A novel design technique for generating building form

تقنية جديدة لتصميم شكل المبنى

by

MAYS N. A. KAYED

Dissertation submitted in fulfilment

of the requirements for the degree of

MSC SUSTAINABLE DESIGN OF THE BUILT ENVIRONMENT

at

The British University in Dubai

November 2018

DECLARATION

I warrant that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.

I understand that a copy of my research will be deposited in the University Library for permanent retention.

I hereby agree that the material mentioned above for which I am author and copyright holder may be copied and distributed by The British University in Dubai for the purposes of research, private study or education and that The British University in Dubai may recover from purchasers the costs incurred in such copying and distribution, where appropriate.

I understand that The British University in Dubai may make a digital copy available in the institutional repository.

I understand that I may apply to the University to retain the right to withhold or to restrict access to my thesis for a period which shall not normally exceed four calendar years from the congregation at which the degree is conferred, the length of the period to be specified in the application, together with the precise reasons for making that application.

____

Signature of the student

COPYRIGHT AND INFORMATION TO USERS

Abstract

This dissertation sets out to develop a novel design technique for generating building form in the early stage of the design. It recognizes generative design approaches potential to explore form integrally with instant evaluation process and questions the potential of enabling the generation of creativity aspects of the design process in such trajectory.

The development of the technique is initiated by proposing an exploratory methodology where a procedural structure for the technique is formulated from four stages: "***backward triangulation***" stage, forward building formation stage, evaluation stage and selection stage. Each stage contains number of processes that replace the traditional design tasks. The development of the technique extended by developing mechanisms to maintain aspects related to design creativity by pure geometric abstraction of basic spatial relations.

The theoretical framework of the technique was successfully tested out and validated by implementing it over serious of examples that are categorized into three groups: examples directed to test out the proposed variables in each process, examples directed to validate the ability of whole technique to generate forms for a hypothetical project requirements and examples of scripting the mechanisms using the algorithmic editor interface and the visual programming language Grasshopper incorporated with Rhinoceros 3D software.

ملخص

يهدف هذا البحث الى تطوير تقنية جديدة لتصميم شكل المبنى في المراحل المبكرة من عملية التصميم. ويوظف من أجل ذلك منهج التصميم التوليدي, وهو منهج جديد يعتمد على مبدأ التصميم الحسابي وعلى الخوارزميات. يتميز هذا المنهج في عملية التصميم بإمكانية التقييم اللحظي لعدة متغيرات لها تأثير كبير على أداء و جودة التصميم, حيث يصعب تقييمها في هذه المراحل المبكرة من التصميم بإستخدام مناهج التصميم الاخرى، وبالتالي تحسين التصميم وتقليل التكلفة والوقت والجهد. كما يُحقق هذا البحث عن إمكانية استحداث خوارزميات تستبدل مهام التصميم المتعلقة بالإبداع البشري.

يبدأ البحث بتعريف المنهج التوليدي في التصميم والاستراتجيات الحالية التي طورها الباحثون لتوظيف هذا النهج لأتمتة بعض مهام التصميم . بناءا على ذلك, تم تحديد التحديات الاساسية التي تواجه تطور هذا الاستراتيجيات وانتشارها. يصيغ البحث التقنية المقترحة بإطار نظري مكون من أربع مراحل أساسية: مرحلة تكوين فكرة التصميم "***backward triangulation***" ، مرحلة تكوين المبنى ، مرحلة التقييم ومرحلة الاختيار .حيث تحتوي كل مرحلة على عدد من العمليات التي تحل محل مهام التصميم التقليدية .كما ان عملية تطوير التقنية شملت تطوير آليات لأتمتة مهام التصميم المتعلقة بالإبداع عن طريق التجريد الهندسي التام للعلاقات المكانية الأساسية.

تم اختبار الإطار النظري للتقنية بنجاح حيث تم التحقق من صحته من خلال تطبيقه على ثلاث مجموعات من : أمثلة موجهة لاختبار المتغيرات المقترحة في كل عملية ، أمثلة موجهة للتحقق من قدرة التقنية ككل على توليد تصاميم لمتطلبات مشروع افتراضي وأمثلة من البرمجة النصية للاليات المقترحة باستخدام لغة البرمجة البصرية Grasshopper والبرنامج الاخراج الحاسوبي Rhinoceros 3D.

For the memory of my beloved grandmother, Thuraiah,

Acknowledgment

This dissertation is the final outcome of a two year master's program in Sustainable Design of the Built Environment. Through this long journey there have been many people to whom I owe deeply for their support, patience and encouragement, their kind and persistence help is what made the completion of this dissertation possible.

First and Foremost, I owe my wholehearted gratitude to my professor, mentor, and main advisor, Dr. Riad Saraiji, who I was fortunate to work with and have his unfailing guidance, expertise and insightful critique; His vast knowledge, dedication to his work and students and his enthusiasm were essential to the development of this dissertation.

I would also like to extend the thanks to my professors in the faculty of Engineering & IT at the British University in Dubai, Prof. Bassam Abu-Hijleh and Dr. Hanan Taleb, as I have learned a lot from them throughout my study period.

To my mother, my biggest advocate and companion, it is with profound love that I thank you for always being there for me and for pushing me all the way through this road.

To my father, for his unlimited faith in me, and for always encouraging me to have a critical mind and an expanded curiosity, you will always have my deepest appreciation and love.

For my dear friend, Suzanne, for all these late nights study sessions, and for English proofing this dissertation, I'm forever indebted for the incredible time spent, the quality of your review, your brutal honesty, and your helpful comments.

Lastly I would like to thank my family and friends for enduring me during these times and for their care and support.

List of content

List of figure

List of tables

Abbreviations

GD	Generative design
SAP	Space allocation problems
MSAP	Multi-floor space allocation problems
SA	Simulated annealing
GA	Genetic algorithm
CFSQP	C Feasible Sequential Quadratic Programming
CA	Cellular automata
IA	Intelligent artificial
CAD	Computer-aided design
SMG	Specialist modelling group
LEED	Leadership in Energy and Environmental design
N_{lines}	Number of lines
D	Distance
r	Radius
BIM	Building information modelling

CHAPTER ONE

INTRODUCTION

Chapter 1 Introduction

1.1 Background information

Design practice in architectural design has been redefining its boundaries, its methods and approaches under the influence of technological development revolution. The fast increasing of computer processing power has proliferated the interest in two main concepts: computerization and computation. Computerization of design process is directly related to the use of computer as a design tool that allows a high level of flexibility and accuracy in performing some design tasks such as drafting, modelling and visualizing (Menges & Ahlquist 2011). While computation is the articulation of the underlying logic of a process or mechanism and addressing a design problem by finite number of steps or algorithm -that is a common language between humans and the machine-computer (Kolarevic 2009). Even though the use of computation in architecture started long before the invention of computers, a convenient way for implementing computational methods has been only provided by computer technologies.

A search of new ways to harness computation has raised the possibilities of growing the design process into a dynamic exploration process that is being called "generative design". The term *generative design* requires a higher level of computation that includes encoding of large number of constraints and the formulation of relationships between design components. In this way, designers become able to explore design situation quickly by navigating in a larger design search space with many alternatives (Krish 2011).

Taken into consideration that the term computation indicates the information processing which does not necessarily generates solutions, generative design refers to a computational method only if the method is able to generate design outcomes automatically (Herr & Arch 2002).

Generative design suggests a fundamental change in the role of the designer (architect) from designing an object to designing the logic behind it and translating it into a set of rules organized by operation sequence. While design procedure in architecture is characterized by the methodology adopted by the designer to achieve defined goals, the outcomes of the process depends on designer's experience and skills used to solve the increasing complexity during design evolution that accrues due to the emerging requirements and problems. This case was dramatically changed by the concept of generative design and that is not due to the utilization of computers to operate the process, in fact that the process is possibly operated by humans. However, in generative design approach, the focus on solving design problems shifts from the design outcomes to include a broader understanding of the notion of design as well as the design process itself (Dİno 2012).

Many generative design systems have been successfully developed to replace some specific design tasks, nevertheless, they failed in finding their border implementations in the architectural design practice. This can be related to the goals that the development of each generative design system is led by, which mostly falls between performances optimization and usability evaluation. The potential of using a generative design approach in the design procedure defiantly exceeds its current applications, to this end, it seems that there is a lack in utilizing the concept of generative design to produce a digital design assistant system or a design tool that helps the architects in the early stages of design process and provides them with design alternatives and solutions.

1.2 Research objectives and aims

The objective of this dissertation is:

- Developing a design algorithm based on generative design approach for generating and automating the concept design of a building form. An algorithm that enables the exploration of many design solutions or ideas according to a defined set of requirements.

The development of such an algorithm requires a deep understanding of the different types of generative design systems as well as breaking down the generative aspects in the normal design procedure that can be encoded to algorithms. This can only be achieved by:

- Examining the existing generative design approaches and identifying the main challenges that limit their implementations.
- Identifying the potential and implications of generative design system for generating building form.
- Establishing an architectural framework for adopting generative design approach for building form exploration process.

1.3 Dissertation contents and structure

This dissertation consists of six chapters:

Chapter 1 (Introduction): This chapter contains an overview of the research topic as well as its main objectives.

Chapter 2 (Literature review): This chapter is divided into three main sections. First section is primarily concerned with design procedure and design thinking, more specifically, the different design methodologies developed during that time. The main aim of this section is to discuss design as ill-structured problem and highlight the need of systemizing the design process in the field of architecture.

While the second section presents a comprehensive review of the previous research and practice in the field of architecture that explores building form by adopting a generative design approach. The section starts with an introduction and an explanation of generative design concepts, systems, techniques and general implementations. Then the discussion focuses on generative design strategies used to generate architectural forms. At the end of the chapter, the main challenges and limitation facing the current development of generative design approaches in architectural field are identified.

Chapter 3 (Methodology): This chapter discusses the methodology that is adopted for achieving the objectives of this dissertation in four sections; the first section presents research approach; while the second section describes the methodological framework of developing the envisioned technique for generating building forms followed by a summary of the technique. The final section discusses how the envisioned technique is validated and the software/ computer program used in the process.

Chapter 4 (A novel design technique for generating building forms): This chapter discusses and describes the proposed technique stage by stage and unfolds the underlying logic behind its organizational structure. The four stages of the technique are described in detail by required input, expected output, design tasks, variables and constraints.

Chapter 5 (Results and discussion): This chapter illustrates the implementation of the proposed generative design technique for conceptual building form exploring in a series of design examples. The chapter investigates the technique capability in formulating diverse design spaces and generating unexpected design possibilities that are functional.

Chapter 6 (conclusion): The conclusion brings together the outcomes in the view of the scope of this dissertation, both in terms of generative design approach from architectural perspective and more detailed aspects concerning the developed technique. In addition, it shows the main contributions of this dissertation to the existing knowledge and proposes further research areas related to the topic.

Chapter two

Literature review

Chapter 2 Literature review

This chapter presents a review on relevant literature within the field of generative design to highlight the current state of art and identify a research gap.

The chapter is divided into three main sections. First section is primarily concerned with design procedure and design thinking, more specifically the different design methodologies developed during the time. The main aim of this section is to discuss design as ill-structured problem and highlight the need of systemizing the design process in architecture field.

The second section presents a comprehensive review of the previous research and practice in the architecture field that explored building form by adopting a generative design approach. The section starts with an introduction on generative design where generative design concepts, systems, techniques and general implementations are explained. Then the discussion focuses on generative design strategies used to generate architectural forms.

At the end of the chapter, the main challenges and limitation facing the current development of generative design approaches in architectural field are identified with respect to technical and design factors.

2.1 Design thinking and design research

The term "Design" has been described in literature by its dual meaning. It refers to the act of designing as well as to the final product or the end result of a design activity (Dino 2012). Many researchers have attempted to formulate a concrete definition of design such as Matchett (1968), Archer (1979) and Goel & Pirolli (1992). However, none of them have successfully captured a common essence without facing some fundamental contradictions. Lawson (2006) in his book ***"how designer think: the design process demystified"*** explains that design as an activity or a way of thinking is very divers across different disciplines which makes it hard to identify common traits. Lawson continued the discussion of this confusion in identifying design by presenting a number of comprehensive definitions suggested by profession in design science field. He concluded that the definitions either held some assumptions that cannot be applied on all disciplines or the definition is too generic that became useless.

In general, design can be understood by analysing its multifaceted nature. According to Archer (1969), design has two main elements: logical element and creative element. In one side, linking design to creativity suggests treating it as a way of thinking where the focus became on distinguishing thinking modes. This can be traced to Bryan Lawson (1979) experiments on two groups of people, one consists of scientists and the other consists of architects. Both were given a "design-like" problem to solve. Lawson observed a completely different approach adopted by the scientists group in comparison with the architects group. The scientists focused on analysing and structuring the problem, while the architects focused on the final outcome by proposing solutions and modifying them accordingly. Scientist approach was considered more effective, however, it limits solution domains.

On the other hand, the logical element recognizes design as a problem-solving activity. Generally, problem-solving activity requires a linear process where all possible next moves are explored and evaluated (Lawson 2006). According to Simon (1973), problem solving activity deals with two types of problems: well-defined problems and ill-defined or ill-structured problems. Jonassen (1997) has conducted a comparison between the two types by collecting and analysing the definitions and properties of ill-structured problems proposed by researchers. He explained that ill-structured problems are multi-objective and subjective making it hard to be described in a limited number of rules. This also makes ill-structured problems' results introduce conflicting values which relate to design problems since they are ambiguous, changeable and contain confusing information.

In problem solving activity, "Problem space" refers to the set of components that organize the process. It includes defining the problem's state along with the goals and a set of operators. For design problems, "problem space" is known as design schema (Jonassen 2000). While design search space refers to all the possible solutions for a design problem. The diversity of the design search space is a direct result of design exploration process (Braha & Maimon 1997). In the traditional design process, the designer forms the design space manually by exploring and evaluating number of design solutions using one of two approaches suggested by Cross (2011): depth-first approach and breadth-first approach.

Depth-first approach is a trial and error approach that is associated with the novice designers. In this approach, one solution is evolved and explored during the whole process, accordingly, the design space is narrow. In opposite, breadth-first approach depends on evaluating and exploring many sub-solutions which allows the exploration of various alternatives (Cross 2011).

Design procedure in architecture is characterized by the methodology adopted by the designer to achieve defined goals. Lowgren (1995) identified three generation of design methodology, started in 1950s, to emphasize the development of engineering approaches to design. The first generation of design methodology attempt to simplify design work by approaching it as object-oriented and problem-solving process that mainly consists of three phases: analysis, synthesis and evaluation. This generation concentrated on developing design methods with giving the highest weight to rationality aspects that is highlighted in the way of gathering and structuring information.

The second generation of design methodology stood against what was called "machine language" of the first generation. The pioneers of the second generation argued that creating logical frameworks for design limits designer role to the role of expert, however, the designer should be as a liberator who explores the needs and requirements of the design. Furthermore, identifying design problems as "wicked" or "ill-structured" problems has influenced the second generation to shift design methodologies into the concept of "satisficing". The concept recognized design process as a process of recognition appropriate solution types instead of problem-solving activity (Lowgren 1995).

The third generation according to Cross (1993) is a combination of the two previous generations which started in the 1990s. This generation was driven by the interest of intelligent design, design automation and the use of the machine to assess the designer. Lowgren (1995) pointed out that the third generation of design methodology in the architectural field dealt with design as a way of thinking and connected it to other disciplines such as philosophy and art.

2.2 Generative design

Introducing computation in design process has allowed the utilization of computers power in information processing to overcome human limitations. It played out a significant role in producing digital tools for drafting and documentation that increased the efficiency of design process in mater of time and effort (Oxman 2008).

Research efforts to employ computation for the purpose of automating design process have led to the birth of dynamic approach for design exploration that is been called “generative design”. Generative design allows the inclusion of a large number of goals and constraints in the design process by formulating and encoding relationships between design components using computation. In this way, designers become able to explore multiple design solutions quickly by navigating in a larger design search space (Krish 2011).

Generative design can be defined as a process or an approach of generating various design solutions using algorithms (Herr & Arch 2002). Terzidis (2004) defines an algorithm as “*a computational procedure for addressing a problem in a finite number of steps*”. The definition of algorithm is tightly combined with the concept of computation where computation describes the process of performing the algorithm. Taken into consideration that the term computation indicates the information processing which does not necessarily generates solutions, generative design refers to a computational method only if the method is able to generate design outcomes automatically (Herr & Arch 2002).

Generative design suggests a fundamental change in the role of the designer (architect) from designing an object to designing the logic behind it and translating it into a set of rules organized by operation sequence. While design procedure in architecture is characterized by the methodology adopted by the designer to achieve defined goals, the outcomes of the process depends on designer’s experience and skills used to solve the increasing complexity during design evolution that accrues due to the emerging requirements and problems. This case was dramatically changed by the concept of generative design and that is not due to the utilization of computers to operate the process, in fact that the process is possibly operated by humans. However, in generative design approach the focus in solving design problems shifts from the design outcomes to include a broader understanding of the notion of design as well as the design process itself (Dino 2012).

Algorithmic systems are considered the base case of generative systems. More specialized versions became well known in the architectural field such as shape grammar, cellular automata and genetic algorithmic. Despite the differentiation, they all consist of component, relations and an environment. The components range from shapes, symbols, cells, agents and genotypes creating a diverge use of such components in the field (Singh & Gu 2012).

These systems proposed a new digitalized way of form finding, for examples, shape grammar generates forms by searching on similarities from database (architectural vocabulary) that was developed by the architect. Although many applications of forms finding where developed based on the concept of shape grammar, the results of these applications are limited by specific architectural style (Colakoglu & Keskin 2010). However, many advantages were identified for utilizing generative design for form finding in architecture such as: the ability of generation of ever-growing number of alternatives and providing an automated evaluation process for the generated solutions (Grobman, Yezioro & Capeluto 2009).

2.2.1 Generative Form finding

The interest in generating forms using computational methods was driven by questioning how raw information can be processed with considering form, function and context together (Agkathidis 2015). Taking into consideration that computation allowed in-moment evaluation of a set of criteria decided by the designer, the form stepped out from being after-the-act task. Here, performance and form are seen as one element for solving the design problem. Grobman, Yezioro & Capeluto (2009) explain that in this way, any process after form generation is categorized as optimization process that will not require any fundamental changes on the design itself. The computer as an intelligent machine had the promise of generating an optimal solution if the design can be fully automated. The implementation, however, for the purpose of generating building forms has faced the challenge of translating designer's creativity in coming up with ideas (Krish 2011). In the traditional design process, many researches have studied the designer's creative work. For example, Donald Schön defines it as the result of designer's interaction with the design task in his "*reflective practice*" model (Plack 2005), Gero & Chase (2001) link designer's creativity to the situation of the designer. Some techniques also were identified by researcher such as brain storming and TRIZ (Singh & Gu 2012).

Other challenge was raised by researcher's attempt to identifying a finite set of functional/ performance criteria. The complexity of combining function with formal properties has been approached by multiple strategies. In the article "*Computer-based form generation in architectural design – a critical review*", Grobman, Yezioro & Capeluto (2009) have classified the strategies implemented by practitioners for Form-finding generative design in the previous research into seven strategies:

1- Space allocation problems,
2- Shape grammar or formal rule- based form generation,
3- Cellular automata,
4- Case-based reasoning/ expert systems,
5- Evolutionary methods
6- Geometric constraints-based form generation and
7- Performance-driven form generation.

According to this classification, a series of examples are explained in this section to form a basis for discussing current state of the art and identifying the knowledge gap.

2.2.1.1 Space allocation problems

Space allocation problems (SAP), which is also known as automated floor plan generation, is the allocation of defined spaces (Rooms) in their best positions with respect to a set of design requirements that maximize their performance (Wong & Chan 2009). Michalek, Choudhary & Papalambros (2002) argue that space allocation problems in architectural layout design hold two aspects: space topology and space geometry. Topology requires decisions based on combinatory relations between layout components, while geometry requires the definition of spatial and dimensional criteria that refers to the position of each component and its size. Accordingly, space allocation problems focus on proposing a rational ordering of the spaces and activities by recognizing the interconnected spaces that have integral functions in a single level (Rodrigues, Gaspar & Gomes 2013).

Building plans in this strategy serve as the main elements of generating building form either by extruding the generated plans (2.5-D approach) which is suitable for small housing projects (Grobman, Yezioro & Capeluto 2009) or by more complex process of solving multi-floor space allocation problems (MSAP). MASP includes the development of multiple algorithms for each floor that specify a shared position for the vertical circulation (stairs and elevators) as constrain for all floors. It also includes another evaluation process according to cost criteria. The evaluation of the generated solutions is done according to the materials and occupants flow between the floors in matter of reducing the cost, time and distance (Rodrigues, Gaspar & Gomes 2013).

One of the researches on the space allocation problems, was conducted by Michalek, Choudhary & Papalambros (2002) has resulted in developing two mathematical models for optimizing both of the geometry and the topology of the architectural floor plan design to generate a single story apartment plan by using different types of algorithms. Optimizing the geometry started by defining a unit that may have one of the three functions: room, boundary and hallway. For each unit, a set of variables were identified to develop a mathematical optimization model, followed by proposing a set of constrains that serve a specific situation (problem) may accrues during the solving process of the mathematical model. Finally, a set of objectives were added to the model to minimize the cost associated with heating and cooling loads and lighting. On the other hand, optimizing the topology had penalty functions in addition to the variables, constrains and objectives. The penalty functions are used to evaluate the feasibility of the generated topology before moving to the geometric optimizer.

For solving the algorithms used in each mathematical models, three methods were proposed. Each method utilizes a different type of algorithm as shown in Table 1.

Table 1: summary of models, methods and algorithms used in Michalek, Choudhary & Papalambros study. Source: (Michalek, Choudhary & Papalambros 2002).

