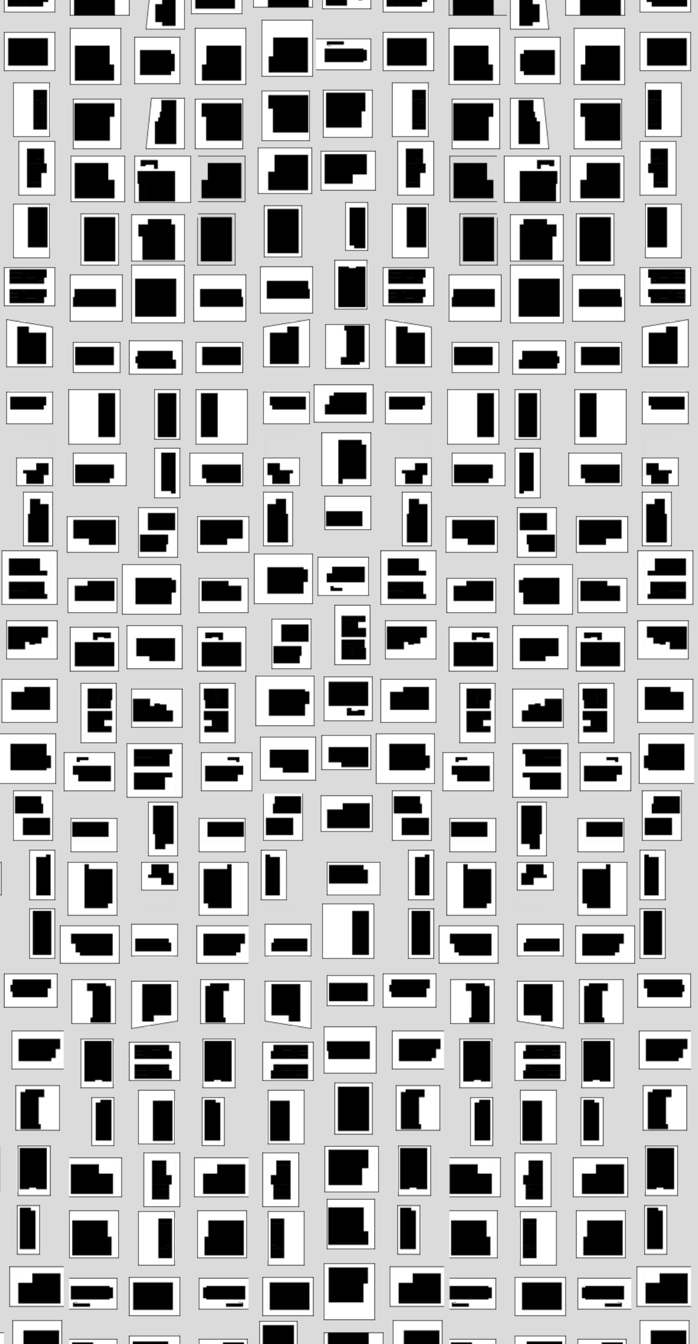
Footprint

The first step in our pipeline tackles the challenge of creating an appropriate building footprint for a given parcel geometry.

To train this model, we used an extensive database of Boston's building footprints and were able to create an array of models, each tailored for a specific property type: commercial, residential (house), residential (condo), industrial, etc.

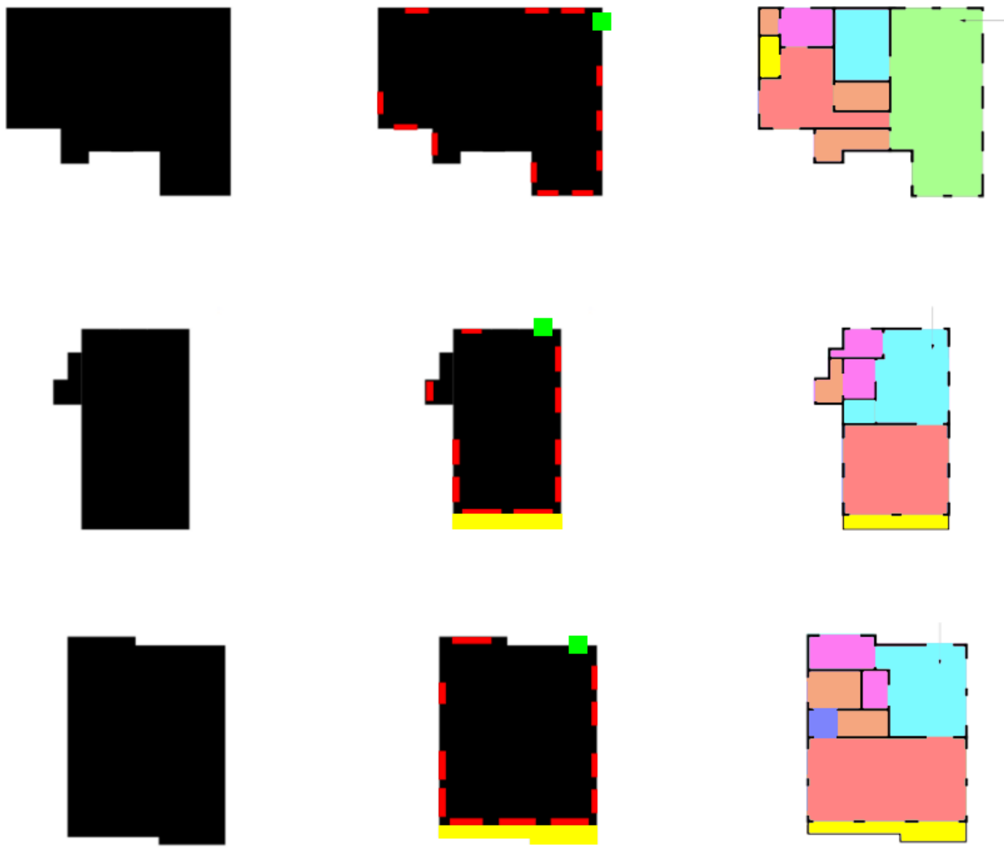


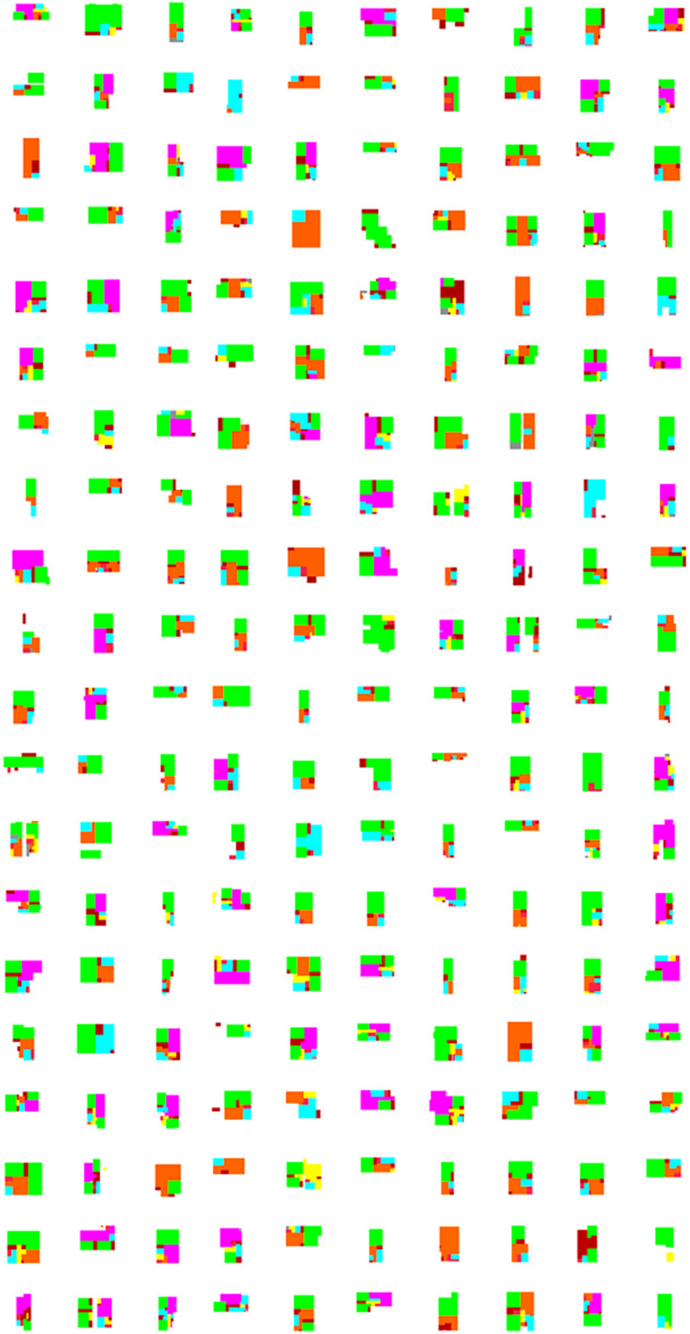
Results: Generated Footprints | Source: Author



Each model is able for a given parcel, to create a set of relevant footprints, resembling in dimension and style the type it was trained for.

9 examples, using the residential model are shown here on the left.

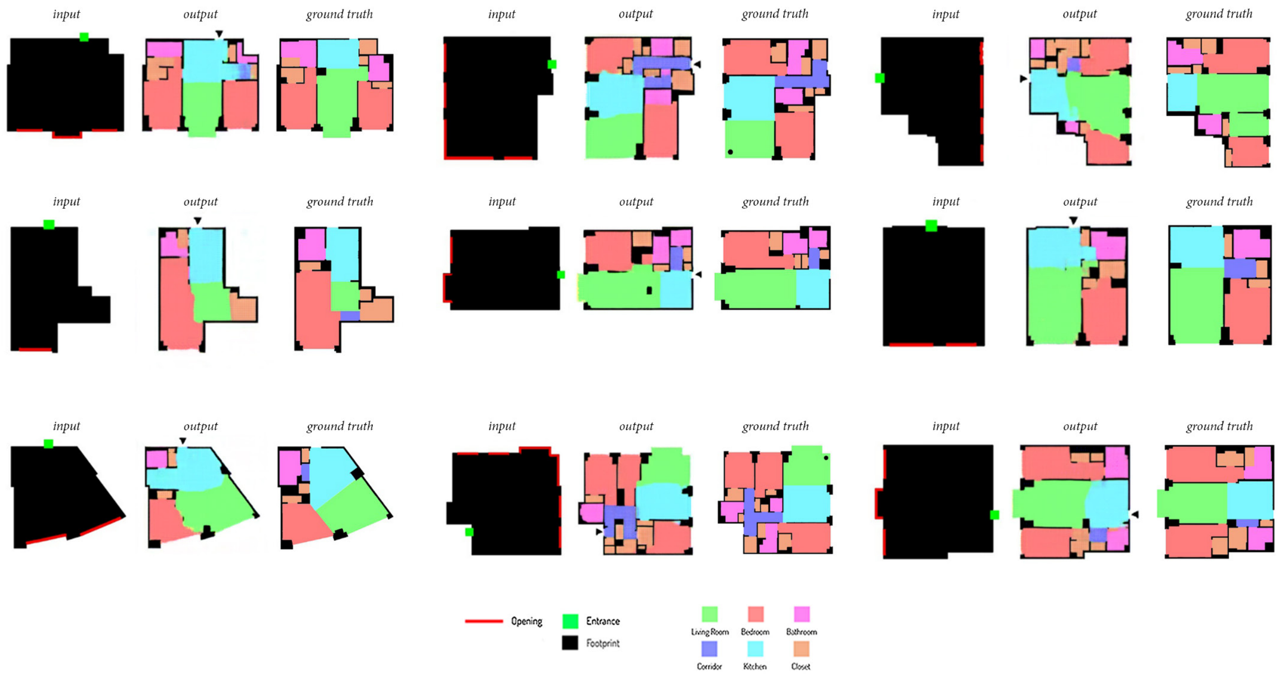




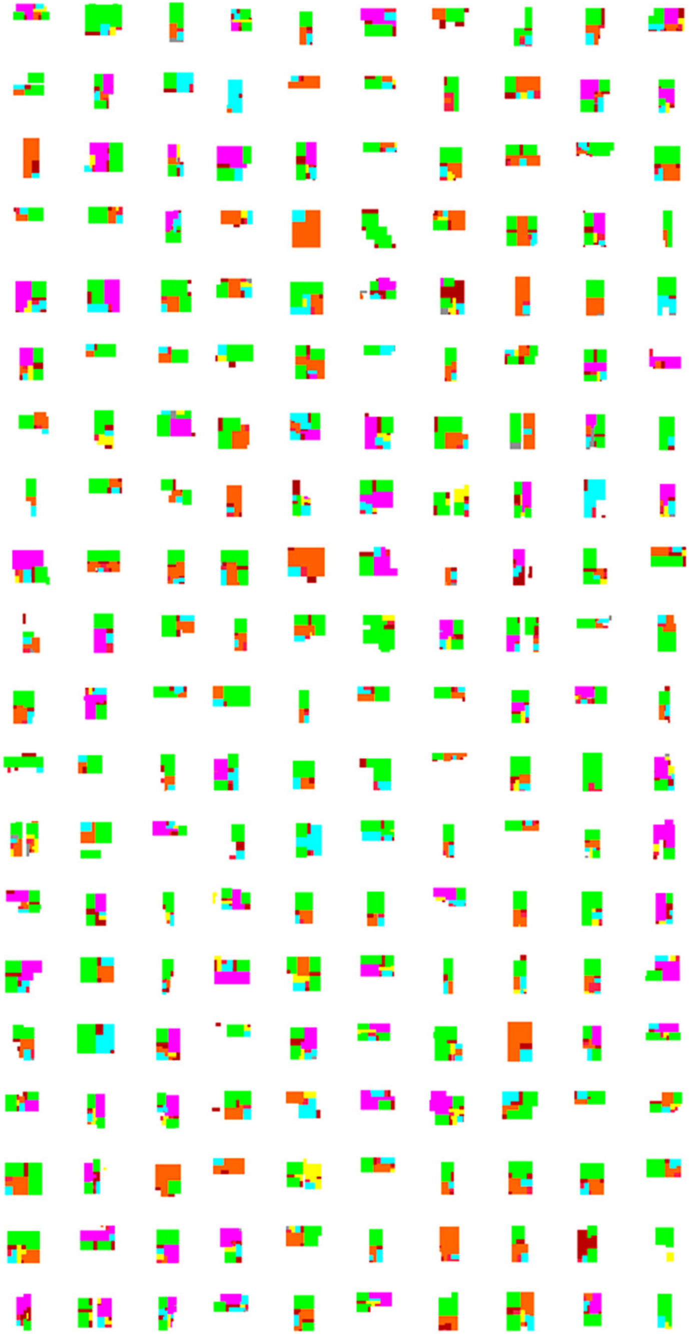
Room Split

The layout of rooms across a building footprint is the natural next step. Being able to split a given floor plan, while respecting meaningful adjacencies, typical room dimensions and proper fenestrations is a challenging process, that GANs can tackle with surprising results.

Using a dataset of around 700+ annotated floor plans, we were able to train a broad array of models.



Results: Generated Program & Fenestration | Source: Author

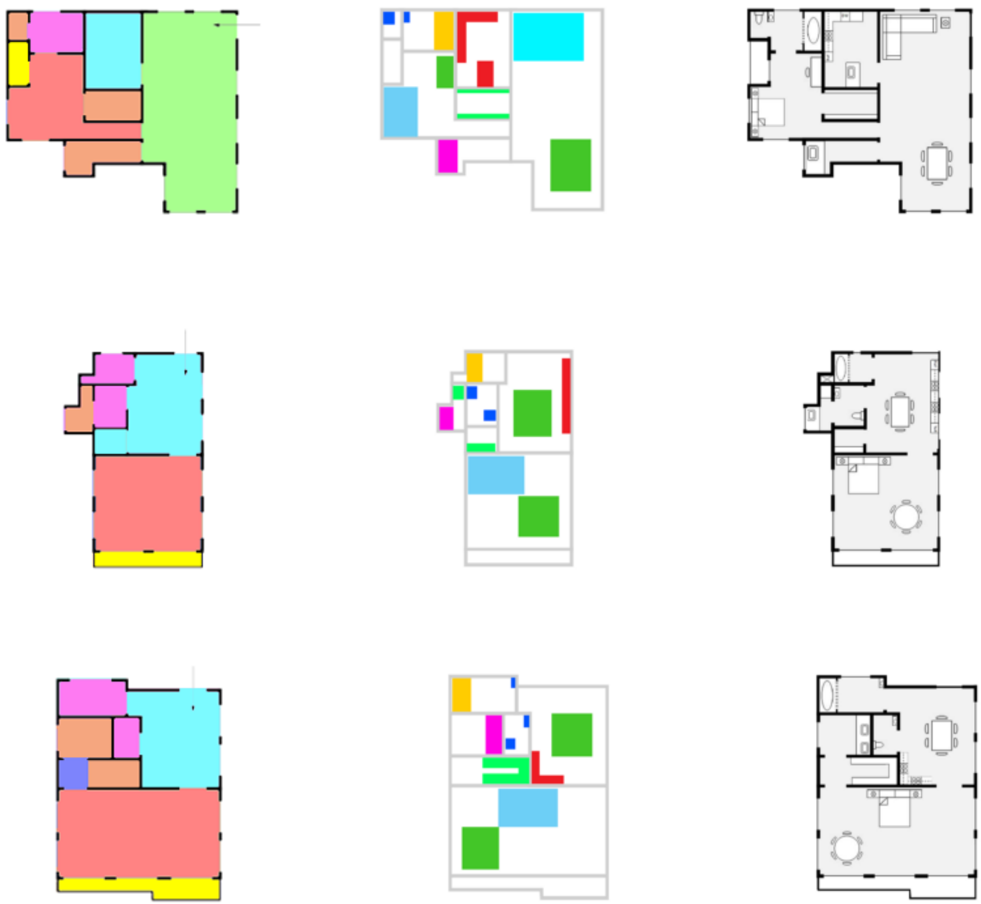


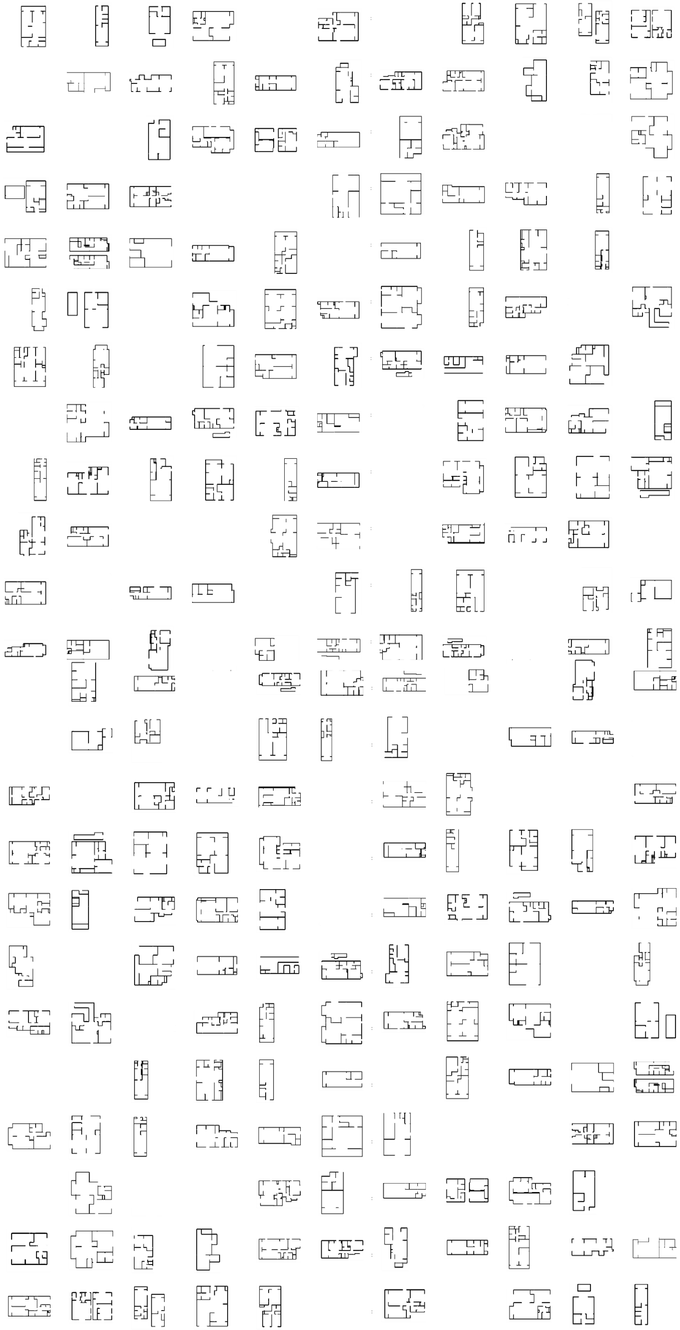
Each model is geared towards a specific room count and yields surprisingly relevant results once used on empty building footprints.

We display here on the left some typical results.

More results can be found at the following address:

http://stanislaschaillou.com/thesis/GAN/unit_opening_results/



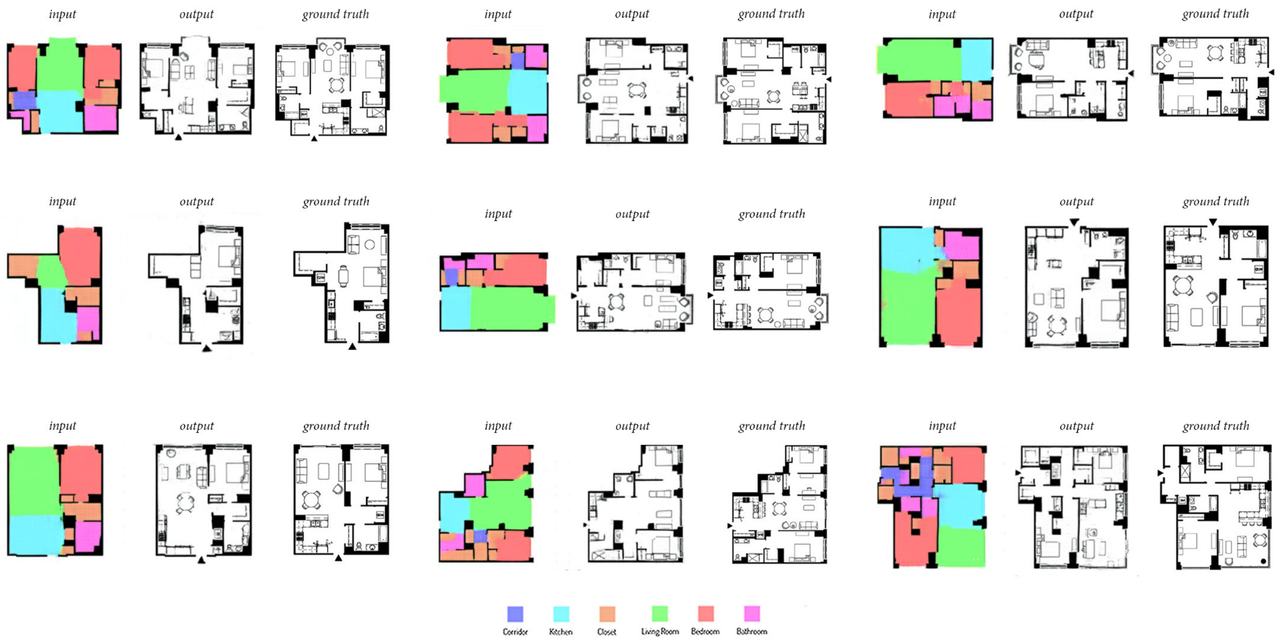


Furnishing

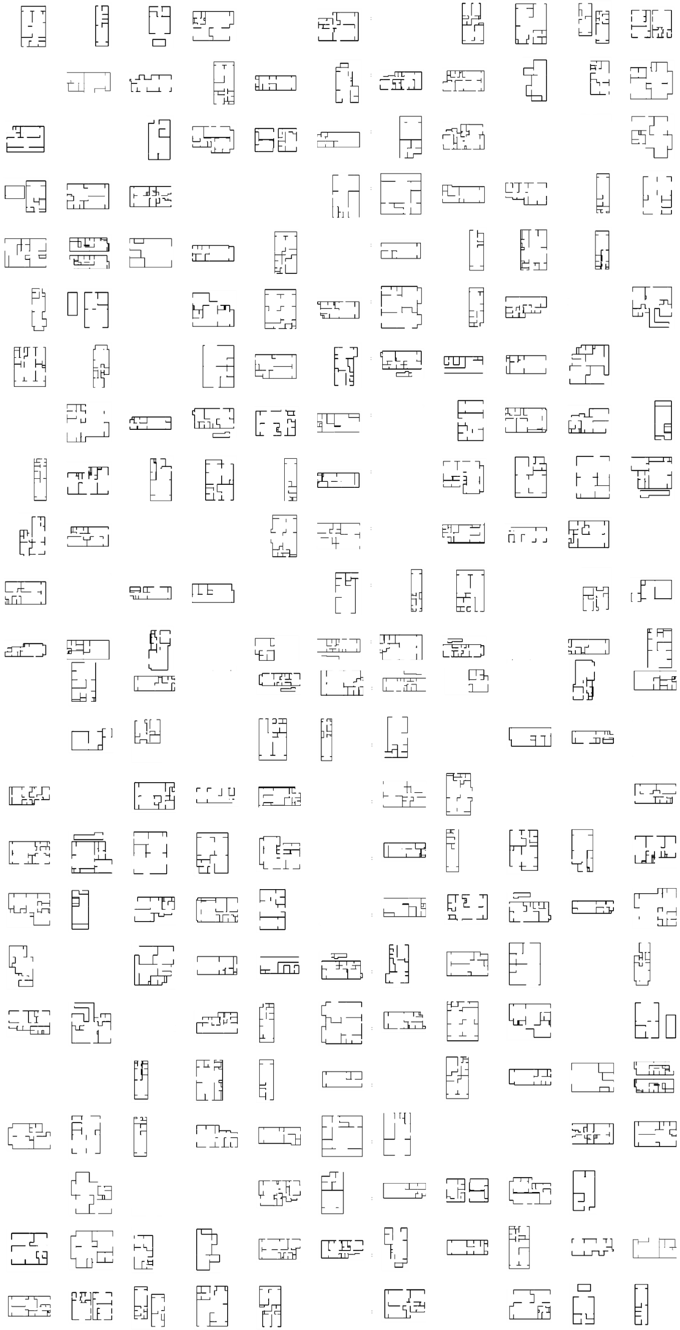
This last step brings the principle of generation to its most granular level: the addition of furniture across space. To that end, we trained at first a model to furnish the entire apartment all at once.

The network was able to learn, based on each room program, the relative disposition of furniture across space, and the dimensions of each element.

For the sake of time and accuracy, we simplified our models to restricted types of furniture.



Results: Furnished Units | Source: Author

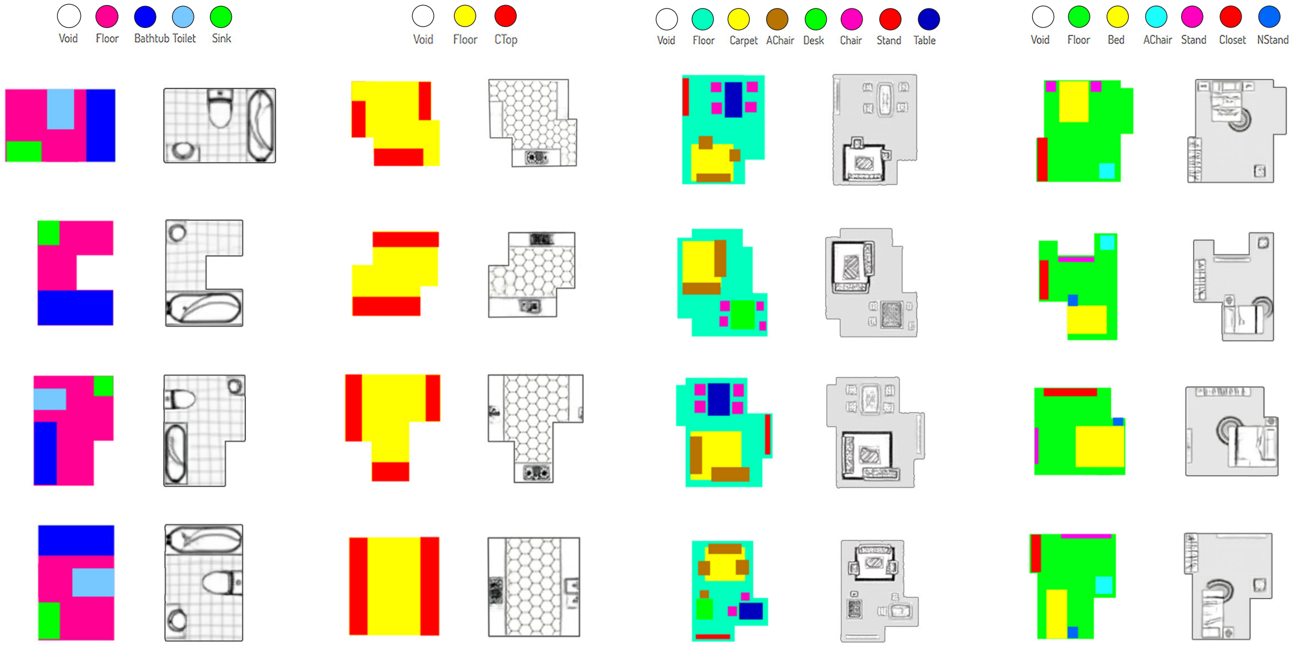


The geometry of furniture is not always perfectly accurate. However furniture types are legible, and their relative position in space is reasonable.

We display here on the left some typical results.

More results can be found at the following address:

http://stanislaschaillou.com/thesis/GAN/unit_furnishing_results/

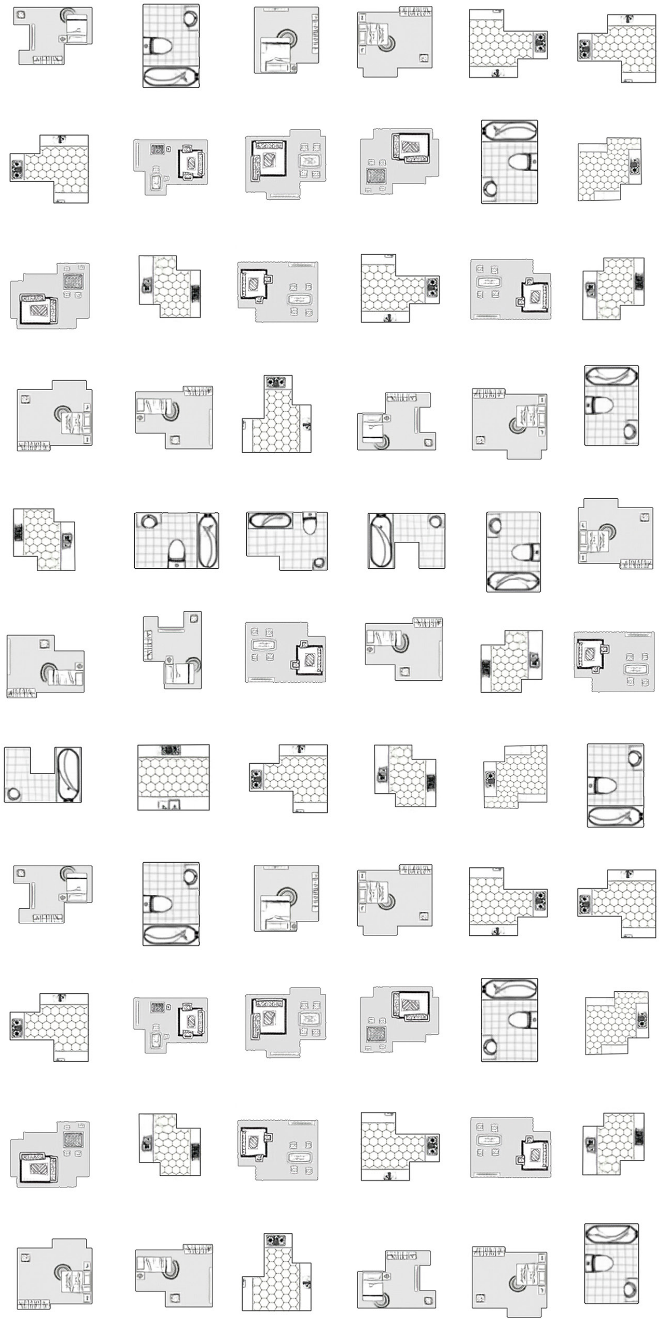


Rendered Rooms: Bathrom, Kitchen, Livingroom, Bedroom | Source: Author

Room Rendering Interface

Simple User Interface for Bathroom, Livingroom, Bedroom & Kitchen Floorplan Rendering

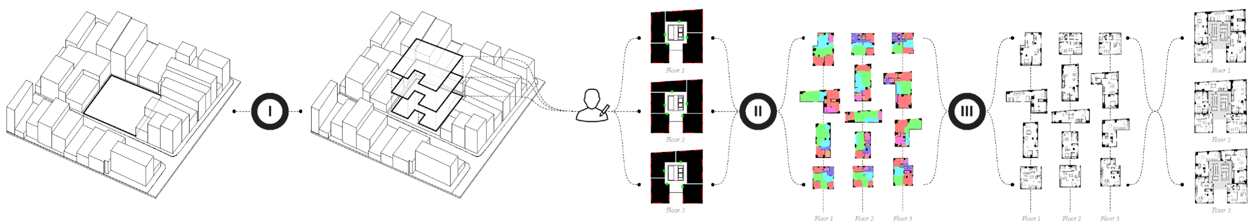
http://stanislaschaillou.com/thesis/GAN/canvas



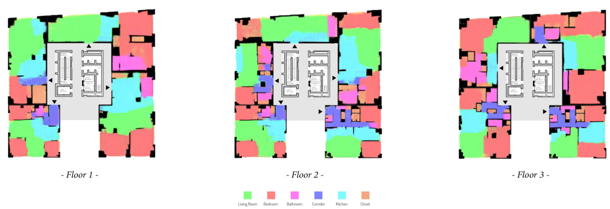
Room Rendering

If the previous results can give a rough idea of potential furniture layouts, the quality of the resulting drawings is still too fuzzy. To further refine the output quality, we have trained an array of additional models, for each room type (living room, bedroom, kitchen, etc...).

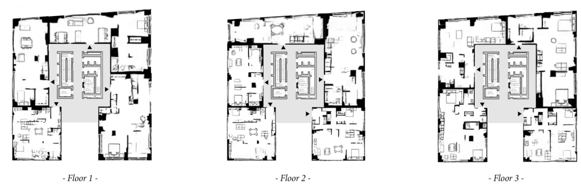
Each model is only in charge of translating a color patch added onto the plan, into a properly drawn piece furniture. Furniture types are encoded using a color code. We display here on the left the results of each model.



Generation Pipeline, Multiple Units | Source: Author



Model II Results | Source: Author



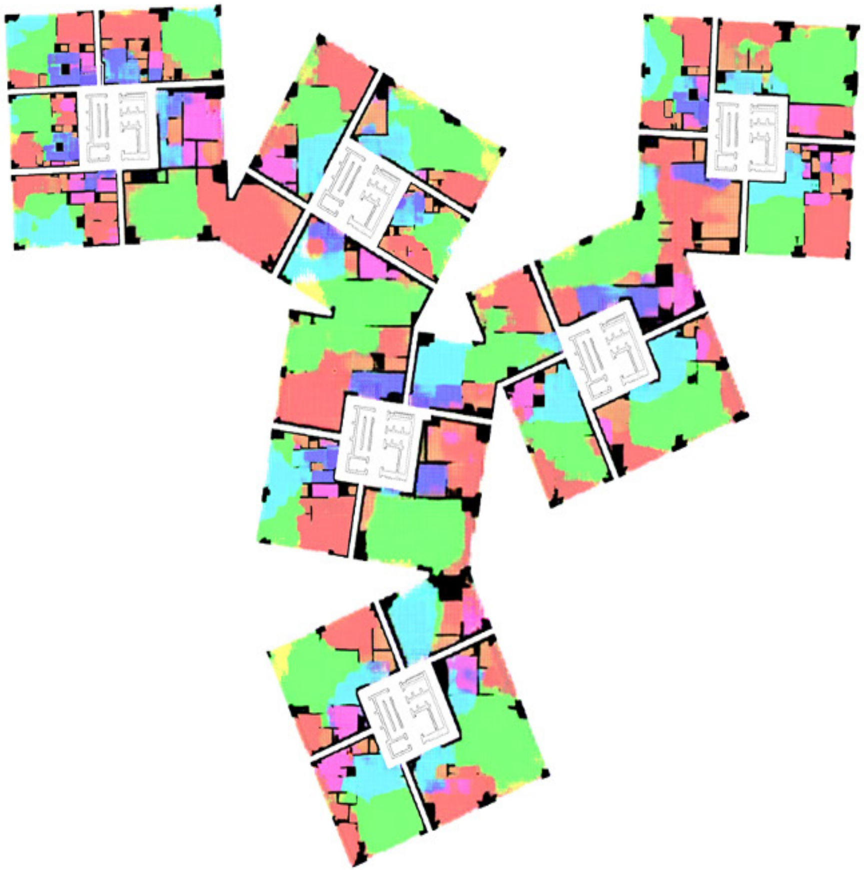
Model III Results | Source: Author

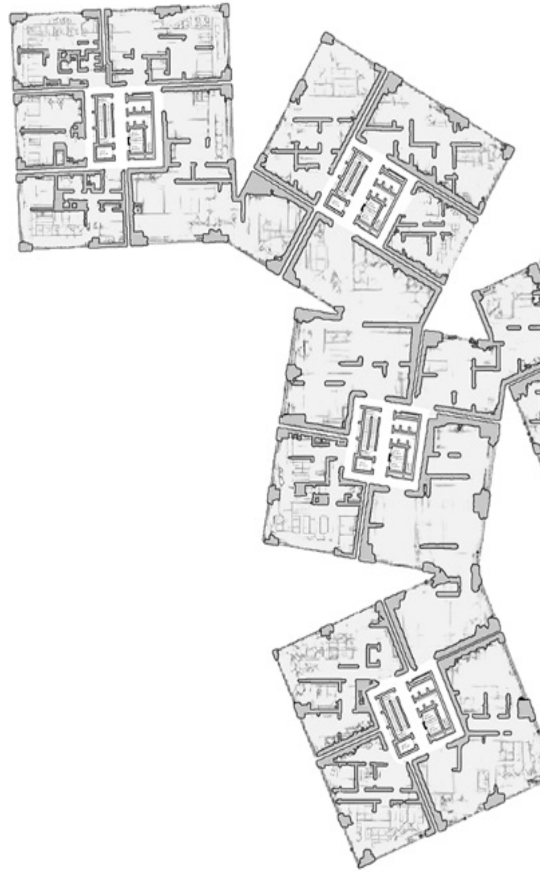
Going Further

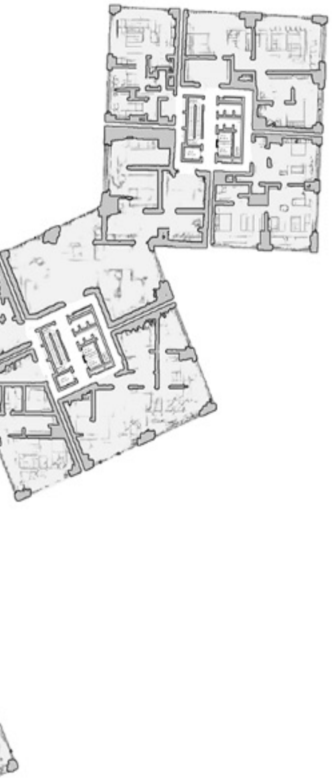
If generating standard apartments can be achieved using our technic, pushing the boundaries of our models is the natural next step. GANs can, in fact, offer quite remarkable flexibility to solve seemingly highly constrained problems. In the case of floor plans layout, as the footprint changes in dimension and shape, partitioning and furnishing the space by hand can be a challenging process. Our models prove here to be quite "smart" in their ability to adapt to changing constraints.

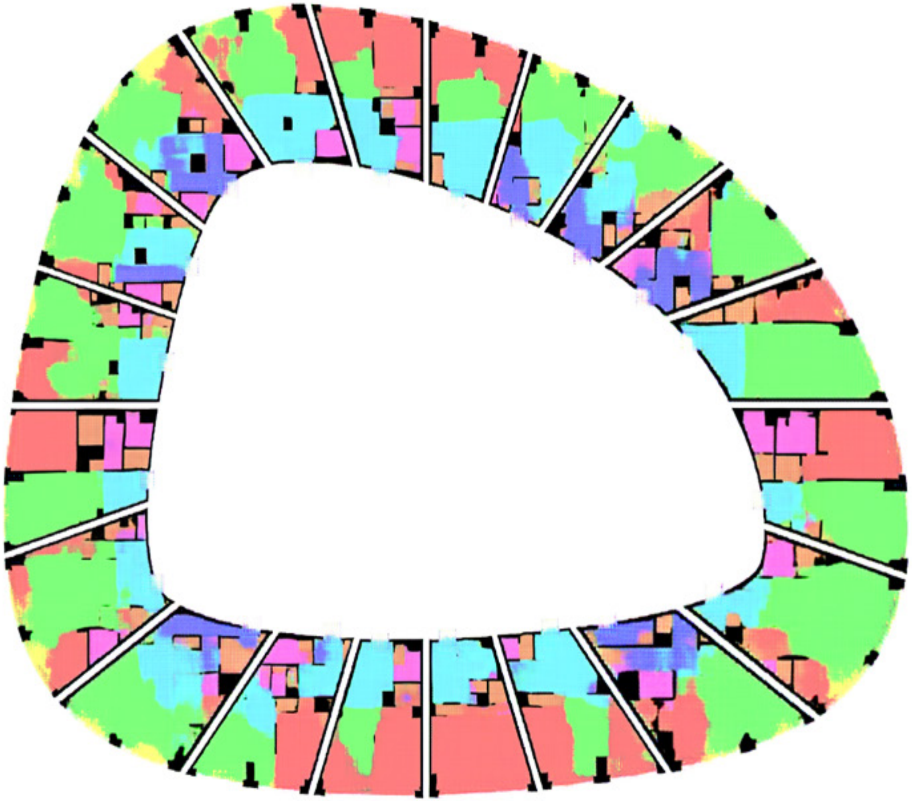
Our ability to control the units' entrance door and windows position, coupled with the flexibility of our models allows us to tackle space planning at a larger scale, beyond the logic of a single unit. By adding some simple algorithms to our generation pipeline (here on the right), we enable users to scale their work to the design of entire buildings, containing multiple apartment units.

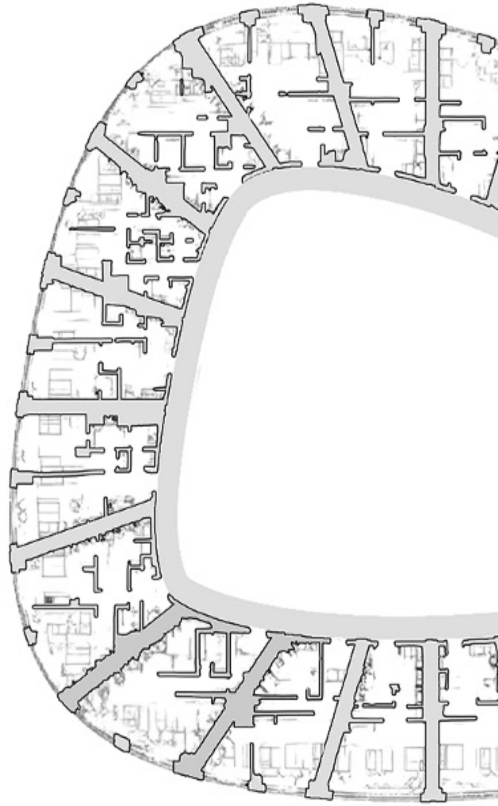
In the next pages, we display the results of our improved generation methodology, as it scales to entire buildings, and neighborhoods.

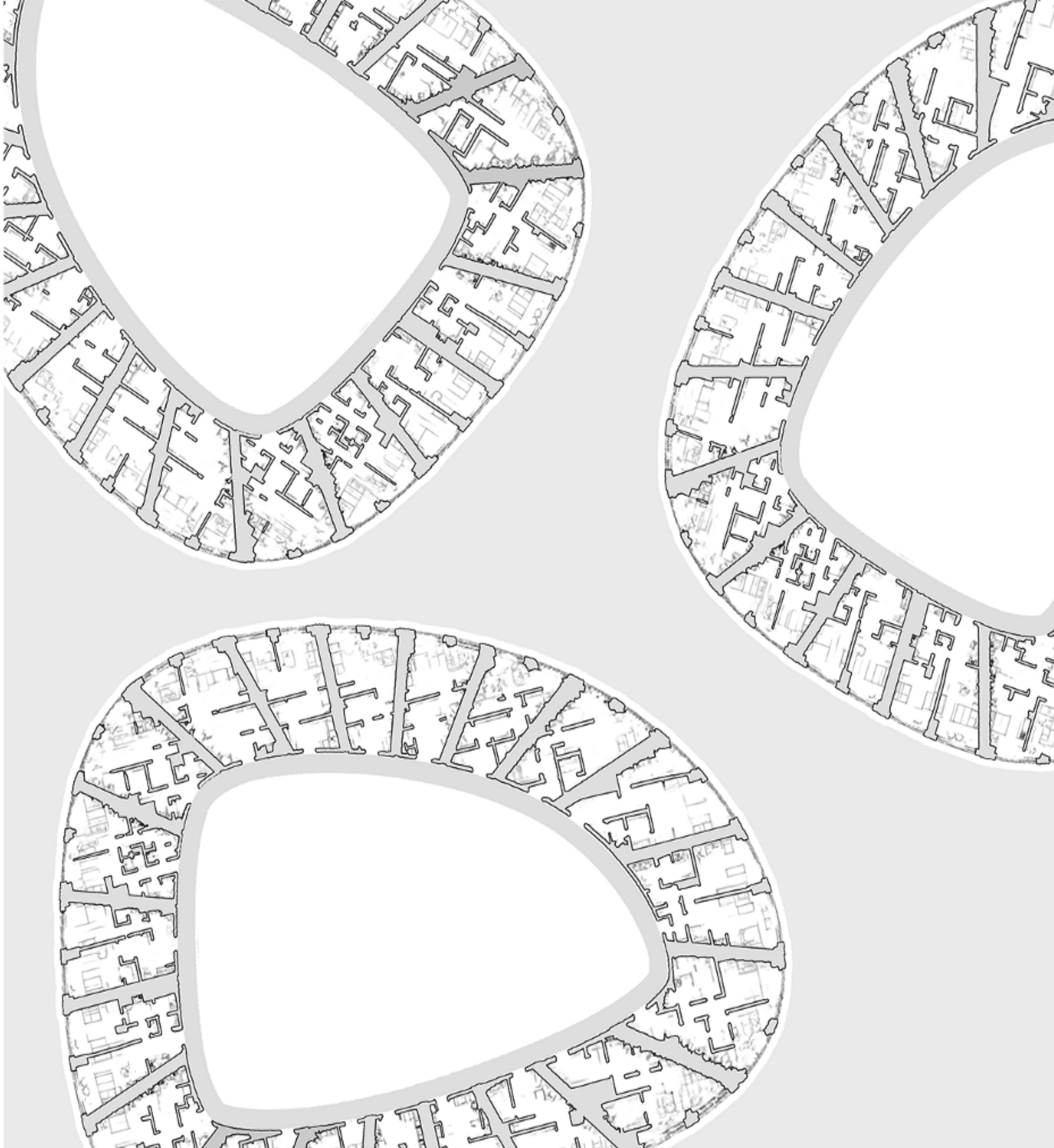


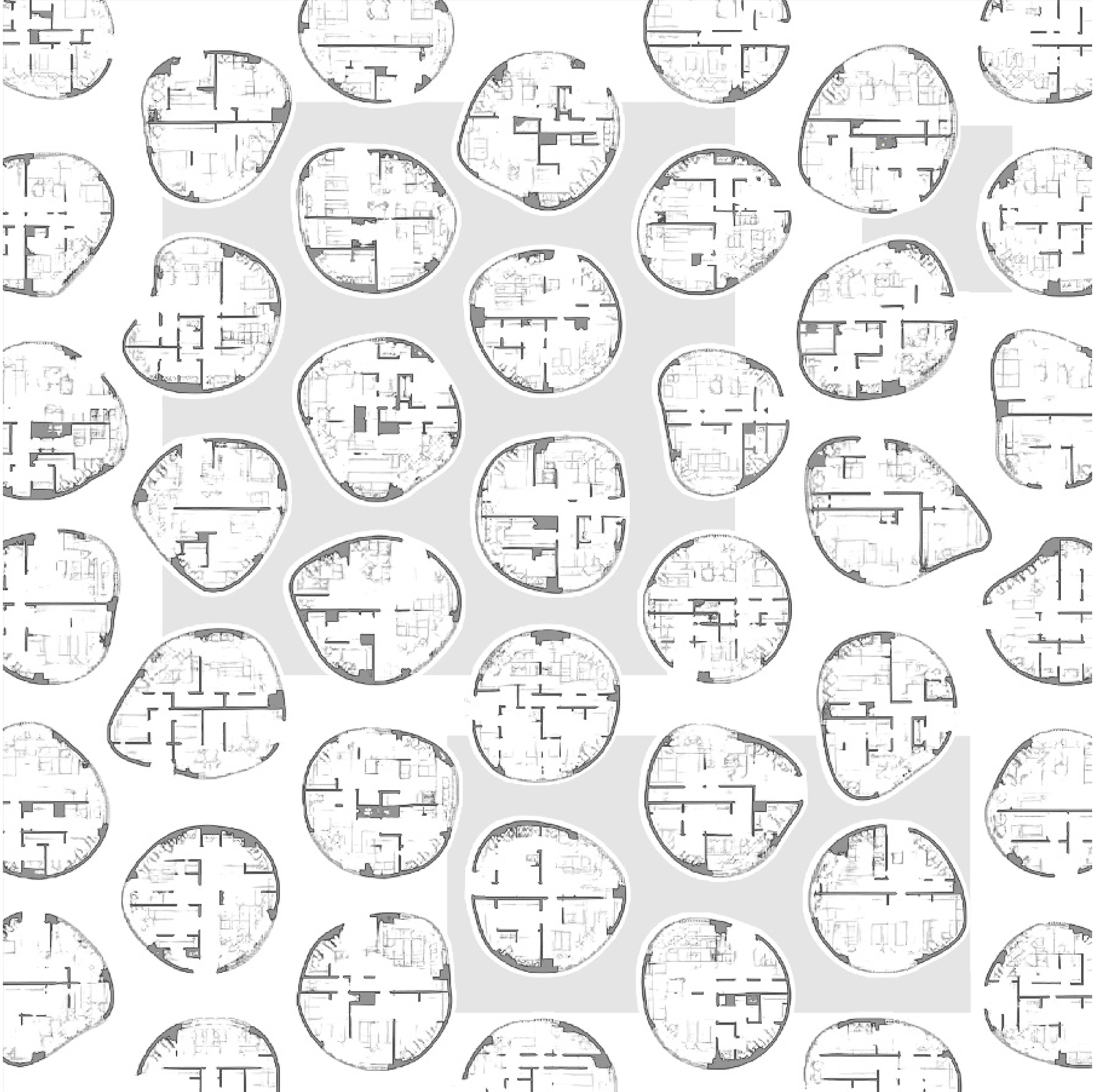


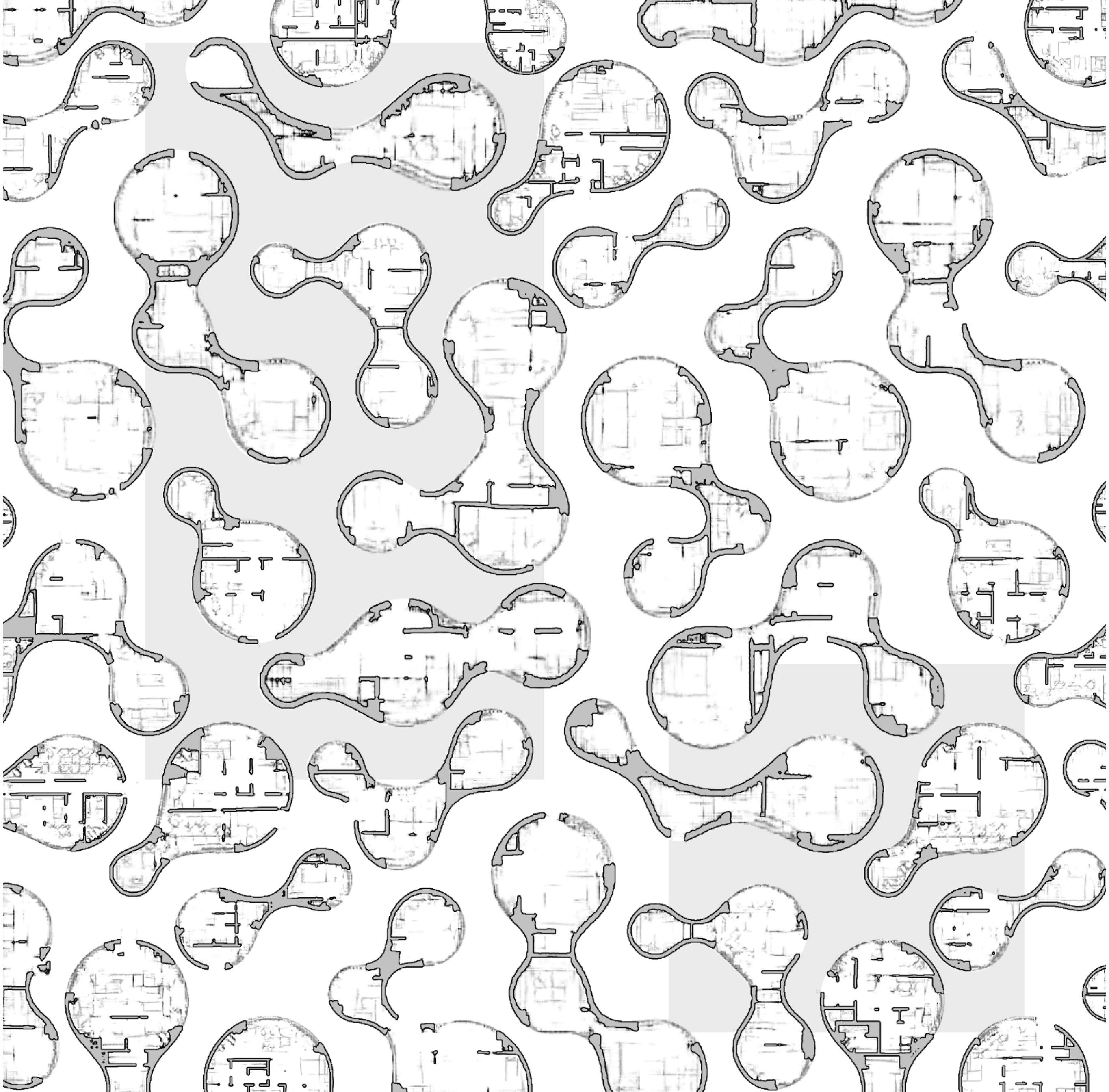


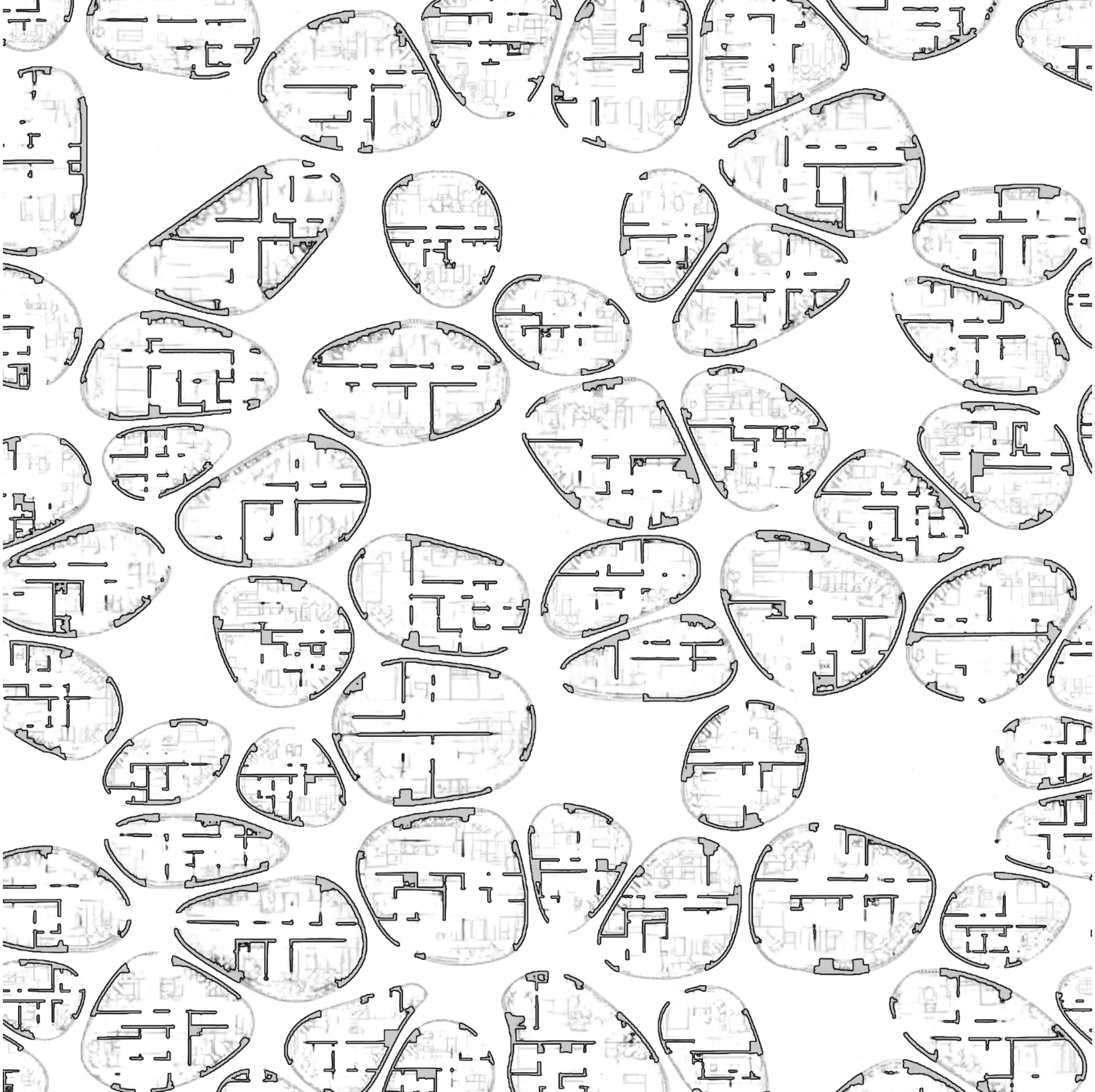












61

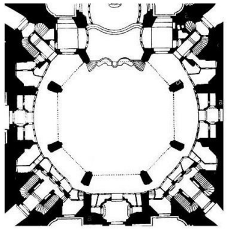
Organization
Takeaways

Beyond the strict development of a generation pipeline, we lay down in this chapter certain intuitions and concepts that will shape the rest of this thesis.

First, we are directionally convinced that our ability to design the right pipeline will condition AI's success as a new architectural toolset. Our choice for the "Grey Boxing" approach, as introduced by Prof. Andrew Witt in Log, will likely secure the best potential results. This method contrasts with the "black box" approach, that only allows users to input information upfront, and to get finished design options at the end of the process, without any control over the successive generation steps. To the contrary, by breaking out our pipeline into discrete steps, "Grey Boxing" permits the user to intervene all along the way. His tight control over the machine is his ultimate guarantee of the design process quality.

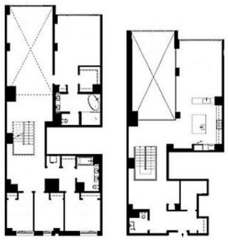
Then, we technically believe that the sequential nature of the application will facilitate its manageability and foster its development. The ability to intervene throughout the generating process is a fundamental dimension: as each step of the pipeline represents a distinct portion of architectural expertise, each model can be trained independently, opening the way to significant improvements and experimentation in the near future. Indeed, improving this entire pipeline end-to-end could be a long and cumbersome task, while amending it step by step remains a manageable process, within the reach of most architects and engineers in the industry.

Finally, we hope our framework will help address the endless breadth and complexity of the models to be trained and those used in any generation pipeline. Tackling parcels-footprint-room split-etc., as we do is one possible approach among, we believe, a large set of options. To encapsulate the necessary steps of space planning, the key is more the principle than the method. And with the growing availability of architectural data, we encourage further work and open-minded experimentation.

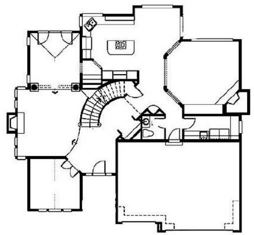


Baroque

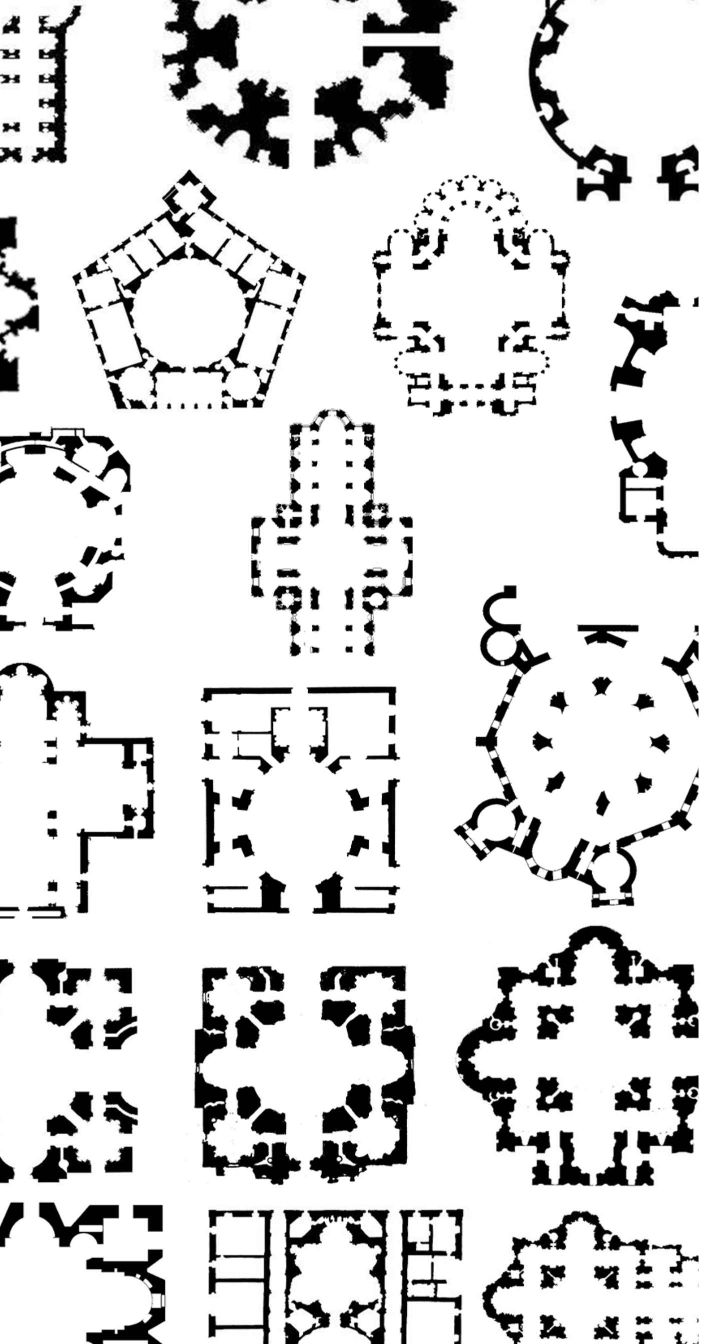
Manhattan Unit



Row-House



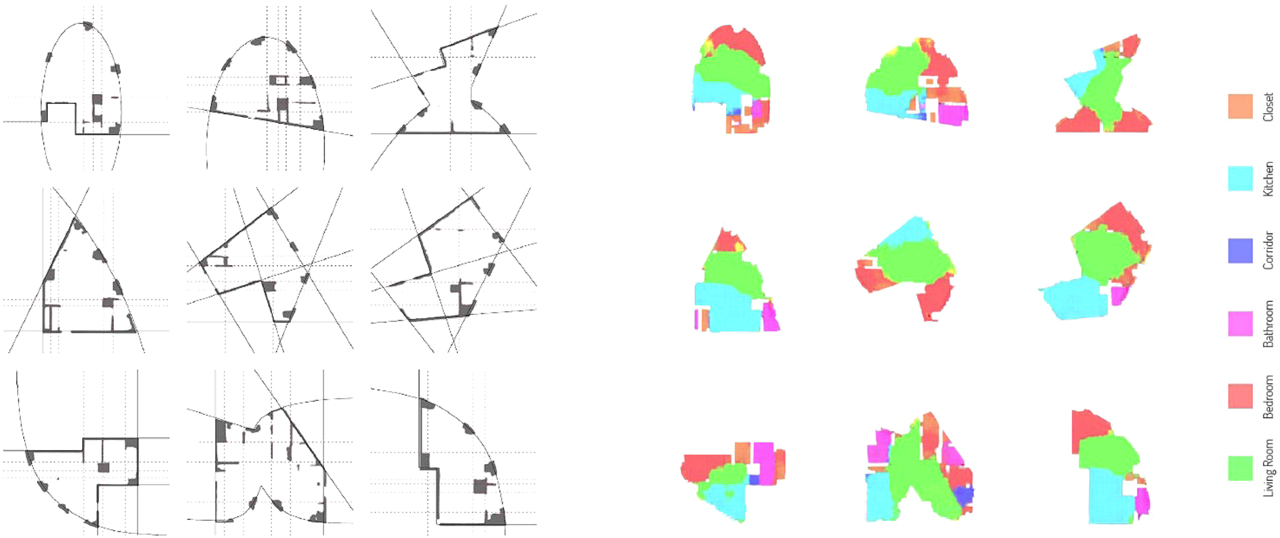
Suburban Victorian



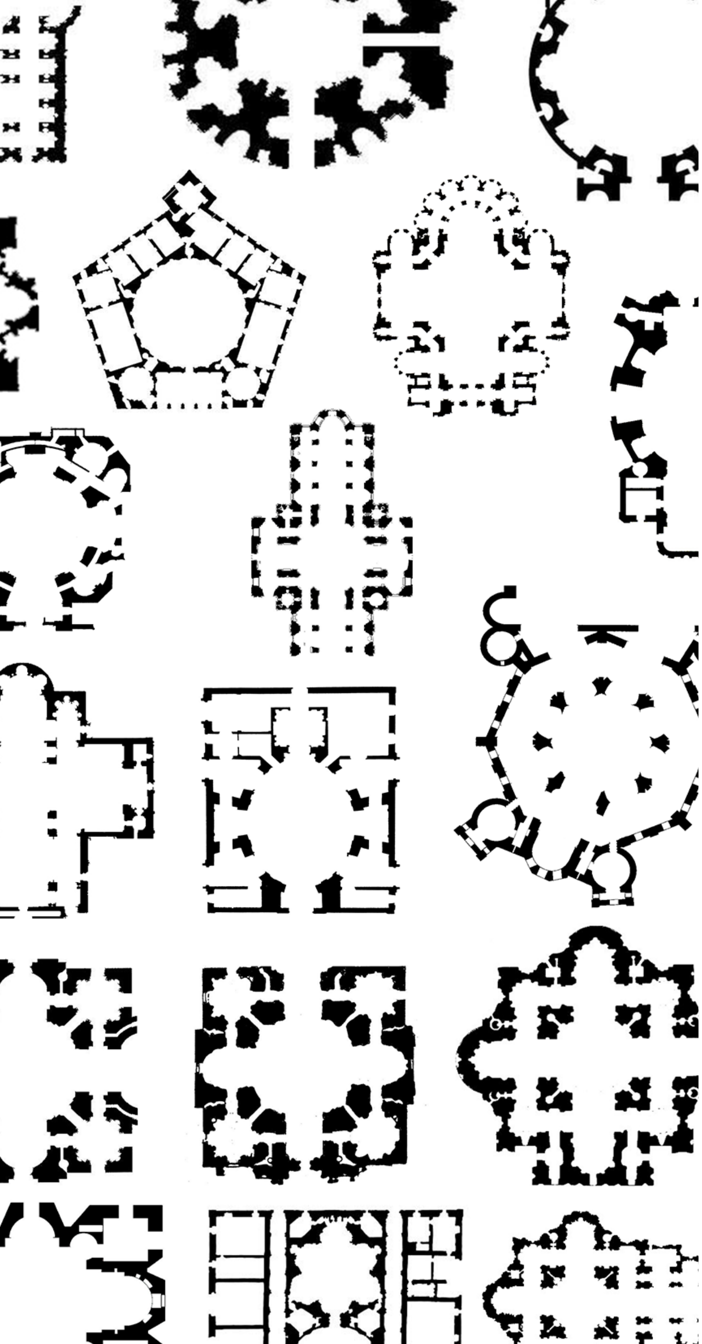
C. Style

As we refine our ability to generate floorplans, we raise the question of the bias intrinsic to our models and offer here to extend our study beyond the simple imperative of organization. We investigate architectural style learning, by training and tuning an array of models on specific styles: Baroque, Row House, Victorian Suburban House, & Manhattan Unit.

Beyond the simple gimmick of each style, our study reveals the deeper meaning of stylistic: more than its mere cultural significance, style carries a fundamental set of functional rules that defines a clear mechanic of space and controls the internal organization of the plan. In his part, we will try to evidence the profound impact of architectural style on the composition of floorplans.



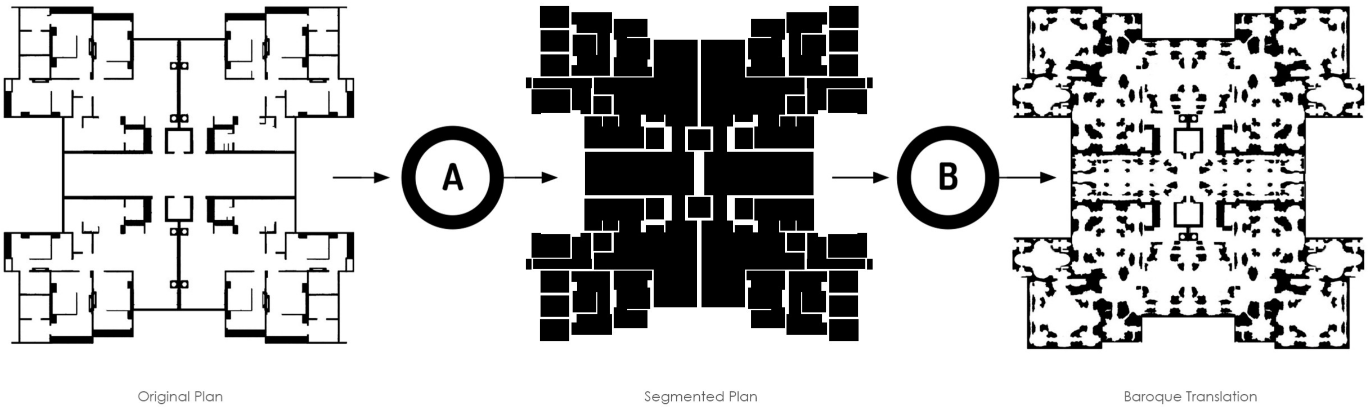
Bias & Style Among Generated Results | Source: Author



Bias & the Emergence of Style

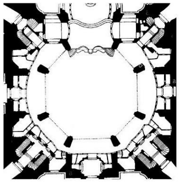
Taking into consideration a batch of generated units, using our initial pipeline, we start to notice some amount of intrinsic bias to our model: the internal wall structure is consistently laid out as an orthogonal system of partitions, disregarding the potential orientation of the units' facades (see image here below). At the same time, the program layout is also consistently set up such that "serving" spaces -bathroom, toilets, closets, kitchen- are packed at the back of the plot, while the odd geometry of the facade gets absorbed by over-dimensioned living rooms and bedrooms.

These characteristics can, in fact, be found all across our initial training set. We understand here this reality as the literal translation of a concept central to the architectural discipline: style.