Model	Method	Algorithm type
geometry optimization model	Local Optimization Method	Gradient-based search algorithms
	Global Optimization Methods	Simulated Annealing (SA) and Genetic Algorithms (GA)
topology optimization model	Global Optimization Methods	An evolutionary algorithm

The researchers have used (CFSQP) - functions of C programming language- to encode the algorithms as well as OptdesX (design optimization software) to allow the interaction of the designer with the optimization models. The results were tested out over a realistic proposed problem, accordingly, no real-world data were collected. At the end of the research, the topology optimization global method shows limitation in providing solution alternatives. The method only generates an "optimal solution" and the model has to be run again to generate other solutions.

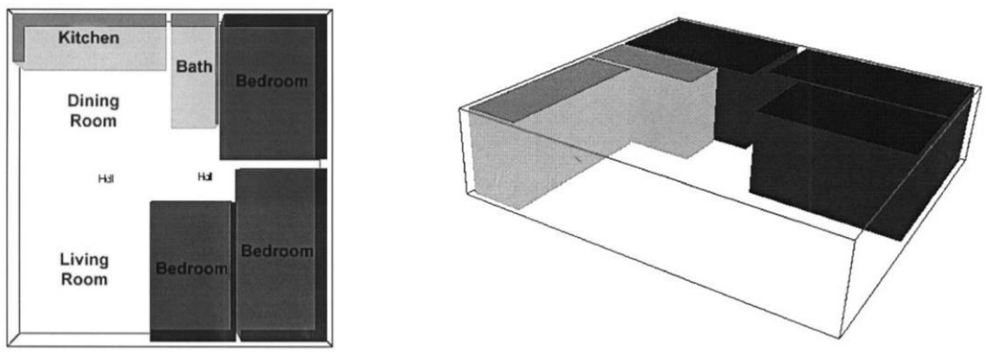


Figure 1: results from using space allocation problem strategy- first example.

Source: (Michalek, Choudhary & Papalambros 2002).

More recently, Merrell, Schkufza & Koltun (2010) proposed a framework for residential building form generation based on space allocation problems strategy that use real-life data. The framework is divided into three stages:

1. Developing relationships according to an architectural program: The first stage was formulated based on data collected from interviews and on-site observations for three residential building projects. The data helped the researchers to indicate the challenges of expanding the high level requirements requested by the client. Since this stage aims to

automate an architectural program into a graphic model, the researchers used a residential building catalogue to collect data. From the catalogue, 120 architectural programs were analysed and encoded. The analysis lead to structuring the data in a way that shows statistical relationships.

2. Optimizing floor plan: the architectural program is turned into a building layout of equally sized rectangular rooms, then a large number of alternatives is generated by adjusting the basic layout using two moves: sliding the walls and swiping rooms. The process used a metropolis algorithm which facilitates a rapid exploration using an objective function. Then each alternative is evaluated by cost function that is defined by specific parts of the architectural program such as accessibility, dimensions, floors and shapes.
3. 3D model visualizing: the building form is visualized by developing architectural templates. Each template presents a specific style that consist of a variety of types and sizes for pass ways, windows, staircases and roofs.

Figure 2 and Figure 3 illustrate the results of implementing the proposed framework. In one hand, the work showed potentials of generating a realistic large number of alternatives especially in the arrangement of the internal spaces. This is mainly related to the extensive amount of data encoded to generate architectural programs. on the other hand, forms generated in the process are very limited by four styles encoded by the researchers which questions its ability to support creativity aspects in the generation process.

Figure 2: Sample of layouts generated in Merrell, Schkufza & Koltun research.
Source: (Merrell, Schkufza & Koltun 2010).

Figure 3: architectural styles encoded to generate forms in Merrell, Schkufza & Koltun study (2010).
Source: (Merrell, Schkufza & Koltun 2010).

Space allocation problems as a strategy for form generation process prioritizes layouts arrangement over form requirements and does not incorporate site consideration. Comparing the presented examples, we can conclude that good quality results are directly related to the amount of data analysed to create the architectural program, which make it difficult to be implemented over large sized projects. Lobos & Donath (2010) have discussed issues related to space allocation problems and they have argued that the best implementation for this approach is after generating the form and defining the outer boundary for the building as the case of the China Central Television Headquarters building in Beijing designed by OMA Architects.

2.2.1.2 Shape grammar or formal rule- based form generation

Shape grammar was first introduced by George Stiny and James Grip in the early 1970s to generate building forms. The strategy consists of a set of shape rules that is applied on initial shape. The rules are visual description of sets of terminal shapes which replace the initial shape if a condition is satisfied. This strategy requires the development of library of shape rules that makes the grammar.

Researches adopted multiple approaches to recreate a different types of grammar. Some researchers have developed a grammar by parameterizing the basic shapes such as Wang & Duarte (2001). Others have taken inspirations from a specific architectural style of historical period or a famous architect work, for example, Cenani & Cagdas (2007) grammar was inspired by the Islamic patterns, Koning & Eizenberg (1981) grammar is a translation of the famous work of Frank Liyod Wright “prairie house” – see Figure 4- and similarly other grammars as those developed by Flemming (1987), Stiny & Mitchell (1978) and Duarte (2001).

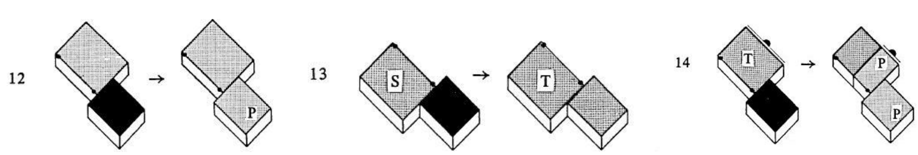


Figure 4: Examples of shape rules inspired from the famous work of Frank Liyod Wright “prairie house”.

Source: (Koning & Eizenberg 1981).

The work of Flemming (1987) is one of the first implementations of shape grammar in architecture form; his grammar is inspired by the houses of Queen Anne. The grammar consists of ten stages that start by generating two-dimensional interior spaces then shifts into three dimensional transformation by generating the exterior parts. The rules of interior generations define the hall as a focal point and accordingly define relationships between the focal point and other shapes that contain functions. However, the handling of the exterior appearance only included rules for generating the roof, porches and chimney breasts. Figure 5 presents examples of building forms generated using Flemming shape grammar.

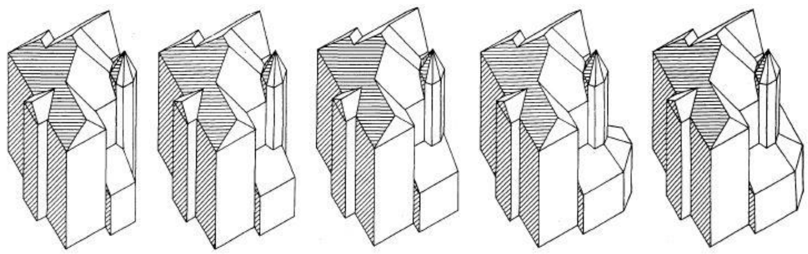


Figure 5: Results of form generated by implementing Flemming shape grammar.
Source: (Flemming 1987).

In comparison, Wang & Duarte (2001) have provided an example of developing more simplified and generalized framework to generate complex forms from few basic shapes. The vocabulary of their proposed shape grammar is inspired by the Froebel building gifts blocks. After identifying spatial relations between the blocks, a number of rules were proposed to generate transformation on the initial shapes by adding other shapes. These shape rules were labelled to avoid ambiguity where labels define the real number of variation created by applying each rule to a basic shape. The grammar was encoded using Java programming languages and visualized using Open Inventor software. However, the results of running the program are compositions that do not express a potential of building forms as shown in Figure 6.

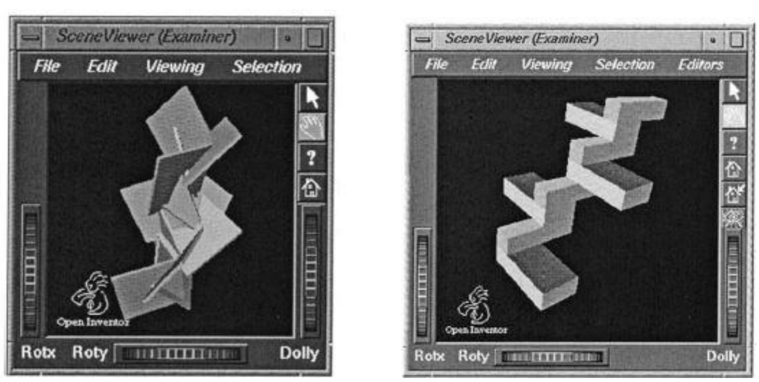


Figure 6: Results of form generated by implementing Wang & Duarte shape grammar.
Source: (Wang & Duarte 2001).

The concept of shape grammar is using initial shape and rules to generate alternatives are usually categorized under what is known as “formal rule-based” approach. This approach is similar to shape grammar as they both use an initial shape to start the process of generation, and different in the type of the rules that make the transformation on the initial shape. One of the famous formal rule-based approaches was proposed by Greg Lynn (1999) in his book *“Animate form”*. In the book, Lynn suggest that the animation and generation of forms should be derived by forces form nature, accordingly, an initial shape modified by rules have encoded dynamic forces such as gravity.

2.2.1.3 Cellular automata

Cellular automata (CA) is a context-sensitive generation process (Singh & Gu 2012). It consists of a group of cells on a lattice grid, each cell has an initial state (alive or dead) that changes according to an interaction with the neighbouring cells. The interaction is directed by a simple set of local rules that define the transformations with relation to the state of neighbour (Petruševski, Devetakovic & Mitrovic 2009; Bhardwaj & Upadhyay 2017).

The theory behind cellular automata (CA) was first introduced by the mathematician John von Neumann in his article " *the general and logical theory of automata*" in 1951. In the article, Neumann proposed a computing mechanism that simulates the growth based on his study of the natural organisms. Although some researches attempted to develop Neumanns’ theoretical model of CA such as Ulam (1962), the model was only popularized by John Conway’s game called “Life” (Krawczyk 2002a). Later in the 1980s, the concept has caught Wolfram eyes- a computer scientist and physicists- and he started studying the application of cellular automata by experimenting a different sets of rules encoded by mathematical equations that define the cells state by colour (black or white) (Packard & Wolfram 1985). Wolfram experiments resulted in patterns display a very complex chaotic behaviour that later was used in developing a random number generator.

Over the last two decades, CA has been utilized in architectural design in several ways. According to recent study by Christiane Herr and Ryan Ford (2016) the adaptation of CA in architectural practice is manifested by one of the following:

- Adopting the CA rules by developing various sets of rules with taking into account the context and introducing obstacles.
- Adopting a CA cell shapes and scales, where in many cases the cell became a representation of architectural element.
- Adapting the CA cell neighbourhoods as a natural sequence of changing cells shapes and scales.
- Adopting CA cell states to allow better response by linking it to cell shape and size or other conditions such as the natural light levels.

Cellular automata is best applied in urban planning level for simulating cities growth pattern or for analysing and simulating pedestrian and traffic flow. For example, Kim, Ahn & Lee (2018) used cellular automata to model handicapped movements according to physical pedestrians conditions. For the purpose of generating building forms, only few recent researches have used cellular automata such as Krawczyk (2002b) who introduced multiple constraints on the cell state that define choices of cell shape, its maximum and minimum allowed size, overall scale, range of distance between cells and other aspects to transfer cells of cellular automata model into architectural elements.

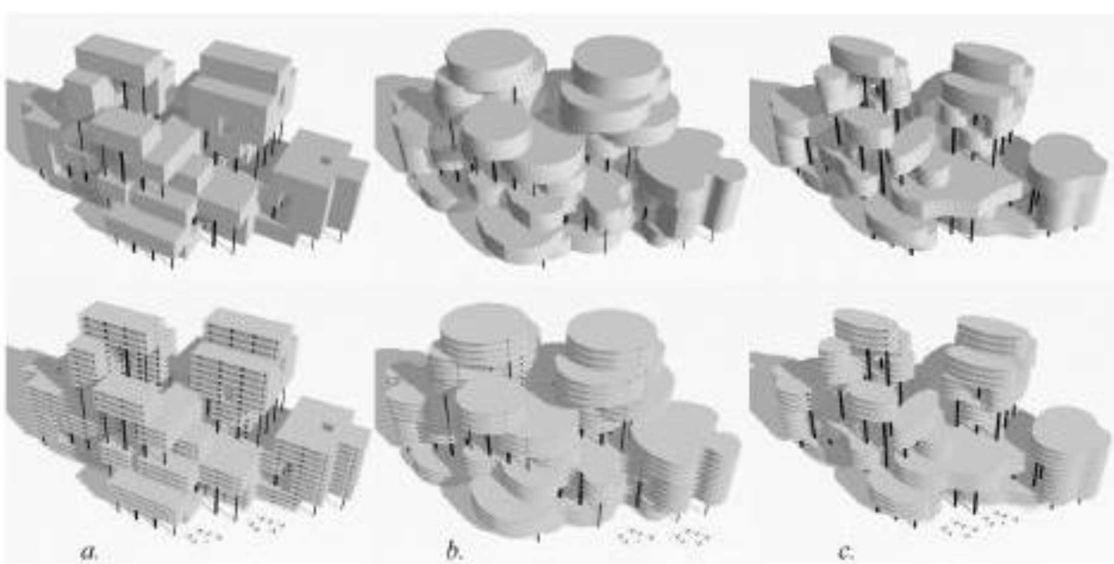


Figure 7: The form series created by Krawczyk.

Source: (Krawczyk 2002b).

2.2.1.4 Case-based reasoning/ expert systems

Case-based reasoning is one of the intelligent artificial approaches that use the solution of previous situations to generate solutions for a new similar cases. However, Aamodt (1994) pointed out that this approach is different from other IA approaches by its ability of utilizing specific knowledge denoted in a case instead of general knowledge with a large domain. Heylighen & Neuckermans (1993) and Chakrabarti et al. (2011) traced back the root of case-based reasoning to Roger Schank work in 1989 when he suggested a problem solving technique depending on previous situations that act as a reminder in a dynamic memory. The strategy mainly consists of two main processes:

1- Selection process: this process aims to find the best matching case from a library for the current problem. The process requires the formulation of a specific description and scripts for each case in the library. According to Akin (2002) a good case description should include the description of progression stages as well as the solution description and identifying the solution implementation boarder by describing a relevant situations. Schmitt et al. (1994) argue that selection of the base case for architectural design problems also depends on architect input which includes the description of the new situation context. The description most of the time consists of site conditions and a programmatic requirement.
2- Adaptation process: this process aims to modify the selected source case to suite the current problem. The process requires determining a set of parameters linked by algorithms to perform the modifications (Grobman, Yezioro & Capeluto 2009). Aamodt (1994) identify two types of adaptation: transformational reuse which refers to reusing the same solution of the source case, and derivational reuse which take the description of the method conducted to solve the previous case and implement it on the new situation.

Although selection process (retrieval stage) is straightforward process, the success in selecting the appropriate source case demands the specification of an extensive amount of features for each case. Adaptation, in the other side, is more difficult as a process due to the fact that the technique only works under supervision which limits its ability to learn from the underlying logic of the solutions. Some discrepancies could results in dealing with large scale and complex problems (Kim, Rudin & Shah 2015).

Figure 8 Illustrates a logical model for case-based reasoning developed by Zhou et al. (2010). As shown in the figure, the process is presented as a cycle of sub-tasks where the adaptation task could be repeated multiple times until the result is accepted.

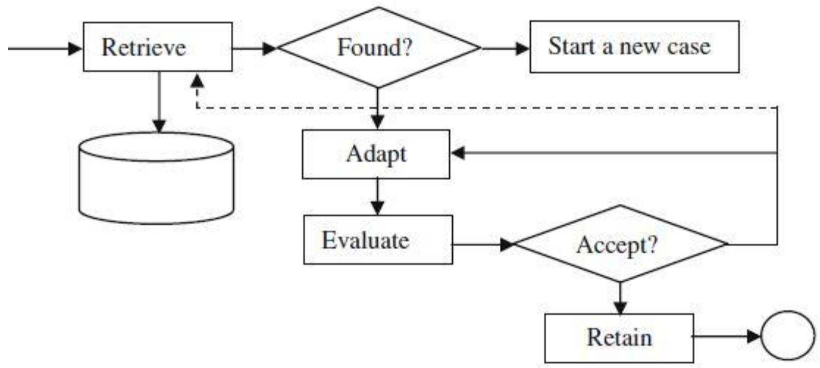


Figure 8: case-based reasoning model example.

Source: (Zhou et al. 2010).

Watson & Perera (1997) have investigated 12 case-based reasoning applications that were developed by a number of universities and institutions for the purpose of building design such as ARCHIE (developed at the Georgia Institute of Technology) and SEED (developed at Carnegie Mellon University). They concluded that the focus of the discussed applications is solving the structural design for the building design and only very few applications deal with architectural design.

A review of related literature shows that no recent researches aiming to use case-based reasoning in the architectural field are found. Moreover, no research in the past has seemed to utilize the approach for the purpose of form finding. However, some examples have achieved progress in this matter such as Garza & Maher (2001) study on combining case-based reasoning with evolutionary technique to generate floor plan layouts for residential building. In their work, they were able to decrease the amount of information needed for adaptation process by developing a genetic algorithm. The algorithm generates a random adaptation throughout combination and modification then evaluates the results instead of selecting and adopting each feature alone. For case selection, they use Frank Lloyd Wright “prairie house” plans as the source case and developed the selection specification based on the Chinese technique for evaluating layouts quality "*feng shui*". Figure 9 shows the result of implementing the proposed model.

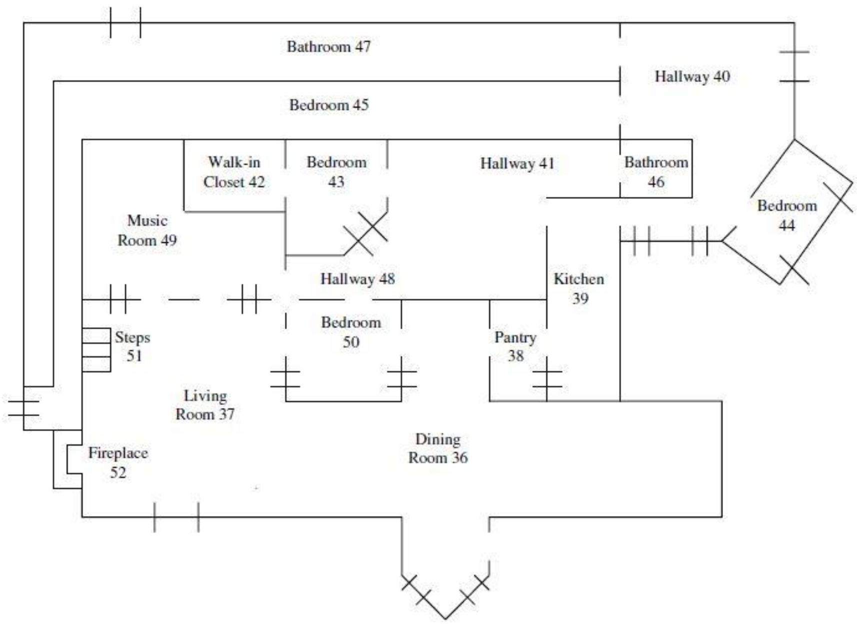


Figure 9: house layout solution found by GENCAD-FS model developed by Garza & Maher.

Source: (Garza & Maher 2001).

2.2.1.5 Evolutionary methods

The term stands for a technique that utilizes multiple types of algorithms inspired and developed by simulating the natural evolution process (Bentley & Wakefield 1997). This technique was first presented in the late 1960s after developing the L-system algorithm by Aristid Lindenmayer and generic algorithm by John Holland. However, the interest in this technique had rapidly grown in 1990s due to its success in solving multi-objective problems which presented a rich material for research in the architectural design field (Grobman, Yezioro & Capeluto 2009).

Evolutionary methods start with identifying a group of possible solutions (*individuals)* in a *population* that is called (phenotype) and a generic operator that measures an evaluation function defines the individual "fitness". The generation of the initial population could be determined by specifying parameters or not determined by allowing a random generation. The technique employs two main principles of natural modifying: selection and variation. Selection is done based on simulating the competition between living being to survive. The process of selection evaluates the quality of each solution with respect to the fitness function, accordingly, only few solutions are selected and the least qualified solutions are eliminated. Variation, on the other hand, is presented by the ability of generating a new set of individuals by recombination and mutation. Hence, variation simulated by process of changing and manipulating with the survived solutions. The variation process is repeated for a number of times defined by the designer (Eiben et al. 2007).

Evolutionary techniques is best known as search mechanism (Jabi & Grochal 2013) or optimization technique. Most of cases that implemented the technique have combined it with other generative techniques such as space allocation and shape grammar (Eiben et al. 2007). Figure 10 presents building form generated by combining space allocation strategy with evolutionary technique in Guo & Li (2017) study.

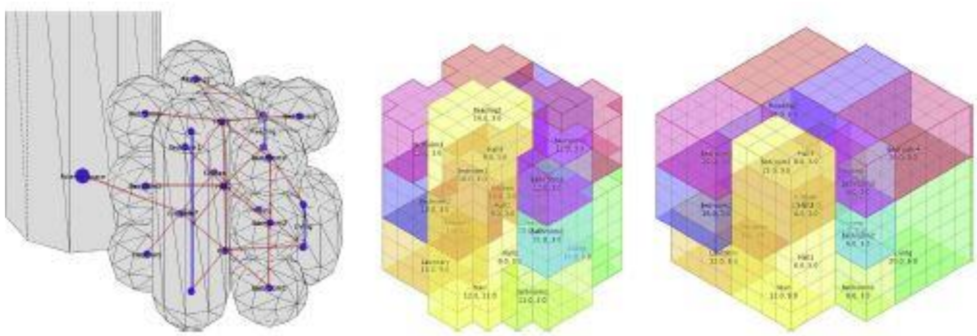


Figure 10: building forms generated by a generative approach combines space allocation problems strategy with evolutionary technique.

Source: (Guo & Li 2017).