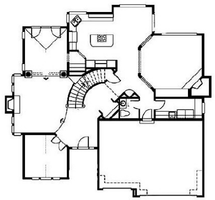
Style Transfer Pipeline | Source: Author

Four Styles

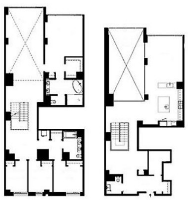


Baroque

Manhattan Unit



Suburban Victorian



Row-House

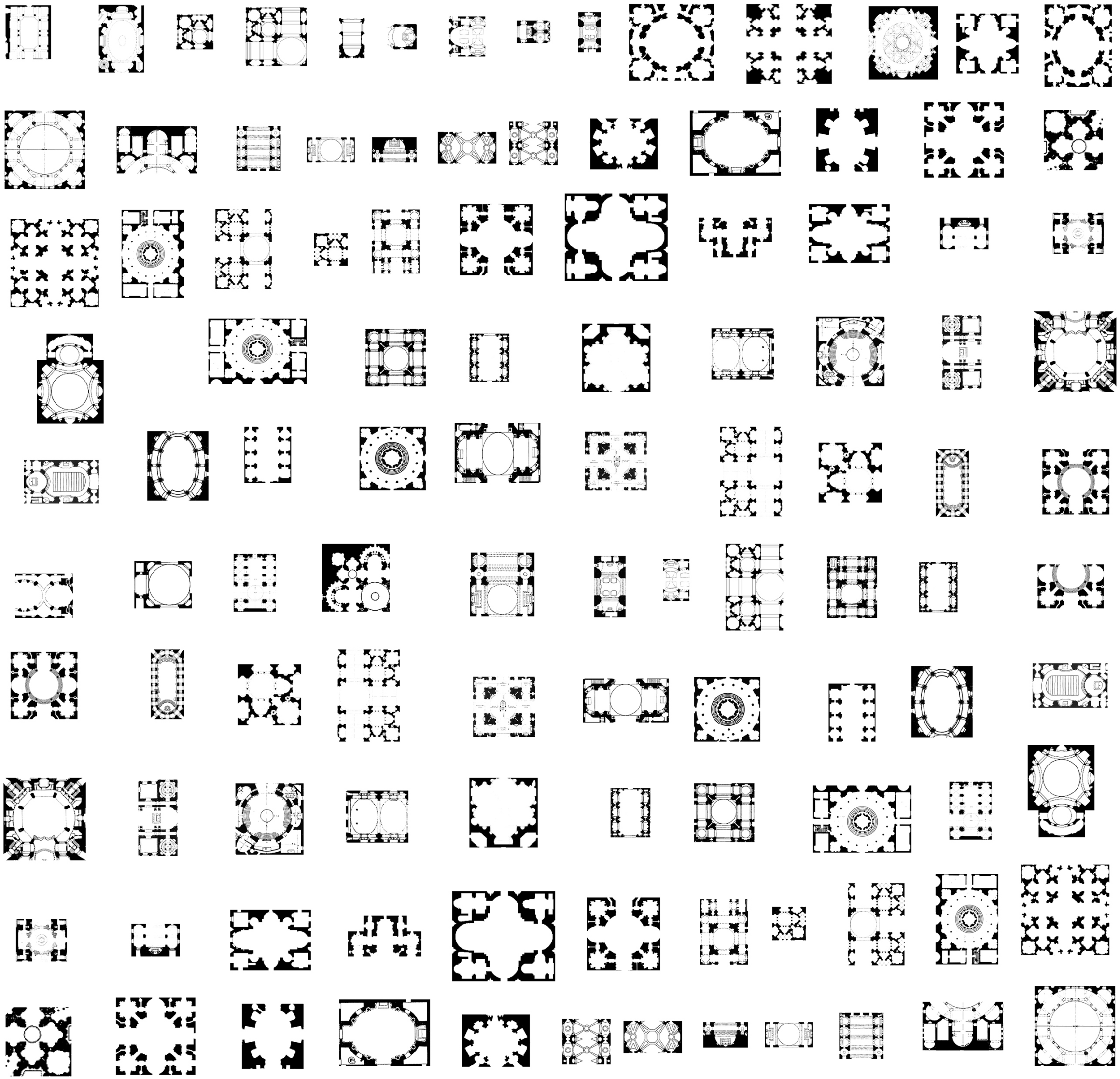
Instead of preventing this bias, striving to create a generic or objective plan generator, we will rather embrace it and study its presence to eventually use it to our advantage.

To that end, we choose to broaden our investigation and extend it to architectural style learning. We create a new pipeline, enabling the conversion of floorplans, from one style to another; here on the left, from Modern to Baroque.

This example reveals all the more the deeper meaning of architectural style: we notice that the translation through model A & B is not simply a new make-over of the existing figure wall, but rather a profound remodeling of internal structures and spatial organizations.

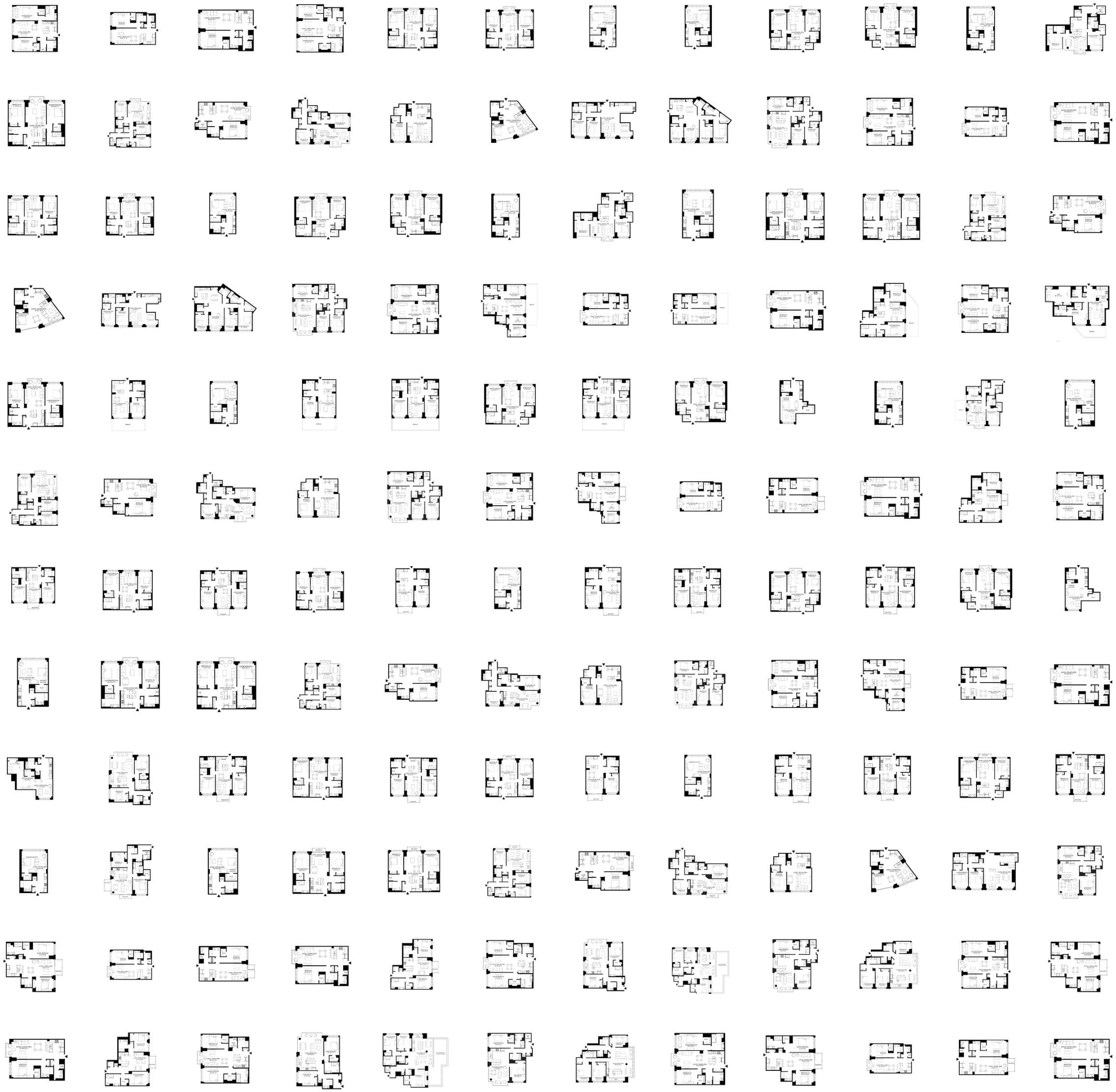
In fact, we evidence here what Farshid Moussavi coined down as the “Function of Style” in her book. Each style, beyond its cultural significance, handles space differently and reacts specifically to similar constraints.

To investigate architectural style learning, we have trained and tuned an array of models on specific styles—Baroque, Row House, Victorian Suburban House, & Manhattan Unit—able to emulate each particular architectural style. The results are displayed in the following pages.



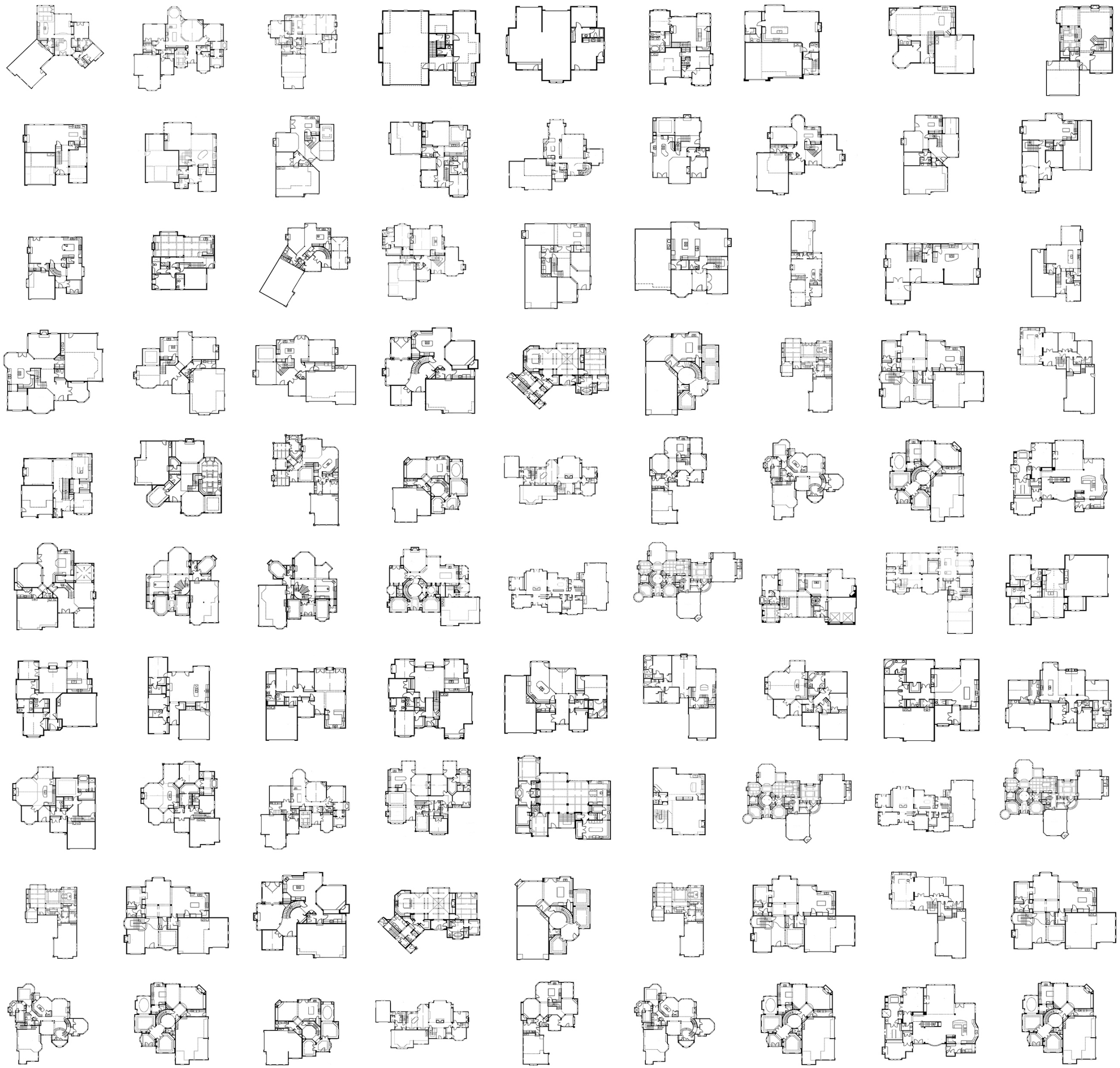
Baroque Style
Training Set

Generated
Units



Manhattan Style
Training Set

Generated Units

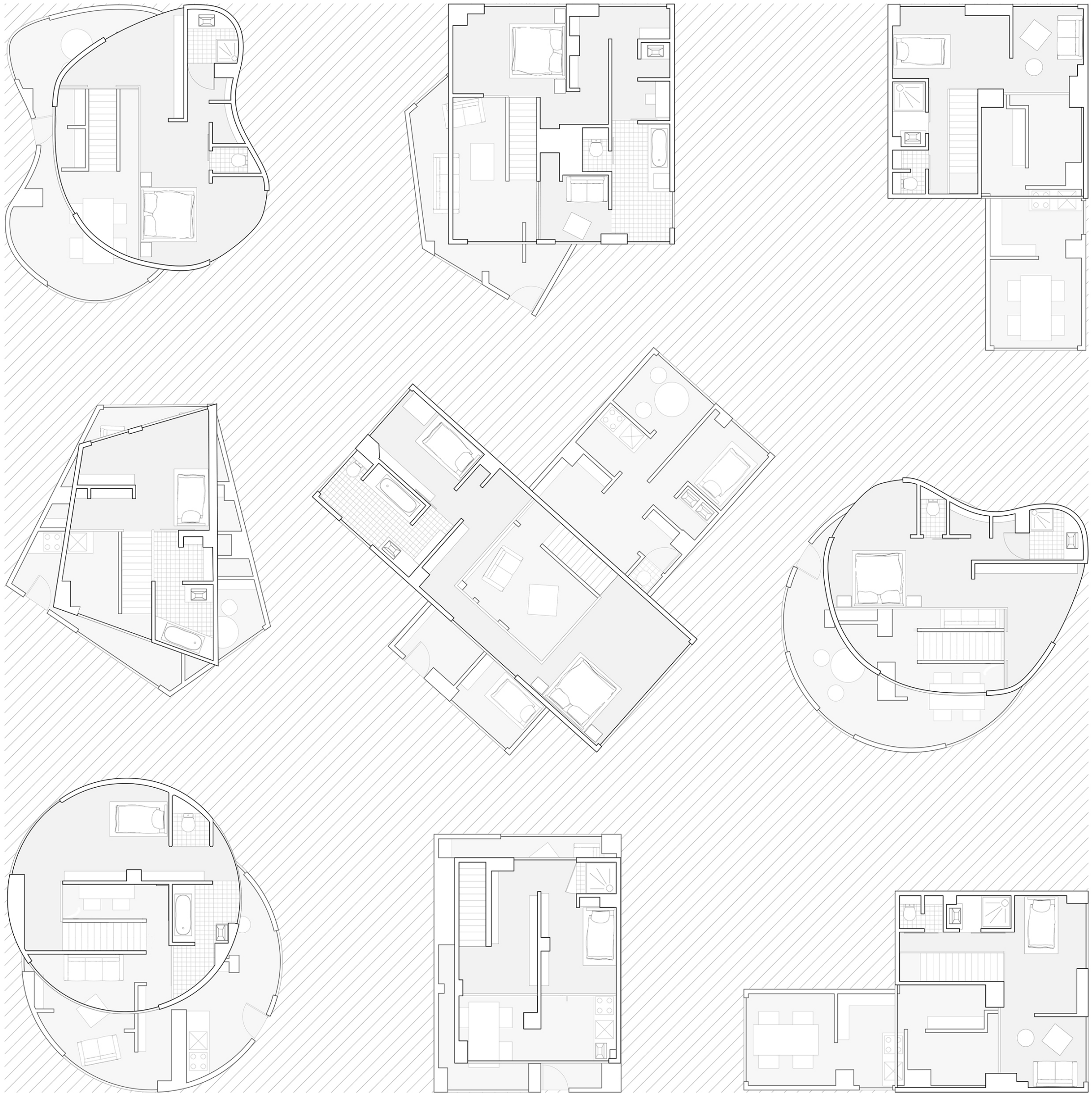


Victorian Style
Training Set

Generated
Units

Row-House Style
Training Set

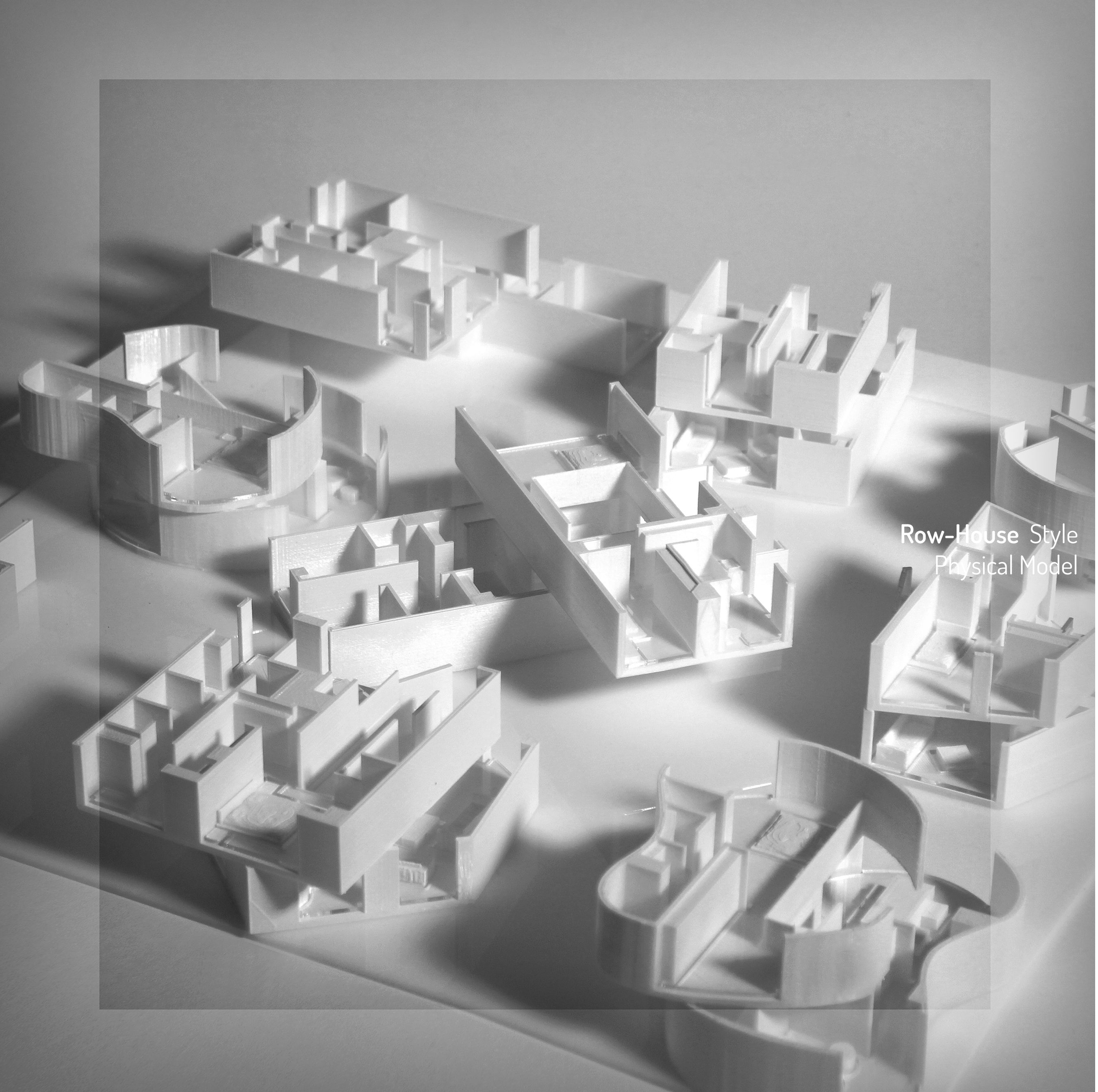
Generated
Units



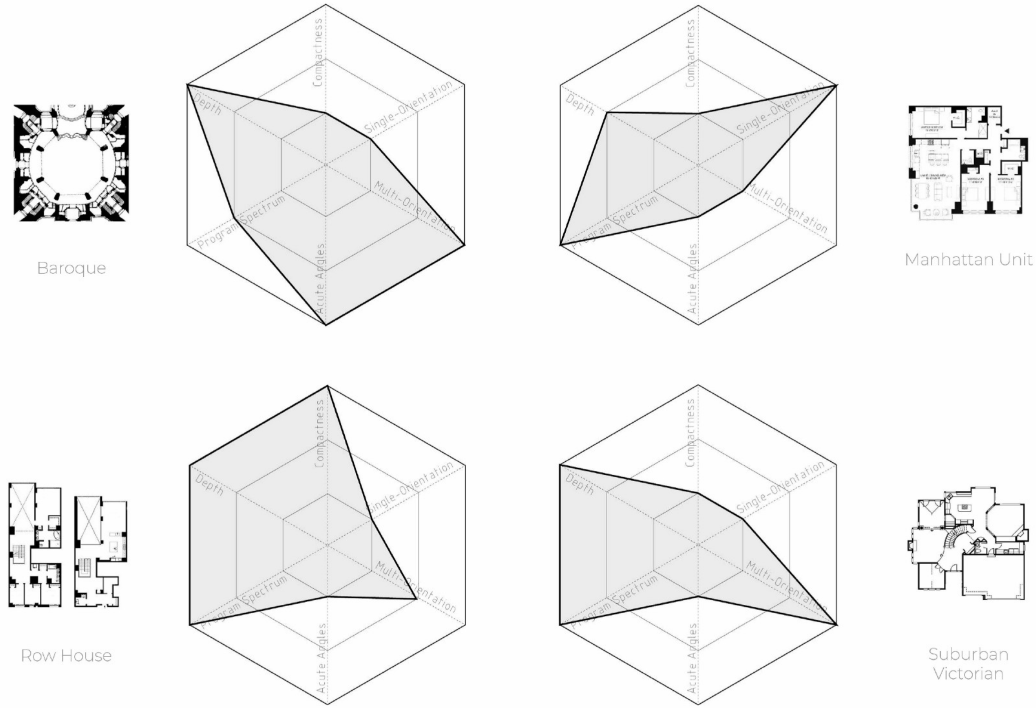
Baroque Style
Physical Model

Manhattan Style
Physical Model

Victorian Style
Physical Model



Row-House Style
Physical Model



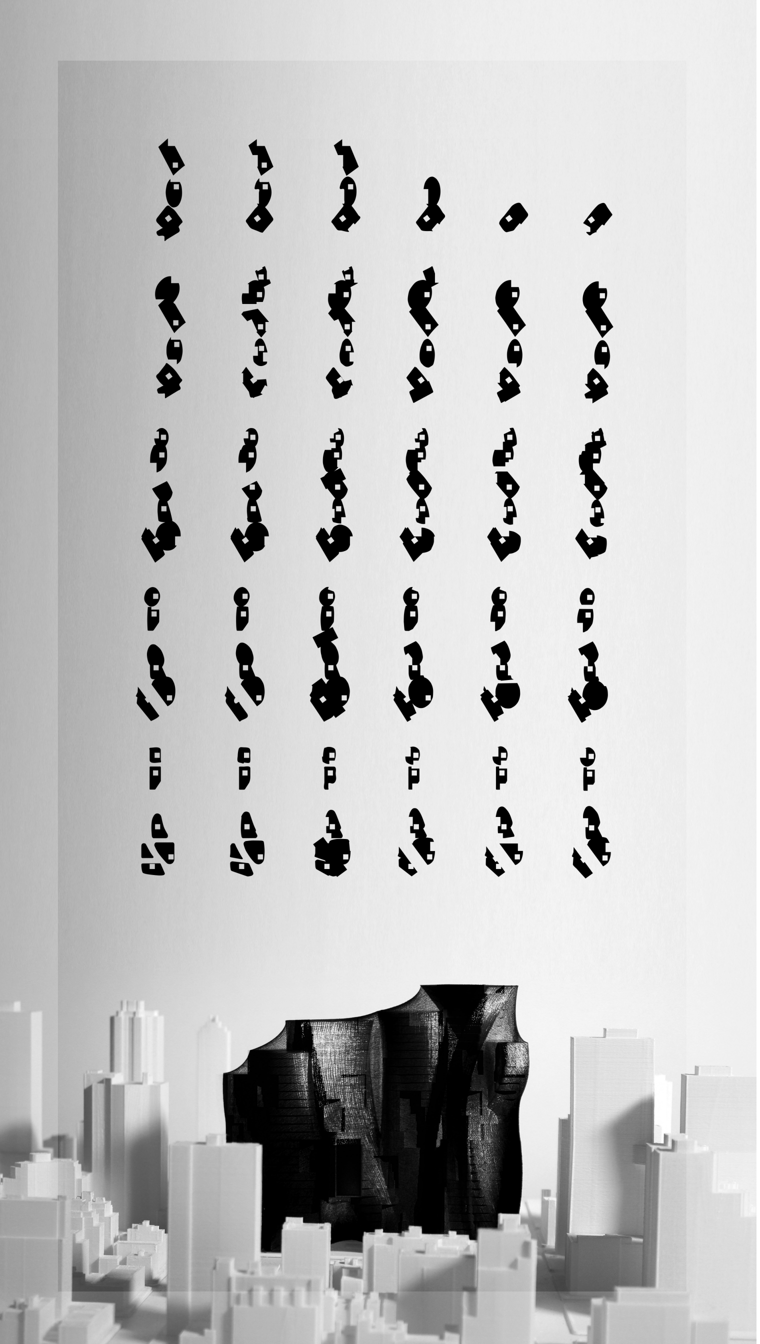
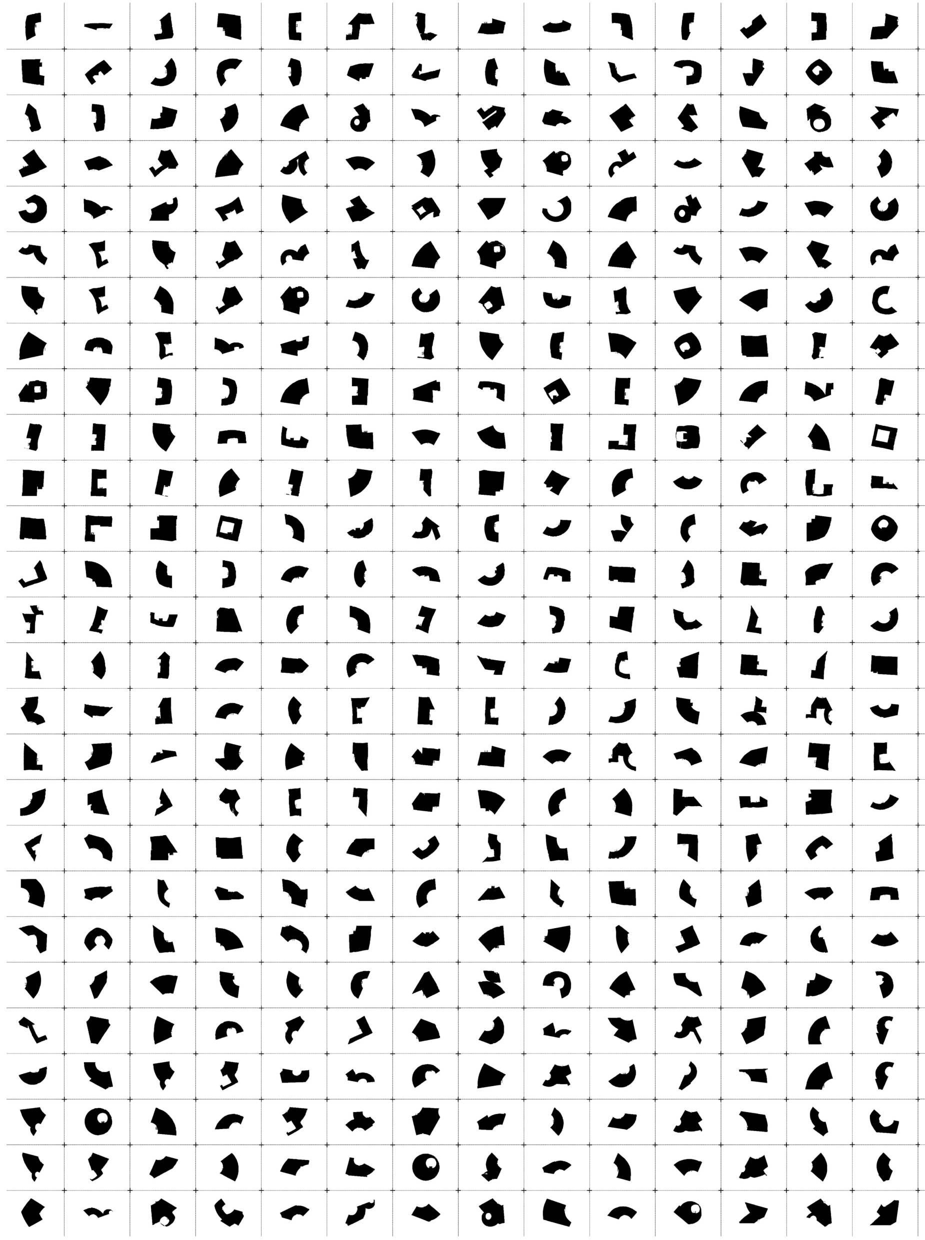
Characteristic Graphs | Source: Author

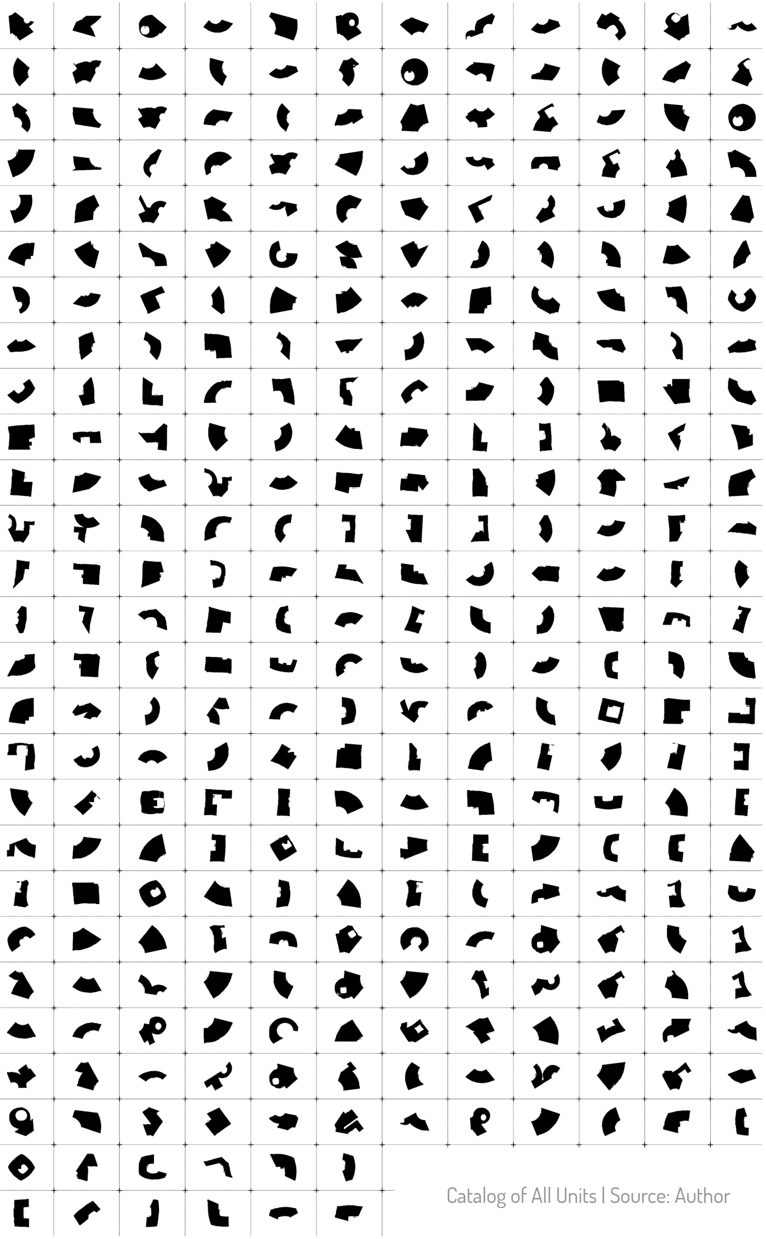
Among the generated units we can identify some clear patterns within each style. This “behavior” proper to each model is a direct translation of each style’s mechanic. More than applying a simple texture across each apartment unit, each model has captured a set of characteristics and rules.

To move beyond this simple observation, we offer to map out each style’s abilities. To each model corresponds a set of strengths & weaknesses, and coining them down will allow us to truly assess the functional reality of each style. In addition, our hope is to expand our understanding of our models’ abilities to allow us later to use each one purposely, given a new set of constraints, and functional requirements.

In clear we propose a six-axis graph, reflecting a given model’s ability to handle six specific types of condition: Depth, Compactness, Single-Orientation or Multi-Orientation (number of facades), Acute Angle (sharp geometry of the boundary), Program Spectrum (breadth of the program).

After thoroughly testing our four models, we propose the graphs here on the left.