2.2.1.6 Geometric constraints-based form generation

In this strategy, the formal properties of geometrical forms are abstracted and the relationships between the components is described by a parametric code (algorithm). This technique is distinguished by the idea of creating representation that depends on single logical model instead of creating architectural program or grammar or cases library (Grobman, Yezioro & Capeluto 2009). Accordingly, the generated solutions are entirely new solutions and do not require searching on similarities. Rong (1999) classified the geometrical constrain-based technique as a modern technique with a symbolic and numeric nature. The process of generating using the technique is driven by equations that calculate the geometrical variables. This equations represent a system of relationship that was reconstructed based on the analysis of the geometrical feature of the product.

One of the earliest studies conducted by Suzuki, Ando & Kimura (1990) investigated geometric constraints-based strategy by developing a geometrical model. The argument of the researchers was that such a model is needed to improve the automation of the design process using CAD system. They recorded the model's capability in capturing the designer's intent. The development of the model took into consideration two aspects: geometrical constraints that start by identifying dimensional constraints in the geometric elements level and continue until it reaches the model level by coding relationships between the individual elements. The second aspect is geometrical reasoning that calculates and manipulates the geometrical properties after identifying it in mathematical equations.

Ault (1999) has discussed and classified number of approaches to use a geometrical constraints-based strategy as a design tool. According to Ault, the strategy depends on formulating:

1. Parametric model or variational model or combining between both. The parametric model define each vertex in a solid model with range of positions that sets its size and shape. However, the variational model defines the vertex by an equation that should be satisfied in order to find a solution for the solid model configuration.
2. Constraints: each geometrical entity in the model is specified by variables. Constraints types are "ground" constraints that specify the relationship between each entity and the total, "dimensional" constraints that specify numerical values usually by equations, geometric constraints specify geometrical properties that could be emerged by combining entitles such as symmetry and algebraic constraints which consists of mathematical equations impose restrictions or logical expressions (IF-THEN) to organize the complex relationship.

Prusinkiewicz & Streibel (2007) have provided several examples to test out the strategy. They focused on developing a geometrical model for generating a polygonally mesh to create smooth surfaces for a number of objects. The main entities of the mesh are edges and faces where the relationships specified faces area and angles between edges. One of the examples presented in their work is the shape of Eiffel tower generated from constructing a geometrical model for a slab as shown in Figure 11.

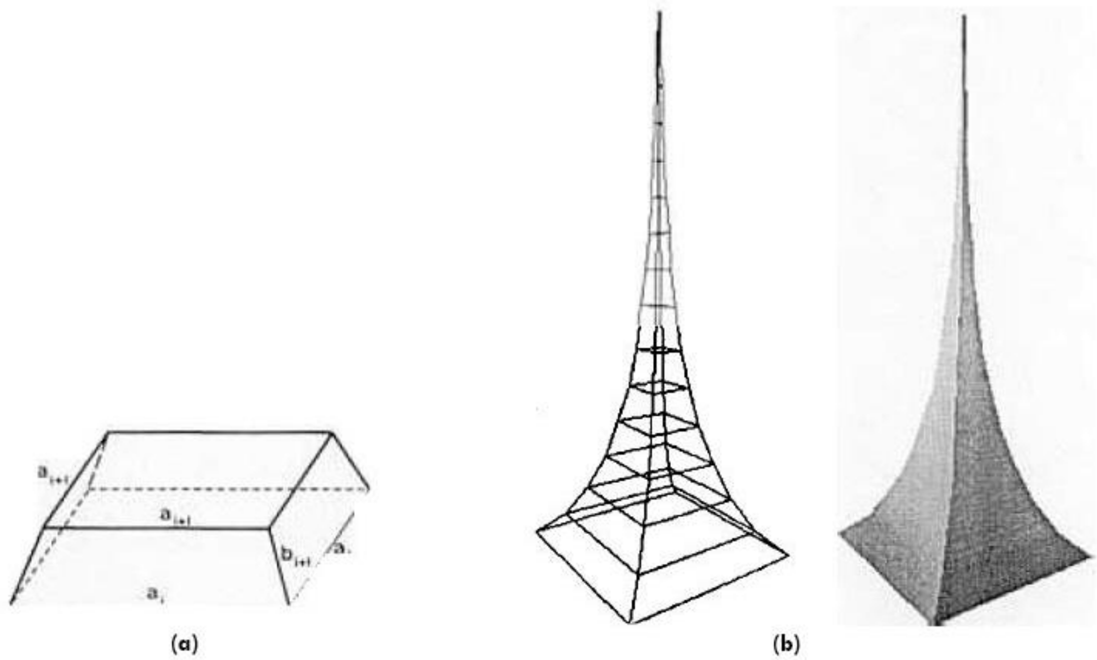


Figure 11: the Eiffel tower shape generated from constructing a geometrical model for a slab.

Source: Prusinkiewicz & Streibel (2007)

Nowadays, numerous international architectural firms use this strategy to generate alternatives in the conceptual phase of the design. Among the firms, Foster and Partners have established a specialist modelling group (SMG) that have worked on developing a geometrical models for over 100 projects such as Swiss Re headquarters and London City Hall. However, they implement the strategy as a control mechanism to provide flexibility in generating alternatives and further environmental analysis and evaluation. Accordingly, the geometrical model differ from project to another and it mainly follows the initial sketches and ideas of the designer.

2.2.1.7 Performance-driven generation technique

Performance- driven form generating concept employs digital simulation tools as a mechanism for generating building form (Oxman 2008). The term performance in this matter is related to the efficiency of the building and its ability to perform. The idea of introducing performance analysis into the design process was first proposed by Negroponte in the 1970s. Negroponte envisioned "architectural machine" that generates, evaluates and adopts the design. The evaluation part is seen as an important step for the machine to adopt the design based on achieving performance criteria (Shi 2010). Negroponte vision at that time was very difficult to implement due to the extensive amount of calculation and analysis the designer have to do. However, with the advancement in the development of the simulation tool, performance-based design started to take its place in the academic research as well as in the practice at the end of 20^{th} century (Shi 2010).

Performance-based design approach utilizes performance analysis data to modify the design after it has been developed in a conventional way. Despite the advantages, the approach is considered to be time consuming since it requires an iterative process to achieve a good solution. Researchers attempt to make transition into prioritizing performance by integrating the simulation tool with the generation model at the beginning of the design. These attempts have extracted a new approach which is known as performance-driven design generation or "performative design" (Oxman 2009).

Both of the approaches (performance-based design and performative design) have been investigated according to two categories of performance:

1. Structural performance: to ensure safety consideration and cost (Ganzerli, Pantelides & Reaveley 2000; Möller et al. 2009).
2. Built environment performance: which is associated to the quality of the building with relation to thermal comfort (solar and heat gain/loss), ventilation (wind and air movement), moisture, acoustics and lighting...etc. this was supported with the adaptation of green building international standards such as LEED. Recent examples include Yi & Malkawi (2012) study in integrating Computational Fluid dynamic and energy simulation for building form optimization using generic algorithm. They proposed a new methodology that assess the designer in reducing the thermal loads in the early stage of the design. Figure 12 illustrates results from the study. Similarly, Zhang, Zhang & Wang (2016) have proposed a method to generates building form based on multi-objectives design criteria that include solar radiation and space efficiency and Si & Wang (2015) developed a workflow for building massing deign based on solar radiation criteria that depends on the evaluation of the total sunlight hours.

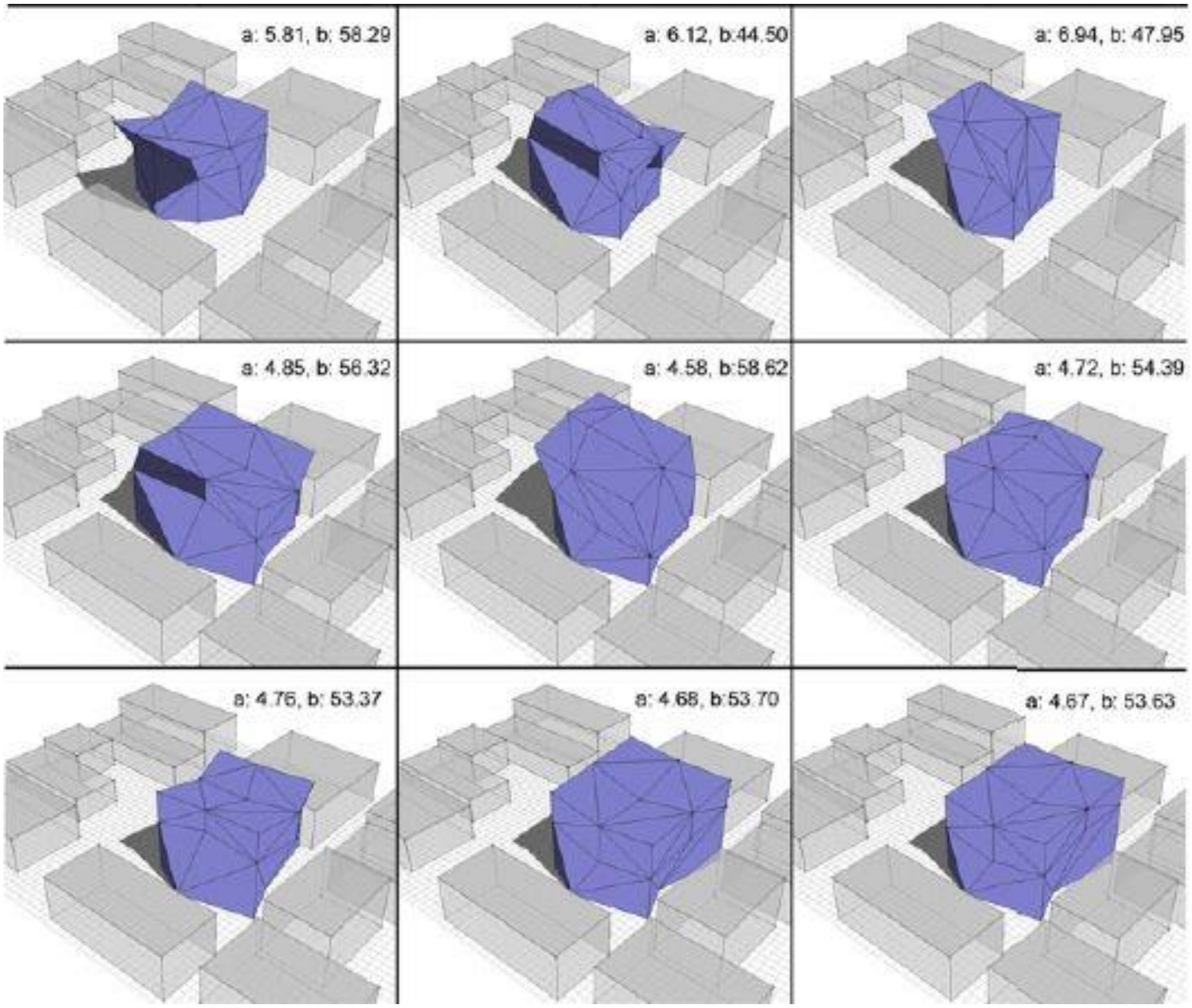


Figure 12: samples of results generated by Yi & Malkawi proposed methodology.

Source: (Yi & Malkawi 2012).

Other performance criteria such as aesthetic and culture also are an important part in the building form process, however, performance-based approaches works with quantitative analysis and failed to incorporate these kind of criteria since it is considered to be subjective and difficult to be quantified.

2.3 Identify the knowledge gap

This section examines the seven common generative design strategies that were presented in the previous section in order to identify the main limitations and implications of each strategy in supporting design exploration. The main aim of this section is forming a basis of the theoretical framework for the envisioned technique by finding a place where new contributions can be made to enhance the ability of generative design approaches in automating the design process and generate alternatives.

The knowledge gap will be identified according to two essential categories: technical factors and design aspects. The overlaps and similarities as well as the differences between the strategies also will be discussed. Before going further with the discussion, a summary of the main concept of each strategy is illustrated in Table 2.

Table 2: generative design strategies- concepts.
Source: author.

Generative design strategy	Concept	Explanation
Space allocation problems	Rational ordering of predefined units that represent spaces and activities	Form is generated by extruding the generated layouts and alternatives are created by following predefined templates.
Shape grammar	Trade off the initial shape with terminal shapes by finding similarities. “element in relations”	Form is generated earthier by extruding the generated layouts or by constructing multidimensional composition from singular unit by using (add, replace, subtract) functions
Cellular automata	Context- sensitive growth based on cells	Form is a pattern of units generated by growth of cells on grid determined by initial cell state.
Case-based reasoning/ expert systems	Utilizing specific knowledge denoted in a case.	Layout is generated by using the solution of previous situations that share similarities with the new case/ using this strategy for building form generation still under research.
Evolutionary methods	Optimization mechanism (initial design improvement)	Forms are generated by modifying a selected initial population that is best achieve fitness criteria
Geometric constraints-based form generation	Manipulating geometrical properties	Form is generated by equations that calculate geometrical variables of basic components
Performance-driven form generation	Performance simulation data inspires formal expression	Form is generated by linking basic geometry to performance analysis data

2.3.1 Knowledge gap - Technical factors

Technical factors mainly indicate the practicality of using each strategy for building form generation. Practicality is one of the main challenges that face the implementation of any new technology or design approach on a large base. This will be measured according to five important technical factors:

1- The requirements of defining and formulating each strategy: they give an insight of the amount of information, effort and time needed to develop the strategy. The requirements are problematic if the strategy is limited to one type of project or for one case which means the use of the strategy for other types of projects will requires the same effort and time.
2- Operation requirements: indicate the amount of effort that is required from the user to use the strategy. Some strategies require a special knowledge to be used that make it inconvenient for the user, other strategies can be time consuming because of the amount of specification required from the user.
3- Applicability of using the strategy on different design scenarios: allows a deeper understanding of the efficiency of developing and using each strategy for the purpose of generating building form.
4- Advantage.
5- Disadvantage.

Table 3: development requirements and operation requirements of generative design strategies. Source: author.

Generative design strategy	Strategy development requirements	Operation requirements
Space allocation problems	• Collection and analysis of large amount of data that specifies programmatic function. • Development of mathematical models for optimizing both of topology and geometry of predefined unit.	User must specifies building requirements that are related to the number of spaces (number of rooms), its function and desirable size for each space.
Shape grammar	• Developing a vocabulary of shapes according to geometrical logic. • Encoding adequate design knowledge into shape rules	User must specifies conditions that match one of the predefined set of initial shapes

Cellular automata	• Mathematical encoding of state rules. • Define constraints to direct growth process	User must define state of the initial cell
Case-based reasoning/ expert systems	• Developing a library of cases. • Formulating a specific description and scripts for each case in the library. • Determining a set of parameters linked by algorithms to perform modification.	User must describe and script the new situation context.
Evolutionary methods	• Developing a mathematical model for optimisation. • Developing generic operator that measures an evaluation function defines the individual "fitness".	User must specify the boundaries of solution space as well as the search space/ select algorithm type/ Specify a fitness criteria- function
Geometric constraints-based form generation	• Abstract geometrical forms. • Developing a parametric model. • Identifying constraints	• Specify project spatial requirements. • Specify site dimensional limitations.
Performance-driven form generation	• Collection and analysis of large amount of performance simulation data. • Developing parametric model that mathematically describe relationship between specific performance criteria and geometry.	• Specify project spatial requirements. • Specify site conditions.

Table 4: applicability on different design scenarios, advantage and disadvantage of each generative design strategy. Source: author.

Generative design strategy	**Applicability on different design scenarios**	**Advantage**	**Disadvantage**
Space allocation problems	• Normally is project-specific which means for each type of project a different algorithm should be developed (each building program is unique) • Only small single-story (maximum two story) residential building. • Difficult to be implemented over large projects.	• Uses functional requirements in the generation process	• Priorities layouts arrangement over form requirements and does not incorporate site considerations. • Applicable for specific type of projects.
Shape grammar	• Depends on the shape rules, if it is only consider geometrical properties then it can be generalized to different design	• Suitability of various project scales. • Able to generate large design search space	• Does not include evaluation of the generated design, hence, it should be incorporated with other

	scenarios, however, if it is constrained by functional requirements then it is project-specific.	• Able to generate complex geometries.	strategies such as evolutionary method • solutions are limited to one architectural style
Cellular automata	• Applicable to different project scales.	• Easy to implement • Suitability of various project scales. • Able to generate complex geometries.	• Does not include evaluation of the generated design. • Limited by cell geometry • Results cannot be modified after the process
Case-based reasoning/ expert systems	• Suitable for small scale projects.	• Uses functional requirements in the generation process. • Includes an evaluation of the generated design.	• Only works under supervision which limits its ability to learn from the underlying logic of the solutions. • Demands the specification of an extensive amount of features for each case. • Some discrepancies could results in dealing with large scale and complex problems
Evolutionary methods	• Applicable to different project scales. • Applicable to different stages at the design process. • Project-specific: the strategy developed for solving one design problems rarely can be used for other problems/ situations.	• Uses functional requirements in the generation process. • Includes an evaluation of the generated design. • Suitability of various project scales.	• Difficult to be used and requires special knowledge. • Solutions are limited to one style.
Geometric constraints-based form generation	• Applicable to different project scales. • Constraints need to be redefined for each design problems.	• Able to generate complex geometries. • Suitability of various project scales. • Uses spatial requirements in the generation process. • Solution can be modified easily.	• Does not include an evaluation of the generated design.
Performance-driven form generation	• Applicable to different project scales. • Suitable for projects that incorporate environmental considerations in the design.	• Able to generate complex geometries. • Suitability of various project scales.	• Difficult to be used and requires special knowledge. • Solutions are limited to one style. • Does not consider functional requirements.

From Table 3 and Table 4, the main limitations of the strategies related to the technical factors are summarized by:

- The use of predefined “design search space” such as space allocation problems, shape grammar, case-based reasoning and evolutionary methods limit creativity and decreases the efficiency of the strategy in solving different design problems.
- These strategies depend on search mechanism that requires the user to specify/ describe/ script an extensive amount of data. As a result, these strategies become less convenient for the user especially when the data has small effects on building forms.
- Strategies that use predefined “design search space” also cannot solve large scale projects. Increasing the scale of projects increases the complexity of the design problem which demands more analysis and efforts.

2.3.2 Knowledge gap - Design factors

Design factors are directly identified by the ability of each strategy to expand the design research space in terms of variability of the generated solutions and their accuracy. Design variability refers to the success of each strategy in generating high numbers of new unexpected solutions that cannot be categorized under limited architectural styles.

Design variability is accounted as main difficulty since an acceptable level of design variability only could be achieved by limiting designer inputs. However, design problems are subjective in its nature and require making decisions during the process by the designer, especially decisions that does not essentially related to logic such as the formulation of design ideas as well as aesthetic aspects.

On the other hand, design accuracy measures the amount of modifications needed to be done after the generation process so the generated alternatives can be utilized. Deign accuracy is directly affected by operation requirements that helps the strategy to adopt different design problems.

Table 5: design factors of generative design strategies.
Source: author.

Generative design strategy	**Design variability**	**Design accuracy**
Space allocation problems	• Variability of solutions is successfully achieved in layouts levels only. • Building form solutions consist of very basic geometry and limited into architecture styles that been encoded in the strategy algorithm.	• Solutions usually require further modification according to the site requirements.
Shape grammar	• All solution share the same architectural style.	• Solutions reflect only form options and need to be validated for functional and site requirements. • Usually require modifications.
Cellular automata	• One solution is generated by each run. • Building geometry is constrained by the shape of cell that is identified by the user.	• Site dimensional constraints as well as spatial requirements of the building can be incorporated in the algorithm.
Case-based reasoning/ expert systems	• One solution is generated by each run. • Adaptation process does not support variability.	• Accuracy of solution depends on the amount of features that been specified as well as the availability of similar case on the library.
Evolutionary methods	• Able to generate large design search space. • Generated solutions have high degree of similarities. • Solutions adopt a specific style since they are essentially an optimization of designer input.	• Solutions are very accurate and generally do not require any further modification.
Geometric constraints-based form generation	• Able to generate large design search space. • Variability of solutions is directly affected by the amount of constraints and variables encoded in the parametric model of the initial geometrical unit.	• Accuracy of solutions depend on the amount of constraints incorporated in the generation process, however, this strategy can be used to achieve high level of design accuracy.
Performance-driven form generation	• Able to generate large design search space. • Generated solutions have high degree of similarities.	• Solutions are accurate with respect to single - performance criteria • Usually require modifications.

CHAPTER THREE

METHODOLOGY

Chapter 3 Methodology

This chapter discusses the methodology that is adopted to achieve dissertation objectives in four sections; the first section presents research approach; while the second section describes the methodological framework of developing the envisioned technique for generating building forms followed by summary of the technique. The final section discusses how the envisioned technique is validated and the software/ computer programs used in the process.

3.1 Research method

Walliman (2011) defines research methods as tools or techniques that are used for enquiry. Essentially, there are three major traditions of research methods: Qualitative research, quantitative research and mixed method research (Gerber, Anthony & Abrams 2007).

Qualitative research is interpretive where things are studied in their "*natural setting*". It includes developing a mean of description for a phenomena or process, as well as, drawing conclusions. While quantitative research provides explanation supported by empirical data (Groat & Wang 2007).

Mixed methods, known as a third methodological movement, combine both of qualitative and quantitative research methods. Researches that used mixed method depend on textual as well as numerical data to conduct the study (Borrego, Douglas & Amelink 2009). As shown in Table 6, mixed methods research is divided according to the relative weight of quantitative and qualitative components into four types: triangulation, embedded, explanatory and exploratory.