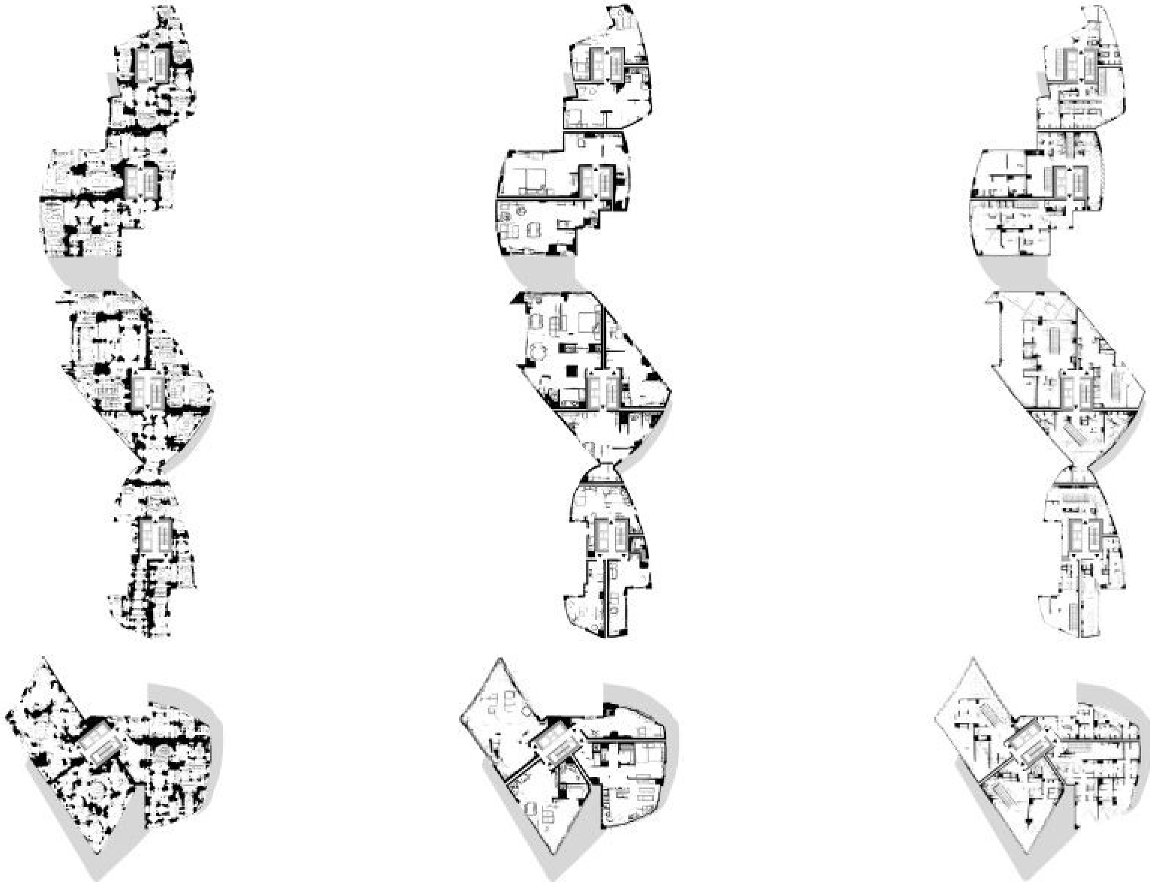





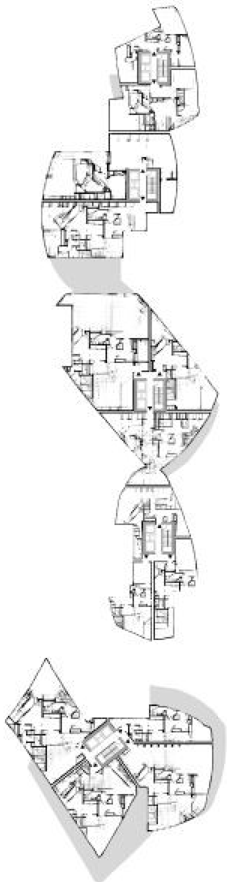
Catalog of All Units | Source: Author

Application

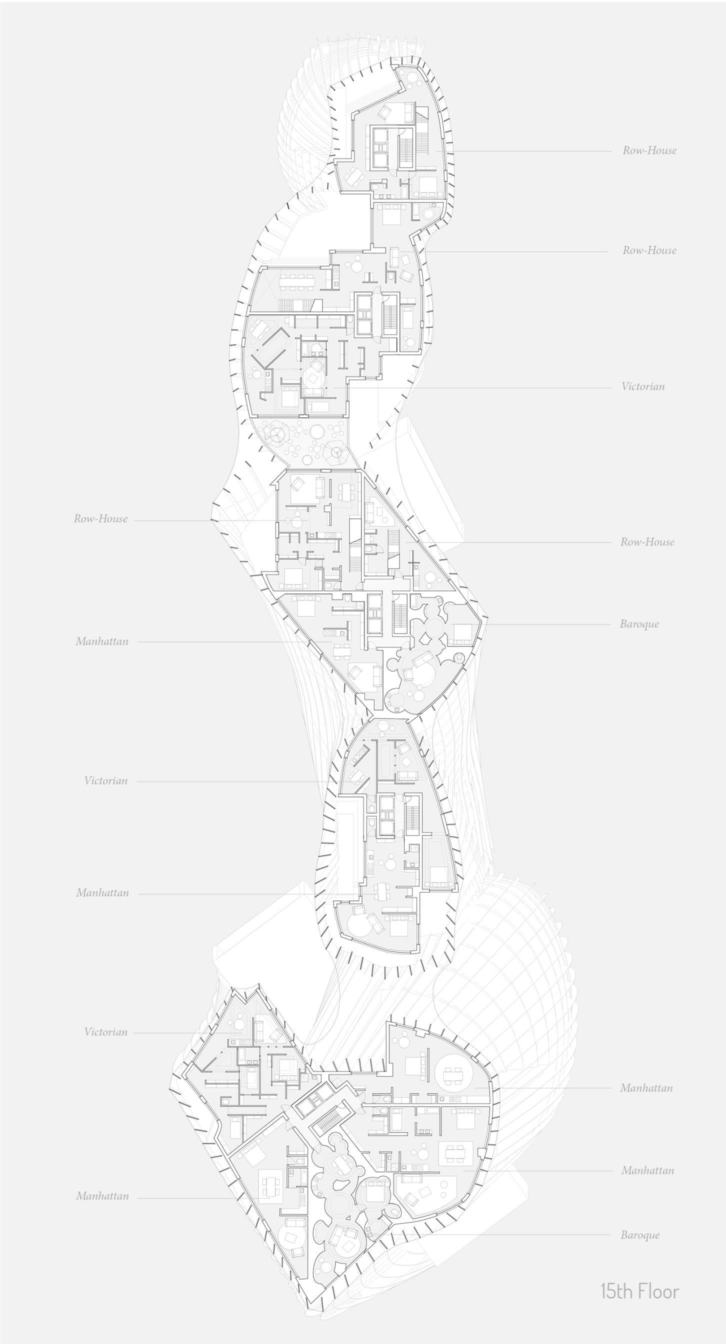
Finally, we have brought all these intuitions together in a final architectural project: a large-scale housing development located in Manhattan's Lower East Side. The complex geometry of the parcel forces a certain amount of complexity on our design. As a result of our massing (on the far left), we obtain a catalog of 380 one-of-kind apartment units (here on the left).

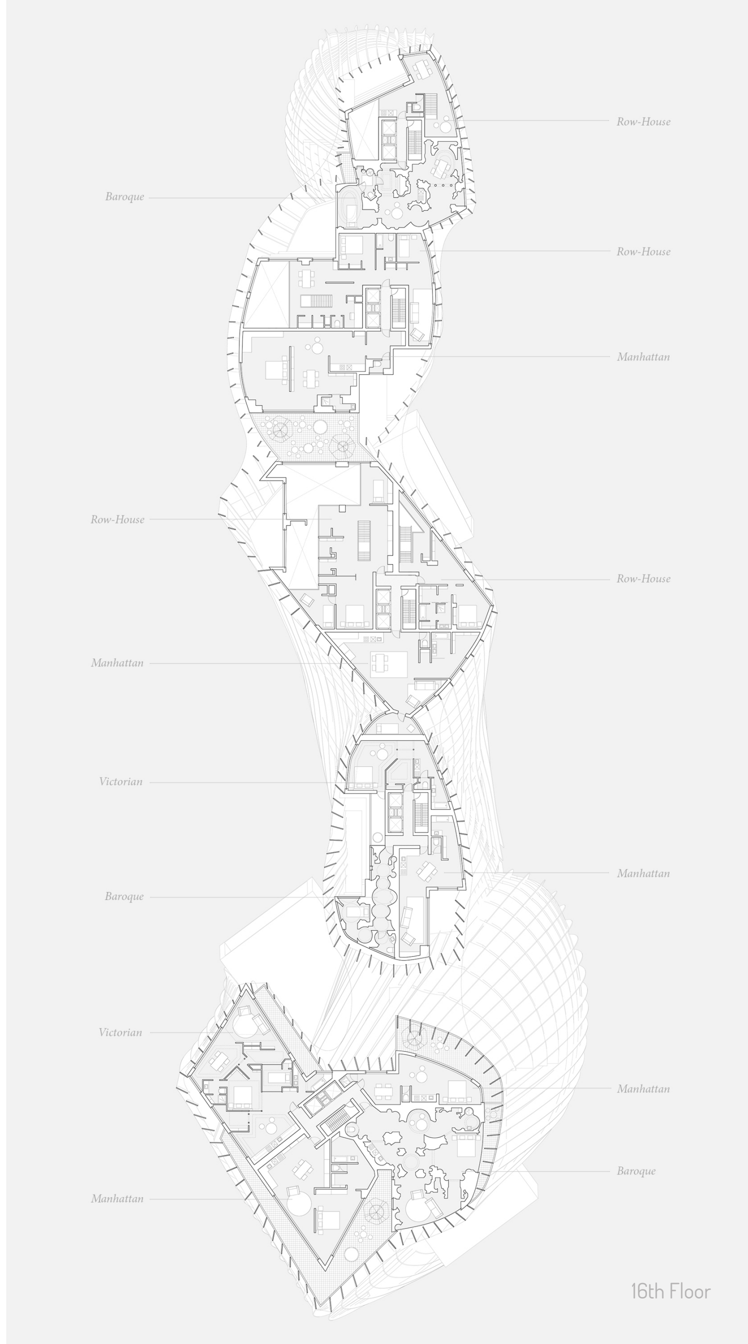


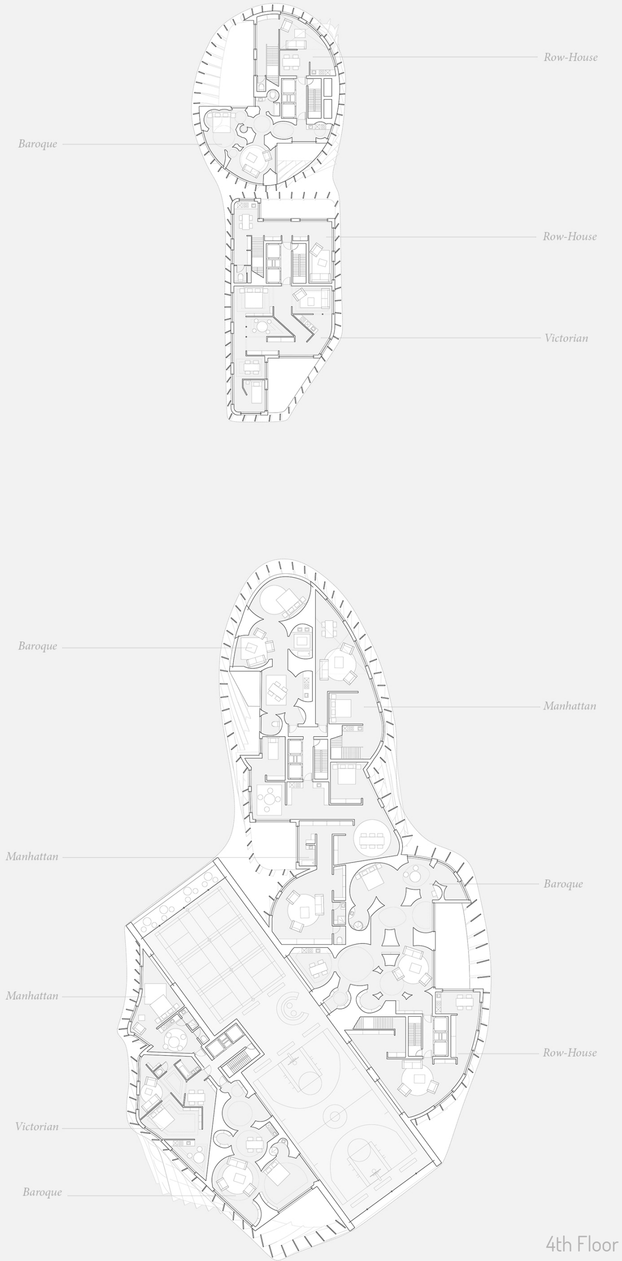
15th Floor Processed Under Each Style: Baroque (Far-Left), Manhattan (Center-Left), Row-House (Center-Right), Victorian (Far-Right) | Source: Author



We first attempt to process an entire floor, each time using a different style. The results displayed here n the left reveal once again the necessity of using styles cautiously, respectful of the constraints and the specificity of each context. If certain units are successfully laid out using a certain style, others fail to find a proper internal organization. By alternating styles across our catalog of apartment units, we hope to find an appropriate answer to each specific spatial condition.

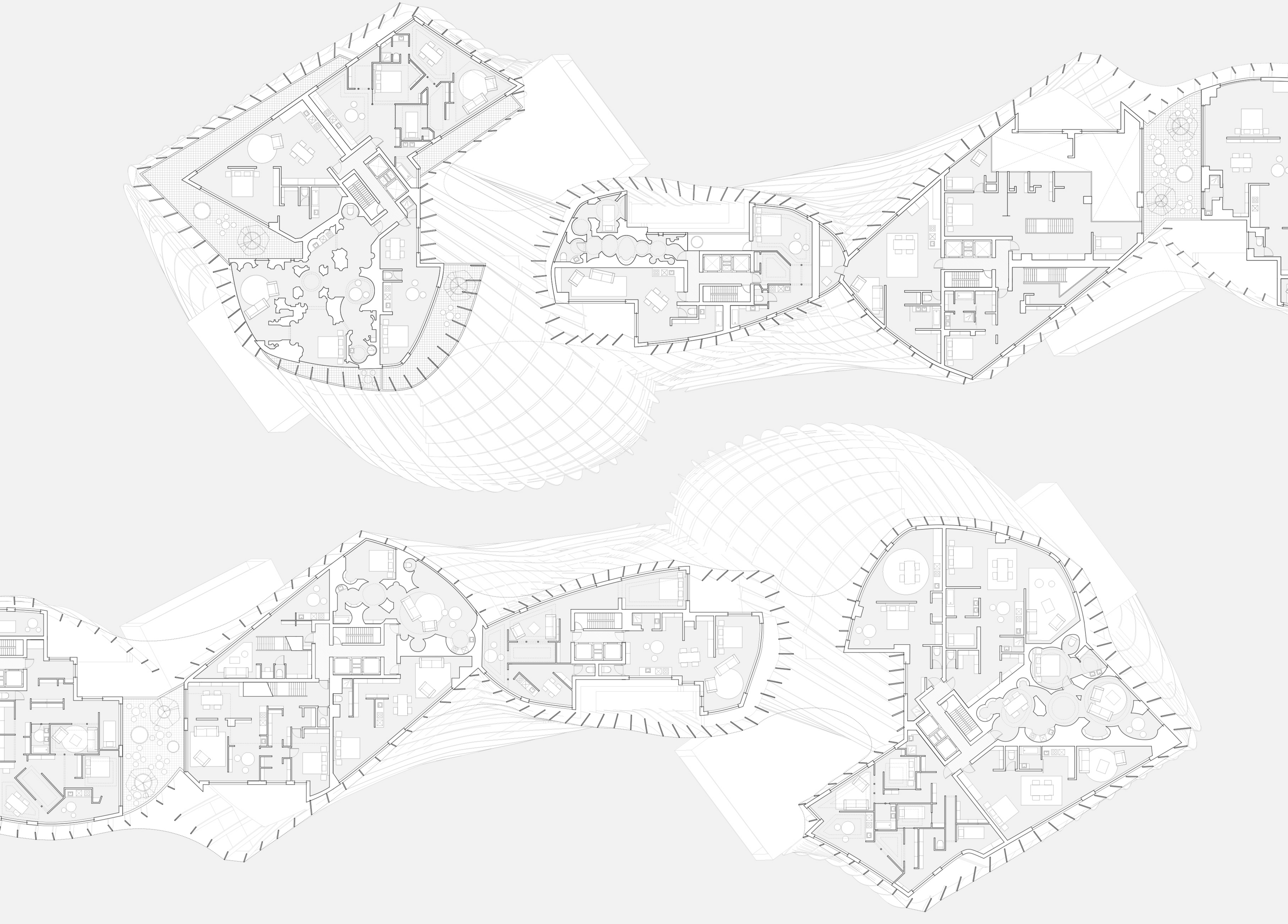

Row-House
Row-House
Victorian
Row-House
Row-House
Manhattan
Baroque
Victorian
Manhattan
Victorian
Manhattan
Manhattan
Manhattan
Baroque
15th Floor


Row-House
Baroque
Row-House
Manhattan
Row-House
Row-House
Manhattan
Victorian
Manhattan
Baroque
Victorian
Manhattan
Manhattan
Baroque
16th Floor



Knowing each GAN-model's strengths and weaknesses, each style's potential and short-comings, we now process every apartment using the most suited model. Each floor is turned into a patchwork of styles. Our goal becomes then to compose our "mosaic" picking for each tile -each unit- the most reasonable model that will best handle the constraints. Out of this selection process, we have isolated some resulting options, displayed here below.

We then narrow down our exploration, precise the selection of units and styles across all floorplates, to finally assemble our final design. Here on the left are three typical floors.



Assumptions & Limitations

We have in fact turned styles into functional tools, able to address specific conditions all across our development.

However, as the above plans suggest, we set aside certain constraints, and make clear assumptions. We would like to clarify these.

The Structure

The structure is left to the structural cores, and the tensions cables running along the facade. The plan is therefore uninterrupted by vertical loads and allows our algorithms to generate freely each unit's partitioning system.

Note: Looking back to our generation pipeline, a potential improvement would take the position of load-bearing walls & columns as inputs for Model I. In such a way, our pipeline would allow designers to control the building structural system. Alternatively, Model I could be unpacked into two successive models, one for laying out load-bearing elements, and one for adding partitioning walls.

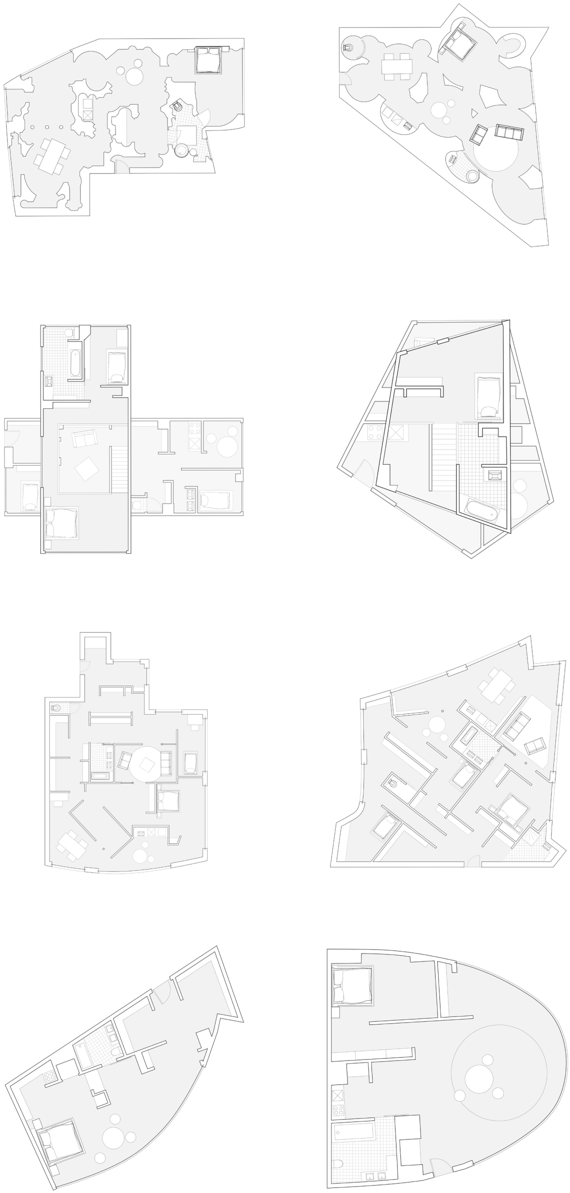
The Strict Imperative of Efficiency

Both the exuberance of each style and the level of freedom given to our models do not address a concern common in our discipline: space efficiency. However, our primary concern here is to maximize the expressiveness of each style, to let each model unfold its mechanic to showcase its "personality".

Note: To reconcile GANs with efficiency, we posit that their outputs constitute a tremendous initialization for standard optimization technics. Parametricism's typical pitfall was to setup a too-broad problem space, coupled with a random initialization. Optimizations run on these settings would often end up converging on locally minimal solutions. The intuition behind our GANs brings a whole new quality of initialization, that narrows down significantly the problem space while setting up a good-enough initial solution.

The Massing

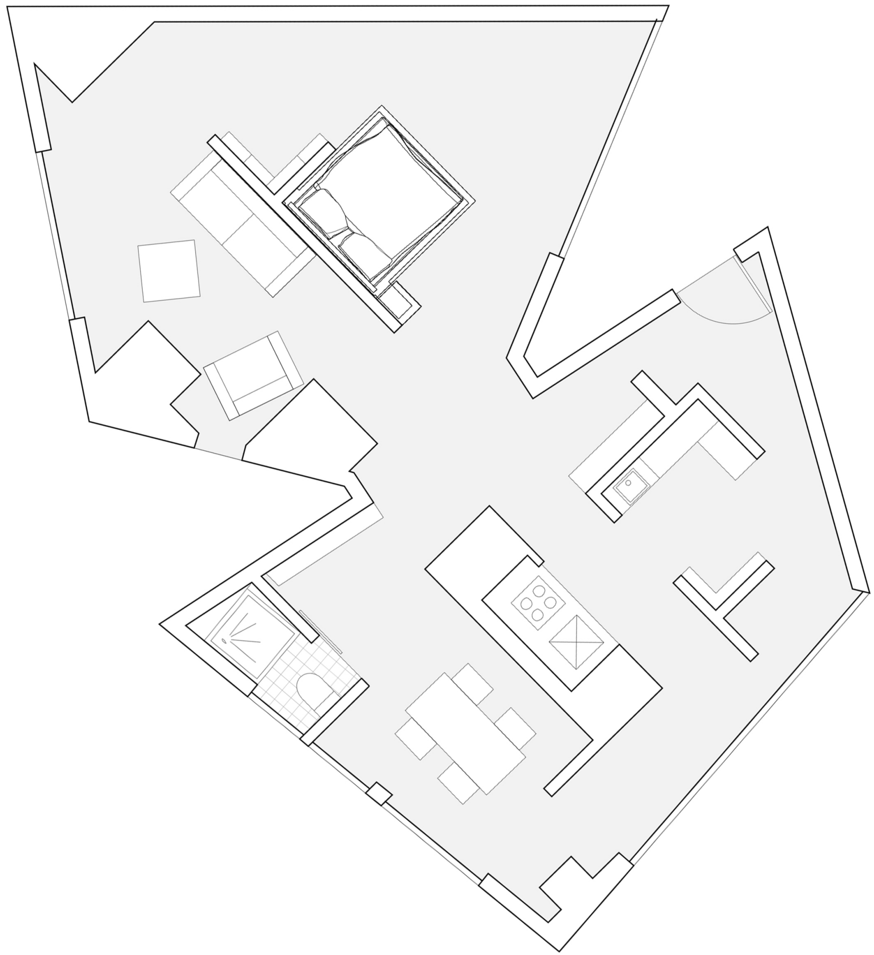
The notion of Massing refers to the outer-shape of our building. The irrational form of the above design is here meant as a trigger for complexity, creating an actual challenge for our models. To a more rationale massing would correspond more tamed & realistic apartment units designs. Our focus here remains to showcase an extreme case and test our models' limits.

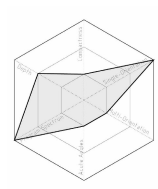


Catalog

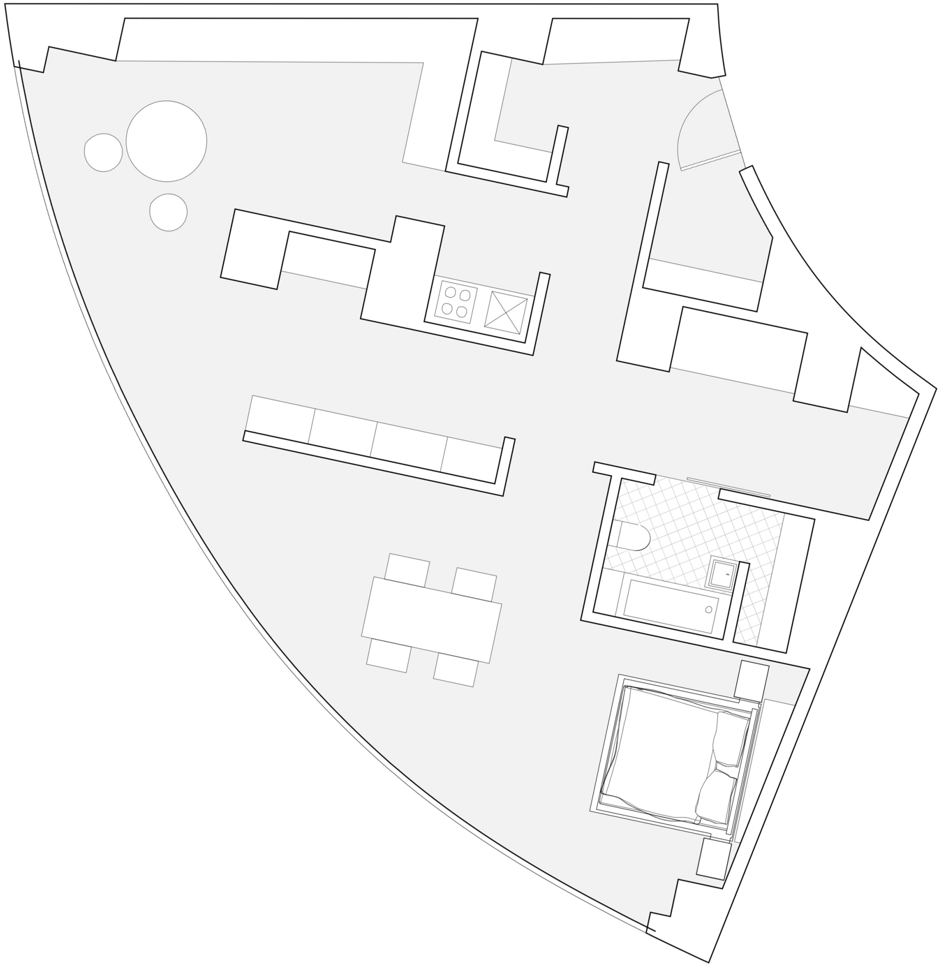
We turn finally to the catalog of units generated, found across our building (here on the left). The coherence and richness of the resulting designs are striking. Moreover, the "intelligence" or formal flexibility displayed in the generated apartments further evidences the validity of the approach: GAN-models can indeed encapsulate some amount of architectural expertise & stylistic that can be later used, depending on the set of constraints at play. The "personality" of each model described in Part II is clearly legible among each subset.

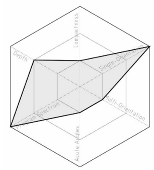
To conclude this section, we offer in the following pages a sequence of key shots, taken across our catalog. To the strict descriptive nature of the plans (left), we associate each time an image of the interior atmosphere (right), as a way to reconcile our process with the more experiential nature of Architecture.



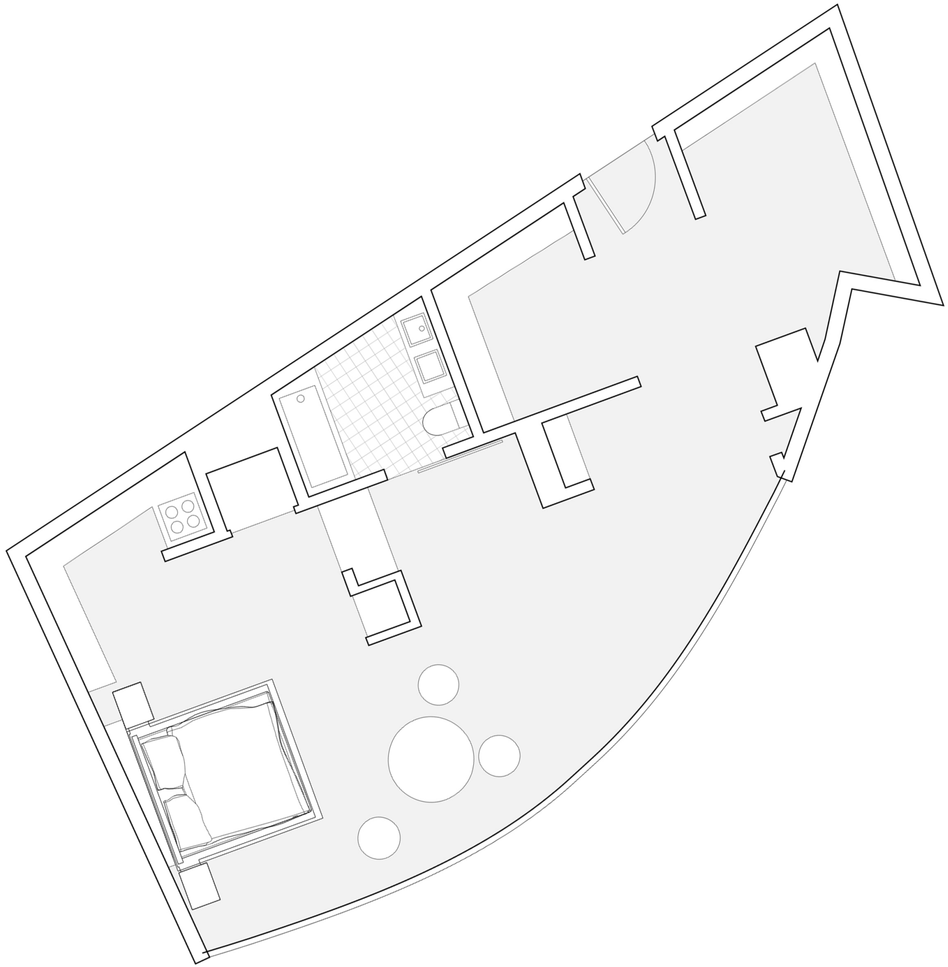

Compactness
Depth
Single-Orientation
Multi-Orientation
Program Spectrum
Acute Angles

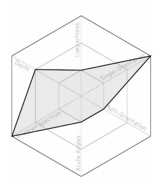
Manhattan



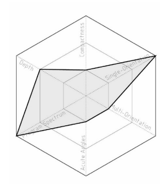

Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Manhattan

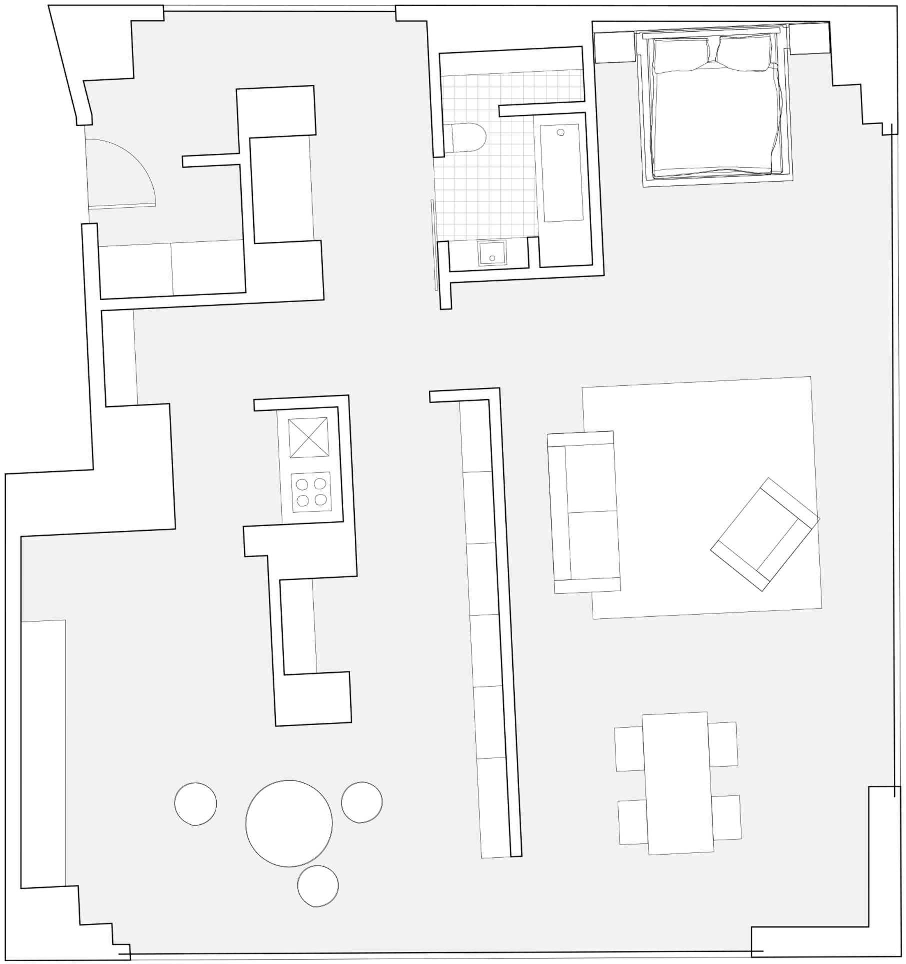


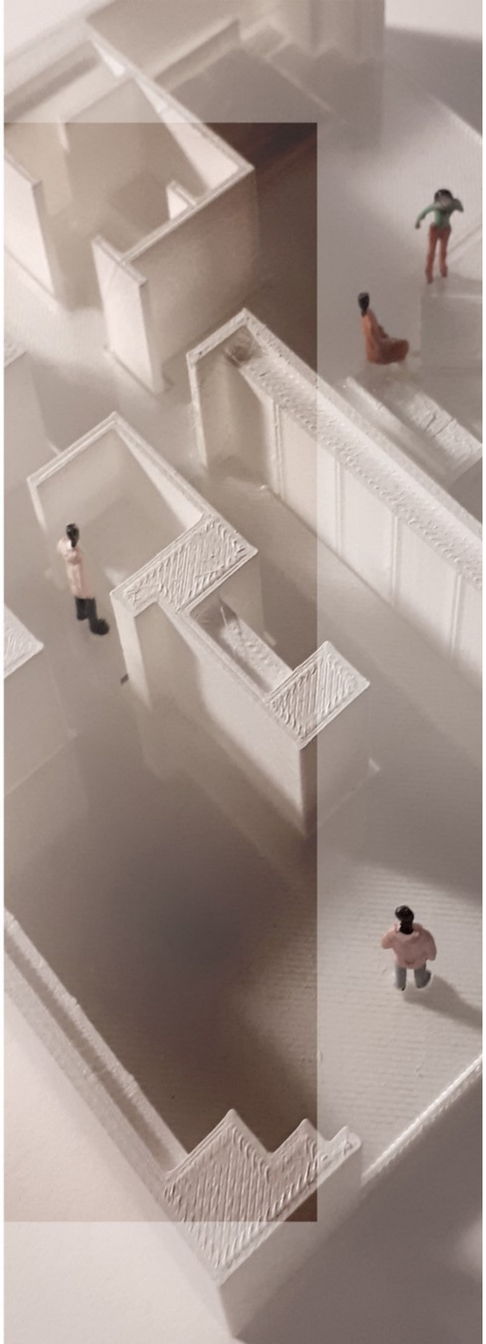

Compactness
Depth
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum

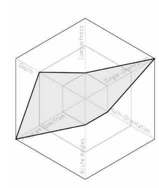
Manhattan


Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

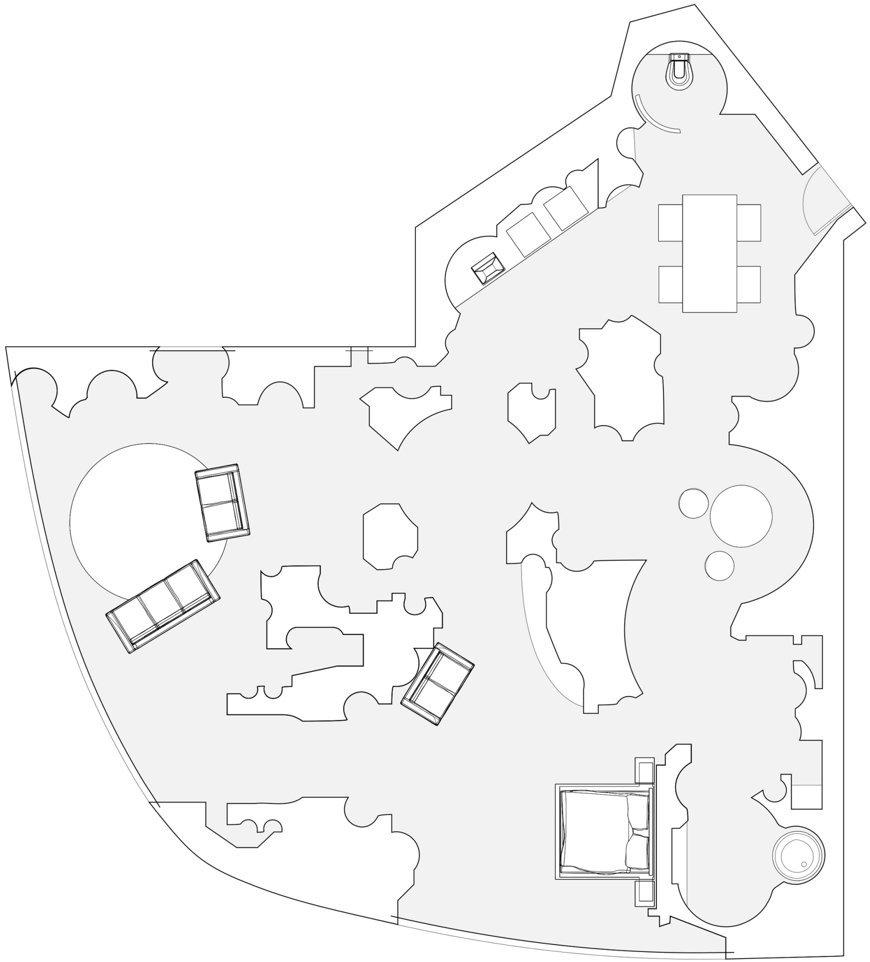
Manhattan

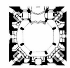


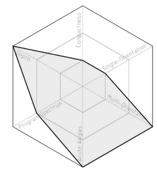



Compactness
Depth
Single-Orientation
Multi-Orientation
Program Spectrum
Acute Angles