Table 6: mixed methods research types.
Source:(Borrego, Douglas & Amelink 2009)

Design Type	Timing of quan and qual phases	Relative weighting of quan and qual components	Mixing – when quan and qual phases are integrated	Notation
Triangulation	Concurrent	Equal	During interpretation or analysis	QUAN + QUAL
Embedded	Concurrent or Sequential	Unequal	One is embedded within the other	QUAN(qual) *or* QUAL(quan)
Explanatory	Sequential, quan then qual	Usually quan is given priority	Phase 1 informs phase 2	QUAN -> qual
Exploratory	Sequential, qual then quan	Usually qual is given priority	Phase 1 informs phase 2	QUAL -> quan

This dissertation follows an exploratory mixed methods research for developing the envisioned technique. According to Borrego, Douglas & Amelink (2009), exploratory approach is human-driven approached mostly assisted by computers. It starts by a qualitative phase that emphasises important factors by reviewing, analysing and comparing value-free data; followed by quantitative phase of measuring and analysing variables.

In same way, this dissertation uses literature review as the primary source of knowledge to create a base for structuring and establishing a framework for the development procedure of the technique. Subsequently, the development of the technique continues with using a quantitative data analysis approach that includes formulating algorithm by the abstraction of geometrical relationship and mathematical equations as presented in chapters 4 and 5.

3.2 Methodological framework

The framework for developing a novel generative design technique is based on the limitation of the current generative design strategies that was identified and discussed in the literature review. This section describes the procedure of developing the generative technique and emphasises the main considerations that were taken along the process. The development of the proposed technique is summarized by three steps:

- **Systemizing the design procedure of building form**: the development starts by analysing the organizational structure of the traditional design procedure and breaking down its underlying logic. Then it is reorganized by defined tasks to form the guidelines of a procedural structure for the envisioned generative design technique. During the process, tasks are categorized by logical tasks and creative tasks. The creative tasks are the tasks that cannot be systemized due to their ambiguity and relation to the human intelligence as they may require decision making. Figure 13 summarizes the organizational structure for the traditional design procedure and the logical and creative tasks.

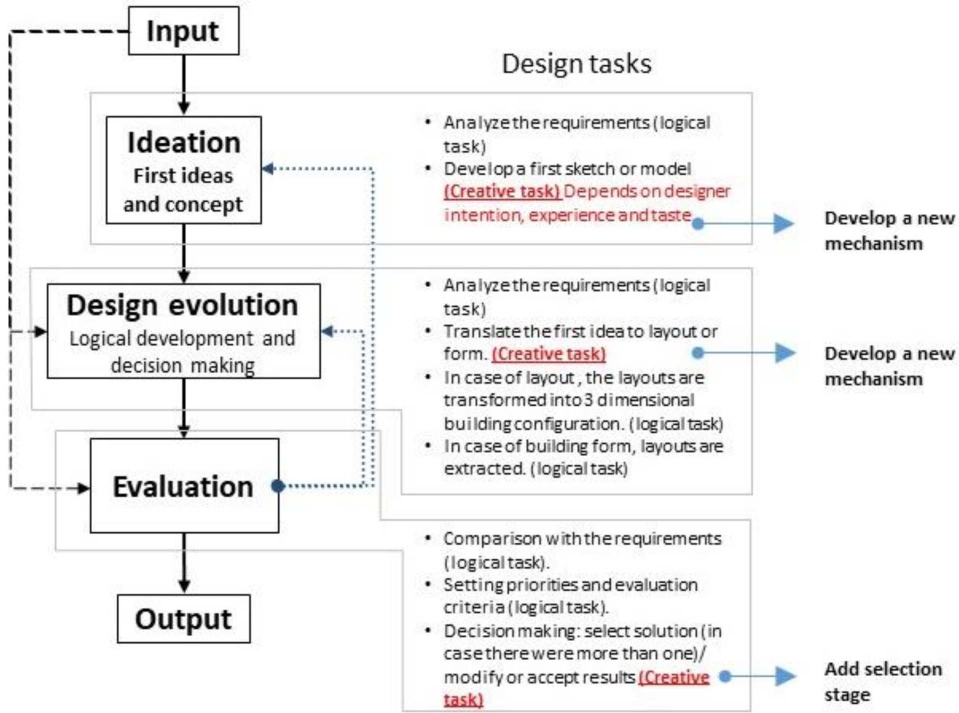


Figure 13: Organizational structure of the traditional design procedure.

Source: author.

As shown in Figure 13, the traditional procedure consists of three main stages: ideation, design evolution and evaluation. In the proposed procedural structure of the technique, the ideation stage and the task of translating the idea in design evolution stage are suggested to be replaced by new computational mechanism. An addition of selection stage is needed to separate the decision making from the evaluation stage. Since one of the goals of the technique is generating a large number of solutions, selection stage is mainly for the designer to select one of the generated solutions.

In the design evolution stage, the designer may starts by creating layouts and constructing the building configuration based on it or vice versa. Similarly, the generative design strategies that were reviewed in the previous chapter either have used layout as the main unit to generate the form or defined a basic geometrical unit for the same. By analysing both cases, it was decided to start from developing the layout then the building form in the design evolution stage of the envisioned technique because of the following:

1. The use of geometrical unit results in the problem of modularity in the generated forms as can be seen in the examples of shape grammar, Cellular automata, Evolutionary methods and Geometric constraints-based form generation.
2. The use of layout supports variability by allowing a nation of methods in transforming the layout into 3 dimensional space.

Table 7 summarizes the main differences in the design procedural stages between generative design strategies and the proposed technique

Table 7: design procedure stages in the generative design strategies vs the proposed technique. Source: author.

<table>
<tr><th colspan="2">Traditional design procedure stages</th><th>Other methods</th><th>The proposed method</th></tr>
<tr><td colspan="2">Ideation</td><td>Limited by requesting a feedback from the designer to specify 2d or 3d geometrical units. (see operational requirements in table 3)</td><td>Computational mechanism to generated a geometrical base pattern based on higher level of geometrical abstraction that doesn’t require the designer feedback</td></tr>
<tr><td rowspan="3">Design evolution</td><td>Translate idea</td><td>Limited by research mechanism that find solutions from predefined design search space.</td><td>Computational mechanism based on the concept of randomness to formulate layouts using the base pattern</td></tr>
<tr><td>In case of form</td><td>Optimizing the geometrical unit based on performance or functional criteria.</td><td>-</td></tr>
<tr><td>In case of layout</td><td>Transformation to 3d configuration limited by extrusion</td><td>incorporating different type of transformation (extrusion, blend , scale and twist or a combination of more than one type)</td></tr>
<tr><td colspan="2">Evaluation</td><td>Most of the methods do not include evaluation stage.
Evaluation in evolutionary method is only part of the optimization mechanism.
Evaluation in performance-driven form generating strategies considers only single performance criteria that essentially is related to environmental design requirements.</td><td>Evolution combines quantitative and qualitative criteria.
Qualitative evaluation is handled by developing indices
Evaluation depends on the principle of exclusion and ranking.
Exclude the inconvenient solutions is better strategy that optimizes the best solution since it supports design variability and allows the inclusion of multiple criteria</td></tr>
</table>

- **Setting a realistic border for inputs data**: essentially, building design process is driven by project requirements and site conditions requested as the main input. In the dissertation, an important reason causing the limitations of the current generative design strategies used for generating building form is directly related to the input that is required for the strategy. Most of the strategies do not consider site conditions (except performance-driven form generation and cellular automata) or some important project requirements such as the needed total built up area.

Furthermore, these strategies use other information that require a special algorithm to process it. For example, strategies such as space allocation problems, shape grammar and case-based reasoning require the development of a “library” or an architectural program to direct the generation process. These kind of inputs are project-specific that cannot be generalized, require a lot of work and analysis but at the end, would have minor effect in terms of generating forms alternatives that may support functionality.

For the envisioned technique, we tried to set out a basic input that has the biggest effects on generating the building forms and lower/ eliminate the need for any further modification on the results.

- **Developing mechanisms to replace creative design tasks**: the originality of this dissertation is manifested in two mechanisms proposed to replace creative design tasks of generating design ideas and layout alternatives. The formulation of this mechanisms depends on analysing the outputs of the traditional tasks of creating design ideas and suggesting a new approach for creating similar outputs by a pure geometric abstraction of basic spatial relations. The challenge was to maintain aspects related to design creativity and avoid falling in the problem of architecture style. In this manner, the principle of randomness in selecting variables was used as a way for supporting creativity.

Since generative design has the advantage of allowing the inclusion of some aspects that are normally considered in advantage stages of the design, another mechanism was added to design development stage to propose structural layout for each generated solution.

3.3 Summary of the proposed generative technique

The proposed design technique of generating solutions for building form design consists of four stages. The first stage ***"backward triangulation"*** deals with the creation of the main units that will be used in the process of generating building layouts. The units are finite elements in multiple sizes and shapes. In this stage, no input from the designer will be needed and the computer will generate different solutions each time by incorporating pseudorandom number generator to provide different data in the generating process for each run.

The second stage ***"forward building formation"*** of the methodology contains four layers of processes for the actual generation of the building form including the generation of layouts. An input of the designer will be used in the processes to ensure all the generated forms exhibit project requirements that are related to the spatial characteristics. Designer inputs are:

1- Plot shape
2- Plot area,
3- Setbacks,
4- Maximum height,
5- Number of floors,
6- The height of the ground floor,
7- Structural span.

The third stage **"evaluation"** is evaluating and ranking the generated solutions a well as excluding the inconvenient ones. The solutions will be ranked based on their complexity and efficiency in fulfilling project's requirements from built up area, footprint, etc. … this stage is important to ensure design practicality.

The fourth and final stage is **"selection"** where the computer presents the final results of the previous stages and the designer could pick one of the solutions based on his/her conviction.

3.4 Validation

To validate the proposed technique for generating building form, it will be implemented over a series of examples categorized into three groups:

1- Examples directed to test out the proposed variables in each process: the examples will cover the implementation of each task in different situations created by changing the proposed variables. These examples supposed to validate the efficiency of the variables in support generating large number of alternatives.

2- Examples directed to validate the ability of whole technique to generate forms for a hypothetical project requirements: a plot is selected randomly to take its regulations with hypothetical project requirements as an input for implementing the whole technique in sequence.

3- Examples of scripting the mechanisms using the algorithmic editor interface Grasshopper incorporated with visual programming language Rhino 3d.

Two softwares are selected to support the implementation of the technique. The first software is AutoCAD and it will be used for drafting or for visualizing the results of applying the proposed tasks. The second software is Grasshopper that will be used for scripting the proposed mechanisms.

AutoCAD software is considered the most popular drafting program used in engineering and construction field due to its simplicity and user-friendliness, as it provides multiple functions for drafting and editing and is very flexible in switching between 2d and 3d environment (Salih & Ahmed 2014).

Grasshopper, on the other hand, is visual design scripting and programming language that is run using Rhinoceros 3D software. Grasshopper contains a wide range of parametric components which makes it architect friendly visual interface that allows real time feedback on changing input and output parameters (Guidera 2011).

Grasshopper is selected for the advantage of its flexibility in specifying/scripting different types of input (geometry/ values/ equations/ list) as well as different types of output format.

CHAPTER FOUR

A NOVEL TECHNIQUE FOR GENERATING BUILDING FORMS

Chapter 4 A novel technique for generating building forms

This chapter discusses and describes the proposed technique stage by stage and unfolds the underlying logic behind its organizational structure. The four stages of the technique are described by required input, expected output, design tasks, variables and constraints. Two types of variable are mentioned in this chapter: designer preference variables and computer preference variables. Designer preference variables aim to direct the design process and generate efficient outputs. This variables are selected after analysing project requirements together with designer's input. While computer preference variables are selected randomly from defined range to ensure a varied output; the randomization plays a significant role in supporting creativity and avoiding the problem of architecture style. However, defining a range of the variables values as a constraint in the random selection process helps in achieving goals (expected outputs) with overcoming complexity.

Figure 14 illustrates a simplified diagram of the proposed technique. The first part of the diagram presents designer inputs for each stage that acts as constraints (restrictions) to direct the design evolution or as exclusion criteria in the evaluation stage. The second part consists of the four stages of the technique and the processes fall under, and the third part shows the output of each stage that is also used as an input for the following stage.

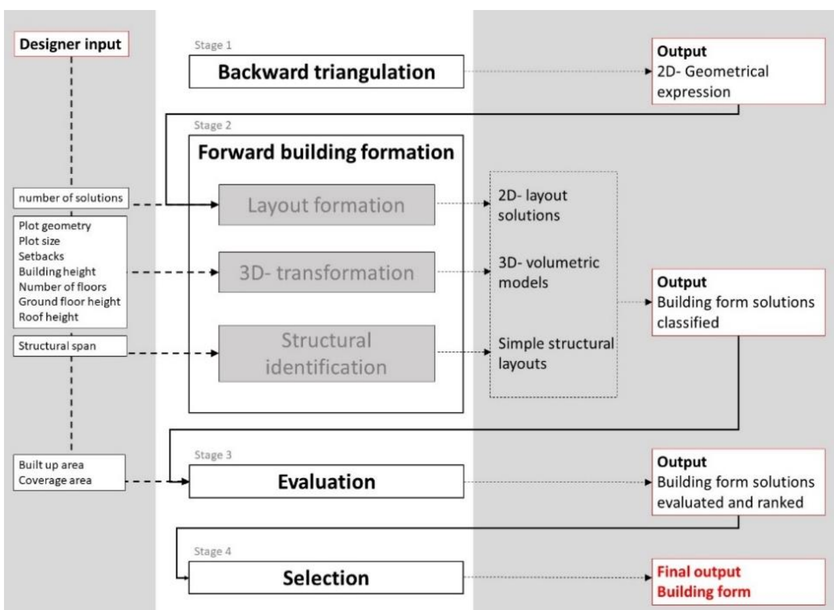


Figure 14: technique simplified diagram.
Source: author.

4.1 Backward triangulation stage.

The traditional design process in architecture always starts with a conceptual phase of developing initial design ideas and inspirations according to the project requirements. Designers in this stage transform their ideas into forms of architectural expression such as sketches or physical models, mainly used as a reference for developing the design configuration. Similarly, backward triangulation stage presents a computational technique which creates an architectural expression that is not part of the outcomes but it works as a base for further form generation processes.

Essentially, the base we need to start the process is a pattern that consists of number of finite shapes/elements which will be used as units to generate the actual layout in following stages. The technique depends on segmenting a basic shape by defining a set of rules/processes for this purpose.

The process of pattern formation starts by creating an initial shape. The initial shape can be any basic shape such as: a circle, a rectangle or a triangle. However, in my thesis, I have set the shape of a circle as initial shape to start the process of form generation. This is justified by the fact that all other shapes can be created by segmenting a circle, but only a circle can generates circular segments as parts of the layout.

Segmentation process relies on two basic tasks: creating lines and connecting nodes. Different pattern options are generated in each run by changing the characteristics of lines created in the first task and changing the definition of nodes in the second task. Lines characteristics are limited to three characteristics significantly affecting the geometrical property of the pattern. Such characteristics include: number of lines (N_{lines}), lines position (whether it is restricted to pass through circle centre or not) and the relationship between lines (whether lines intersection creates equal angles between lines). While nodes could be defined as one or more of the following: lines' midpoints and points of intersection (line intersection with the initial shape or line intersection with other line). After applying each task, lines will be redefined as the distance between two points of intersection when no other intersection point exists between them. This is very important since it will lead to the formation of new nodes and makes the repetition of "connecting nodes" task valid. For example, if two lines were created in the second task (creating lines), they will be defined as four lines in the third task. So if nodes in the third task are identified as lines midpoints, connecting them will create results different from the previous task as shown in Figure 15. A loop is created in the process that allows repeating "connect nodes" task multiple times with redefining nodes in each time.

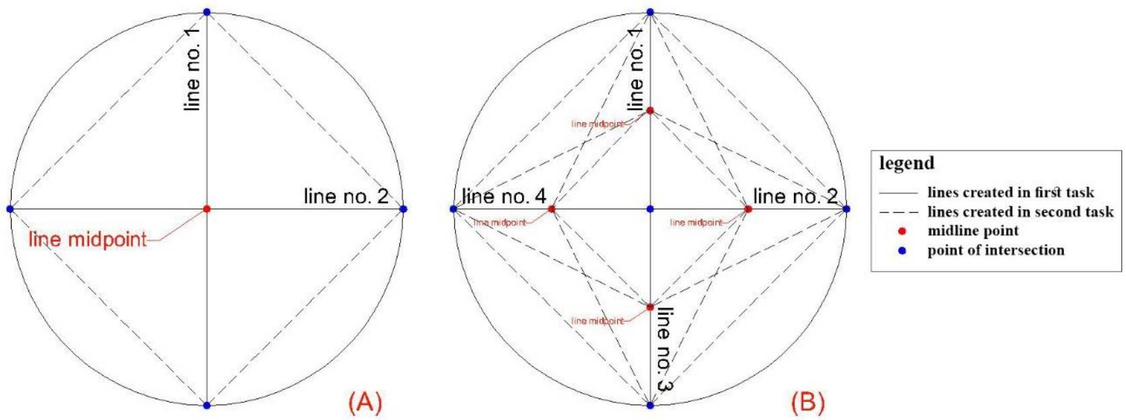


Figure 15: Effect of redefining lines after each task in the backward triangulation stage.

(A) Result of connecting nodes where nodes are identified as line intersections and line midpoints without redefining lines. (B) Result of connecting nodes where nodes are identified as line intersections and line midpoints after redefining lines. The hidden lines present lines created by applying the task of connecting nodes. Source: author.

While experimenting and exploring the technique, we found that the density of lines varies significantly from the inner parts of the circle to the parts around the boundary. In order to distribute it, all lines created during the process are extended to intersect with circle's boundary at the end of the process. Figure 16 shows the results of three examples of patterns generated in backward triangulation stage before extending the lines. The detailed process of generating the pattern for the three examples are discussed in chapter 5. As shown in the figure, in all examples the generated lines are dense in the inner parts of the circle.

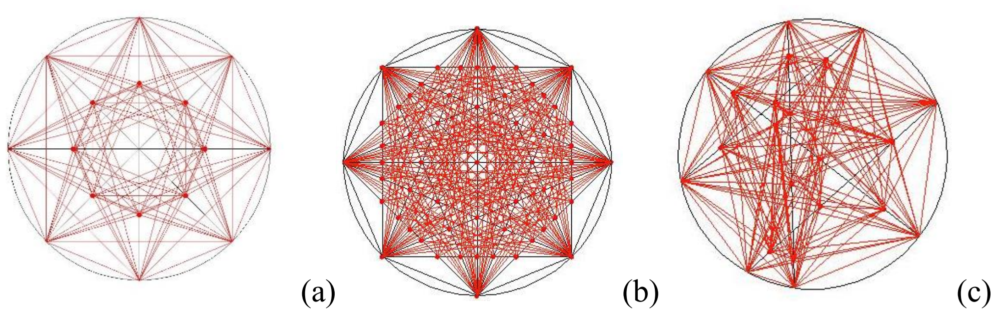


Figure 16: a, b, c are examples of patterns generated in backward triangulation stage showing lines density concentrated in the middle of the initial shape.

Source: author.

While this stage doesn't require designer' input, the output of the stage is a composition of geometrical units. Figure 17 summarizes the component of backward triangulation technique.

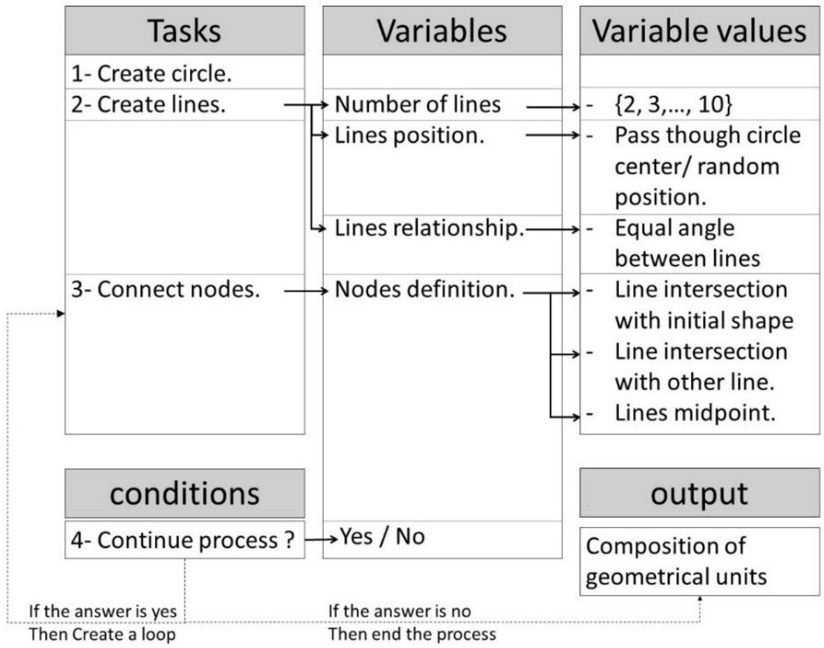


Figure 17: Backward triangulation components.
Source: author.

Backward triangulation stage tasks:

This stage consists of a starting task (creating circle as initial shape), tasks for segmenting the initial shape (the second task of creating lines and the third task of connecting nodes) and finally a decision making task to end the whole procedure or to continue.

1- **Create circle**: as discussed before, shape of the circle is the most convenient shape to start pattern generation process with for various reasons. First of all, circle has the same distance between the centre and any other point (self- centralised) which makes it easier to create symmetrical pattern as well as decreases the number of segmentation tasks. Secondly, it gives the chance to incorporate curves in building's boundary and finally, all other shapes can be easily created from segmenting the circle. The area of circle is not important in this stage, therefore no numerical data is needed to start the task hence circle's area will act as a measuring unit.

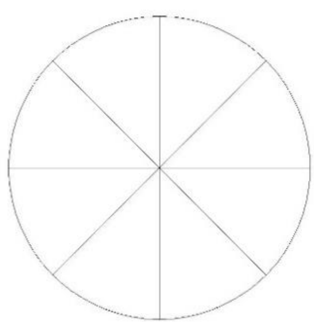
2- **Create lines**: lines created in this task must pass through two points on circle boundary. Mathematically, two points on the line $P_{\circ}$ ($x_{\circ}$, $y_{\circ}$) and P_1 (x_1, y_1) must satisfy the equation of a circle created in first task. If the circle has a centre (h, k) and radius r, then:

$$(x_{\circ} - h)^2 + (y_{\circ} - k)^2 = (x_1 - h)^2 + (y_1 - k)^2 = r^2 \ldots \text{Eq.1}$$

Lines created in this task are controlled by three variables: number of lines (N_{line}), line position and relationship between lines. The computer selects a random value for each variable using a pseudorandom number generator to create different compositions of lines.

- **Number of lines** (N_{line}) could be any number >1 because one line does not create enough number of nodes to be connected in the next task. Also increasing the number of lines will increase the number of nodes and create a very dense composition. However, N_{line} can be instructed to be less than ten and the computer will select "N_{line}" randomly (from values range between 2 to 10) using pseudorandom number generator.

  This number doesn't express the total number of lines created by the whole procedure using the segmentation process. "N_{line}" here is the initial number of lines created by "creating line" task taking into consideration that lines can be created as well from "connecting nodes" task as shown in Figure 18.


(a)

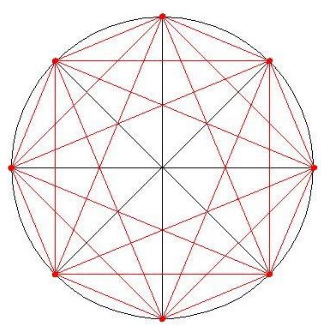

(b)

Figure 18: lines created form the segmentation process.

(a) 4 lines created from "creating lines" task. (b) An additional 24 lines created from "connecting nodes" task. N line =4 while the total number of lines= 28. Source: Author.

- **Lines position** refers to their position in relation to the circle's centre only. In this case, Line position affects two important characters of the pattern: centralization and symmetry. Lines passing through the centre will create a centralized composition and symmetrical pattern.

  If we consider D as the distance between points $P_{\circ}$ and P_1 of the line created in the second task (which are points on circle boundary); then D should be equal to 2r (where r is circle radius) so the line passes through the centre and if d < 2r then the line will not pass through circle's centre as shown in equation number 2.

$2r = \sqrt{(x1 - x_{\circ})^2 + (y1 - y_{\circ})^2}$ … Eq.2

- **Relationship between lines** is controlled by the angles created from lines intersection. The relationship is not controlled by the value of the angles itself, but by the chance of restricting angles values to be equal or not. Equal angles create a radial symmetrical pattern.

3- **Connect nodes**: other lines are created in this process by connecting the points that are considered to be "nodes". Nodes are grouped into two categories: lines midpoints and lines intersections either with the circle or with another line. Before connecting nodes, computer will define what category of nodes will be connected. The selection may vary from selecting all nodes categories or selecting one category, which results in a larger creation of alternative lines compositions.

4- **Decision** (continue process or end): The computer will randomly select to either continue the process of creating lines or to end it. In case of selecting "to continue process", a loop is created and the third task (connect nodes) is repeated with a new definition of nodes, while selecting "end process" will end the procedure and the lines are extended to intersect with the circle. The generated pattern will be provided as an input for the next stage.

4.2 Forward building formation stage

This stage presents the second stage of the proposed technique and it aims to translate patterns that were made in the first stage (backward triangulation stage) into building form by creating the third dimension. Important considerations were taken during the development of the stage to ensure realistic outcomes (solutions) to represent building form without limiting creativity, therefore, it was essential to revisit the traditional design procedure and understand designer's way of thinking and making decisions in the literature review. Despite the fundamental changes proposed in this approach, the output still has to fulfil project requirements as well as designer's intent.

The technique solves building's overall configuration and its functionality in this stage by proposing a process containing three layers. The layers of this process are:

1- Creating two dimensional layouts (layout formation process).
2- Creating of the third dimension (transformation process).
3- (Structural identification process) where structural layouts for the building are proposed.

The results of this processes are alternatives of building forms that can be evaluated in the following stage. Unlike the first stage, this stage requires inputs from the designer. The inputs are mainly used in the second process (transformation process) as restrictions (constraints) which include:

1- Plot area,
2- Plot geometry,
3- Setbacks,
4- Number of floors,
5- Building total height,
6- Ground floor height,
7- Top floor height and
8- Structural span range.

Other inputs are optional to be provided such as desirable number of solutions that will help in directing the process of layout formation.

4.2.1 Layout formation process

In the backward stage, computer has created a pattern consisting of a circle divided into finite elements. As discussed before, this pattern does not present the actual layout of the building; however, it will be used as a base for creating the layout in this stage. Layout formation process starts with taking the pattern as an input and generates number of layouts by selecting a finite element from the pattern and connecting it to another selected one. A selection criteria was developed to direct computer selection of finite elements that includes elements' type and size. Element type refers to the shape of the element such as triangle, rectangle, polygon and circular segment, while size ranges from large element to small element and includes a medium element size. Moreover, a reference and a relationship between two elements should be defined to ensure a real spatial connection before connecting the elements. The reference could be either a side shared between the two elements or a node. The process of selection will keep repeating until goal is reached, in this case, circle area. Figure 19 summarizes the component of forward layout formation process.

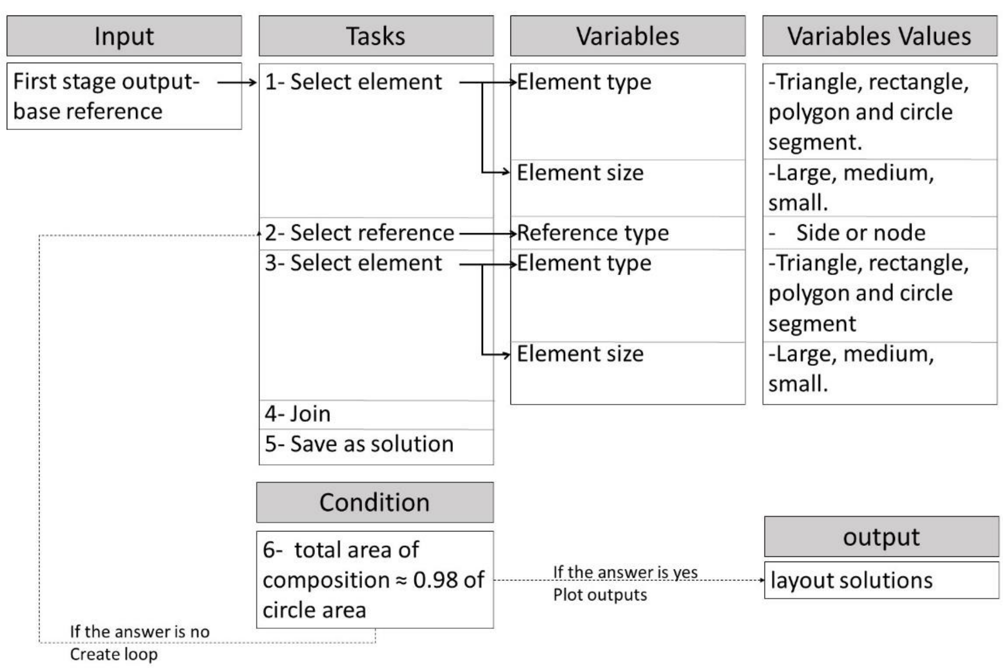


Figure 19: layout formation process components.

Source: author.

“Select element” task is an easy one when applied by humans due to their intelligence. However, for computers, the task must rely on a mathematical logic in order to be understood. This was taken into consideration as the main factor during the development of the process, especially in deciding on the variables. Starting from process input, the base pattern is a network of connected points which can be organized by a tree map to identify group of points that are connected to each point by a path. When selecting the first variable (shape type), the computer selects the number of connected points that will be creating a surface; for example, if shape type is a triangle, the computer will select three connected points and create a surface between them with an area greater than zero.

The second variable (element size) prevents the computer from selecting only the nearest points; moreover, it plays a significant role in creating at least the minimum number of required solutions. In this case, element size is a relative measurement which measures the percentage of element area to the circle (initial shape) area. If the designer requests at least one hundred solutions, then element selection process shall be repeated at least one hundred times before composition area becomes 0.98 of circle area (which is the condition for ending the process).

Consequently, medium sized element is identified as (1/number of solutions) of circle area. For example, if the requested minimum number of solutions is 100, then we need at least one hundred medium sized elements to be selected to satisfy the condition of ending process. Elements sizes are determined as shown in the following:

Ending condition is

Area of the composition = 0.98 of the circle area … Eq.3

Taking into consideration that area of the composition is equal to total area of selected elements that can be calculated by:

Total area of selected elements = Number of repetition × average area of the elements … Eq.4

From equations 3 and 4:

Number of repetition × average area of the elements = 0.98 of the circle area

Average area of the elements= (1/ number of solution) × 0.98 of the circle area

Assuming that average area of the elements defines the medium size and 0.98 of the circle area is almost equal to the circle area;

Medium size = (1/number of solutions) × circle area … Eq.5

Thus, if we consider each third of the total selected elements in the process falls under one category of the element sizes (according to the probabilities) then small size shall be 0.5 medium size (150 small elements at least are needed to satisfy the condition in the example) and large element shall be 1.5 medium size (50 large elements at least is needed to satisfy the condition):

Small size = ½ medium size= 0.5 × (1/number of solutions) × circle area … Eq.6

Large size= 1 ½ medium size = 1.5 × (1/number of solutions) × circle area… Eq.7

Table 8 summarize the variables of "select element" task in the forward building formation process.

Table 8: variables of "select element" task list.

Source: author.

List of variables		
Variables' number	The variable	Variable value
1	Element shape	triangle
		rectangle
		polygon
		circle segment
2	Element size (with relation to circle area and number of requested solutions)	large= 1.5 × (1/number of solutions) × circle area
		medium= (1/number of solutions) × circle area
		small= 0.5 × (1/number of solutions) × circle area

In the second task "select a reference", the variable also relates to the number of points shared between the selected elements in task 1 and 3. By selecting side as reference, two points used in the creation of the first surface (of the first element) will be used again to create the second surface (of the second element).

The output of this process is a large number of solutions instead of one layout solution. As presented in Figure 19, the result of each run is saved as a solution and the process will continue until the condition is satisfied.

4.2.2 3D Transformation process

In 3D transformation process, computer generates the third dimension of the building. The process includes analysing designer's inputs (project requirements) that are related to the building configuration such as; plot geometry and size, setback, number of floors, maximum building height, top height and ground floor height. These requirements guide the 3D transformation process to avoid the generation of chaotic geometries that are hard to be interpreted and translated into a realistic building design. However, in the process, the amount of restrictions was minimized to avoid the repetition of particular style which at the end weakens creativity.

The process starts with creating a generic template based on designer's input. This template presents the dimensional limitations (limitations form regulation and project requirements) as well as the area involved in the process. Figure 20 illustrates an example of a generic template.

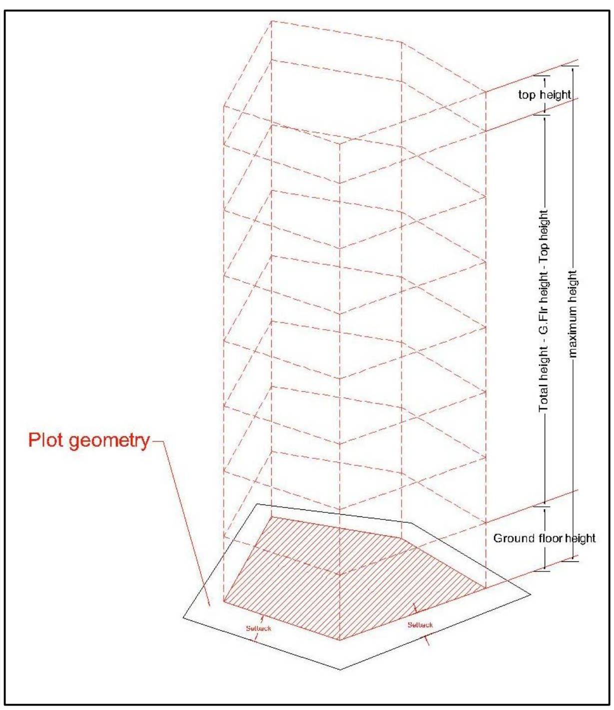


Figure 20: example of generic templet generated in 3d transformation process.

Source: author.

After creating the template, the actual process of creating building form will start by selecting operation type, and accordingly inserting number of layouts needed for the operation, while the process continues with arranging them and finally finishes the operation. The result will be saved as a solution and the process will keep repeating until reaching a specific number of repetitions that are specified based on a number of alternatives requested by the designer. Figure 21 summarizes the main component of the process.

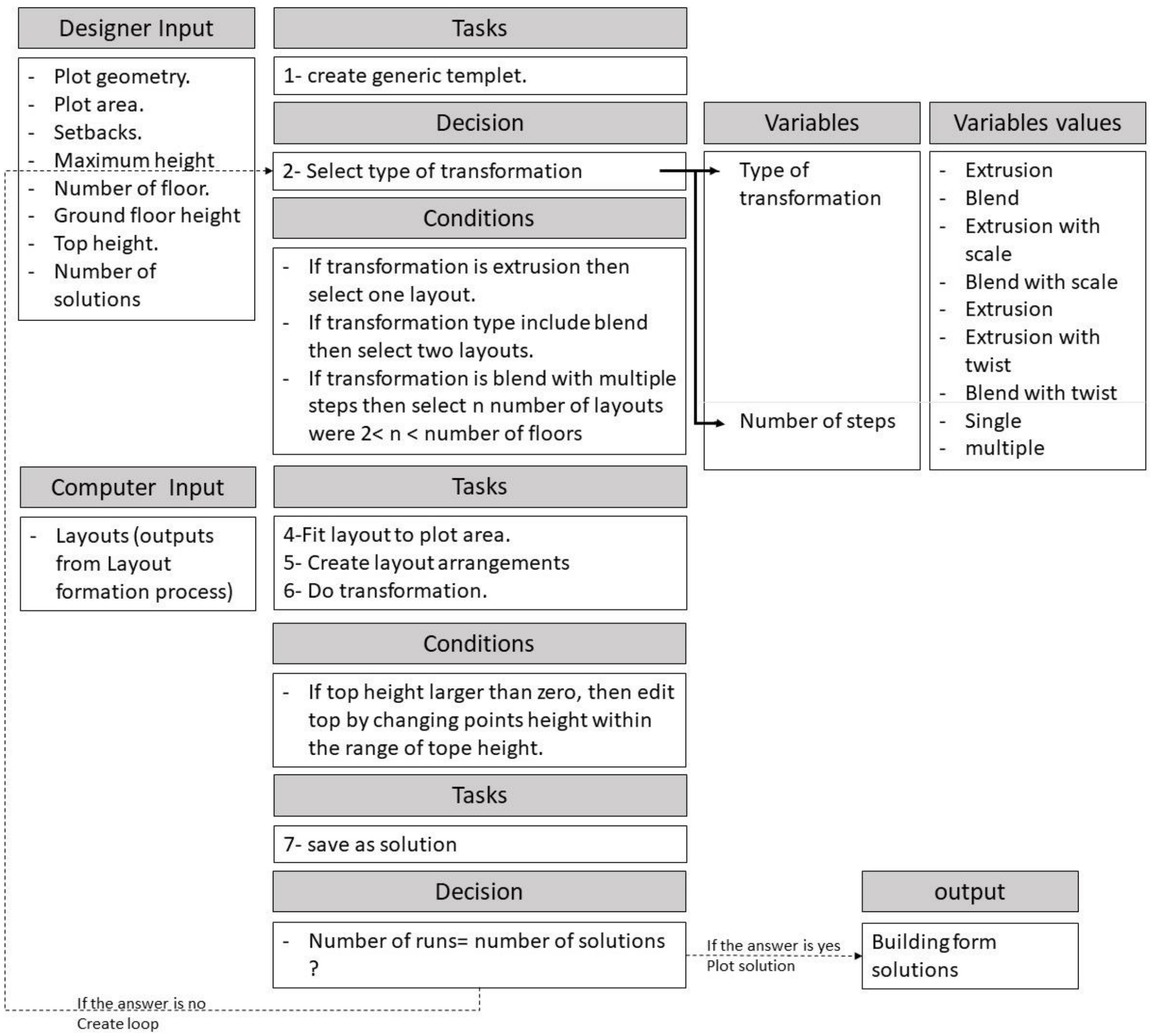


Figure 21: 3d transformation process components.

Source: author.

As shown in Figure 21, two variables control selecting transformation: type of transformation and number of steps. Type of transformation is either extrusion or blend, where both of them can be combined with two other types of transformation which are scale and twist. While the number of steps depends on transformation type, in case of blend transformation, multiple steps mean different blends will be created between floors and in case of extrusion multiple steps mean that shapes creating the layout will be extruded to different levels (different heights).

Transformation selection controls the number of layouts needed to create the transformation, hence, number of layout will be taken as an input from previous process's output. Extrusion transformation type requires only one layout, in contrast, blend transformation type requires two layouts and if the blend transformation is done in multiple steps then number of layout needed for the transformation can be any number between 2 and the number of floors. Before doing the transformation, layouts are placed in the generic template and scaled to fit the plot boundary.

Regardless of the transformation type, all created forms will have a flat roof. In order to create a more dynamic shape for the roof, the computer can define points on the top layout and change its height randomly within the range of "top height" input that is provided by the designer as shown in Figure 22.

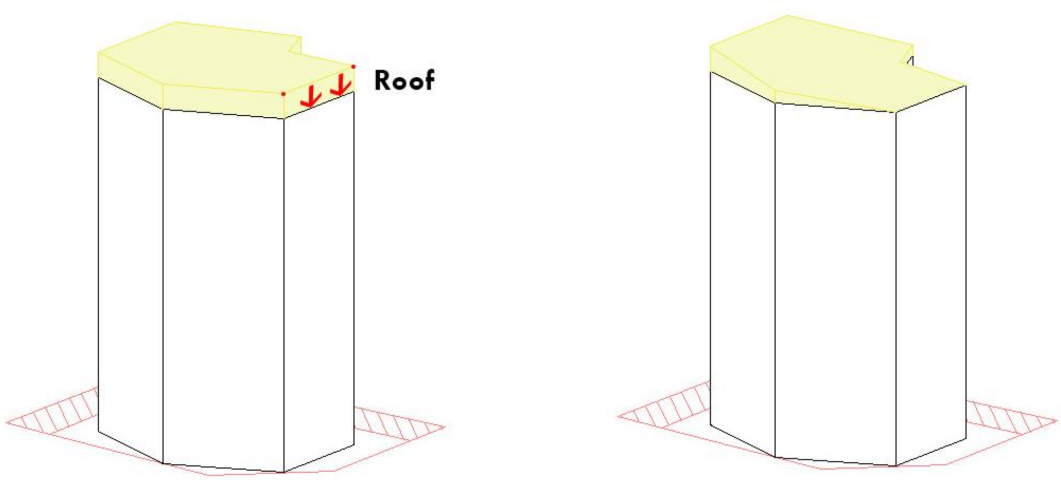


Figure 22: editing roof shape in 3d transformation process.

Source: author.

The final generated form is saved as a solution in the output and the process is repeated until the system saves enough number of solutions to be evaluated in next stage. If the designer wants to have one hundred solutions, the repetition will probably be much more than hundred since many solutions will be excluded in the evaluation process. The estimation of minimum number of solutions shall be created before excluding process requires a mathematical analysis of large number of examples that implements this technique.

4.2.3 Structural identification process

The final process in forward building formation stage is identifying potential structural layouts in where a simple layout consists of columns and beams is proposed. This process depends on the reference plan created in backward triangulation stage and used as a base for layout formation stage. This mainly happens by proposing algorithms that picks up nodes and calculates the spacing and the span; accordingly suggests different arrangements. The spacing and span between columns shall be given as input from the designer, so the computer uses it as criteria for the structural identification. The picked nodes are the points that share the same location in all floors – points that are vertically aligned. The computer can also suggest minor changes on certain point's locations when necessary to create a structural layout, only when the changes will not be affecting the form generated earlier.

4.3 Evaluation stage

In this stage, each solution generated from the previous stages is quantitatively and qualitatively evaluated based on multiple criteria before presenting it to the designer in the next stage. The criteria include:

1 Spatial criteria,
2 Structural criteria and
3 Environmental criteria.

The importance of the evaluation process is to help the designer to choose the most convenient solution according to his/her personal preference by making comparison between solutions. Moreover, it acts as a filter for the results by measuring the efficiency of each solution as a "design proposal" of building form and eliminates the inconvenient ones.

The evaluation stage depends on exclusion and ranking strategies. A number of variables falls under the mentioned criteria will be analysed and calculated for each solution such as:

4 Built up area,
5 Coverage area,
6 Best-fit,
7 Footprint,
8 Structural potential and
9 Complexity.

All of the variables fall under the parameters which the designer decides to maximize or minimize in the generated solution except for the built up area or the coverage area. Built up area generally is a goal that must be achieved during the design process while coverage area is constrained by maximum value in plot regulations. Therefore, both of them were set as exclusion criteria that will be evaluated first in order to select best solutions and to exclude the rest.

4.3.1 Built up area

Built up area is considered amongst the most significant requirements of a project hence it indicates the maximum utilization of plot area as well as other cost considerations. In general, the required built up area is decided based on the client's requirements either by achieving the maximum area from the potential design or by specifying a cost limitation.

In case of maximum utilizing:

Built up area= plot area without setbacks × total number of floors

In the case of cost limitation:

Built up area= defined cost ⁄ cost of meter built up area

The computer calculates floors area for each generated form and compares them, to the required. Consequently, the solutions falling out the required built up area are excluded. The number of solutions resulted after exclusion was specified earlier by the designer in the inputs.

Other important measurement can be calculated from the built up area is the functional area. The functional area is calculated by extracting all areas with small height that building user cannot stand comfortably under it. These areas are created in some transformation types such as twist and blend where the slope of the wall occupies some of floor's area. If the minimum height of the functional area is specified in project requirements, for instance, minimum height is 2.7m then all area falling under this constrain is continued off from the built up area.