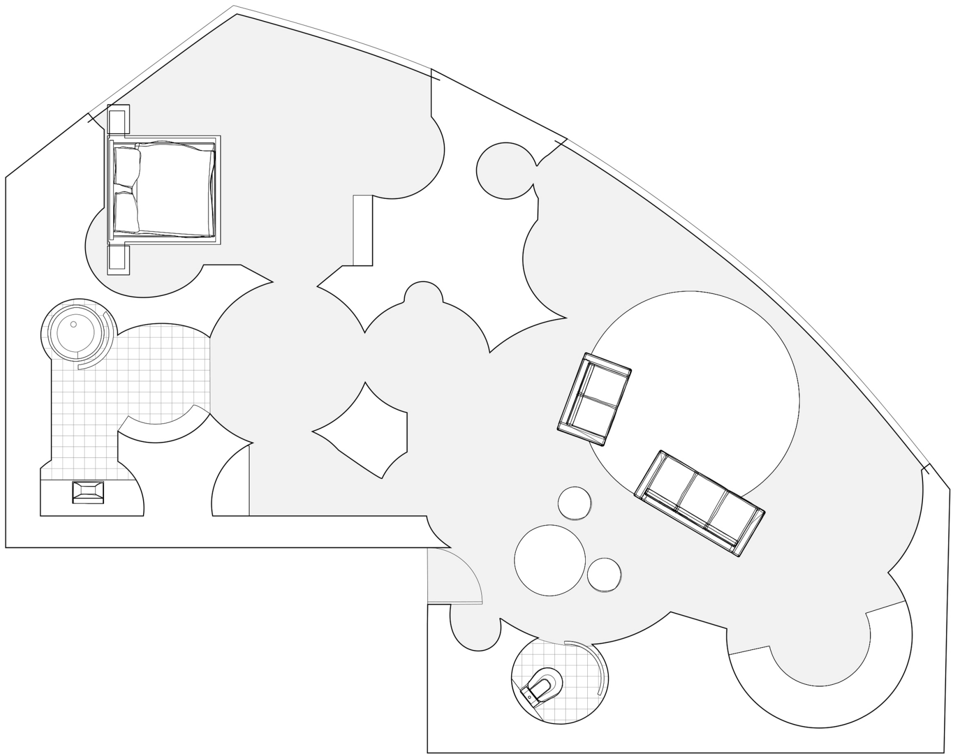
Manhattan

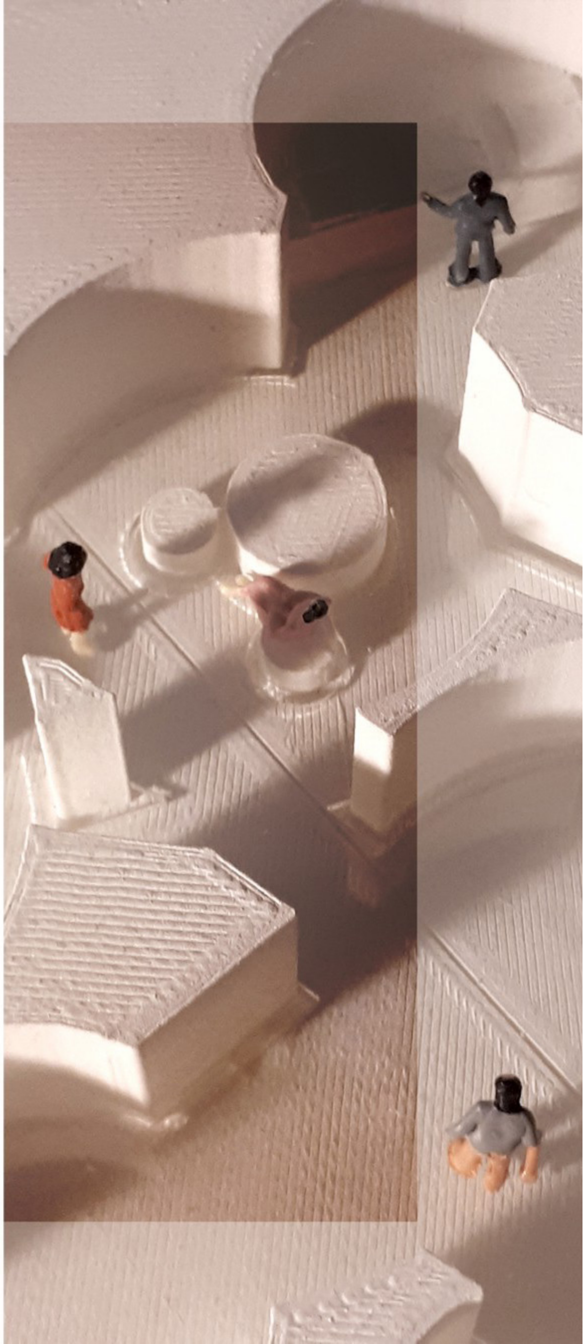


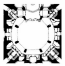


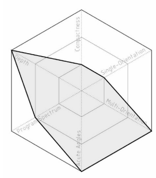

Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Baroque

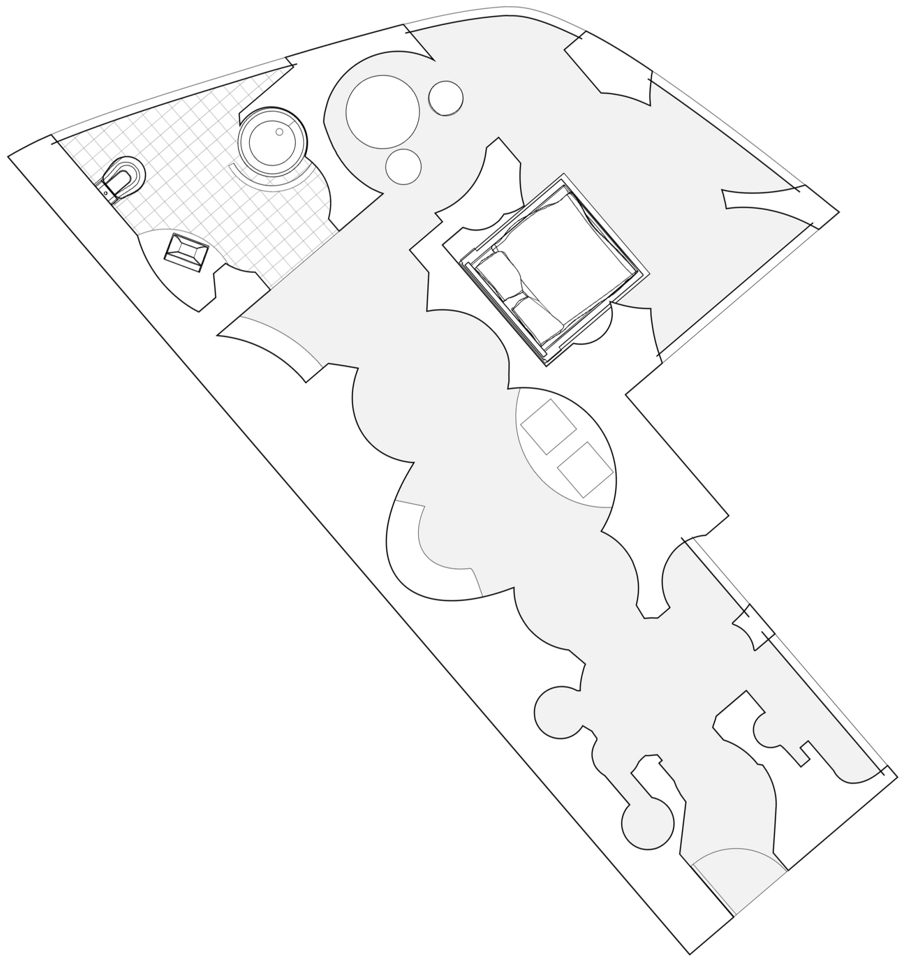


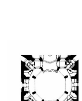


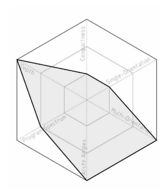



Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

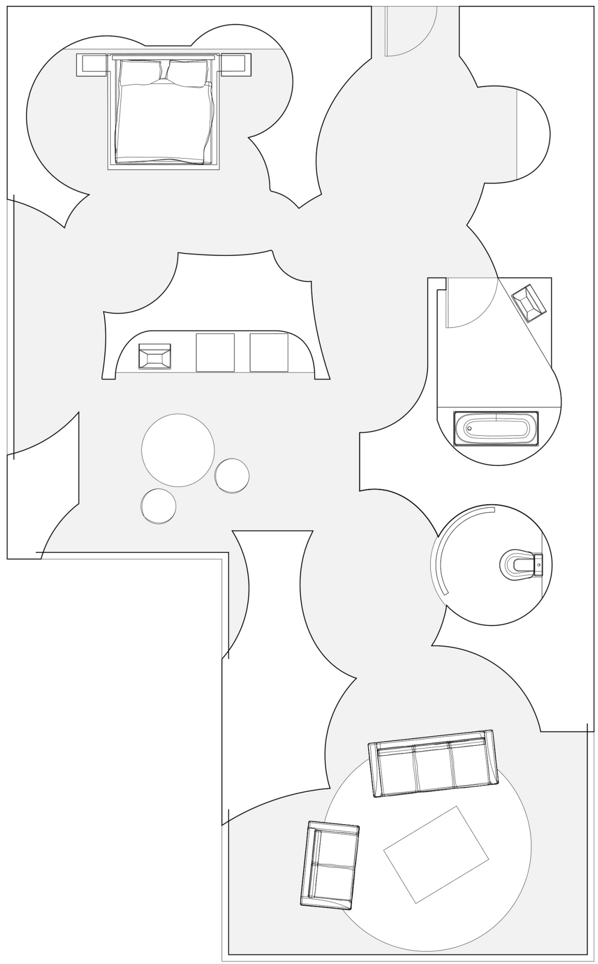
Baroque

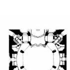


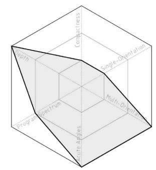



Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

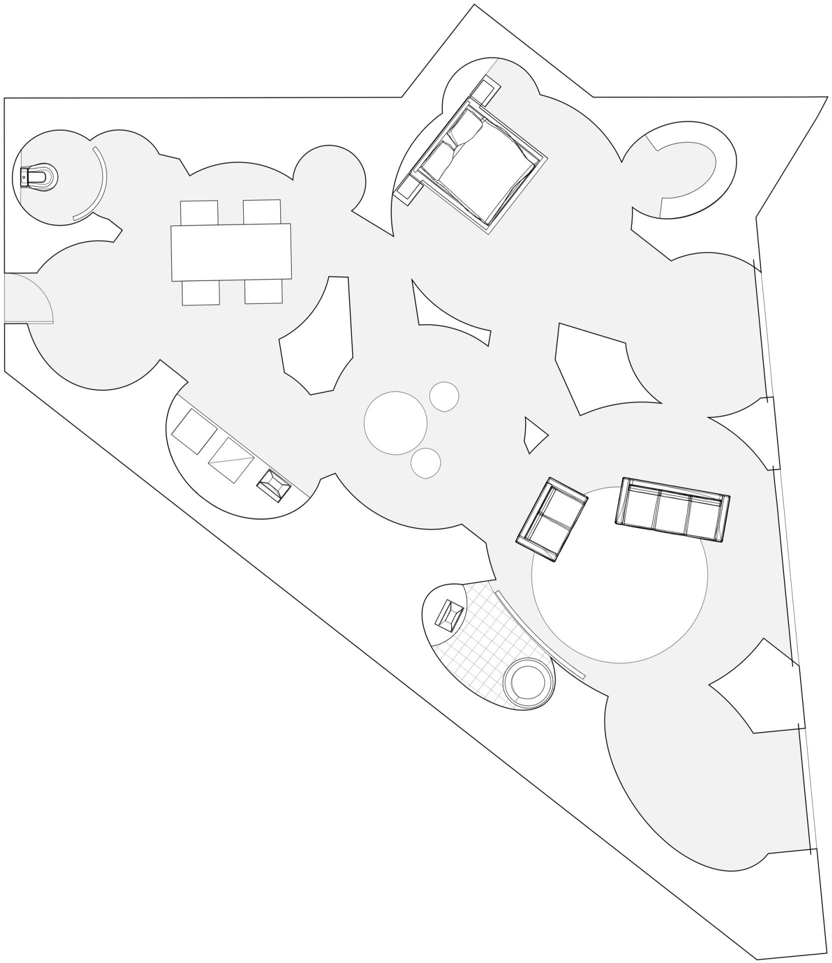
Baroque

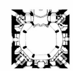


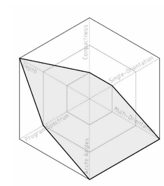



Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

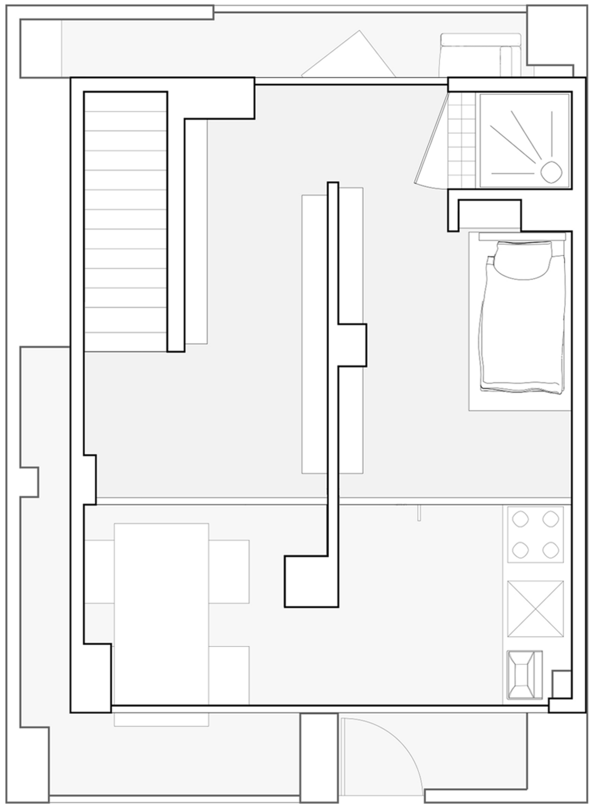
Baroque

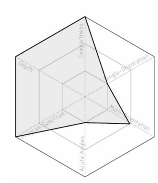




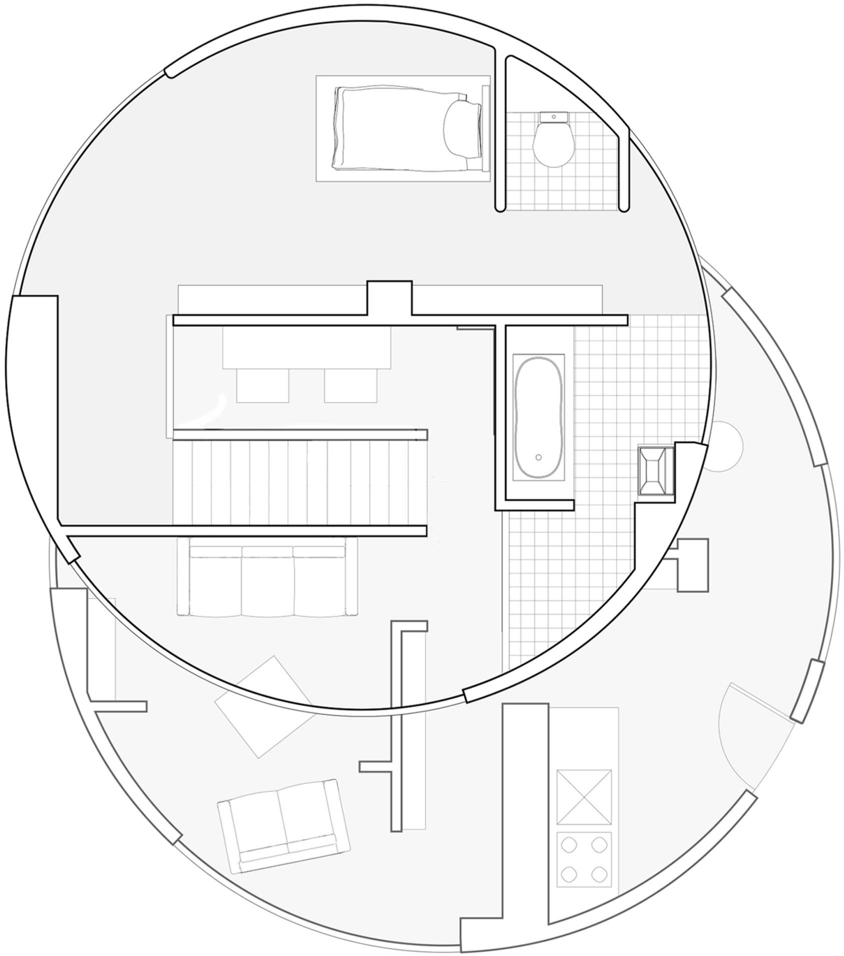

Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

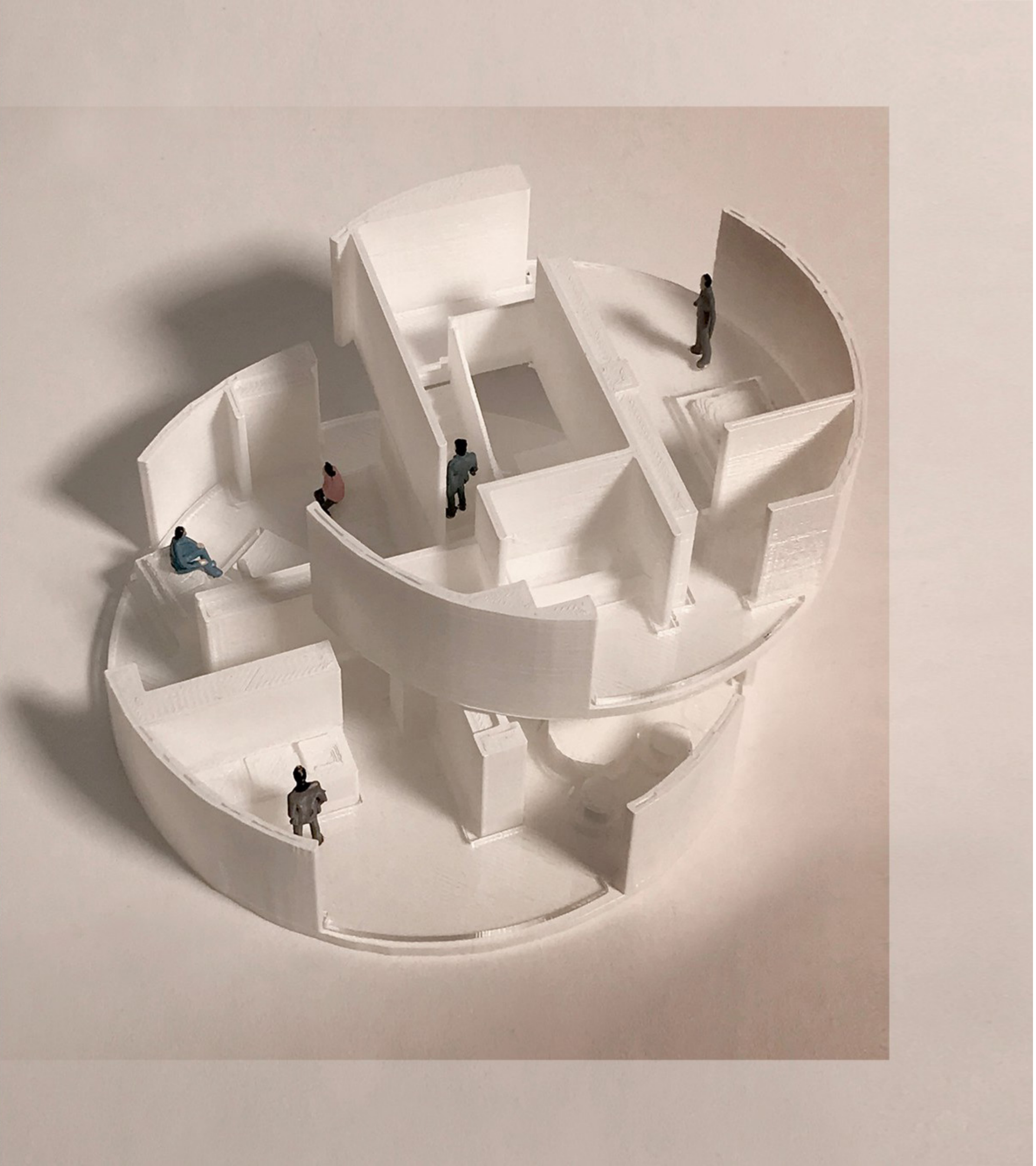
Baroque

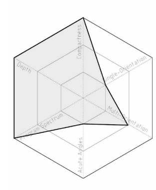



Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

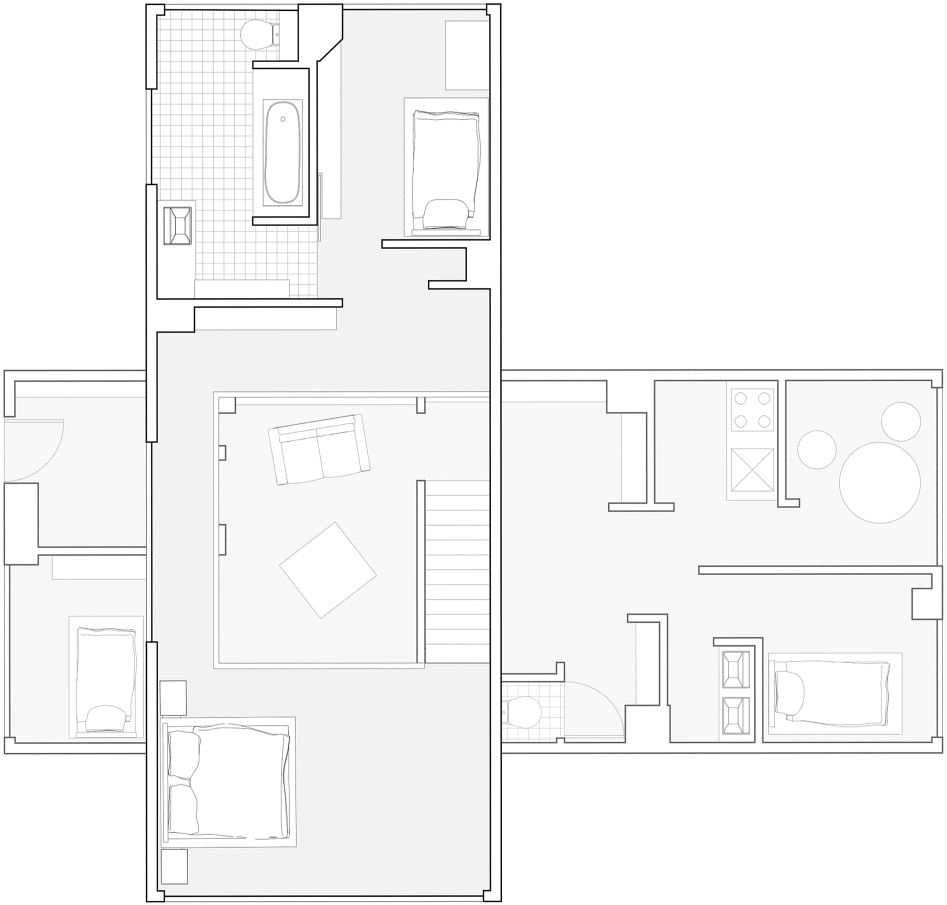
Row–House

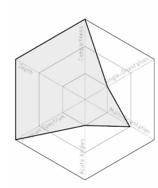




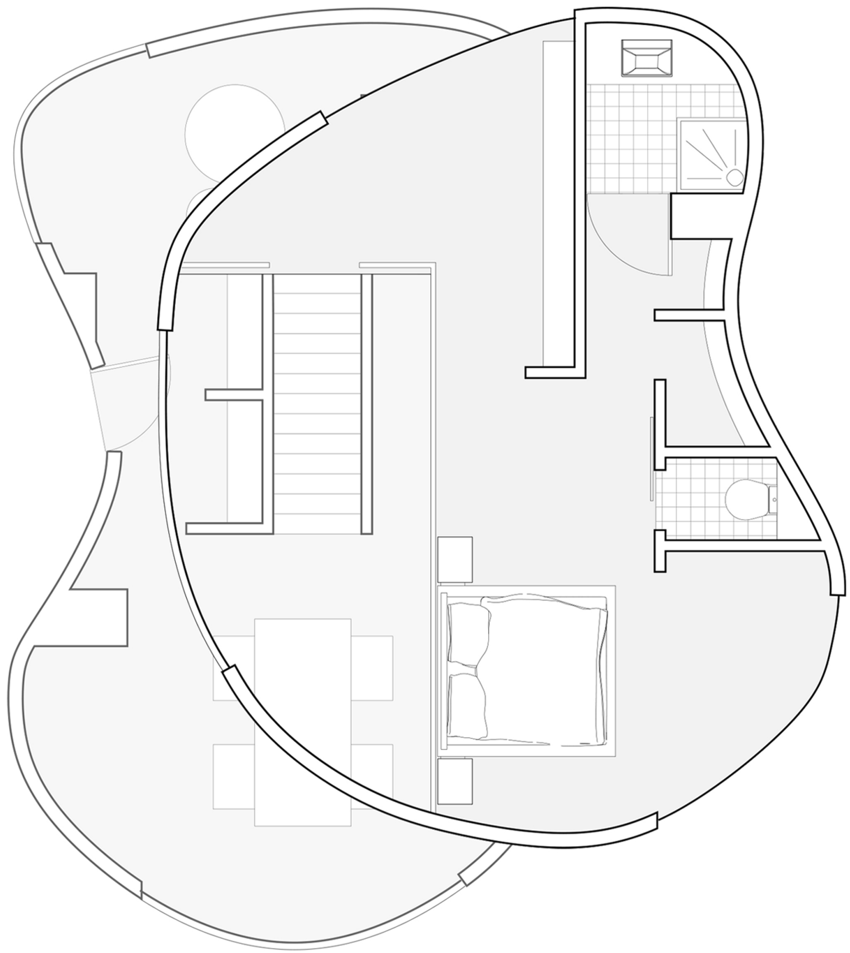


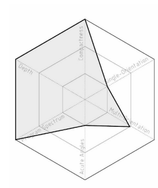
Row–House



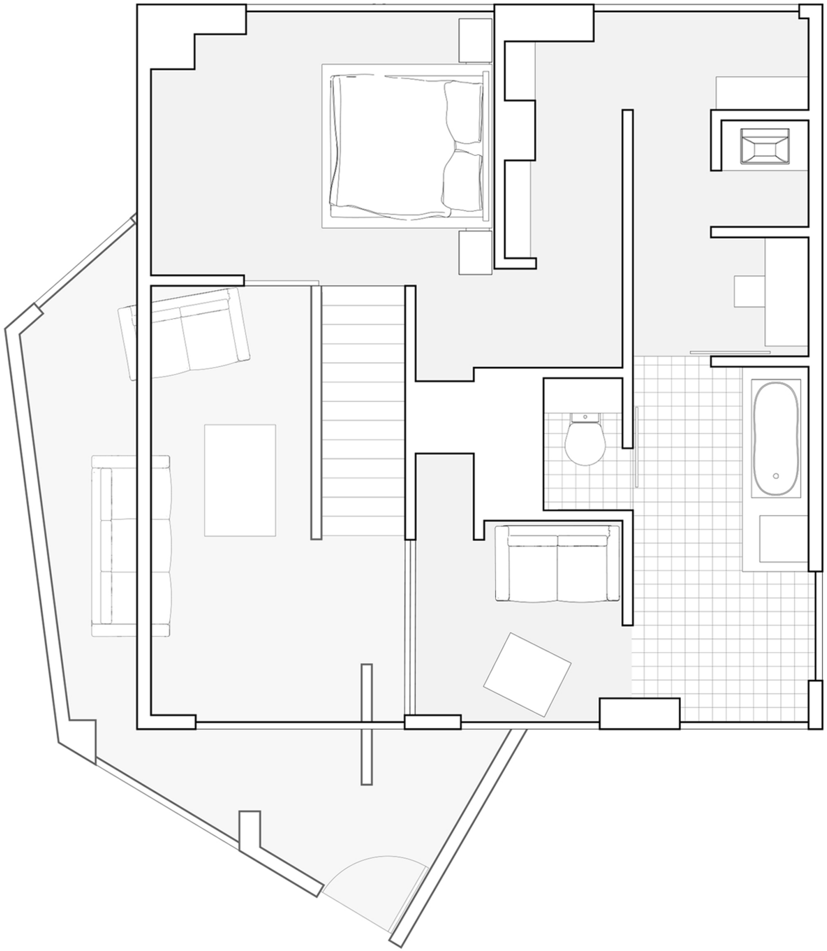


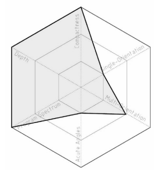
Row–House



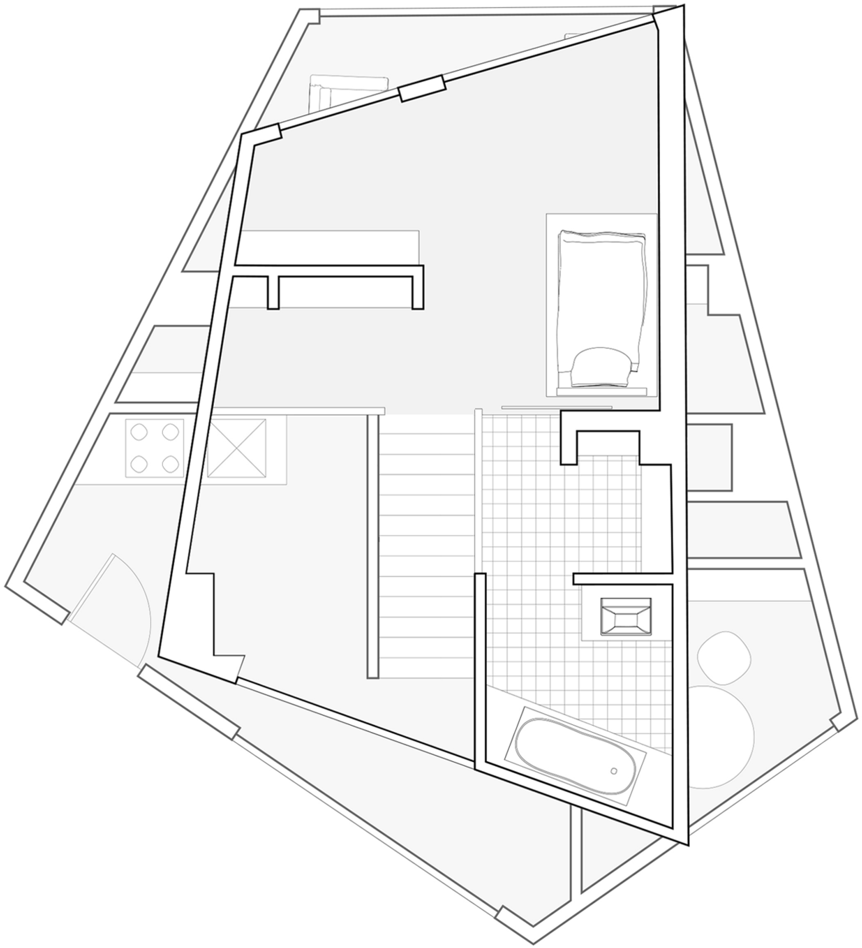

Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

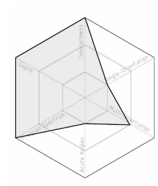
Row–House




Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

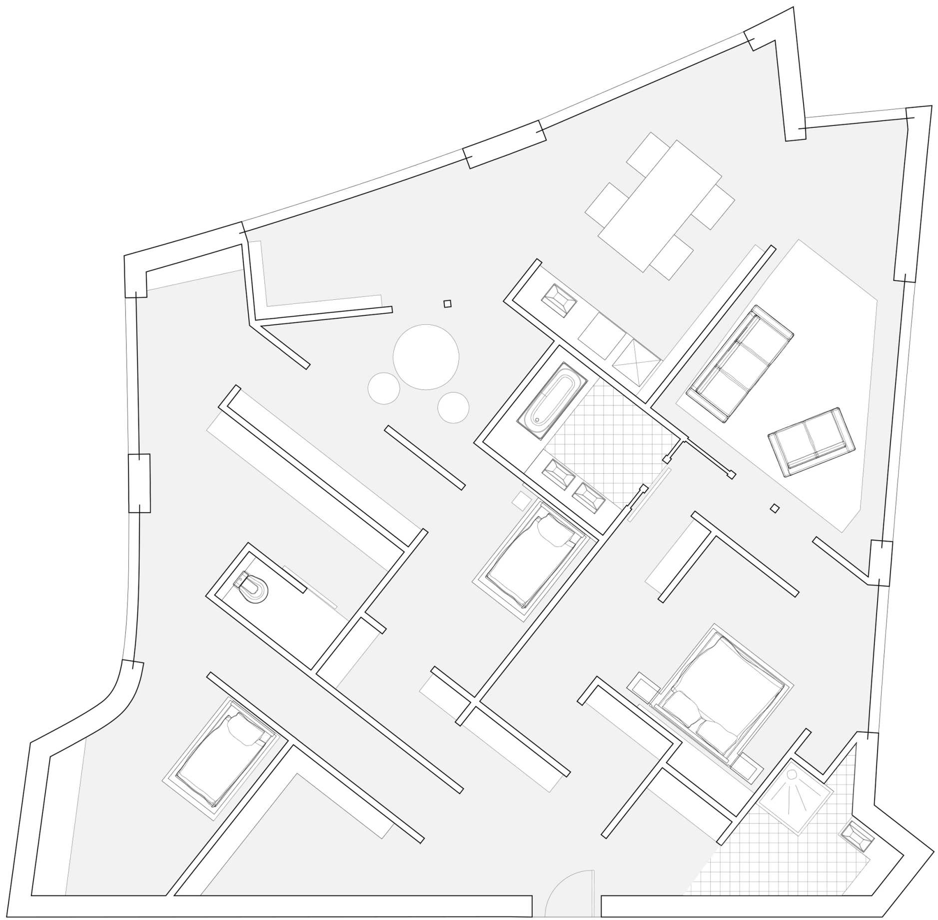
Row-House

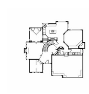


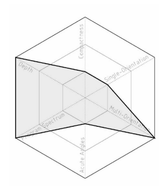


Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Row-House

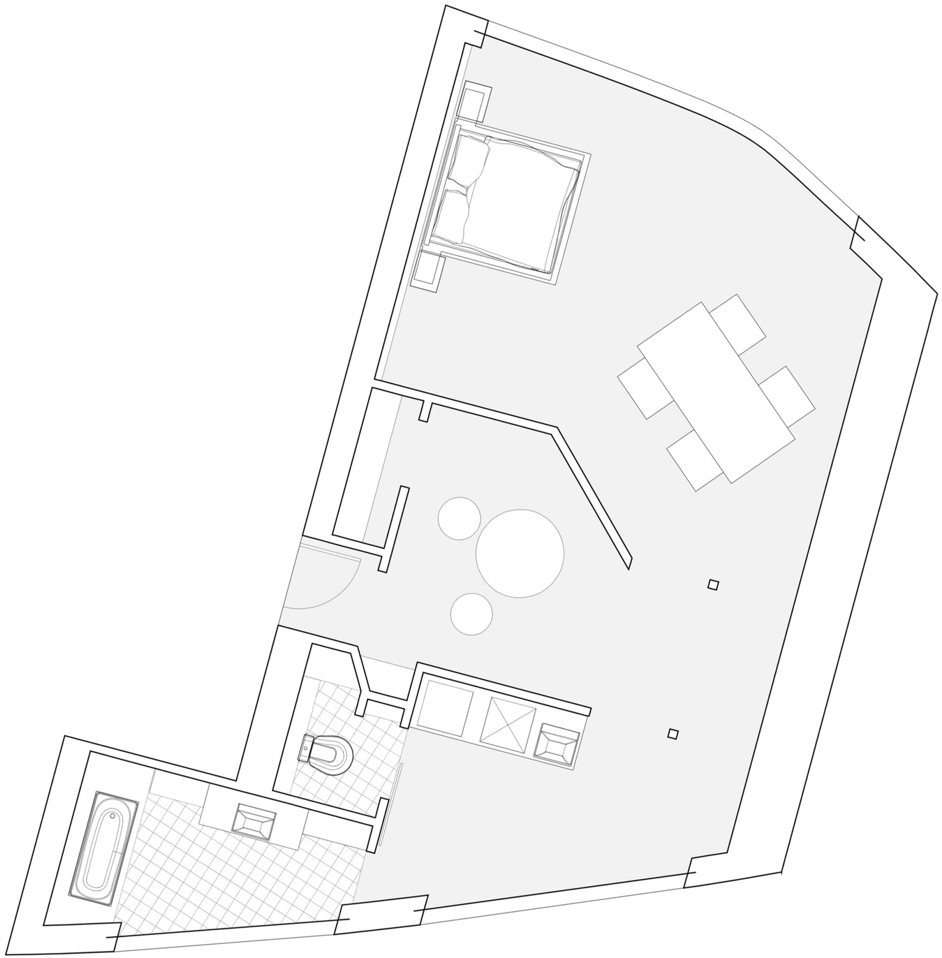


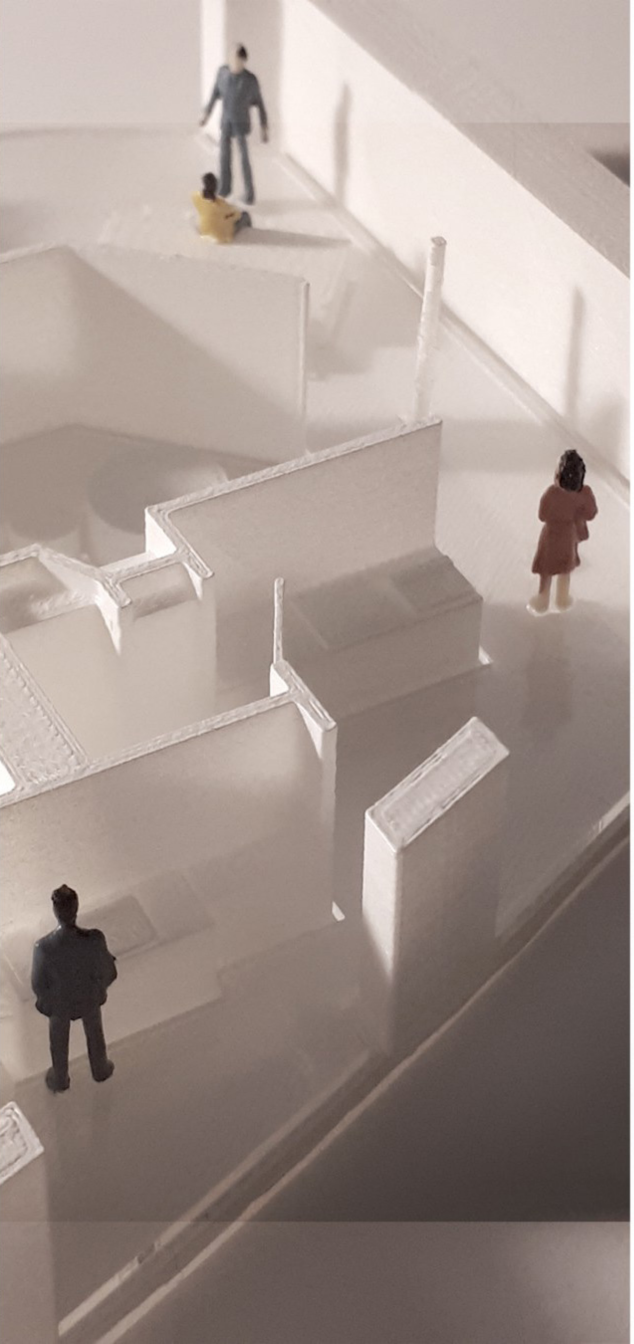


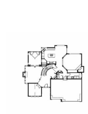


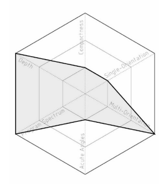
Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Victorian