4.3.2 Coverage area evaluation

Converge area is another important measurement specified most of the times in plot regulations and is referred to the building's projected area on the plot surface. Coverage are considers to be a spatial limitation for the building that is mostly affect the floors above the ground where ground floor dimension in reality is limited by the setbacks, the upper floors areas may exceed setbacks if its projection in the ground still within the regulated coverage area.

4.3.3 Best fit

This measurement refers to the relationship between the plot shape and the layout shape and it is measured by calculating the percentage between ground floor areas to the plot area where the largest percentage is considered to be the best fit. Best fit indicates the maximum utilization of ground floor area for construction. However, if the designer wants to utilize some area for landscaping then solutions that have the least (best fit) percent show better opportunity to incorporate landscaping in the design.

4.3.4 Building Footprint

Footprint is considered as an environmental measurement that indicates the impact of human activity on land. Building footprints refers to the area of the plot that is occupied from the building and is equal to the ground floor area.

4.3.5 Structural potential

This variable is linked to the structural identification process where structural layouts were proposed for each generated form. The computer will rank the solutions by analysing the forms with structural layouts best matching the required structural span/spacing and require the least changes on nodes locations.

4.3.6 Complexity (Archi-form complexity index)

Mathematically, complex shape is a shape created from two or more shapes. However, in architecture, building form consisting of two masses is still considered a simple form. Complexity of building form increases with adding more masses, however, it cannot be easily categorised as simple or complex forms. In other words, complexity in architecture forms is a relative scale more than a quantitative measurement. In order to classify the results based on their complexity, Archi-form complexity index was developed.

The index deals with two levels of complexity: the complexity of the layout; (mainly used in generating the forms) and complexity of the transformation process. Four factors are proposed to measure complexity which are: number of layouts used in the transformation process, number of edges for each layout, number of operations and type of operation. Points system is developed for evaluating the four factors as shown in Table 9.

Table 9: Points system to evaluate vertical complexity factors for building form.
Source: author.

Points system for evaluating complexity for each factor		
Factor	**Details**	**Points**
Number of layouts (N layouts)	For extrusion = 1 layout For single blend = 2 layouts For multiple blends = variable	Total points = N layout × point
Number of layout edges	For each edge, 0.5 point If there is more than one layout used in the operation then the layout that have the largest number of edges is used for calculating the total number of points	Total points = N edges × 0.5 point
Number of operations	For single extrusion, blend= 1 For extrusion, blend combined with scale or twist= 2 For multiple extrusions or blends= variable. each twist with 45˚ consider as one operation	Total point =N operation × point For multiple extrusions = N operation × 0.25 point
Type of operation	The operations are ranked as the following from the simplest to the most complex transformation: Extrusion, Scale, Blend and Twist.	Total point Extrusion = 1 point, Blend =3 points Combined with scale ×2 Combined with Twist ×5

The index range between 0 and 1 where form that has the largest number of points will be given the value "1", and the complexity index for the other forms will be estimated accordingly. For example, to measure the complexity of solution number 1 which was generated by multiple blends for 3 layouts: first layout with 5 edges, second layout with 7 edges and the third layout with 8 edges.

According to Table 9:

Points from number of layouts= 3 points.

Points from number of layout edges= 0.5× 8= 4 points

Points from number of operations= 1× 3= 3 points

Points from type of operations= 3 points

Total number of points= 13 points

If we considered solution 2 and solution 3 were generated in the same process but with different operations have a total point of 19 points and 8 points respectively, then solution 2 index is considered as 1 and accordingly index values of both solution number 1 and 3 are been calculated:

Solution 1 index= 12/19 = 0.68

Solution 3 index= 8/ 19= 0.42

It is important to say that complexity may be desirable by some designers who view complexity as an indication of design's uniqueness and creativity in the generated form. Subsequently, it could be part of exclusion criteria as it may increase building costs.

4.4 Selection stage

The final result of the process is presented at this stage for the designer with solution evaluation and ranking. The designer then, is given the choice to pick any of the options based on his/her conviction. If the designer is not satisfied with the presented solutions, the process can be repeated and a different set of solutions will be displayed as a result of the changeable computer preference variables used in the process.

CHAPTER FIVE

RESULTS AND DISCUSSION

Chapter 5 Results and discussion

This chapter illustrates the implementation of the proposed generative design technique for conceptual building form exploring in a serious of design examples. The chapter mainly aims to investigate the technique capability in formulating a diverse design spaces and generating unexpected design possibilities that are also functional.

The results of implementing the proposed processes for each stage are organized in four sections. In each section, examples are developed to test out variables efficiency in support the process to generate alternatives first, then other serious of examples were indicated to generate a building forms for a hypothetical project requirements that was developed according to plot regulations presented in Figure 23.

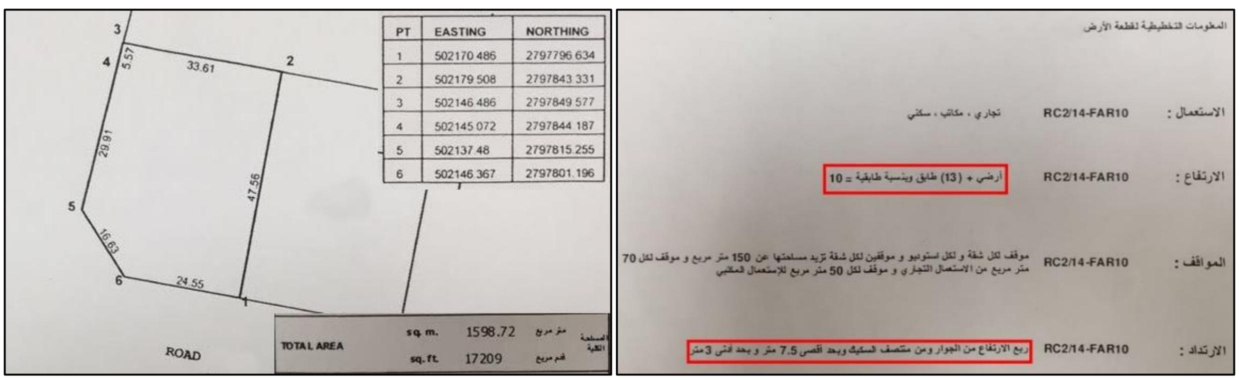


Figure 23: Plot affection plan show plots' information and regulation.
Source: Dubai municipality (2017).

All examples illustrate a direct implementation of the processes by the researcher which were visualized using AutoCAD. Moreover, an algorithmic editor interface Grasshopper incorporated with visual programming language Rhino3d were used to formulate the associations of some parts "variables" of the generative technique into code. This is mainly used in backward triangulation stage and layout formation process in the forward building form stage. These two parts of the technique are proposed to replace tasks that their results are unexpected, unmeasurable and totally depends on aspects related to the designer taste, style and experience in the traditional design procedure; which makes them extremely complex design problems.

In the previous chapter, both of them were developed by analysing the outputs of the traditional tasks of creating design ideas and suggest a totally new approach for creating similar outputs by a pure geometric abstraction of basic spatial relations. By comparing the general practice of design in architecture and the new technique that is proposed in this thesis, these two parts of the methodology are not a direct translation of the design procedure into computation logic. For this reason, a judgmental based on a direct implementation of tasks by researcher in these two parts will not provide a clear insight of their tendencies in the process. While the rest of technique stages and processes aim to systemise the design procedure of building form in order to provide a theoretical framework. The encoding for these parts are not a problematic since they require functions that have been already encoded in a wide range of software such as Revit, AutoCAD and BIM.

The hypothetical project requirements used to provide an input data for the examples is presented in Table 10. Information of plot area, building maximum height and setbacks were taken from the affection plan of the plot and the ground/ floor/ roof heights as well as structural span and number of solution are assumptions.

Table 10: plot information and project requirement definition.
Source: author.

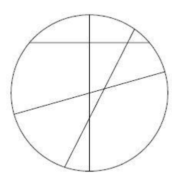
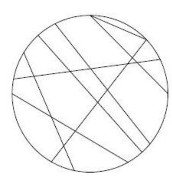
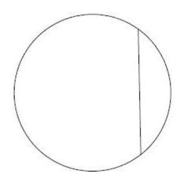
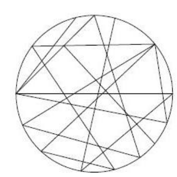
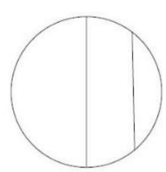
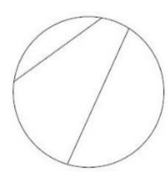
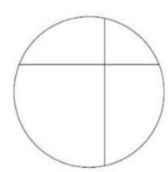
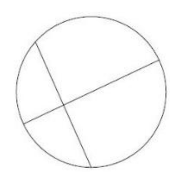
Plot information and project requirements	
Plot area	1598.72 m2
maximum height	ground +13 floor+ roof
Setbacks	fourth the height from neighbors (maximum =7.5m and minimum = 3m)
Building hieght	53 m
Number of floor	15 (round + 13 floor + roof)
Floor hieght	3.5 m
Ground floor hieght	4 m
Roof hieght	3.5 m
Structural span	6- 8 m
Number of solutions	16

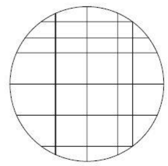
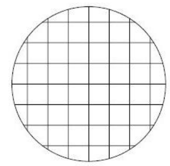
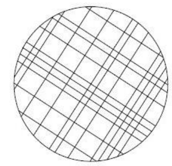
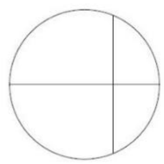
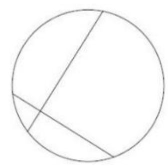
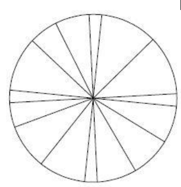
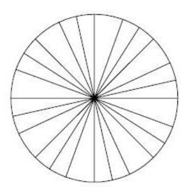
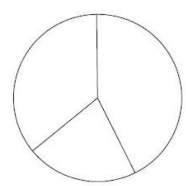
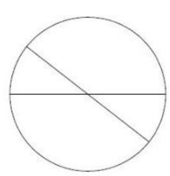
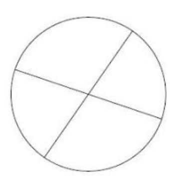
5.1 Backward triangulation

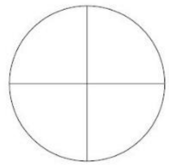
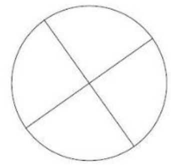
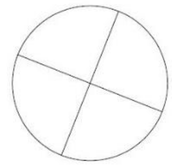
Before developing examples on segmentation processes that follow backward triangulation technique to create a base pattern, the different formulation of each task generated from changing the variables was separately presented in Table 11 and Table 12.

The first task in backward triangulation stage “create lines” is controlled by three line characters. In the first row of Table 11 all lines characters are not defined. This allow the generation of unlimited number of results that some of them cannot be used for creating the pattern. This happens when number of nodes generated from the task is not enough to make connections, such as result number 3 in the first row. Going further with the examples, more constraints are incorporated in the task description. Even though introducing constrains lowers the number of alternative generated from implementing the task, it generates more organized patterns. Moreover, comparing the results in each row shows that the three line characters are efficient in creating different styles of patterns.

Table 11: results of task 1 "Create line" in the backward triangulation stage.
Source: author.

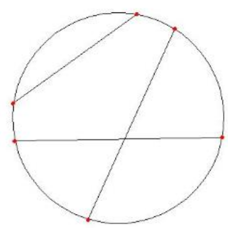
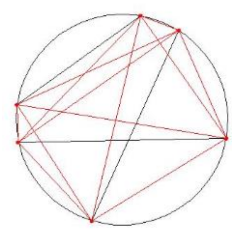
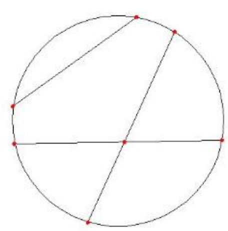
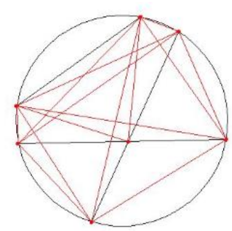
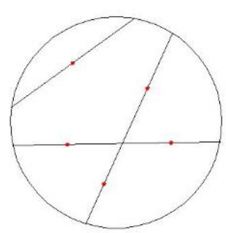
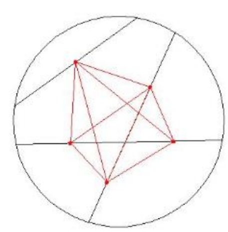
<table>
<tr><th colspan="2">Task description</th><th colspan="4">Result</th></tr>
<tr><td colspan="2">Examples present the case of no variables are defined
“Create lines”</td><td rowspan="4">1
</td><td rowspan="4">2
</td><td rowspan="4">3
</td><td rowspan="4">4
</td></tr>
<tr><td>N lines</td><td>undefined</td></tr>
<tr><td>Line position</td><td>undefined</td></tr>
<tr><td>lines relationship</td><td>undefined</td></tr>
<tr><td colspan="2">Examples present constraints the number of lines to 2 lines.
“Create two lines</td><td rowspan="4">1
</td><td rowspan="4">2
</td><td rowspan="4">3
</td><td rowspan="4">4
</td></tr>
<tr><td>N lines</td><td>two lines</td></tr>
<tr><td>Line position</td><td>undefined</td></tr>
<tr><td>lines relationship</td><td>undefined</td></tr>
</table>

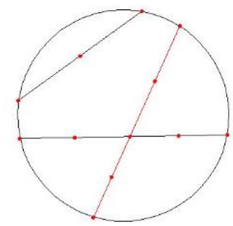
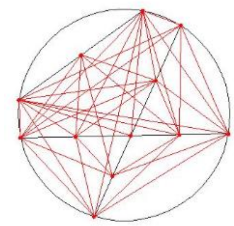
Examples of Restriction the relationship between lines “Create lines with equal angles between them”		1	2	3	
N lines	undefined				-
Line position	undefined				
lines relationship	equal angles				
Examples of Restriction both of number of lines and relationship between lines “Create two lines with equal angles between them”		1	2		
N lines	two lines			-	-
Line position	undefined				
lines relationship	equal angles				
Examples of Restriction line position. “Create lines pass through the centre”		1	2	3	
N lines	undefined				-
Line position	pass through the centre				
lines relationship	undefined				
Examples of Restriction both of number of lines and position “Create two lines pass through the centre”		1	2		
N lines	two lines			-	-
Line position	pass through the centre				
lines relationship	undefined				

Examples on Restrict all lines characters “Create two perpendicular lines pass through the centre”		1	2	3	
N lines	two lines				-
Line position	pass through the centre				
lines relationship	equal angles				

In Table 12, results of applying “connect nodes” task are presented. Each row in the table shows samples of composition created by changing the definition of nodes for the same input.

Table 12: Results of task 2 "connect nodes" in the backward triangulation stages.
Source: author.

Task description	**Identifying nodes**	**Connect nodes result**
Examples of defining nodes as lines intersections with initial shape		
Examples of defining nodes as lines intersections with initial shape and lines intersections with other lines		
Examples of defining nodes as lines midpoints		

Examples of defining nodes as lines intersections and midpoints		

“Connect nodes” task in the technique is repeated for random number of times. The results of applying the task as shown in the examples are various but limited. If nodes were defined as lines midpoints and the process was repeated with the same nodes definition, then nothing will be added to the composition. This was solved by broaden the process from only creating lines by connecting nodes to includes lines segmentation which is been discussed in the previous chapter as the necessity to redefine lines after each task. Line segmentation creates a new lines midpoints to be defined as nodes and allows the process to efficiently create wider range of alternatives.

5.1.1 Examples on backward triangulation

In order to understand how tasks work together to generate different compositions, three sets of backward triangulation processes are presented in Table 13, Table 14 and Table 15. All sets share the same number of lines variable in “create line” task - 4 lines- and differ in lines position and relationship variables as well as nodes definition. The choices of variables for each set was done in a way that allow further comparison between the results. Figure 24 presents the similarity and differentiation between the three examples.

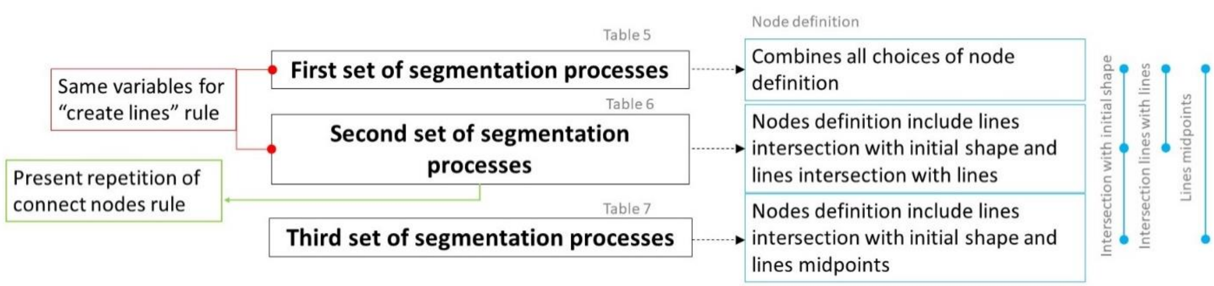


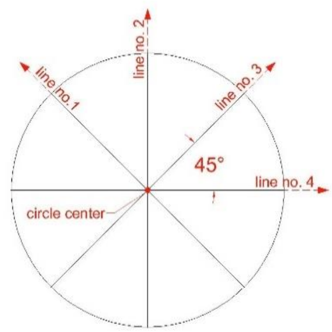
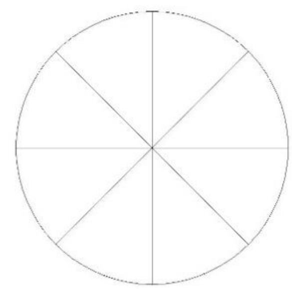
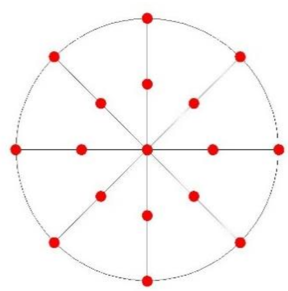
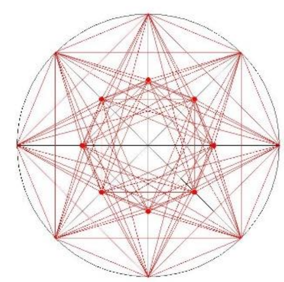
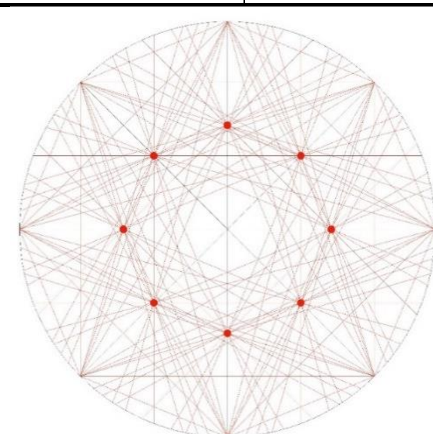
Figure 24: backward triangulation examples structure.

Source: author.

- First set of backward triangulation processes.

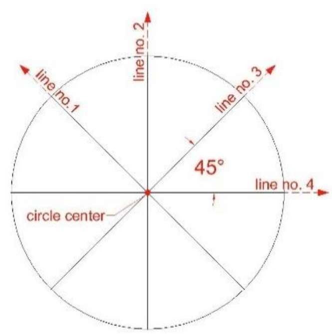
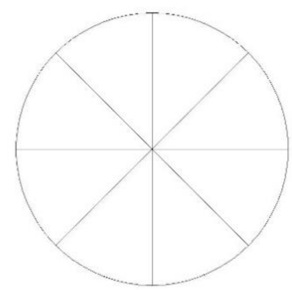
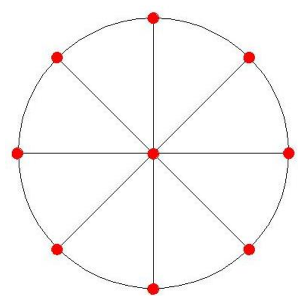
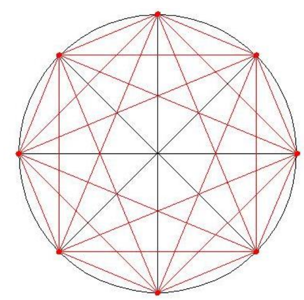
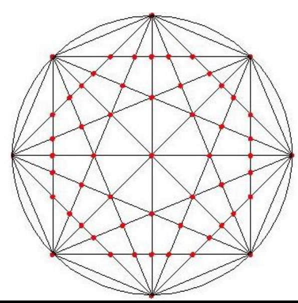
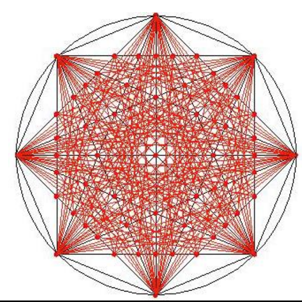
Table 13: First set of backward triangulation processes- example 1.

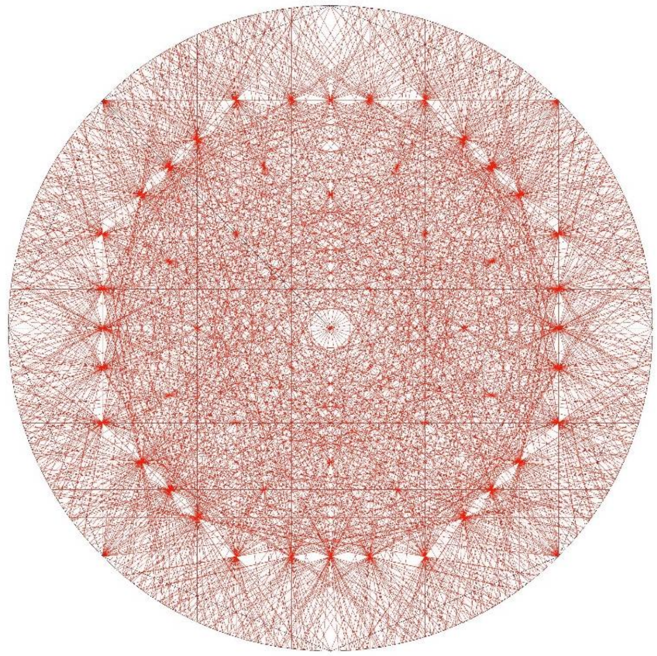
Source: author.

Step no.	Details	Controlled variables	Analysis of the process	Result
Step 1	Create circle	-	-	-
Step 2	Create lines	Number of lines: 4 Lines position: pass through centre Line relationship: Equal angles.		
Step 3	Connect nodes	nodes are defined as: • Line intersection with initial shape • Line intersection with line • Lines midpoint		
Step 4	Continue process?	No then extend lines		
Step 5	**End process**			

- Second set of backward triangulation processes.

Table 14: Second set of backward triangulation processes- example 2.
Source: author.

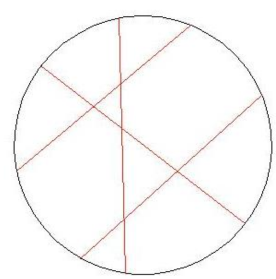
Step no.	Details	Controlled variables		Analysis of the process	Result
Step 1	Create circle	-		-	-
Step 2	Create lines	Number of lines	4		
		Lines position	pass through centre		
		Line relationship	Equal angles.		
Step 3	Connect nodes	nodes are defined as line intersection with initial shape			
Step 4	Continue process?	yes		-	
Step 5	Connect nodes	nodes are defined as line intersection with line			

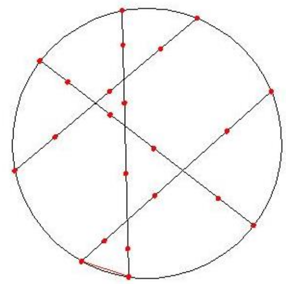
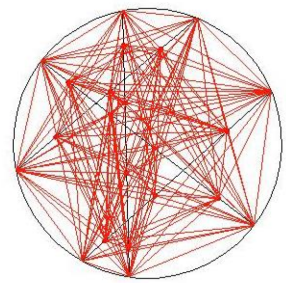
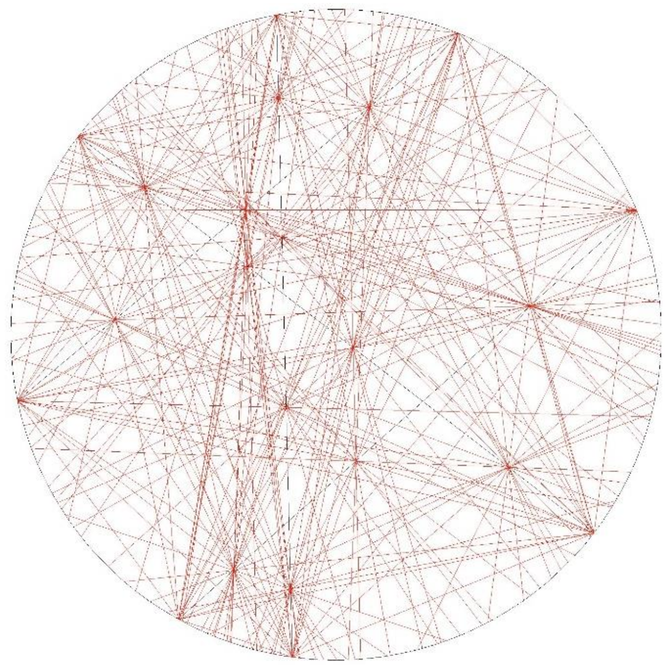
Step 6	Continue process?	No then extend lines	
Step 7	**End process**		

- Third set of backward triangulation processes.

Table 15: third set of backward triangulation processes- example 3

Source: author.

Step no.	**Details**	**Controlled variables**		**Analysis of the process**	**Result**
Step 1	Create circle	-		-	-
Step 2	Create lines	Number of lines	4		
		Lines position	undefined		
		Line relationship	undefined		

Step 3	Connect nodes	nodes are defined as: • Line intersection with initial shape • Lines midpoint		
Step 4	Continue process?	No then extend lines		
Step 5	**End process**			

The first and second sets of backward triangulation processes have the same variables of "create lines" task where both are constrained by the position of lines to pass through initial shape centre and the relationship between the generated lines (equal angles between lines). In comparison, the third set of processes is not constrained with lines position and relationship. This has an effect on types of pattern generated from the process where results of the first and second sets are self-centralized and radially symmetrical unlike the result of the third set.

For "connect nodes" task, even though there was a common definitions of nodes between sets but none of them created exact similar connections. Moreover, the result of the second set shows a denser pattern than the one resulted from the first set despite the fact that nodes definition of the first set included more nodes than the nodes definition in the second set. This is related to the repetition of “connect nodes” task in the second set. In each repetition, lines are exponentially multiply and so the number of nodes.

Finally, all examples generated compositions with large areas around the boundary that need to be segmented before the end of the process. The addition of extend lines function balance the final results and solve the problem.

5.1.2 Grasshopper/ Rhino3d application to illustrate the system

The implementation of backward triangulation stage continued with developing parametric model in Grasshopper/ Rhino3d application to illustrate the systematic variation generated from manipulating the variables by using mathematical functions.

Figure 25 shows the complete visual scripting components for the backward triangulation stage. The model consists of four groups of components;

- The first group is to generate the initial shape (circle),
- The second group is to generate lines,
- The third group is to identify nodes and make connection between them and finally
- The fourth group is to repeat connect nodes task.

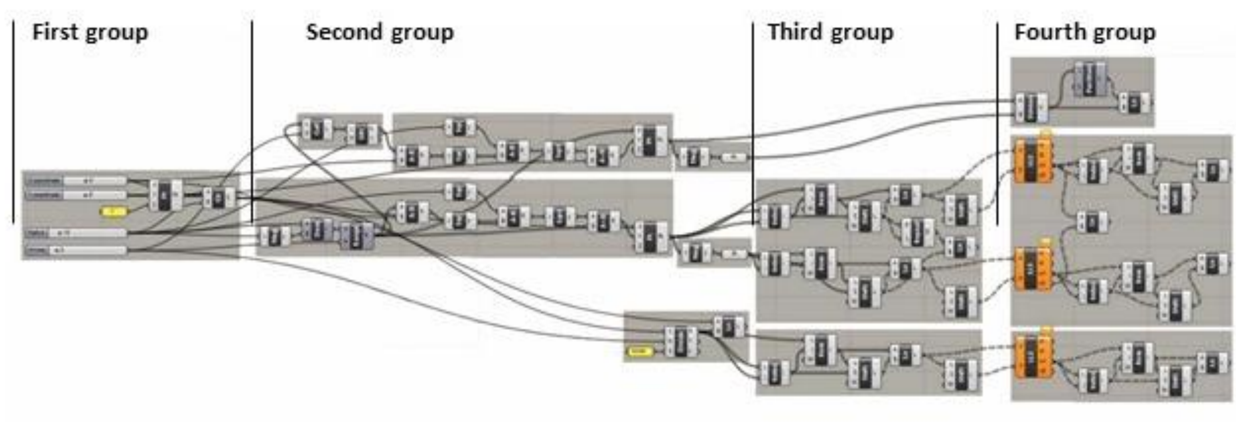


Figure 25: Grasshopper visual scripting components for the backward triangulation stage.

Source: author.

The model is divided into three paths, each represents a case of variables definition for lines characters. All paths share the same start which is the initial shape (circle) components as shown in Figure 25. The circle is controlled by its radius and (x, y) coordinate of the centre point. The use of number slider component is to ensure that any change on radius value or circle centre position will not affect the generated patterns which supports the argument that no input data is needed for the backward triangulation stage.

The circle radius and (x, y) coordinates of the centre point also connected to other components in each case model as shown in Figure 26. In general, these variables is used to find the range of x coordinate and y coordinate of points that are located on the circle boundary which are needed to create lines inside the initial shape.

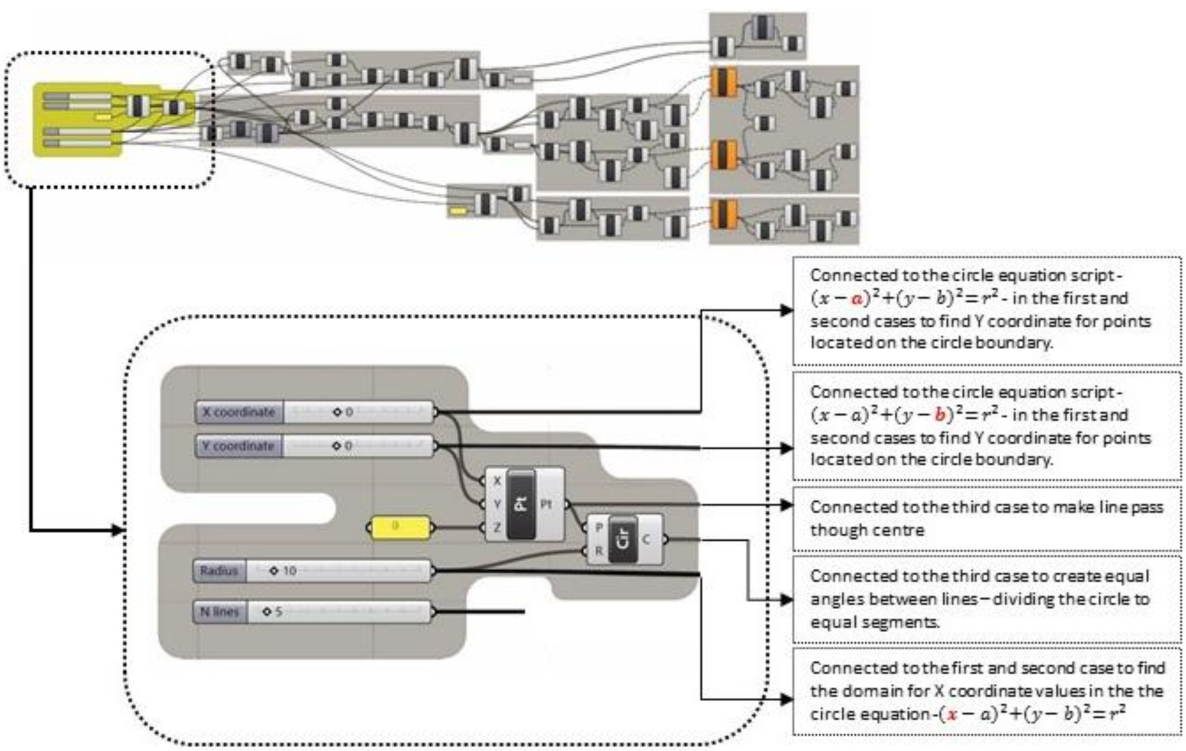


Figure 26: first group of component- initial shape encoding.

Source: author.

The first path introduces the choice of creating lines with no constrains on lines position and relationships, also, nodes are defined as lines intersections. Figure 27 presents the first case script.

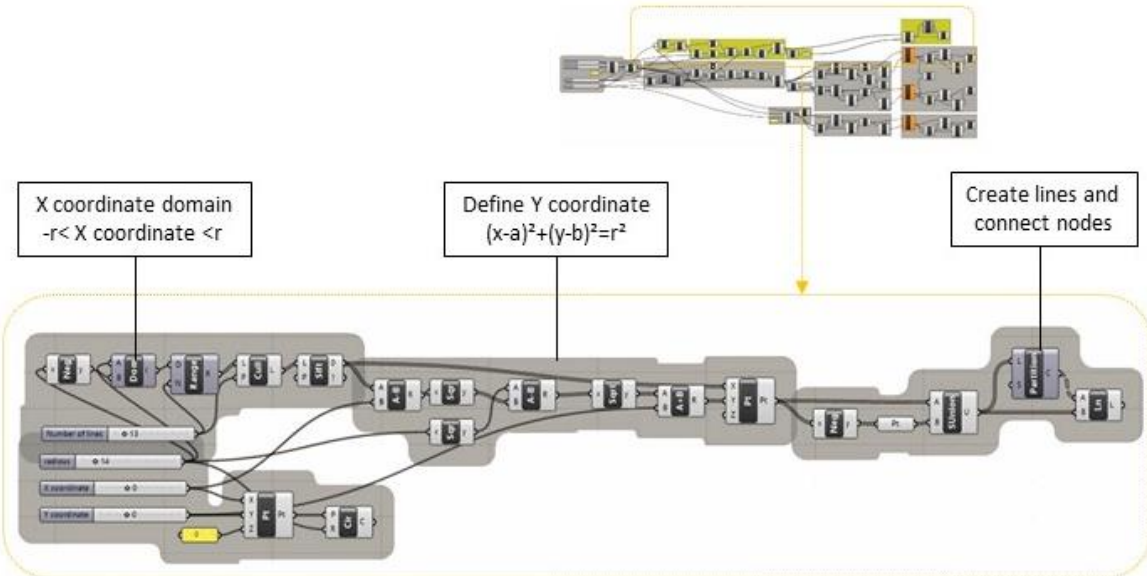


Figure 27: first case visual script.
Source: author.

Figure 28, Figure 29 and Figure 30 illustrate three examples of pattern resulted from the first path. The difference between the examples is number of lines created in “create lines” task.

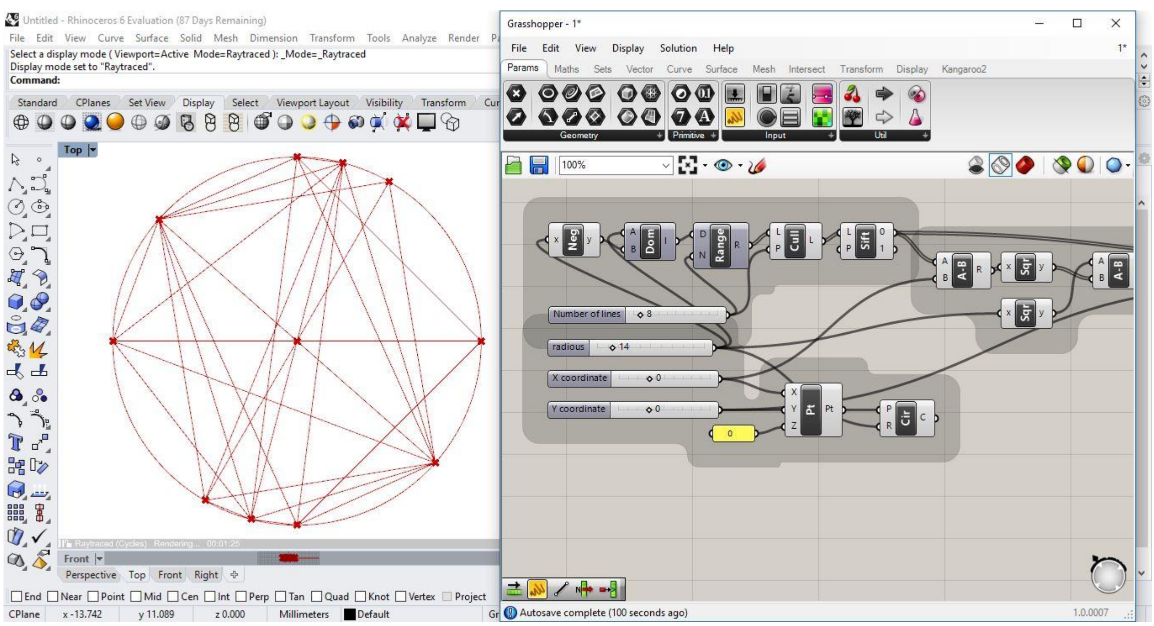


Figure 28: Results of the first case script for 8 lines.

Source: author.

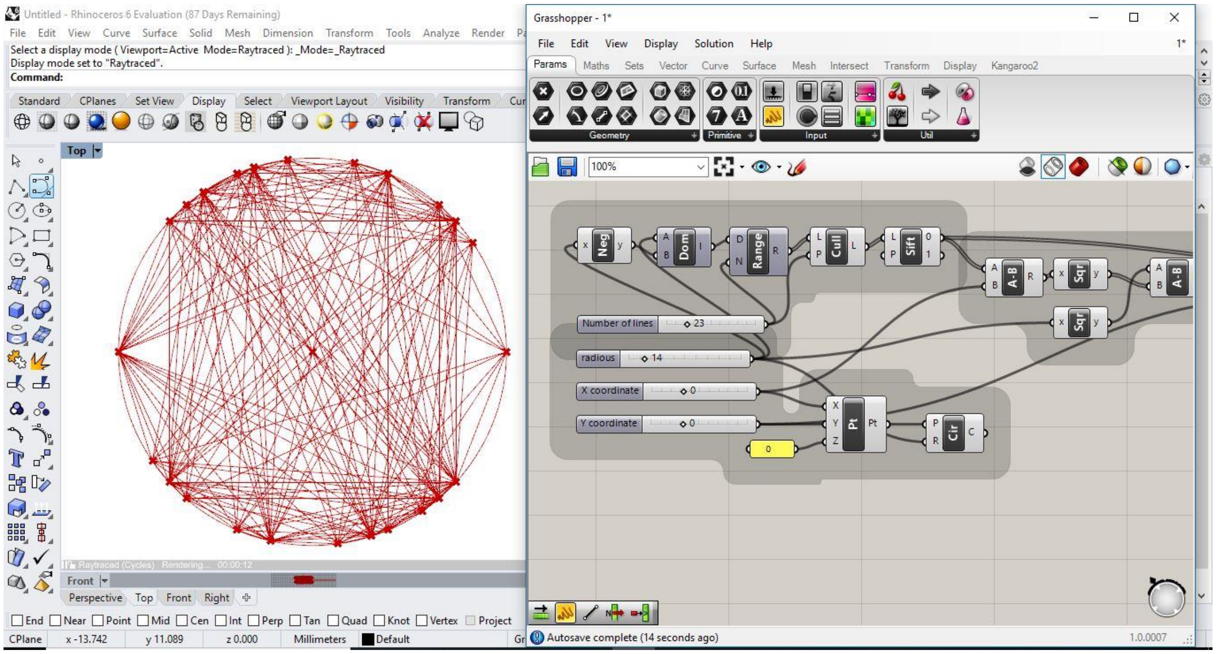


Figure 29: Results of the first case script for 23 lines.

Source: author.

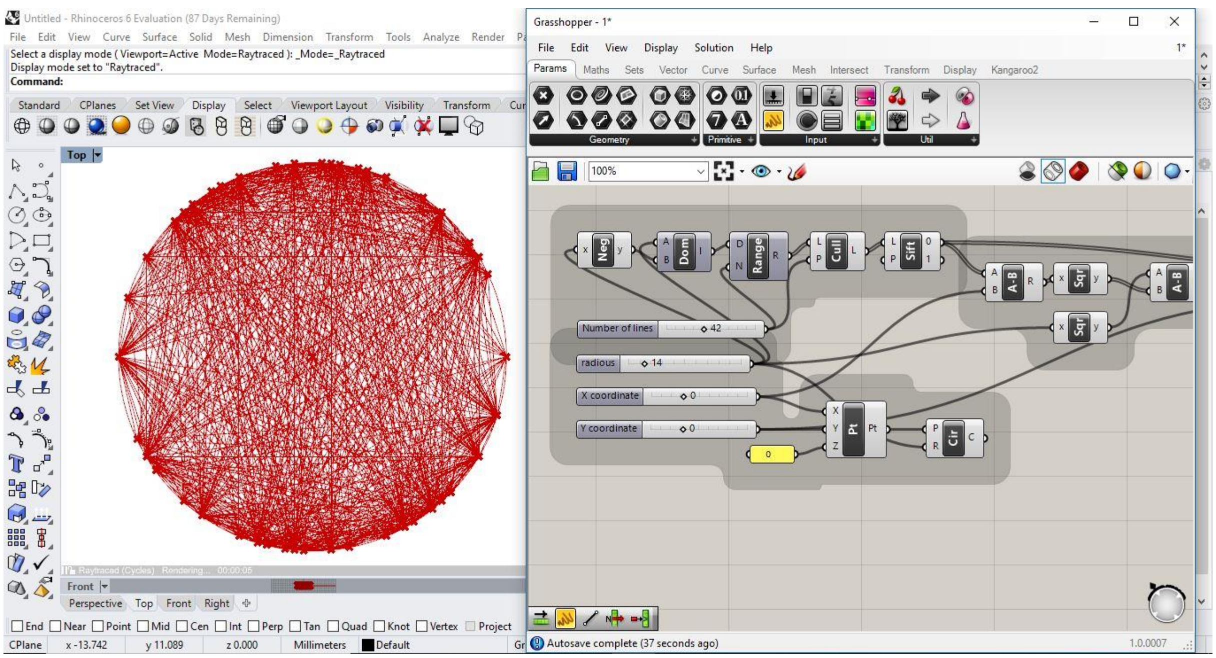


Figure 30: Results of the first case script for 42 lines.

Source: researcher

The second path introduces the choice of creating lines that pass through the circle centre as shown in Figure 31. However, the relationship between lines (the angles between intersected lines) is not constrained. Nodes are defined as lines intersection and the process was repeated.