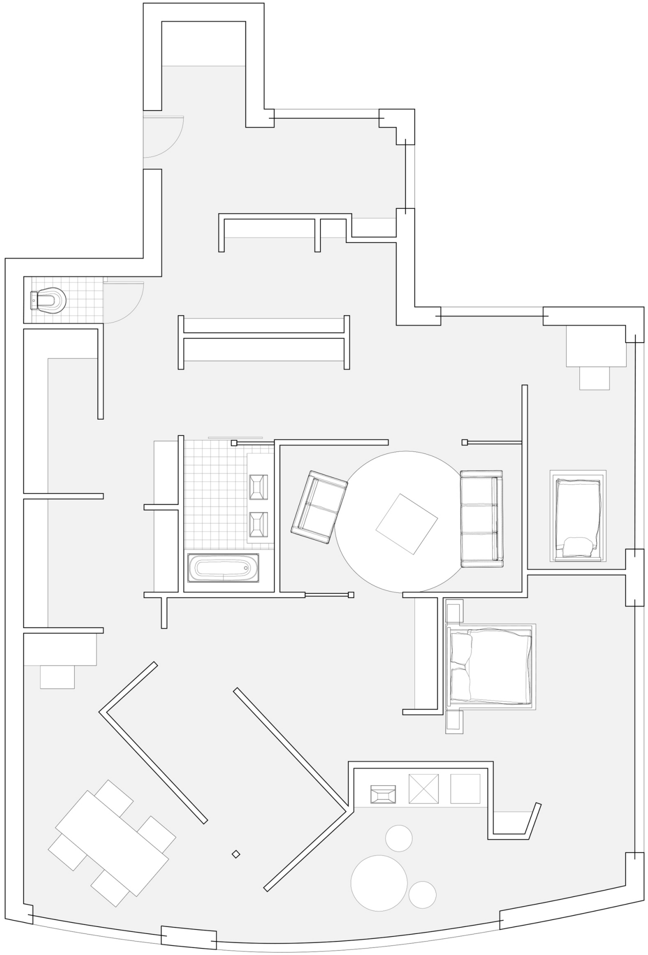


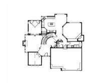


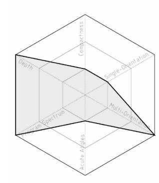



Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

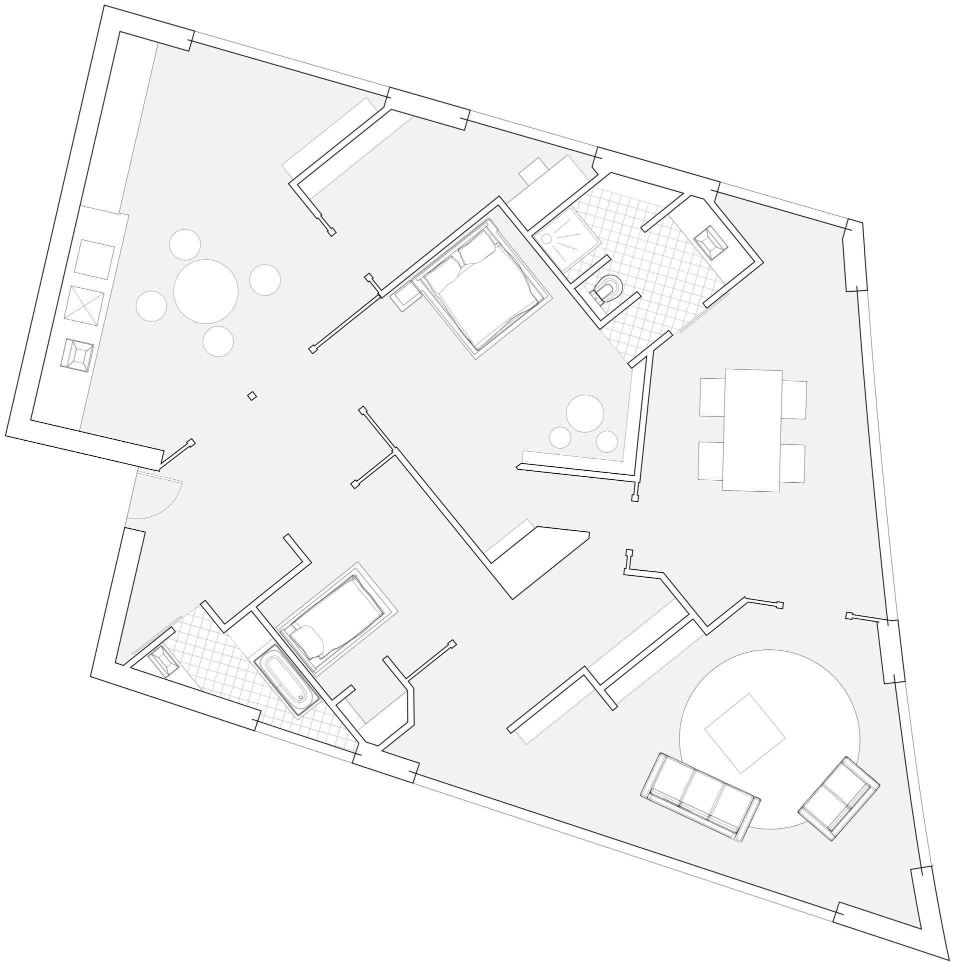
Victorian

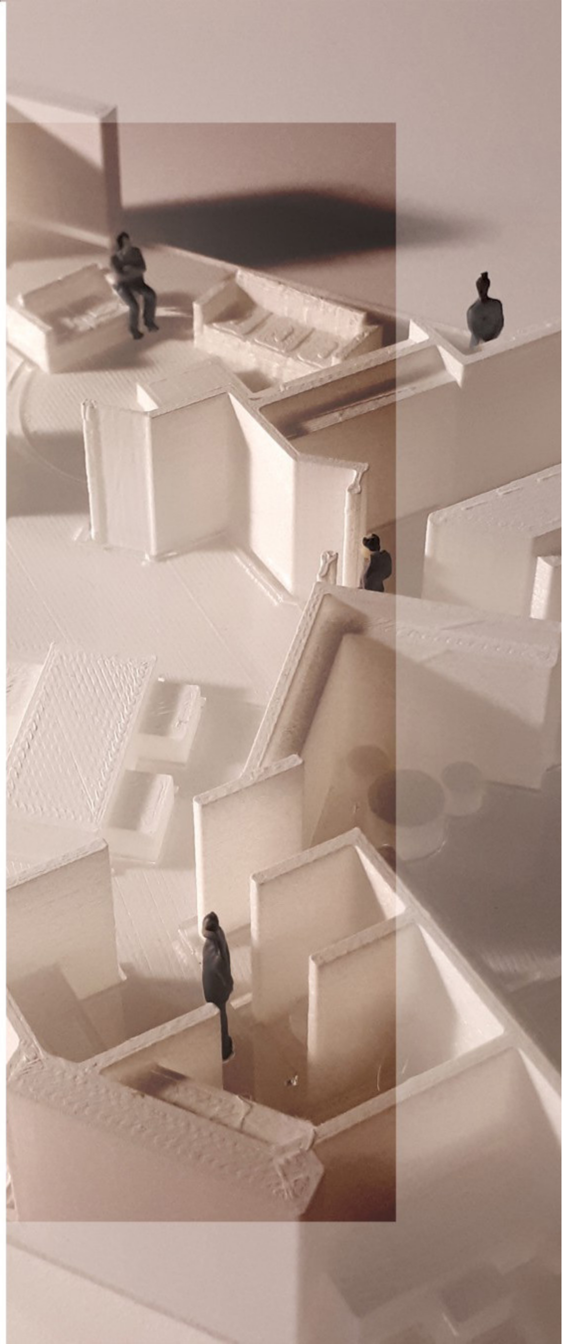


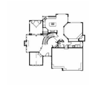


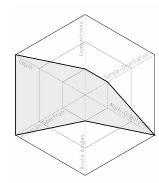

Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Victorian








Compactness
Single-Orientation
Multi-Orientation
Acute Angles
Program Spectrum
Depth

Victorian

Style

Takeaways

If we can think of floorplans first as compositions, before being strictly the product of engineering, then studying the driving forces of the composition is maybe where AI can offer us some meaningful answers. Following this intuition, we have evidenced in this chapter that architectural styles carry at a deeper level an implicit mechanic of space, that significantly impacts any floorplan's composition. In clear, there are spatial consequences to choosing a given style over another.

At a more fundamental level, we can think of styles as being the by-product of architectural history. If there is within each style a deeper set of functional rules, then studying architectural history could potentially be about understanding the evolution over time of these implicit rules. Being able to encapsulate each style could allow us to go beyond the study of precedents, and complement it by unpacking the behavior of GAN-models such as the ones trained here. Their ability to emulate some of the unspoken rules of architecture could allow us to address the "quality with no name" embedded in buildings that Christopher Alexander defines in his book The Timeless Way of Building. AI is simply a new way to study it.

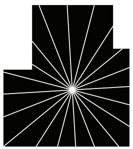
Finally, the inherent presence of style within each GAN-model constitutes a key takeaway: far from the promise of an agnostic & objective practice of generative design, it seems that style permeates irrevocably the very essence of any generative process. In clear: Style is not an ancillary, superficial or decorative addendum. Style is at the core of the composition. Recognizing this evidence is a prerequisite to understanding what AI can bring to Architecture. In other words, there will be no agnostic-AI for Architecture, no style-less machine, no objective generative design. On the contrary, each model or algorithm will come with its flavor, its personality, its know-how.

Access Code Source

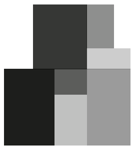
III Qualify

"*Mal nommer les choses c'est ajouter au malheur du monde*" – Albert Camus

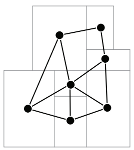
"Failing to name things adds to the World's disarray"



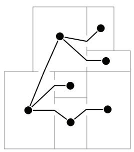
Footprint



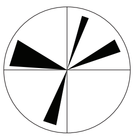
Program



Connectivity



Circulation



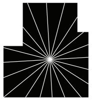
Orientation

Thickness & Texture

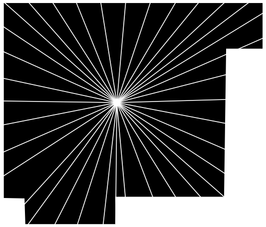
The study of Architecture requires us to agree on terms and adjectives to make sure computers metrics can eventually grasp the different facets of building design. This imperative becomes most crucial, as the Architect plans to rely on computers to help their design process. Transforming adjectives into quantifiable metrics and encoding them becomes the necessary bridges between the human and the machine.

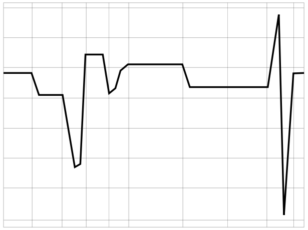
To that end, we offer in this chapter six key metrics, qualifying six essential aspects of floorplan design: Footprint, Program, Orientation, Thickness & Texture, Connectivity, and Circulation. These metrics work together as a comprehensive framework, addressing both the stylistic and organizational dimensions of floorplans. Each has been developed as an algorithm, thoroughly tested, and released as an open source set of tools.

Access Code Source



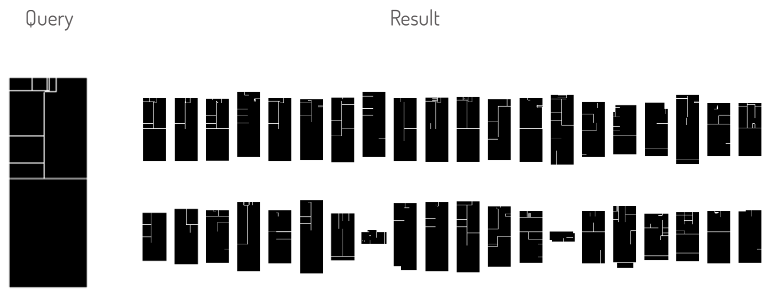
Footprint

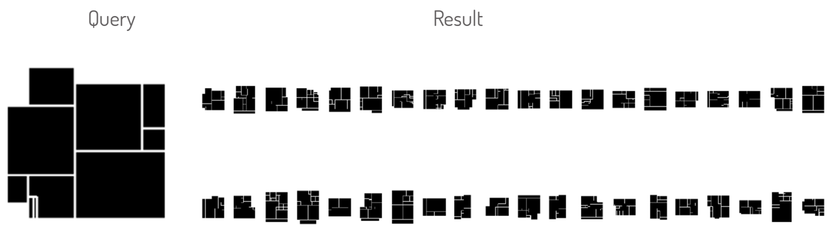


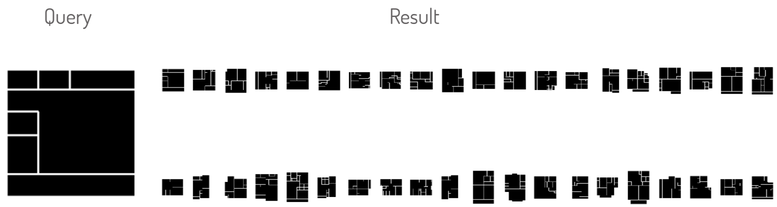


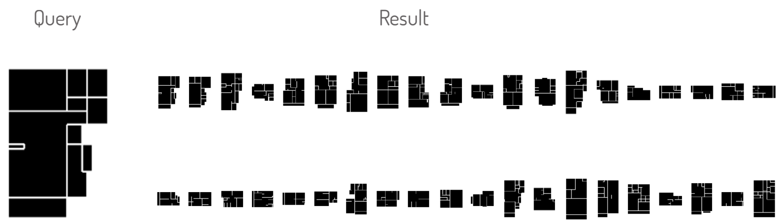
The shape of a building is the simplest and most intuitive proxy to qualify its style. The “Footprint” metric analyzes the shape of a floorplan perimeter and translates it into a histogram. This descriptor, while encoding the shape of a building, can translate common adjectives—“thin”, “bulky”, “symmetrical”, etc.—used by architects into numerical information, in order to communicate with a computer about building shapes.

From a technical standpoint, this metric uses polar convexity to turn a given outline into a list of discrete values (vectors) that can then be compared to other floorplans. We use a polar array of lines, stemming from the center of the plan, to extract the area of the plan captured by each slice of space obtained. This methodology has proven to yield satisfactory results, as shown in the queries here on the left. This technic can also be employed to qualify indoor spaces’ shape as well as building perimeter’s geometry.

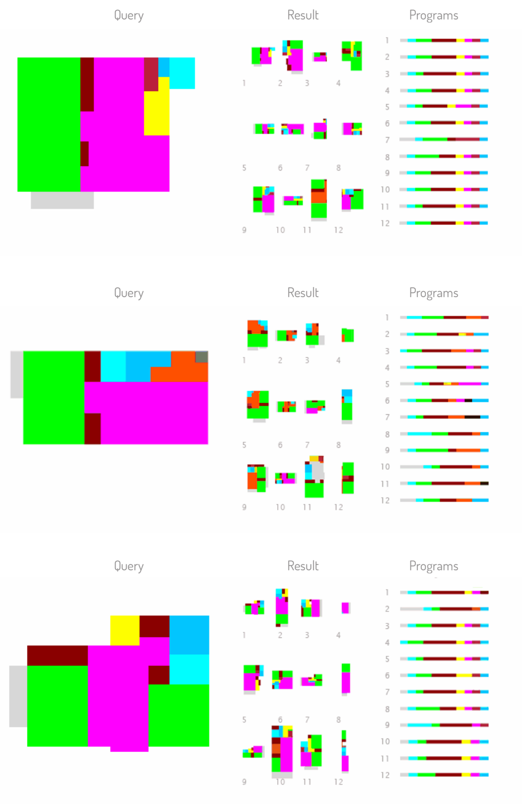

Query
Result


Query
Result


Query
Result


Query
Result

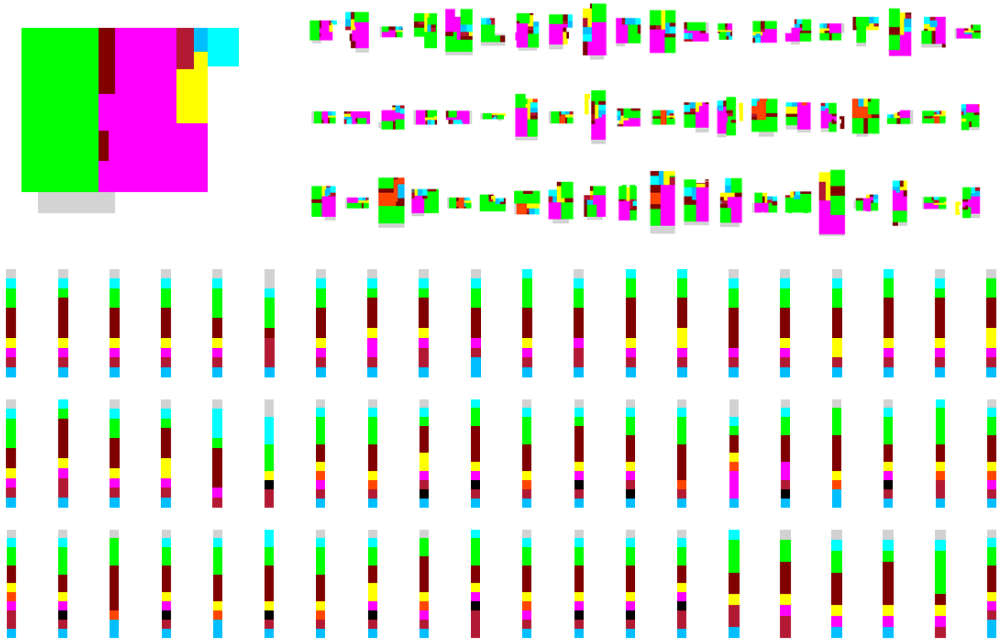
Access Code Source

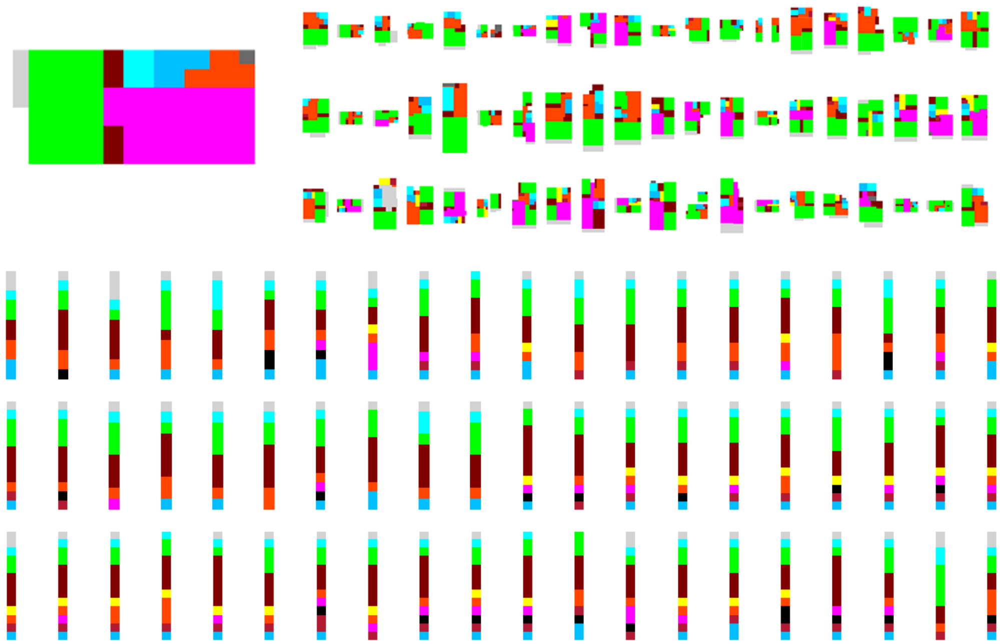


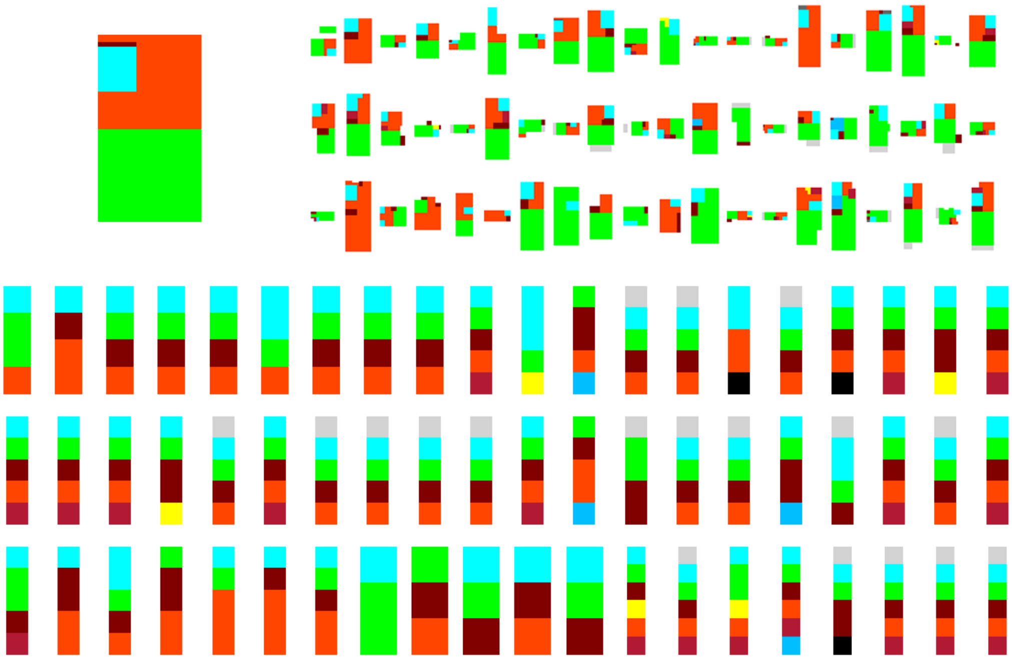
Program

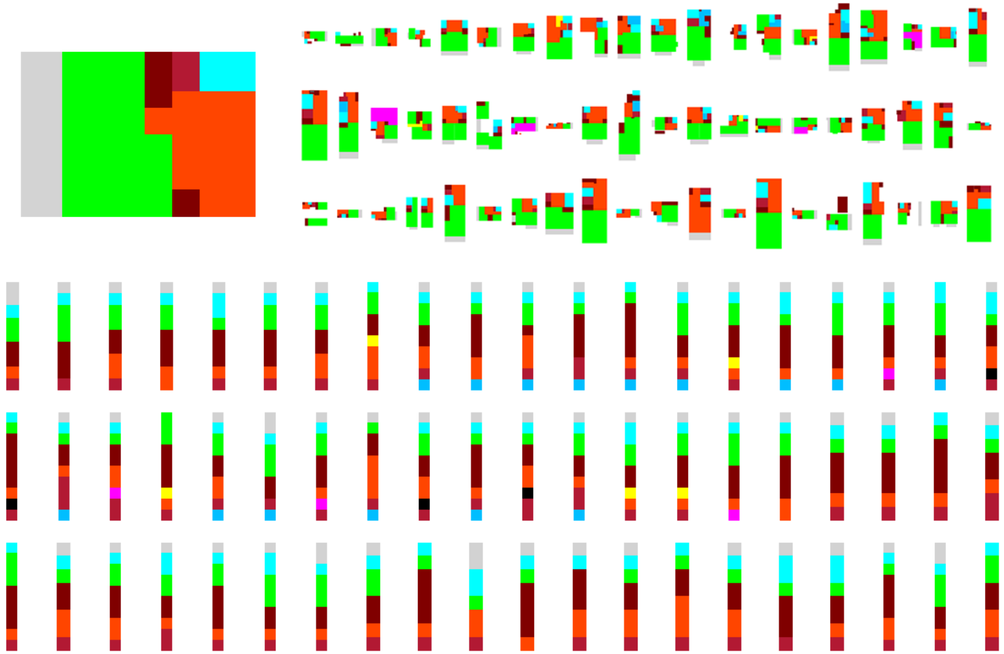
The program of a building, displaying the type of rooms it contains, is a major driver of its internal organization. Capturing this reality is central to our approach. To describe the "mix" of rooms we represent through a color code the list of rooms contained in any given floorplan. This colored band becomes the proxy, to describe the program. It acts as a template, aggregating both the quantity and the programmatic quality of the rooms within the floorplan. It is an intuitive visual description for humans which can translate into a reliable encoding technic for machines.

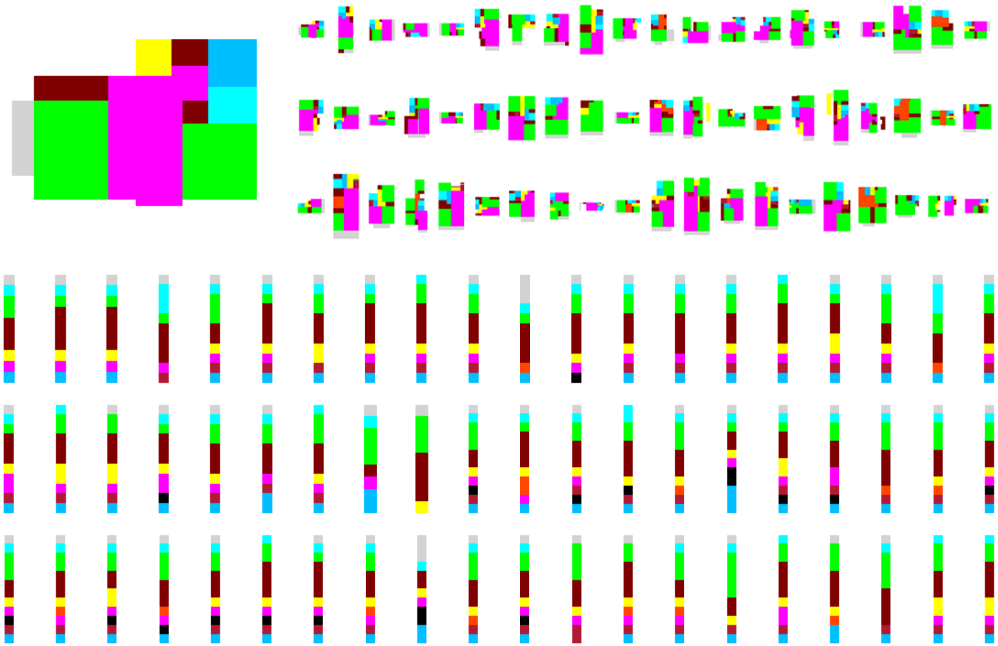
From a technical standpoint, using this colored band enables us to compute the programmatic similarities and dissimilarities between any given pair of floorplans. To visualize results, each plan is reported as both a colored floorplan and one-dimensional color vector of its program.

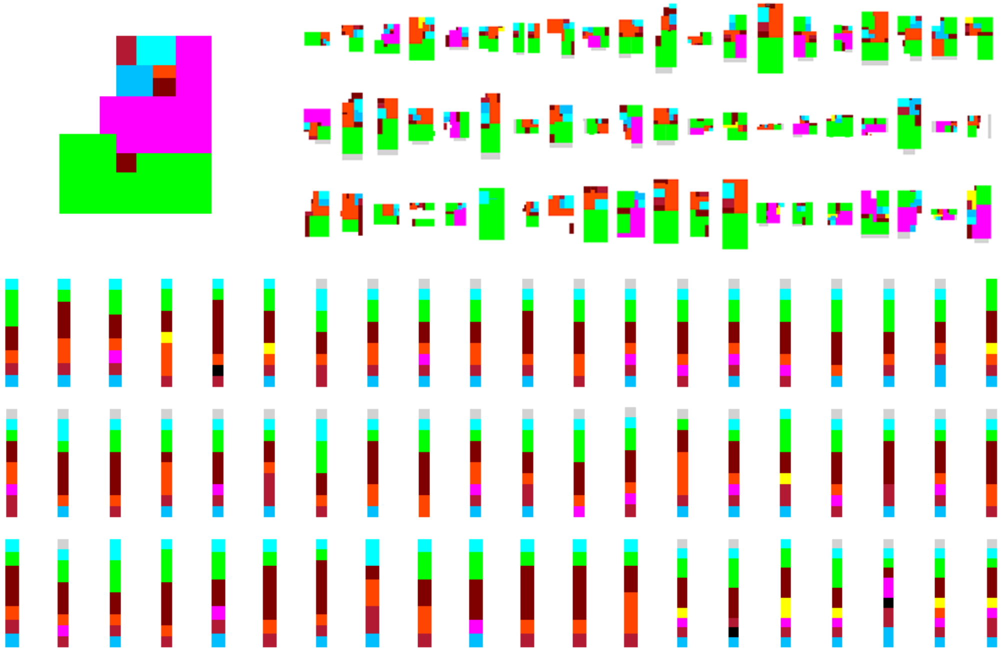




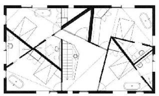


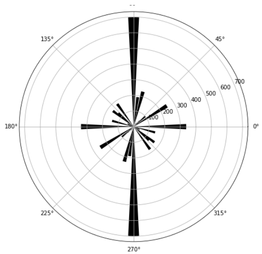


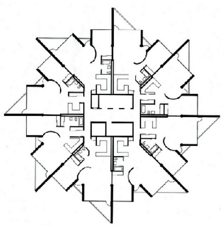


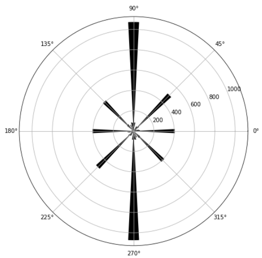


Access Code Source





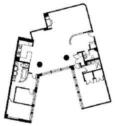


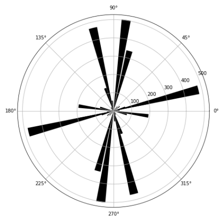


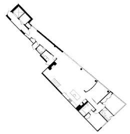
Orientation

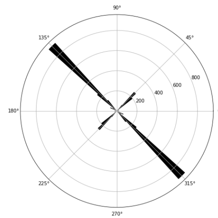
The orientation of walls in a plan is a valuable source of information. It can describe both the enclosure of a plan (how secluded spaces are due to the walls' presence) and the style of plan. In fact, using this metric, we can easily differentiate a modern house-pavilion from a gothic cathedral, simply by extracting the histogram of the walls' orientation.

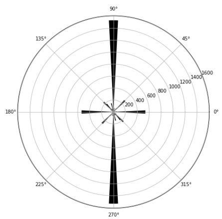
From a technical standpoint, orientation extracts the walls of a given floorplan and sums their length along each direction of space, from 0 the 360 degrees. The resulting list of values is an assessment of the overall orientation of the plan. It can be averaged to get a single descriptor or used as a vector to compare across plans.

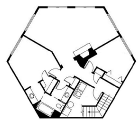


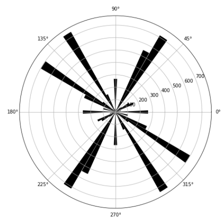

90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500

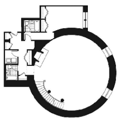
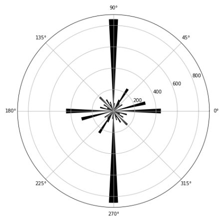
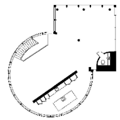
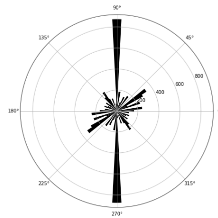


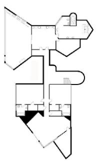
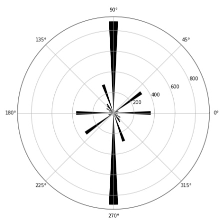
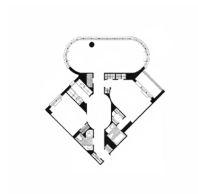
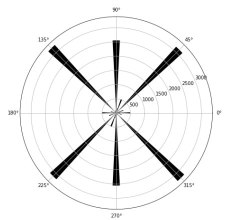

90°
135°
45°
180°
0°
225°
315°
270°
200
400
600
800

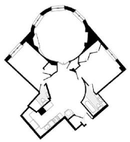

90°
135°
45°
180°
0°
225°
315°
270°
200
400
600
800
1000
1200
1400
1600

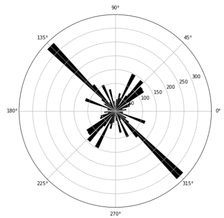


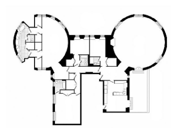

90°
135°
45°
180°
0°
225°
315°
270°
200
300
400
500
600
700

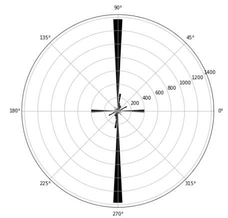


90°
45°
135°
180°
0°
225°
315°
270°
200
400
600
800


90°
45°
135°
180°
0°
225°
315°
270°
200
400
600
800

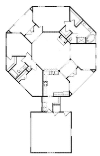


90°
45°
135°
180°
0°
225°
315°
270°
200
400
600
800


90°
45°
135°
180°
0°
225°
315°
270°
500
1000
1500
2000
2500
3000

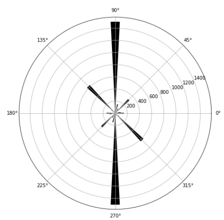


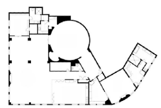

90°
45°
135°
180°
0°
225°
315°
270°
100
150
200
250
300

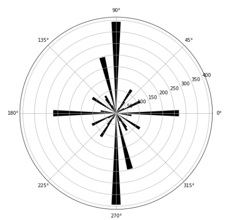


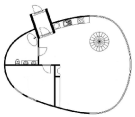

90°
45°
135°
180°
0°
225°
315°
270°
200
400
600
800
1000
1200
1400

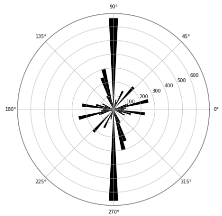


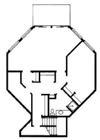

90°
45°
135°
180°
0°
225°
315°
270°
200
400
600
800
1000
1200
1400

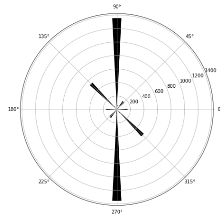


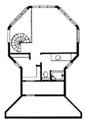

90°
45°
135°
180°
0°
225°
315°
270°
150
200
250
300
350
400

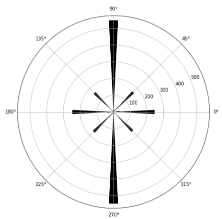


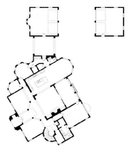

90°
45°
135°
0°
180°
315°
225°
270°
100
200
300
400
500
600

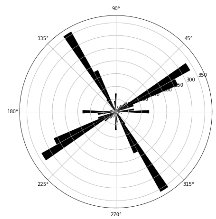


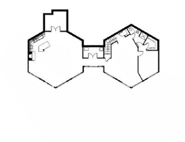

90°
45°
135°
0°
180°
315°
225°
270°
200
400
600
800
1000
1200
1400

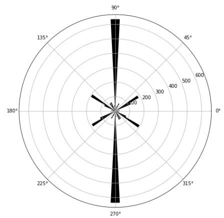


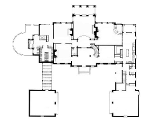

90°
45°
135°
0°
180°
315°
225°
270°
100
200
300
400
500

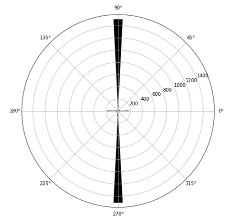


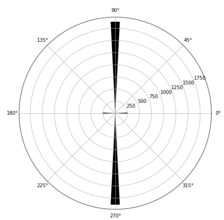

90°
45°
135°
0°
180°
315°
225°
270°
300
350

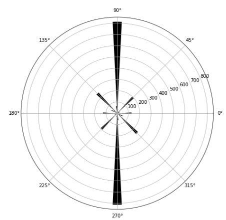


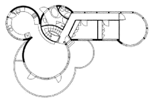

90°
135°
45°
600
500
400
300
200
180°
0°
225°
315°
270°

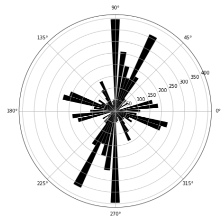


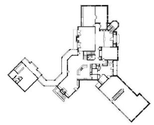

90°
135°
45°
1400
1200
1000
800
600
400
200
180°
0°
225°
315°
270°

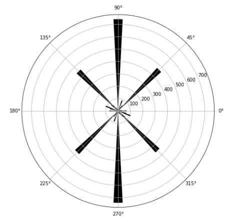

90°
135°
45°
1750
1500
1250
1000
750
500
250
180°
0°
225°
315°
270°

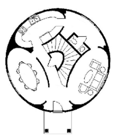

90°
135°
45°
800
700
600
500
400
300
200
100
180°
0°
225°
315°
270°

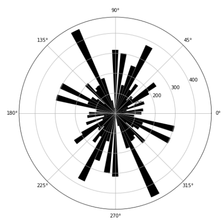


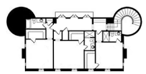

90°
135°
45°
180°
0°
225°
315°
270°
100
150
200
250
300
350
400

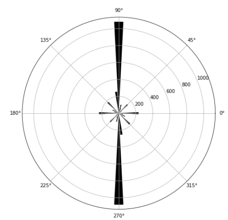



90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500
600
700



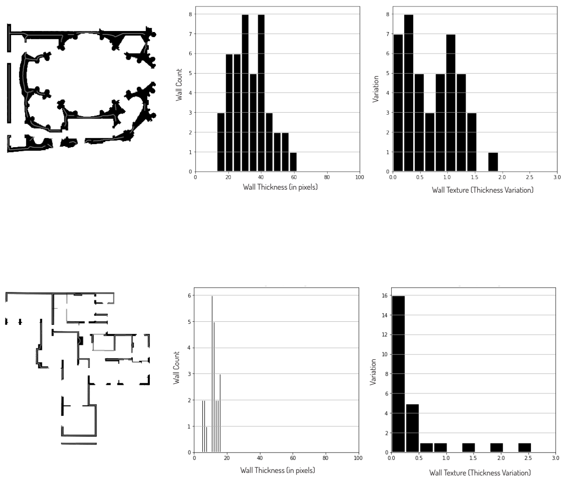

90°
135°
45°
180°
0°
225°
315°
270°
200
300
400




90°
135°
45°
180°
0°
225°
315°
270°
200
400
600
800
1000

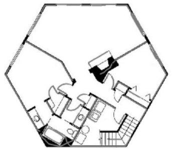
Access Code Source

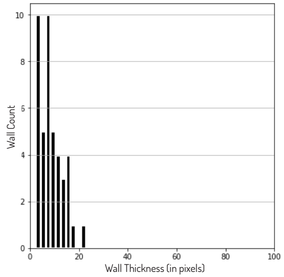
Thickness & Texture

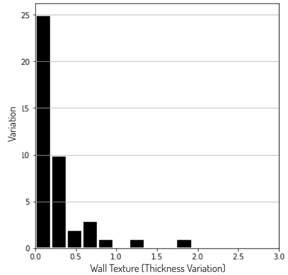


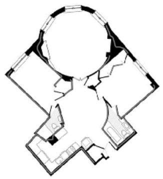
Thickness & Texture qualify the “fat” of the plan: its wall thickness and the variation of this thickness. The thickness of walls across a plan, as well as the geometry of the wall surface—Texture—can vary drastically from one style to other. A Beaux Arts Hall would display columns and indented thick walls when a villa from Mies van der Rohe would display thin rectilinear walls, which our metric would grasp easily.

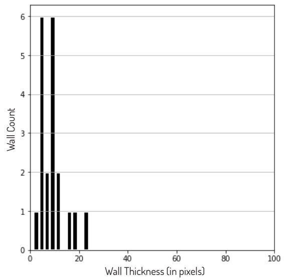
From a technical standpoint, this metric isolates all the walls of a given plan and outputs a histogram of wall thicknesses. At the same time, the algorithm computes the variation of the thickness, to better describe the wall texture (ie. flat walls versus mullions).

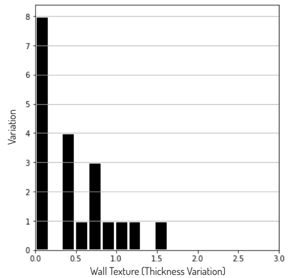


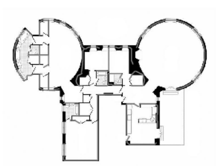

Wall Count
Wall Thickness (in pixels)

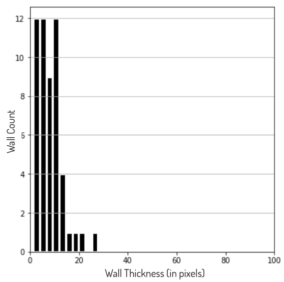

Variation
Wall Texture (Thickness Variation)

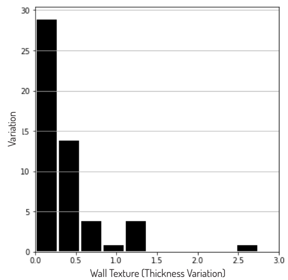


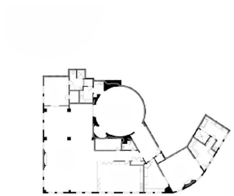

Wall Count
Wall Thickness (in pixels)

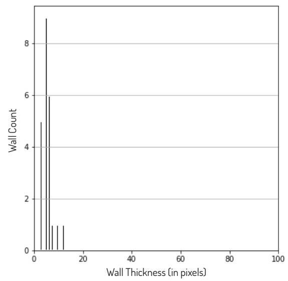

Variation
Wall Texture (Thickness Variation)

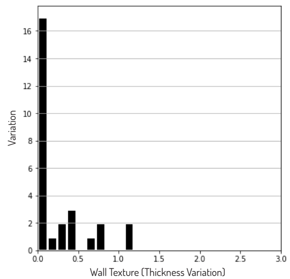


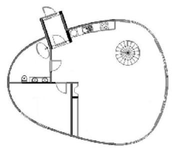

Wall Count
Wall Thickness (in pixels)

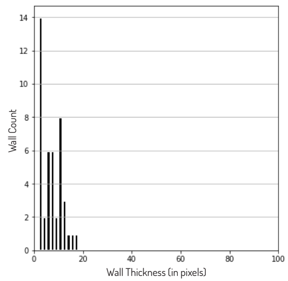

Variation
Wall Texture (Thickness Variation)

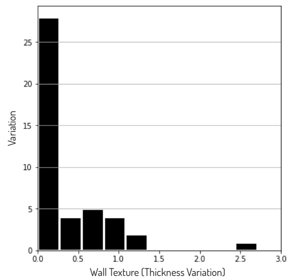


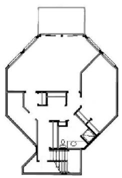

Wall Count
Wall Thickness (in pixels)
0 20 40 60 80 100
0 2 4 6 8

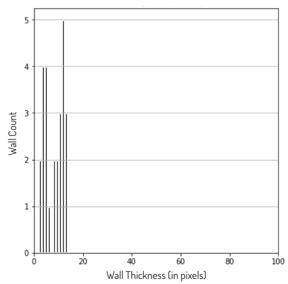

Variation
Wall Texture (Thickness Variation)
0.0 0.5 1.0 1.5 2.0 2.5 3.0
0 2 4 6 8 10 12 14 16

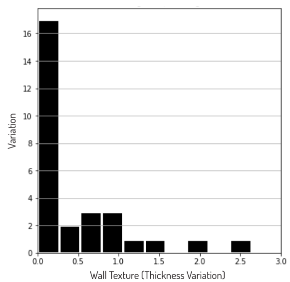


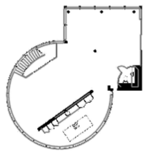

Wall Count
Wall Thickness (in pixels)
0 20 40 60 80 100
0 2 4 6 8 10 12 14

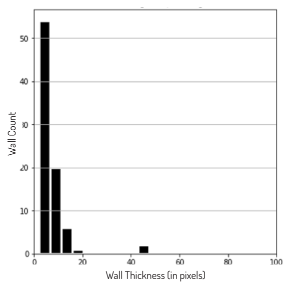

Variation
Wall Texture (Thickness Variation)
0.0 0.5 1.0 1.5 2.0 2.5 3.0
0 5 10 15 20 25

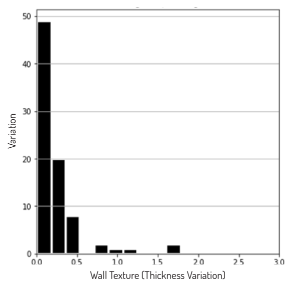


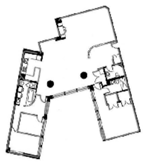

Wall Count
Wall Thickness (in pixels)
0 20 40 60 80 100
0 1 2 3 4 5

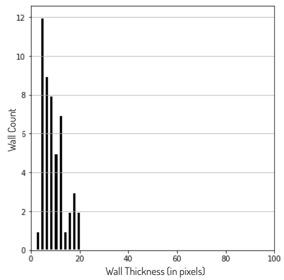

Variation
Wall Texture (Thickness Variation)
0.0 0.5 1.0 1.5 2.0 2.5 3.0
0 2 4 6 8 10 12 14 16

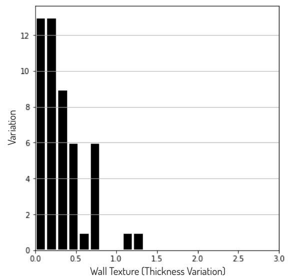


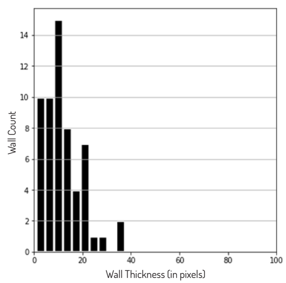

Wall Count
Wall Thickness (in pixels)

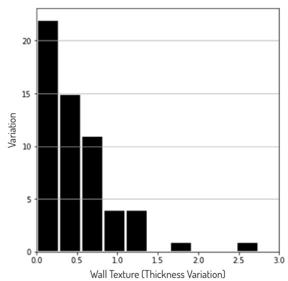

Variation
Wall Texture (Thickness Variation)

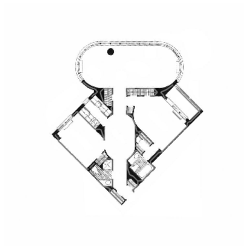


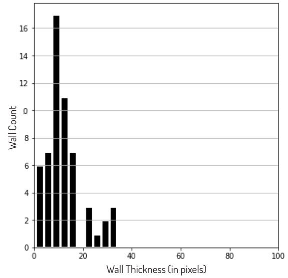

Wall Count
Wall Thickness (in pixels)

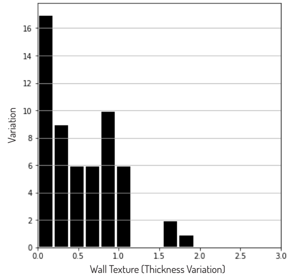

Variation
Wall Texture (Thickness Variation)

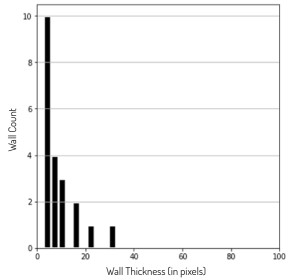

Wall Count
Wall Thickness (in pixels)

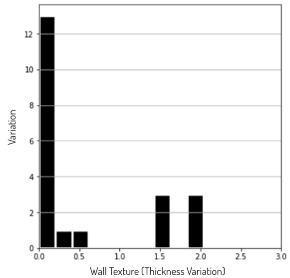

Variation
Wall Texture (Thickness Variation)

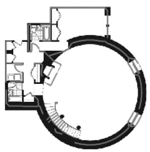


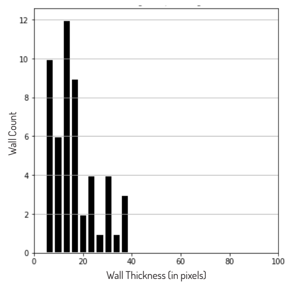

Wall Count
Wall Thickness (in pixels)

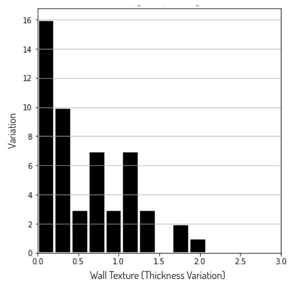

Variation
Wall Texture (Thickness Variation)


Wall Count
Wall Thickness (in pixels)

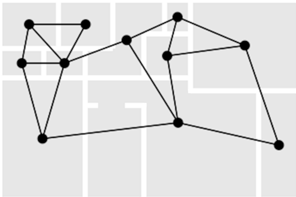

Variation
Wall Texture (Thickness Variation)

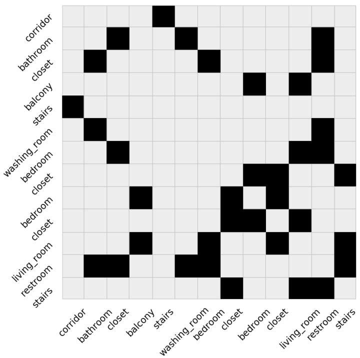


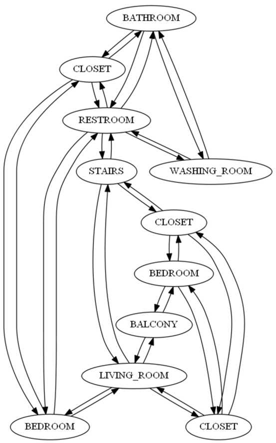

Wall Count
Wall Thickness (in pixels)


Variation
Wall Texture (Thickness Variation)

Access Code Source



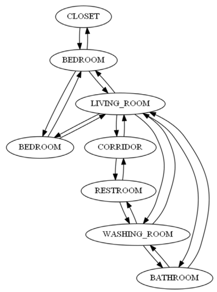


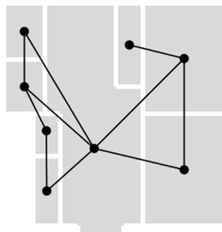


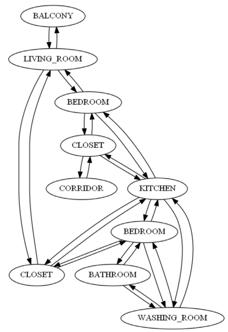
Connectivity

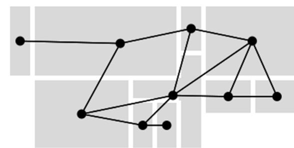
Connectivity tackles the question of room adjacency. The proximity of rooms to one another is a key dimension of a floorplan. Moreover, their connection through doors and corridors defines the existence of connections between them. Connectivity investigates the quantity and quality of such connections, by treating them as a standard graph.

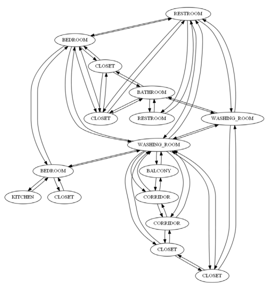
From a technical standpoint, by using the fenestration from the plan, we can deduce the graph of existing relationships among rooms. The Connectivity metric then builds an adjacency matrix, reporting these connections. A graph representation is finally generated. Using this graph, we can compare floorplans, taking into account the similarity of connections among rooms.

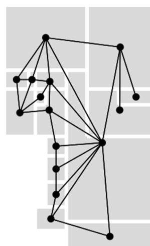

CLOSET
BEDROOM
LIVING_ROOM
BEDROOM
CORRIDOR
RESTROOM
WASHING_ROOM
BATHROOM

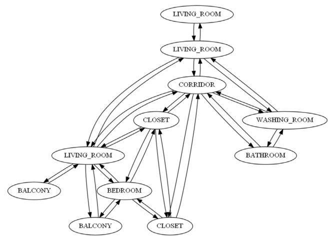


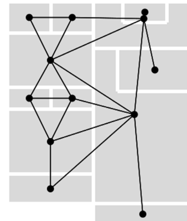

BALCONY
LIVING_ROOM
BEDROOM
CLOSET
CORRIDOR
KITCHEN
BEDROOM
CLOSET
BATHROOM
WASHING_ROOM

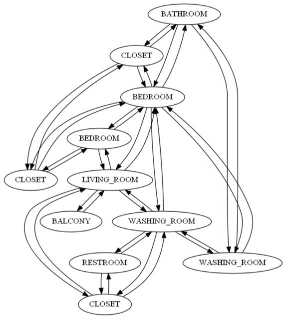


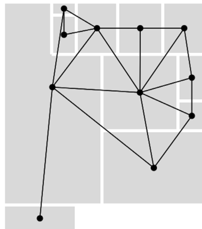

RESTROOM
BEDROOM
CLOSET
BATHROOM
CLOSET
RESTROOM
WASHING_ROOM
WASHING_ROOM
BEDROOM
BALCONY
KITCHEN
CLOSET
CORRIDOR
CORRIDOR
CLOSET
CLOSET

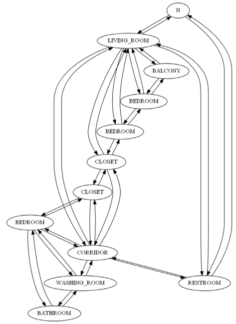


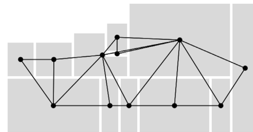

LIVING_ROOM
LIVING_ROOM
CORRIDOR
CLOSET
WASHING_ROOM
LIVING_ROOM
BATHROOM
BALCONY
BEDROOM
BALCONY
CLOSET

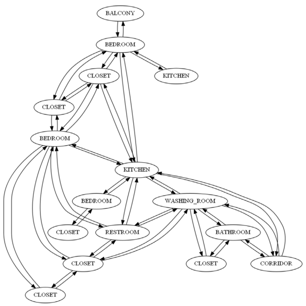


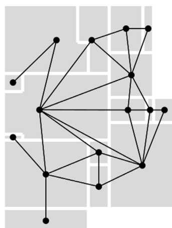

BATHROOM
CLOSET
BEDROOM
BEDROOM
CLOSET
LIVING_ROOM
BALCONY
WASHING_ROOM
WASHING_ROOM
RESTROOM
CLOSET

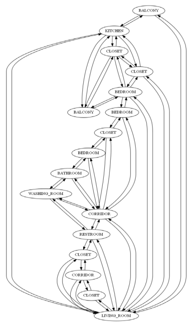


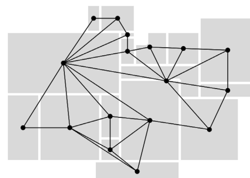

N
LIVING_ROOM
BALCONY
BEDROOM
BEDROOM
CLOSET
CLOSET
BEDROOM
CORRIDOR
WASHING_ROOM
RESTROOM
BATHROOM

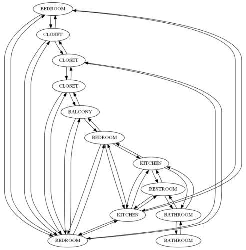


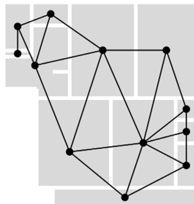

BALCONY
BEDROOM
CLOSET
KITCHEN
CLOSET
BEDROOM
KITCHEN
BEDROOM
WASHING_ROOM
CLOSET
RESTROOM
BATHROOM
CLOSET
CLOSET
CORRIDOR
CLOSET

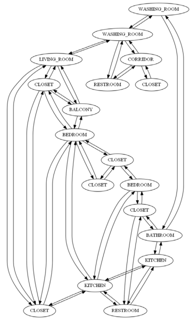


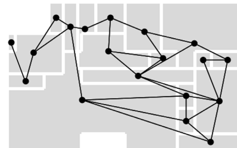

BALCONY
KITCHEN
CLOSET
CLOSET
BEDROOM
BALCONY
BEDROOM
CLOSET
BEDROOM
BATHROOM
WASHING_ROOM
CORRIDOR
RESTROOM
CLOSET
CORRIDOR
CLOSET
LIVING_ROOM

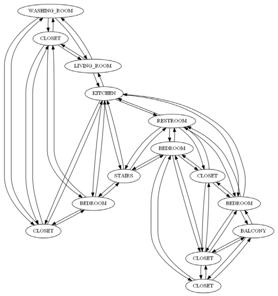


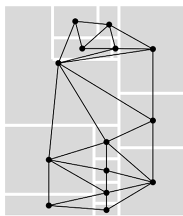

BEDROOM
CLOSET
CLOSET
CLOSET
BALCONY
BEDROOM
KITCHEN
RESTROOM
KITCHEN
BATHROOM
BEDROOM
BATHROOM

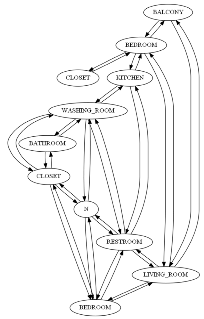


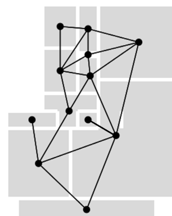

WASHING_ROOM
WASHING_ROOM
LIVING_ROOM
CORRIDOR
CLOSET
RESTROOM
CLOSET
BALCONY
BEDROOM
CLOSET
CLOSET
BEDROOM
CLOSET
BATHROOM
KITCHEN
KITCHEN
CLOSET
RESTROOM

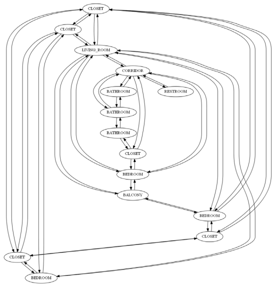


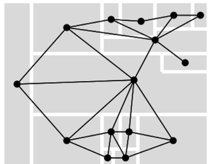

WASHING_ROOM
CLOSET
LIVING_ROOM
KITCHEN
RESTROOM
BEDROOM
STAIRS
CLOSET
BEDROOM
BEDROOM
CLOSET
BALCONY
CLOSET
CLOSET

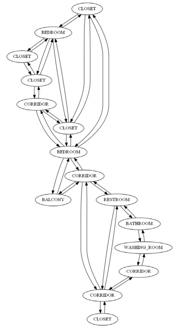


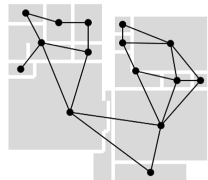

BALCONY
BEDROOM
CLOSET
KITCHEN
WASHING_ROOM
BATHROOM
CLOSET
N
RESTROOM
LIVING_ROOM
BEDROOM

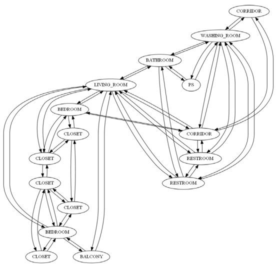


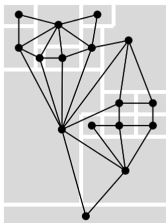

CLOSET
CLOSET
LIVING_ROOM
CORRIDOR
BATHROOM
RESTROOM
BATHROOM
BATHROOM
CLOSET
BEDROOM
BALCONY
BEDROOM
CLOSET
CLOSET
BEDROOM

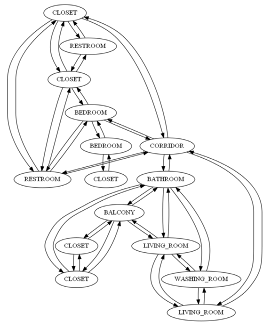


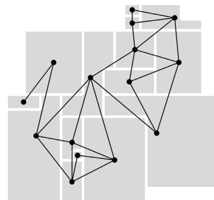

CLOSET
BEDROOM
CLOSET
CLOSET
CORRIDOR
CLOSET
BEDROOM
CORRIDOR
BALCONY
RESTROOM
BATHROOM
WASHING_ROOM
CORRIDOR
CORRIDOR
CLOSET



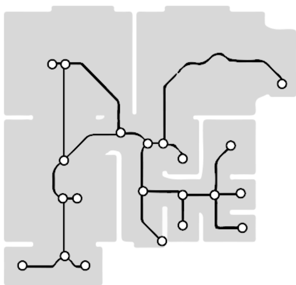

CORRIDOR
WASHING_ROOM
BATHROOM
PS
LIVING_ROOM
BEDROOM
CLOSET
CORRIDOR
CLOSET
RESTROOM
CLOSET
RESTROOM
CLOSET
BEDROOM
CLOSET
BALCONY

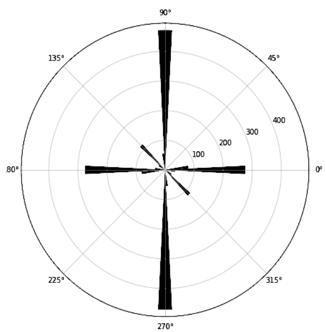



CLOSET
RESTROOM
CLOSET
BEDROOM
BEDROOM
CORRIDOR
RESTROOM
CLOSET
BATHROOM
BALCONY
CLOSET
LIVING_ROOM
CLOSET
WASHING_ROOM
LIVING_ROOM



Access Code Source

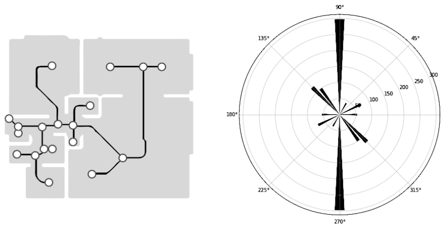


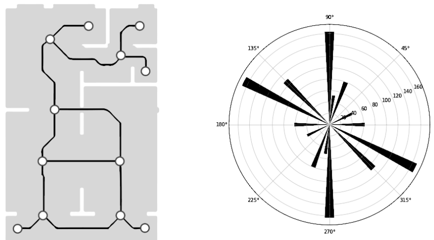


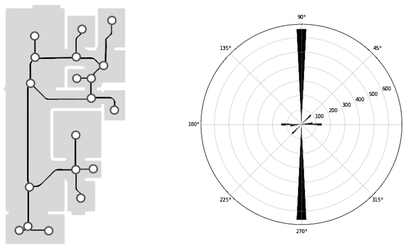
Circulation

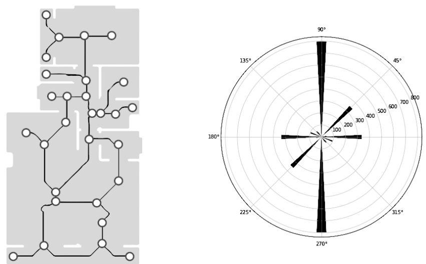
The circulation in floorplan captures how people move across it. By extracting a skeleton of the circulation, or in other words, a wireframe of the circulatory network, we can both quantify and qualify people's movement across a floorplan.

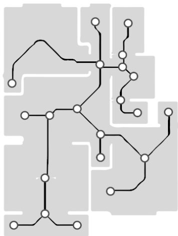
From a technical standpoint, Circulation extracts the skeleton of circulations of a given floorplan and sums its length along each direction of space, from 0 the 360 degrees. The resulting histogram is an assessment of the circulatory network geometry and can be used to be compared against other floorplans' circulation.

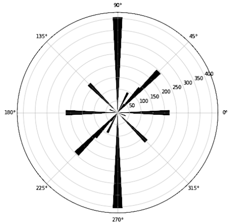

90°
135°
45°
180°
0°
225°
315°
270°
100
150
200
250
300


90°
135°
45°
180°
0°
225°
315°
270°
60
80
100
120
140
160

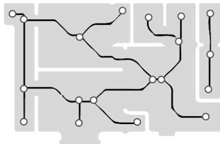

90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500
600

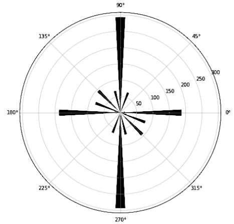

90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500
600
700
800



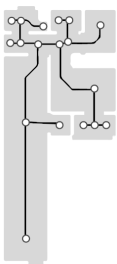


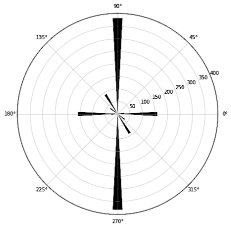
90°
45°
135°
180°
0°
225°
315°
270°
50
100
150
200
250
300
350
400



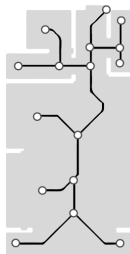


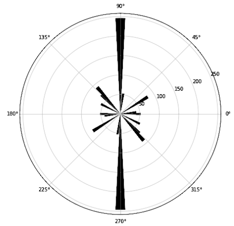
90°
45°
135°
180°
0°
225°
315°
270°
50
100
150
200
250
300



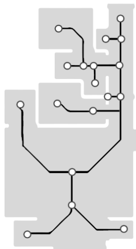


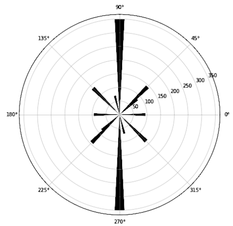
90°
45°
135°
180°
0°
225°
315°
270°
50
100
150
200
250
300
350
400



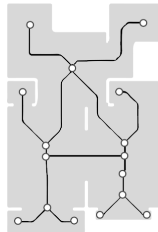


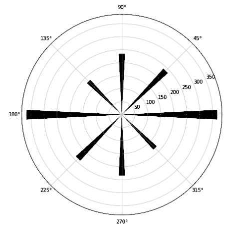
90°
45°
135°
180°
0°
225°
315°
270°
50
100
150
200
250



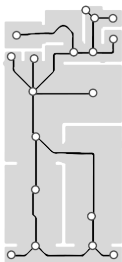


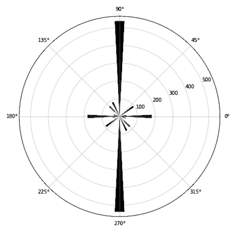
90°
45°
135°
180°
0°
225°
270°
315°
50
100
150
200
250
300
350



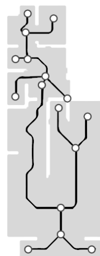


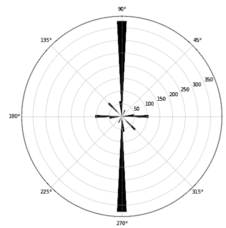
90°
45°
135°
180°
0°
225°
270°
315°
50
100
150
200
250
300
350



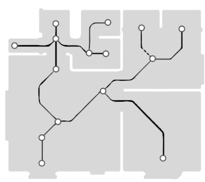


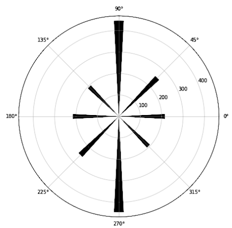
90°
45°
135°
180°
0°
225°
270°
315°
100
200
300
400
500

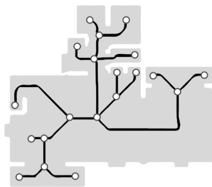


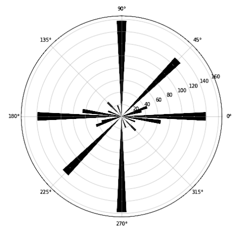


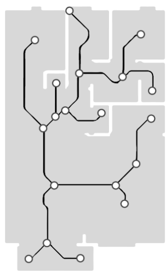
90°
45°
135°
180°
0°
225°
270°
315°
50
100
150
200
250
300
350

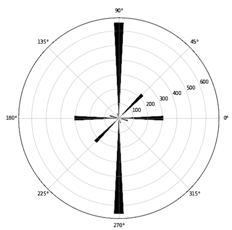


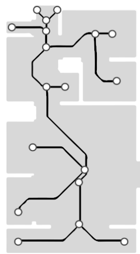

90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400

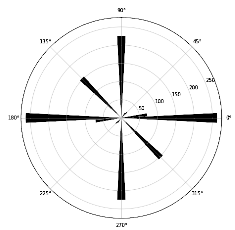


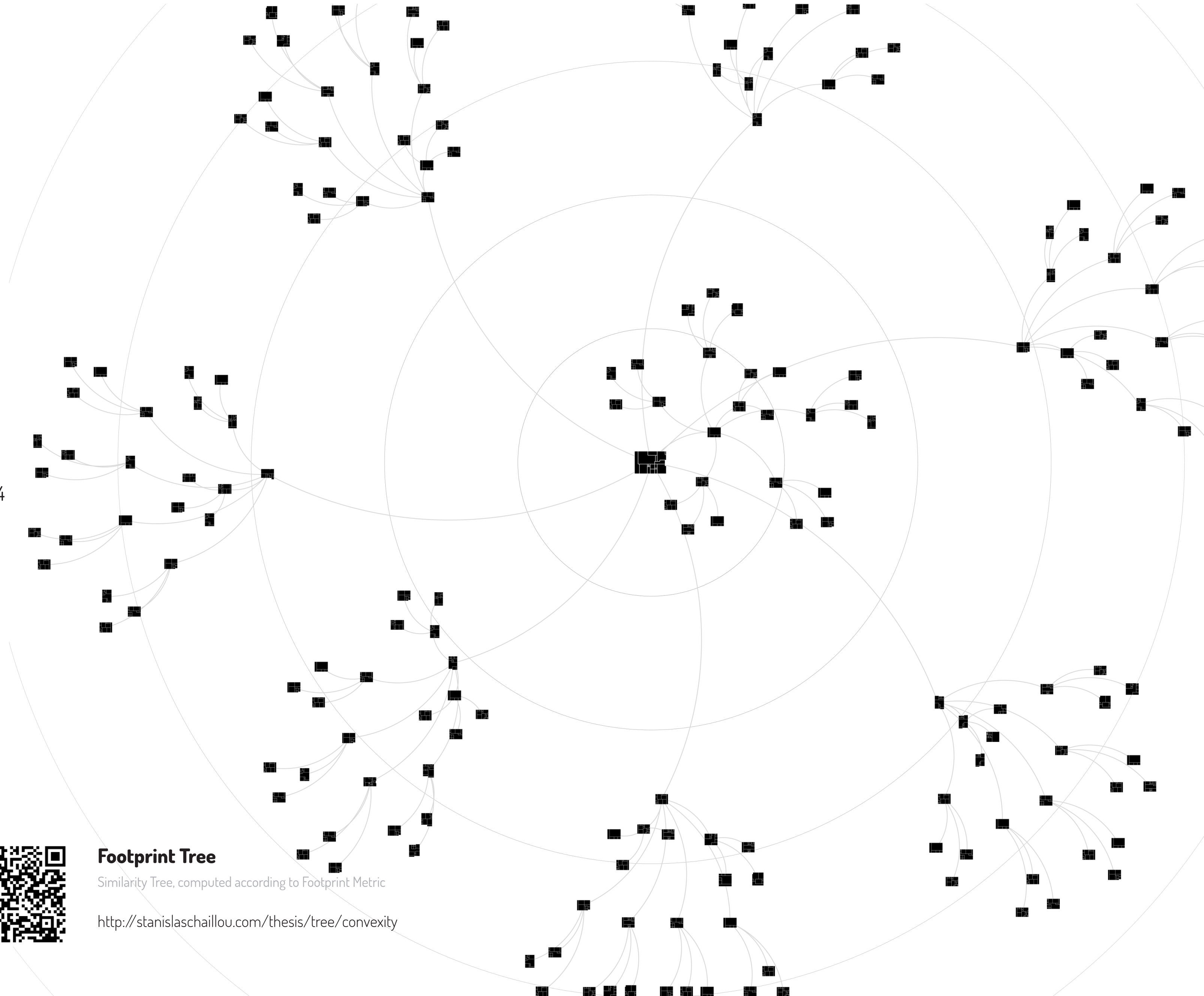

90°
135°
45°
180°
0°
225°
315°
270°
20
40
60
80
100
120
140
160




90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500
600



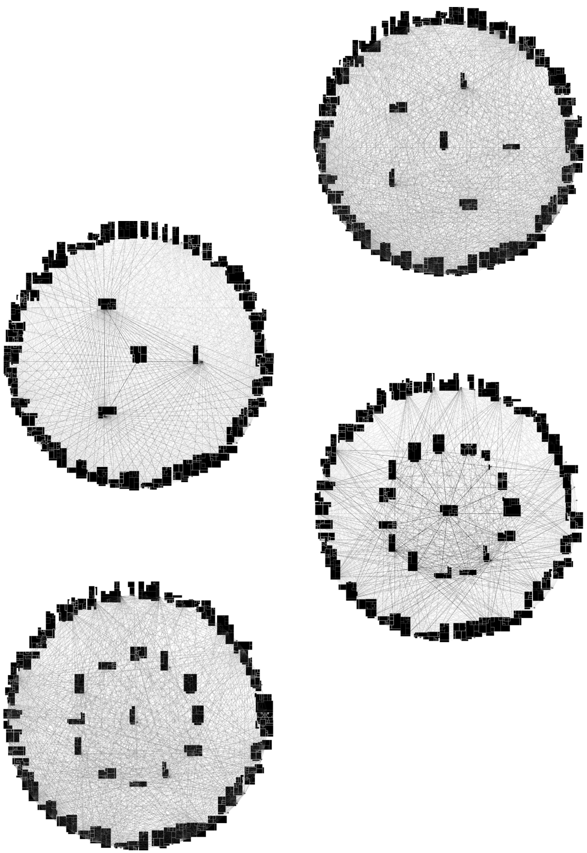

90°
135°
45°
180°
0°
225°
315°
270°
50
100
150
200
250



Footprint Tree

Similarity Tree, computed according to Footprint Metric

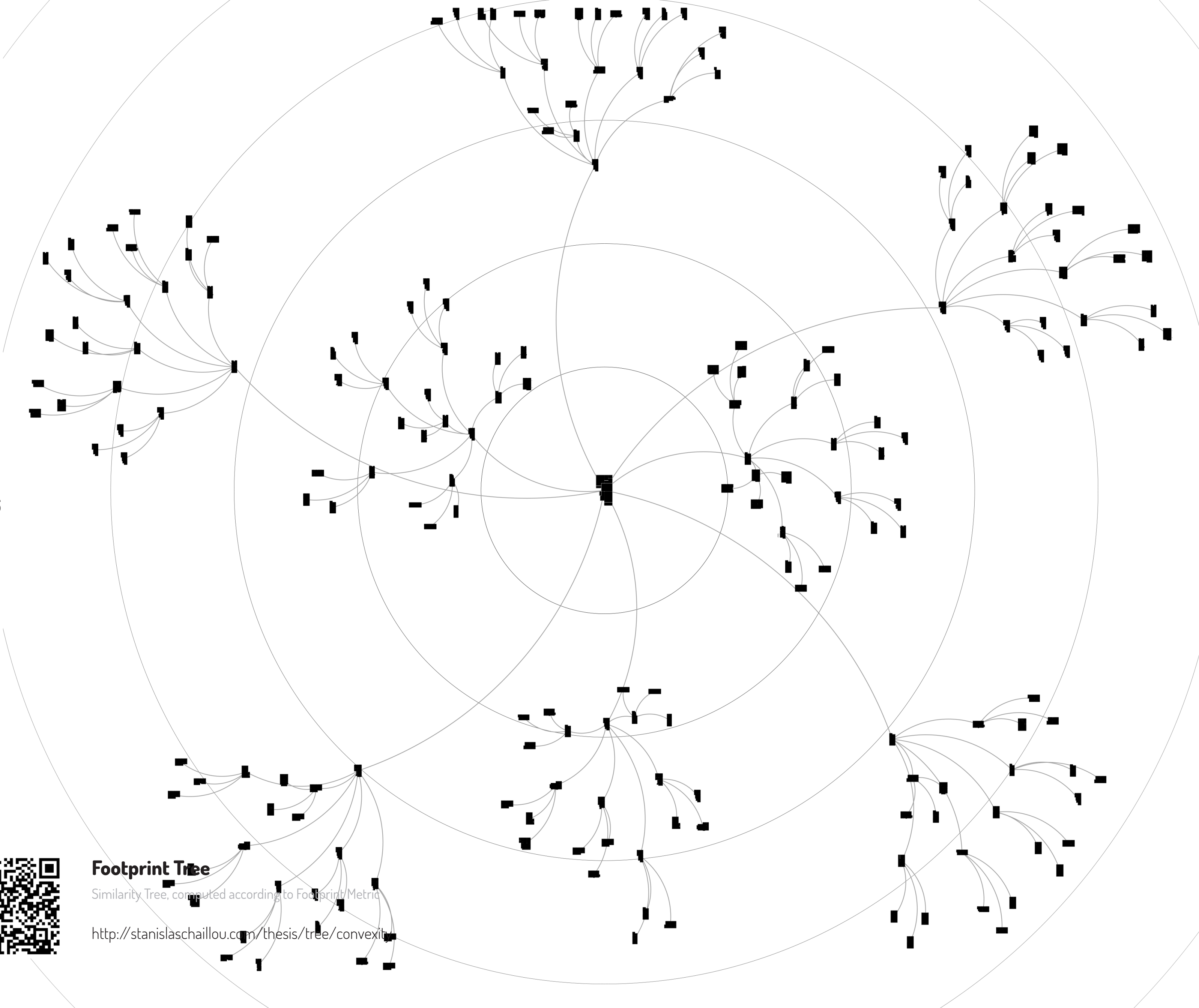
http://stanislaschaillou.com/thesis/tree/convexity



V Mapping & Browsing

Looking back at our GAN-models, each one actually outputs multiple options at each step of our generation pipeline. The designer is then invited to "pick" a preferred option, modify it if needed, before actioning the next step. Browsing through the generated options however can be frustrating, and time-consuming. To that end, the set of metrics defined in the "Qualify" chapter can demonstrate their full potential here and complement our generation pipeline. By using them as filters, the user can narrow down the range of options and find in a matter of seconds the relevant option for its design. This duality of Generation-Filtering is where the value of our work gets all the more evidenced: we provide here a complete framework, leveraging AI while staying within reach of a standard user.

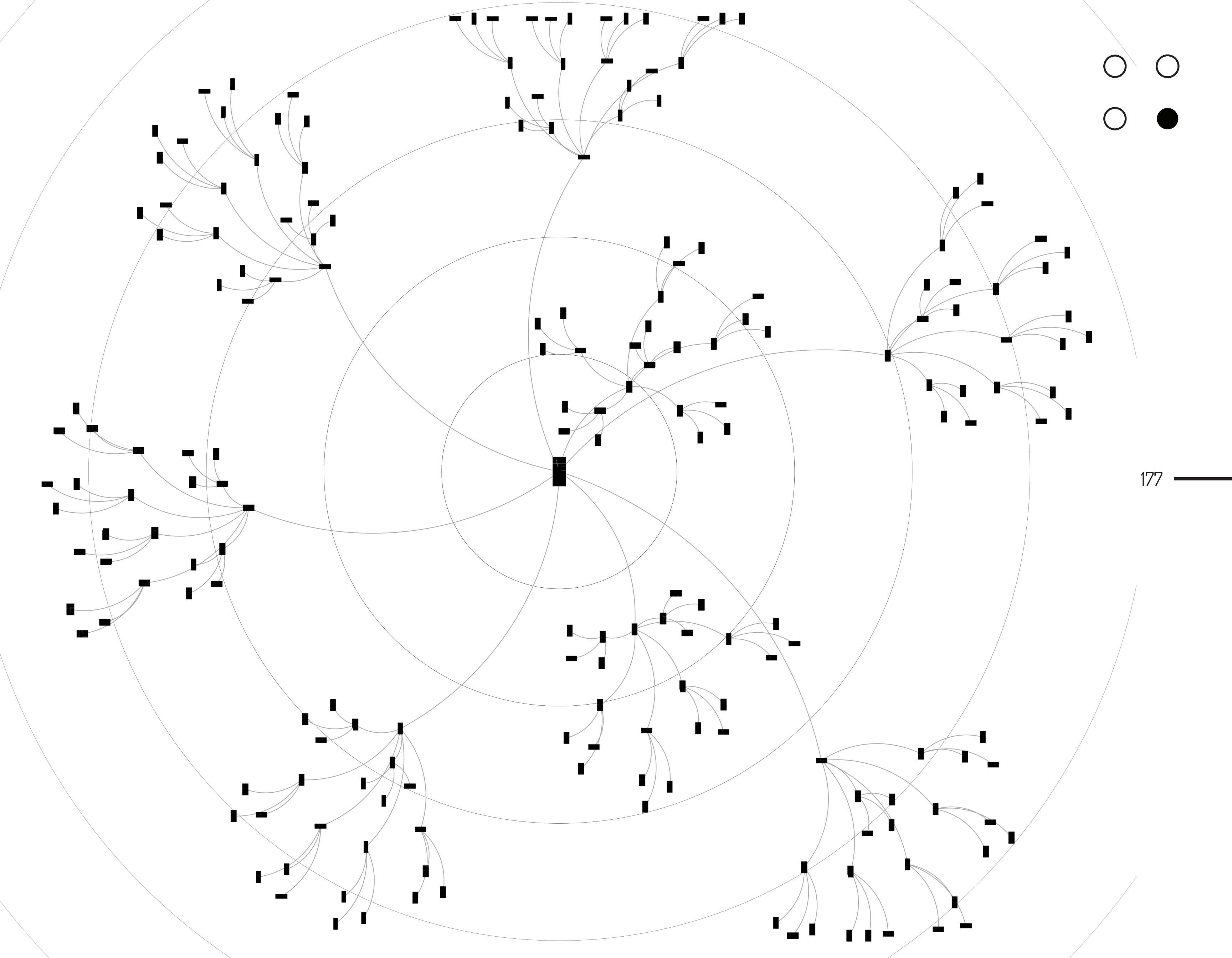
Once filtered according to a given criterion (Footprint, Program, Orientation, Thickness & Texture, Connectivity or Circulation), we provide the user with a tree-like representation of her/his choice. At the center is a selected option, and around it, its nearest neighbors classified according to a user-selected criterion. The user can then narrow down the search and find its ideal design option, or select another option within the tree, to recompute the graph.

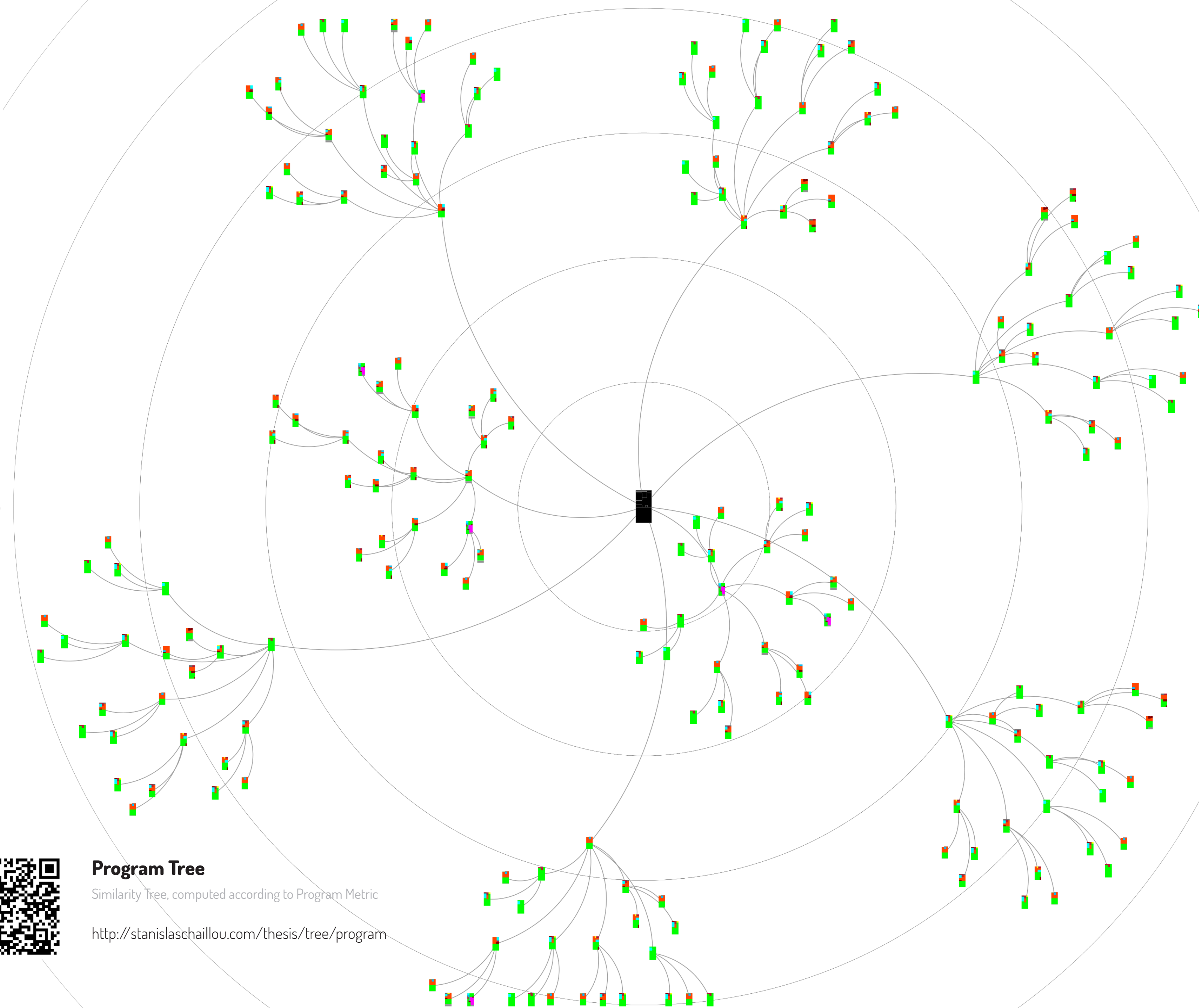


Footprint Tree

Similarity Tree, computed according to Footprint Metric

http://stanislaschaillou.com/thesis/tree/convexity

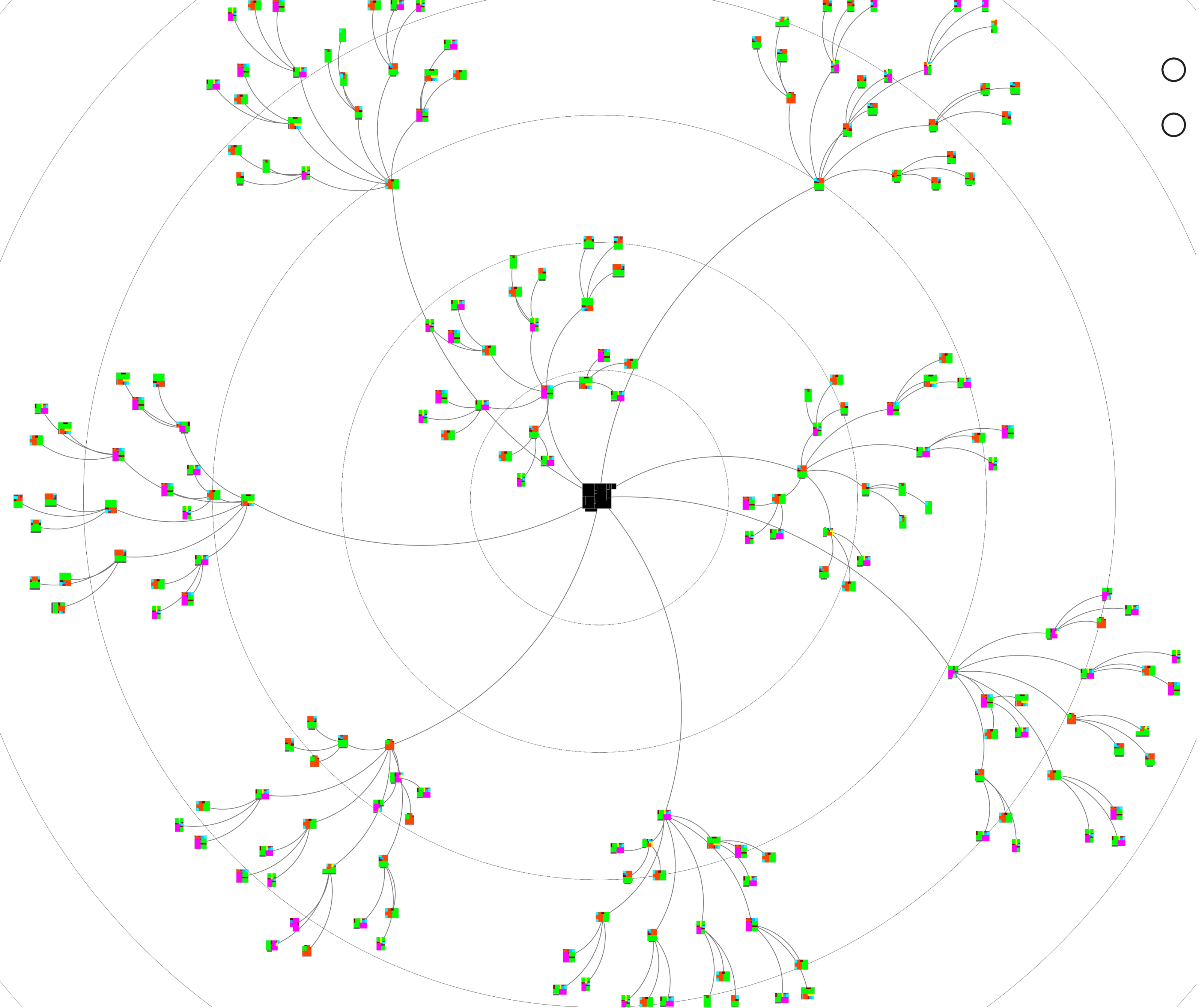




Program Tree

Similarity Tree, computed according to Program Metric

http://stanislaschaillou.com/thesis/tree/program

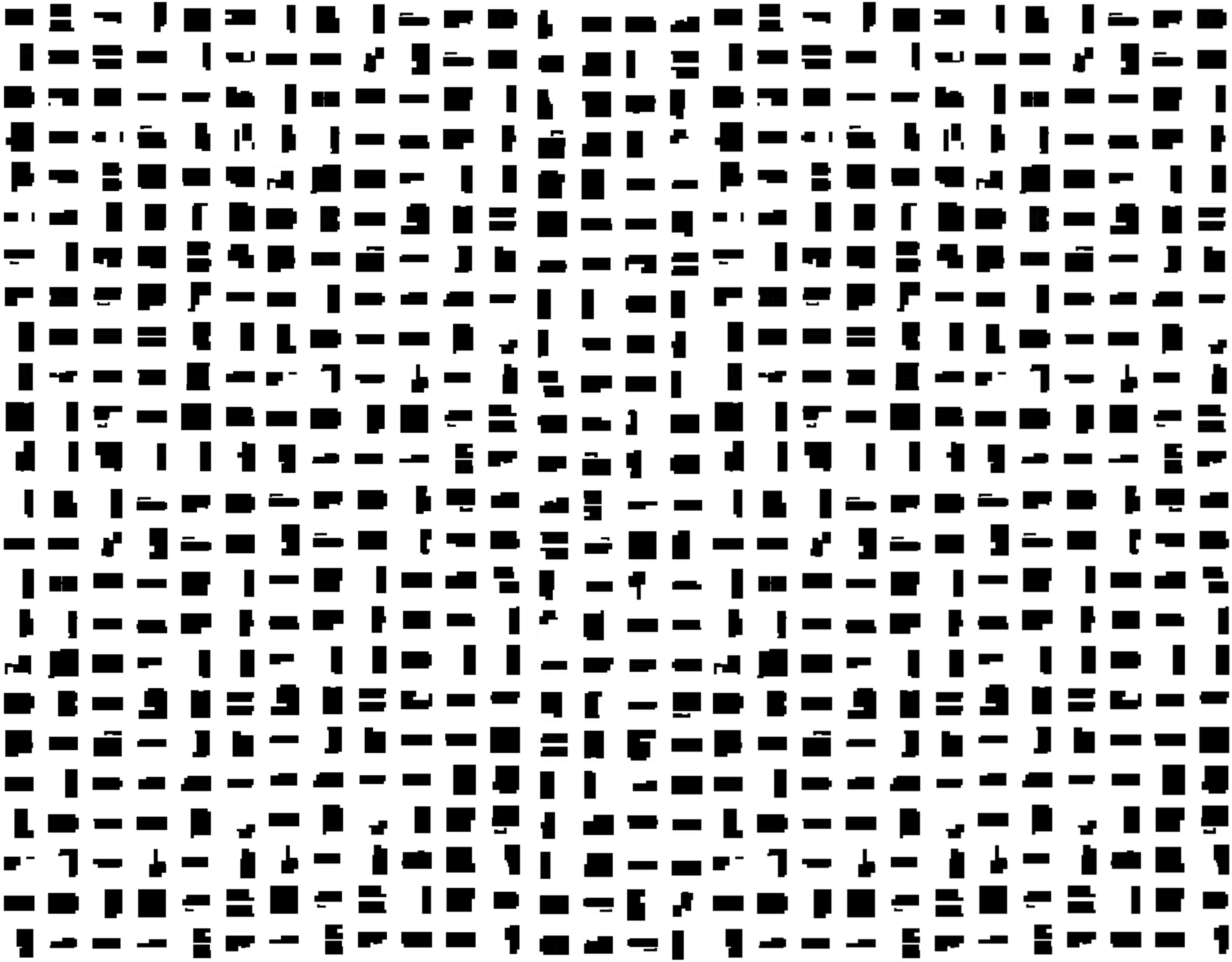


179

Connectivity Tree

Similarity Tree, computed according to Connectivity Metric

http://stanislaschaillou.com/thesis/tree/graph



Conclusion

AI will soon massively empower architects in their day to day practice. As such potential is about to be demonstrated, our work participates to the proof of concept while our framework offers a springboard for discussion, inviting architects to start engaging with AI, and data scientists to consider Architecture as a field of investigation. However, today, our manifesto could be summarized in four major points.

Conceptually first, our belief is that a statistical approach to design conception shapes AI's potential for Architecture. Its less-deterministic and more-holistic character is undoubtedly a chance for our field. Rather than using machines to optimize a set of variables, relying on them to extract significant qualities and mimicking them all along the design process is a paradigm shift.

Then, nesting models one after the other will ultimately allow **(1) to encapsulate relevant pieces of expertise, (2) while designers will be able to intervene** between each model, thus achieving the back-and-forth between humans and machines

However, AI is also the opportunity for our discipline to embrace a new set of investigative methodologies. If we applied here our GAN-models to architectural style learning, many other phenomena found in Architecture can be studied. AI simply brings new means to architectural research, allowing us to unpack the complexities found in the built environment. We see here the possibility for rich results, that will complement our practice and address some blind spots of our discipline.

Finally, Artificial Intelligence is the opportunity for Architecture to reinstate simplicity and clarity as a driving principle of our discipline. As we craft methodologies, a new ethos based on investigative rigor and scientific rationality can offer a valid intellectual anchor to architectural research. Far from the confusion that sometimes prevails in o ur field, **we have here the opportunity to build a new sound foundation for the architectural discourse.** It is a necessity that will ultimately bring back Architecture where it belongs: as an Art & a Science, a product of intellectual sensibility, and technical rigor.

[redacted]

Theoretical References

Digital Architecture Beyond Computers, Roberto Botazzi, Bloomsbury

Data-Driven Design & Construction, Randy Deutsch, Wiley

Architectural Intelligence, How Designers and Architects Created the Digital Landscape, Molly Wright Steenson, MIT Press

Architectural Google, Beyond the Grid—Architecture & Information Technology pp. 226-229, Ludger Hovestadt, Birkhauser

Algorithmic Complexity: Out of Nowhere, Complexity, Design Strategy & World View pp. 75-86, Andrea Gleiniger & Georg Vrachliotis, Birkhauser

Code & Machine, Code, Between Operation & Narration pp. 41-53, Andrea Gleiniger & Georg Vrachliotis, Birkhauser

Gropius' Question or On Revealing And Concealing Code in Architecture And Art, Code, Between Operation & Narration pp. 75-89, Andrea Gleiniger & Georg Vrachliotis, Birkhauser

Soft Architecture Machines, Nicholas Negroponte, MIT Press

The Architecture Machine, Nicholas Negroponte, MIT Press

A Pattern Language, 1968, Christopher Alexander

Notes on the Synthesis of Form, 1964, Christopher Alexander

Cartogramic Metamorphologies; or Enter the RoweBot, Andrew Witt, Log #36

Grey Boxing, Andrew Witt, Log #43

Suggestive Drawing Among Human and Artificial Intelligences, Nono Martinez, Harvard GSD Thesis, 2016

Enabling Alternative Architectures : Collaborative Frameworks for Participatory Design, Nathan Peters, Harvard GSD Thesis, 2017

Technical References

DANIEL: A Deep Architecture for Automatic Analysis and Retrieval of Building Floor Plans, Divya Sharma,Nitin Gupta,Chiranjoy Chattopadhyay,Sameep Mehta, 2017, IBM Research, IIT Jodhpur

Automatic Room Detection and Room Labeling from Architectural Floor Plans, Sheraz Ahmed, Marcus Liwicki, Markus Weber, Andreas Dengel, 2012, University of Kaiserslautern

Generative Adversarial Nets, Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, June 2014 ,Universite de Montreal

Pix2Pix, Image-to-Image Translation with Conditional Adversarial Networks, Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros

Automatic Interpretation of Floorplans Using Spatial Indexing, Hanan Samet, Aya Soffer, 1994, Univeristy of Maryland

Parsing Floor Plan Images, Samuel Dodge,Jiu Xu ,Bjorn Stenger, 2016, Arizona State University, Rakuten Institute of Technology

Project Discover: An Application of Generative Design for Architectural Space Planning, Danil Nagy, Damon Lau, John Locke, Jim Stoddart, Lorenzo Villaggi, Ray Wang, Dale Zhao and David Benjamin, 2016, The Living, Autodesk Studio

Raster-to-Vector: Revisiting Floorplan Transformation, Chen Liu,Jiajun Wu,Pushmeet Kohli,Yasutaka Furukawa, 2017, Washington University, Deep Mind, MIT

Relational Models for Visual Understanding of Graphical Documents. Application to Architectural Drawings, Llus-Pere de las Heras, 2014, Universitat Autonoma de Barcelona

Shape matching and modeling using skeletal context, Jun Xie,Pheng-Ann Heng,Mubarak Shah, 2007, University of Central Florida,Chinese University of Hong Kong

Statistical segmentation and structural recognition for floor plan interpretation, Lluís-Pere de las Heras, Sheraz Ahmed, Marcus Liwicki, Ernest Valveny, Gemma Sánchez, 2013, Computer Vision Center, Barcelona, Spain

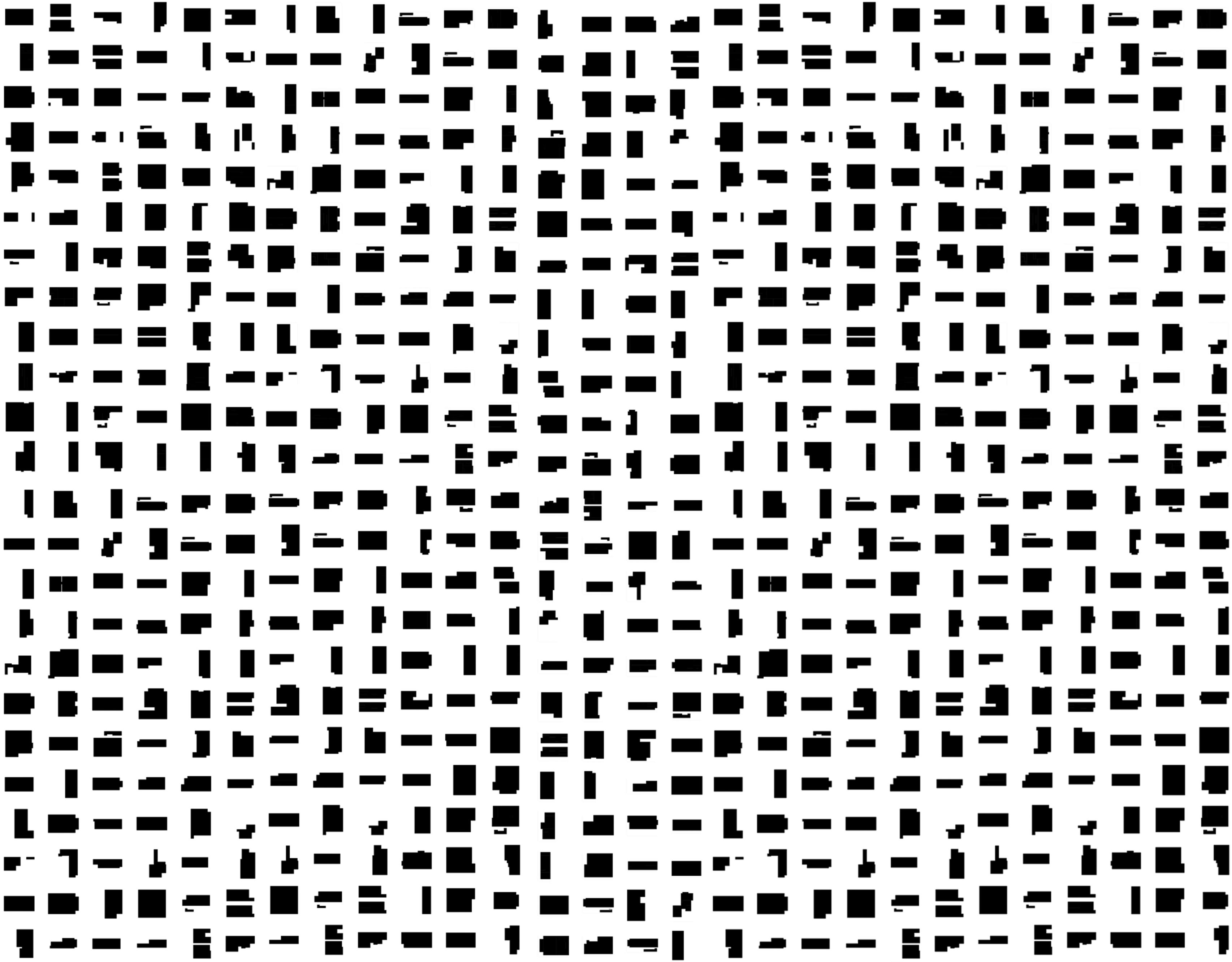
Unsupervised and Notation-Independent Wall Segmentation in Floor Plans Using a Combination of Statistical and Structural Strategies, Lluıs-Pere de las Heras, Ernest Valveny, and Gemma Sanchez, 2014, Computer Vision Center, Barcelona, Spain

Path Planning in Support of Smart Mobility Applications using Generative Adversarial Networks, Mohammadi, Mehdi, Ala Al-Fuqaha, and Jun-Seok Oh. , 2018

Automatic Real-Time Generation of Floor Plans Based on Squarified Treemaps Algorithm, Fernando Marson and Soraia Raupp Musse, 2010, PUCRS

Procedural Modeling of Buildings, Pascal Muller,Peter Wonka,Simon Haegler,Andreas Ulmer,Luc Van Gool, 2015, ETH Zurich, Arizona State University

Generative Design for Architectural Space Planning, Lorenzo Villaggi and Danil Nagy, 2017, Autodesk Research



Thesis Website
Main Website of Thesis Project

http://stanislaschaillou.com/expliquer

Github Repository | Github
Main Code Repository of Thesis Project

https://github.com/StanislasChaillou/OpenPlan

Connectivity Tree | Visualization
Similarity Tree, computed according to Connectivity Metric

http://stanislaschaillou.com/thesis/tree/graph

Program Tree | Visualization
Similarity Tree, computed according to Program Metric

http://stanislaschaillou.com/thesis/tree/program

Footprint Tree | Visualization
Similarity Tree, computed according to Footprint Metric

http://stanislaschaillou.com/thesis/tree/convexity

Mapping Similarity | Notebook
Maps and Visualizations of floor plans similarities

https://github.com/StanislasChaillou/OpenPlan/blob/master/Mapping_Similarity.ipynb

Footprint Metric | Notebook
Algorithm & Applications of Footprint Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Footprint.ipynb

Connectivity Metric | Notebook
Algorithm & Applications of Connectivity Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Connectivity.ipynb

Orientation Metric | Notebook
Algorithm & Applications of Orientation Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Orientation.ipynb

Program Metric | Notebook
Algorithm & Applications of Program Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Program.ipynb

Thickness & Texture Metric | Notebook
Algorithm & Applications of Thickness & Texture Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Thickness_Texture.ipynb

Circulation Metric | Notebook
Algorithm & Applications of Circulation Metric

https://github.com/StanislasChaillou/OpenPlan/blob/master/Circulation.ipynb

AI + Architecture

Towards a New Approach

Stanislas Chaillou | Harvard GSD | 2019

Harvard University
Graduate School of Design