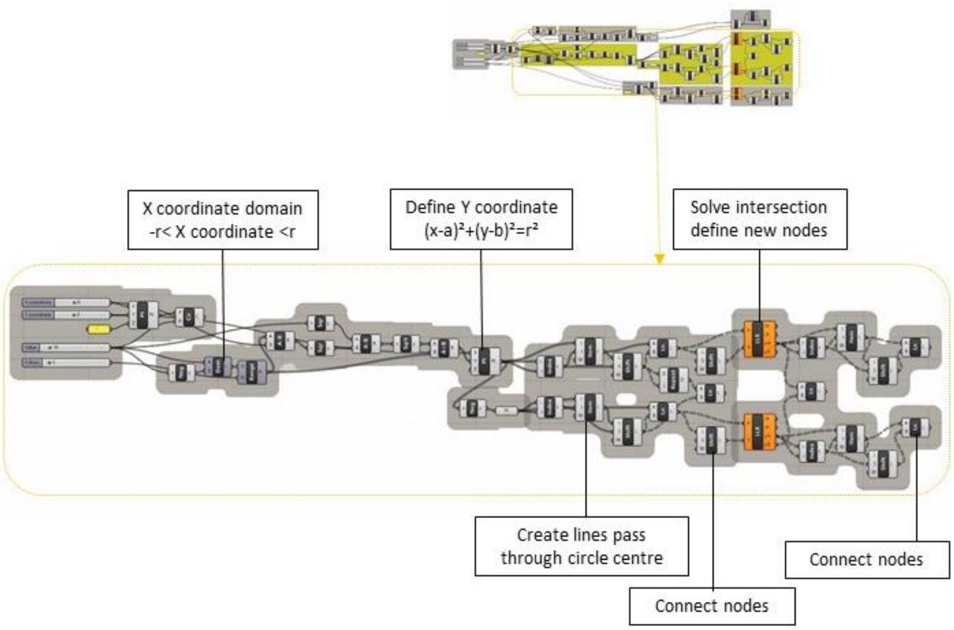


Figure 31: Second case visual script.

Source: author.

Figure 32, Figure 33 and Figure 34 illustrate results of testing out the second case script.

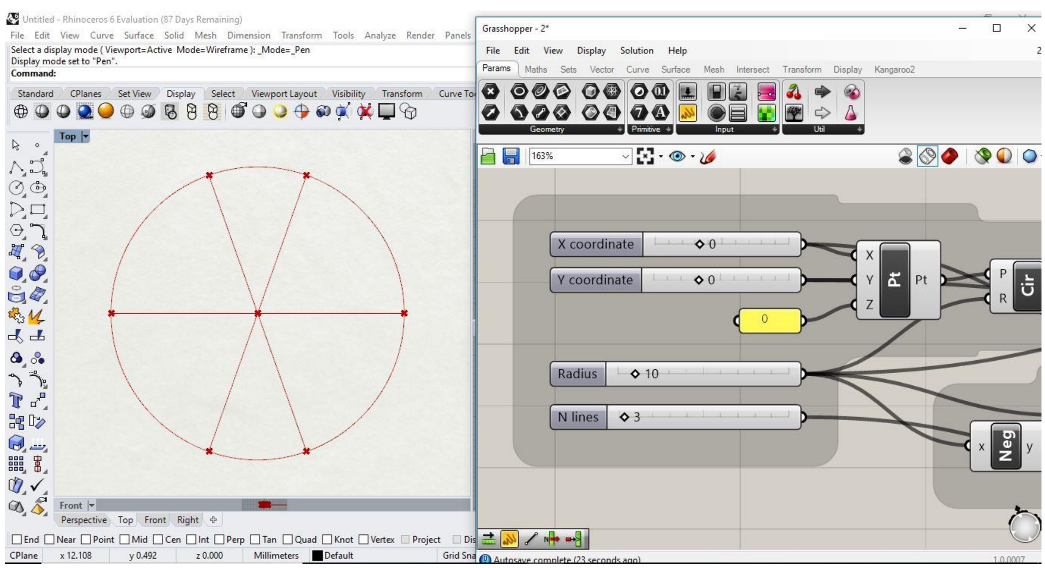


Figure 32: results of first step (create lines) by the second case in grasshopper model.

Source: author.

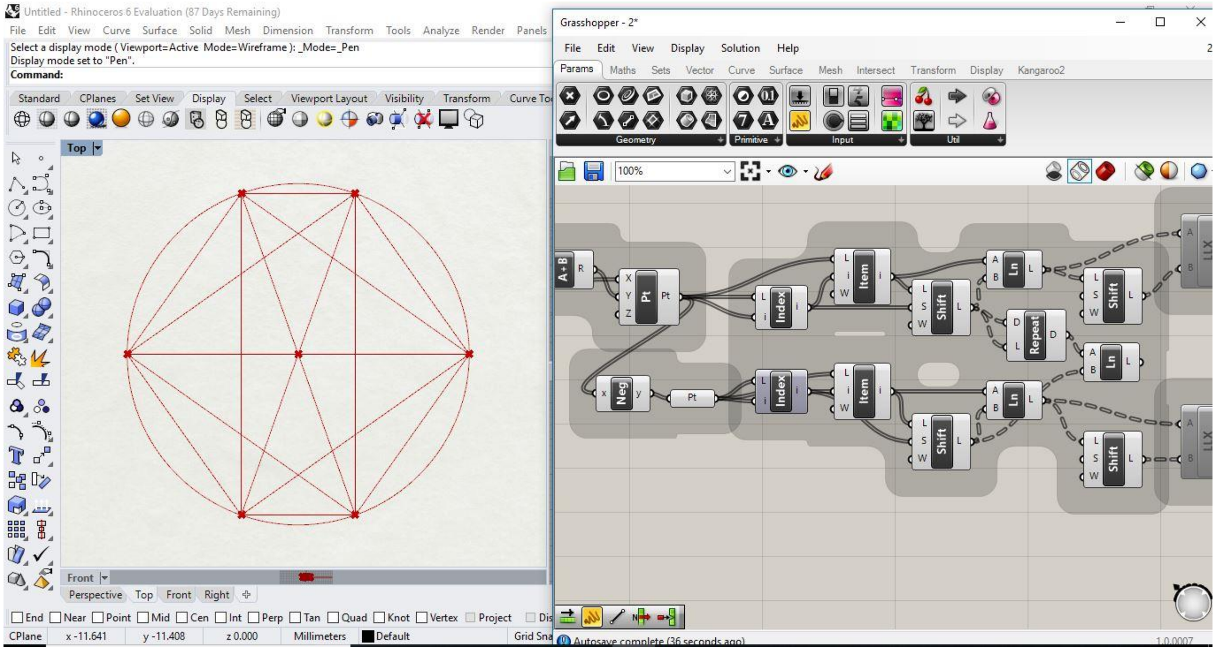


Figure 33: results of second step (connect nodes) by the second case in grasshopper model.

Source: author.

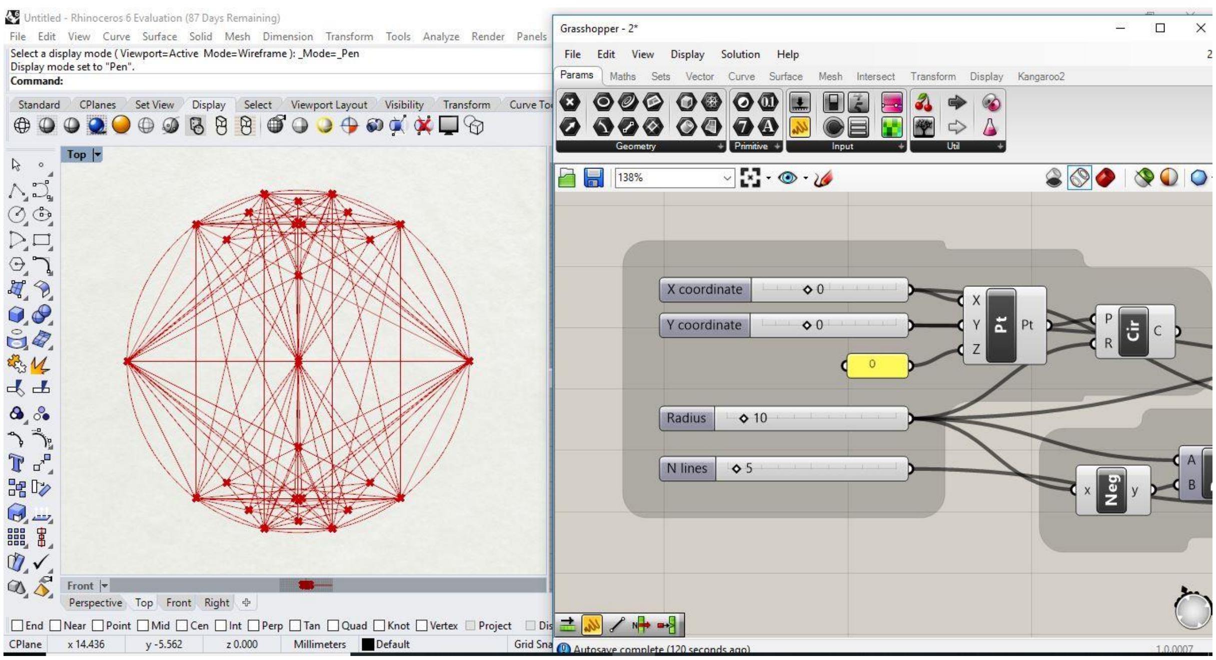


Figure 34: example of results of the second case script with increasing the number of lines to 5.

Source: author.

The third path introduces the choice of creating lines that pass through the centre and create equal angles between the intersected lines as shown in Figure 35. Nodes are defined as lines intersection in the first apply for “connect nodes” rule and as lines midpoints in the repetition.

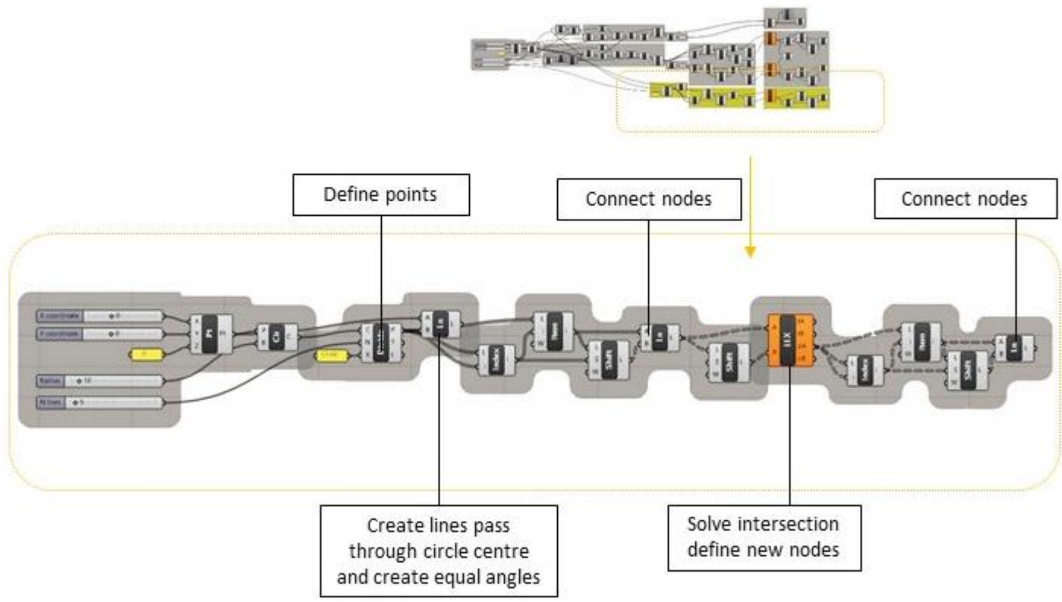


Figure 35: third case visual scripting.

Source: author.

Figure 36, Figure 37, Figure 38 and Figure 39 illustrate results of testing out the third case script.

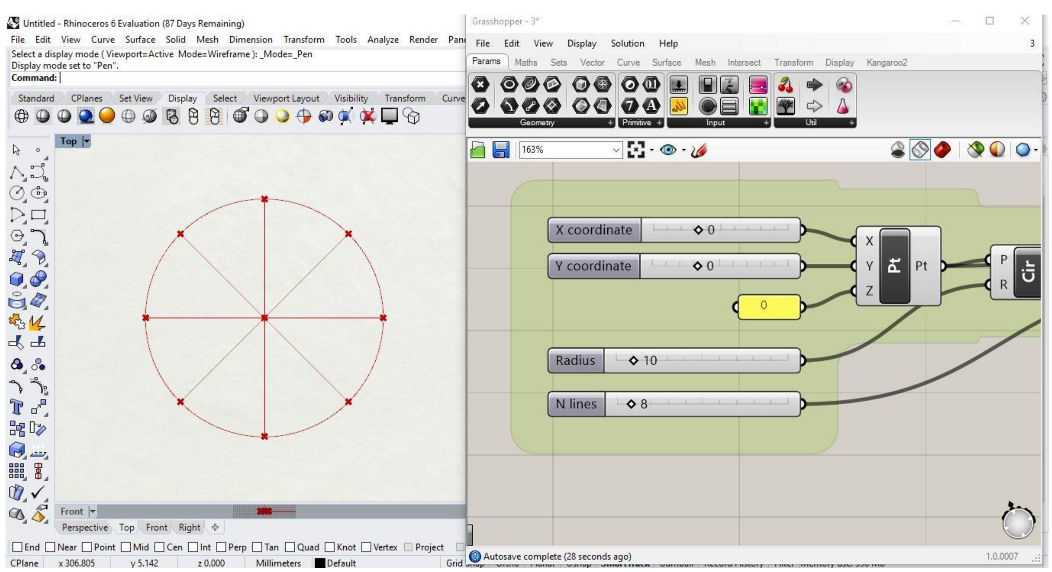


Figure 36: results of first step (create lines) by the third case in grasshopper model.

Source: author.

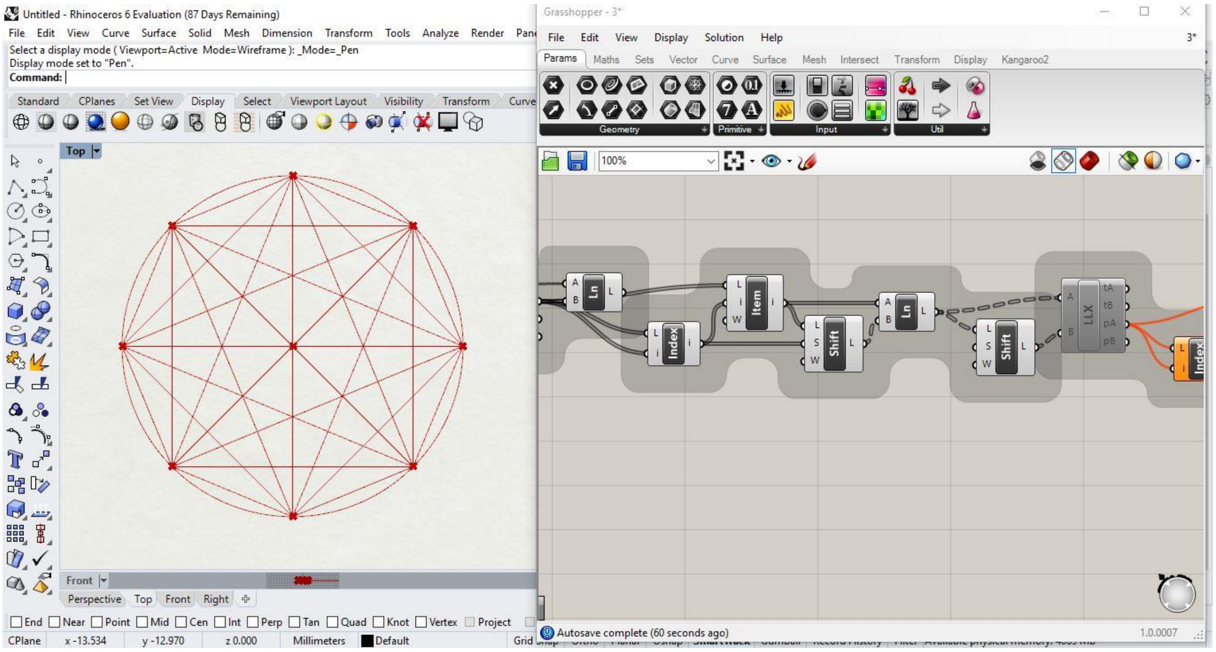


Figure 37: results of second step (connect nodes) by the third case in grasshopper model.

Source: author.

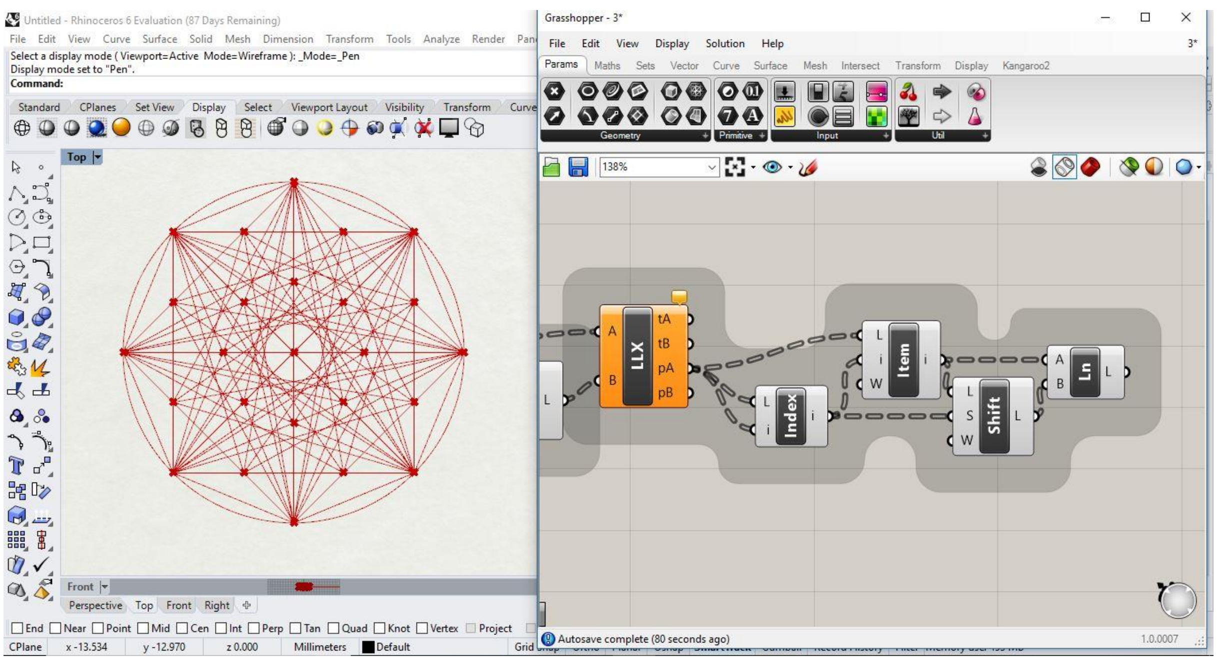


Figure 38: results of the repetition of the second step (connect nodes) in the third case with new definition of nodes.

Source: author.

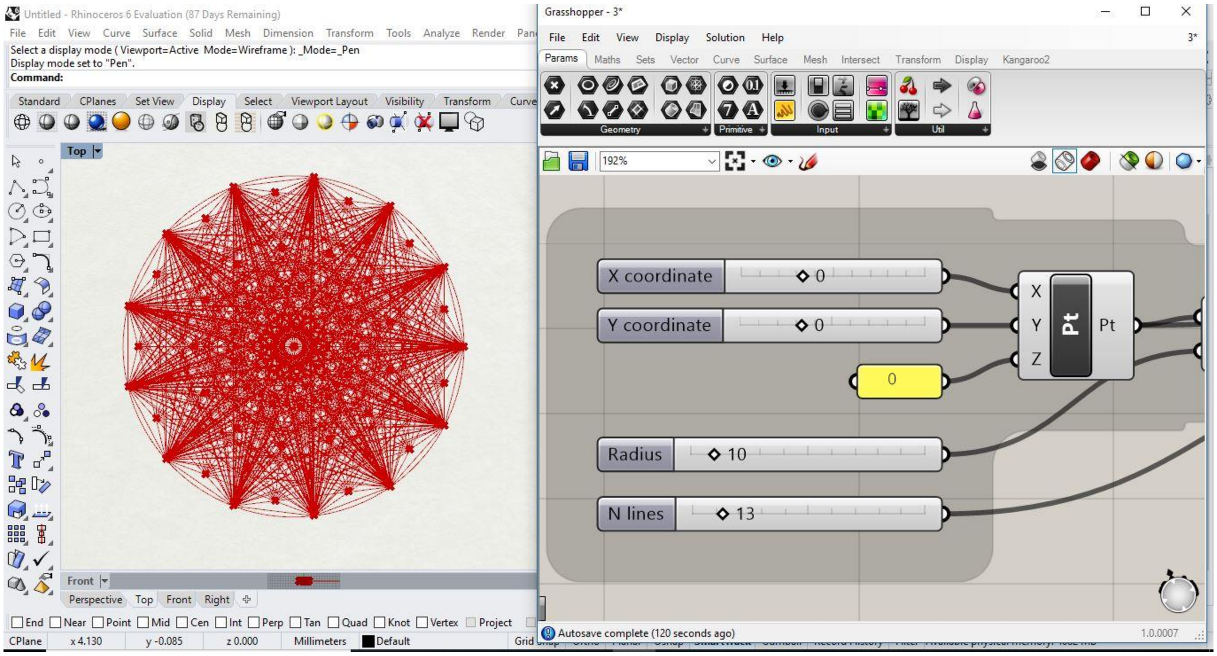


Figure 39: example of results of the third case script with increasing the number of lines to 13.

Source: author.

5.2 Forward building form

From this point forward, the examples that will be presented are been built up on the results of the first set of backward triangulation processes (Table 13) as a base pattern, in addition to the hypothetical project requirements that were mentioned earlier in this chapter. Before going further with the first process example in this stage, values that define the range of element sizes is been calculated with taking into consideration that 16 solutions are the minimum numbers of solutions requested by the designer:

Small size= 1/32 of circle area = 0.03125 circle area

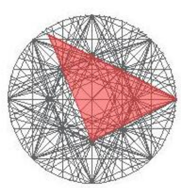
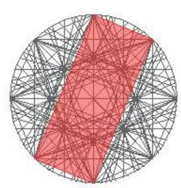
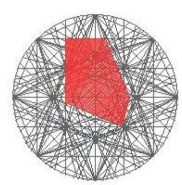
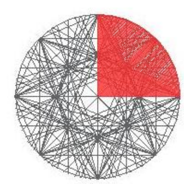
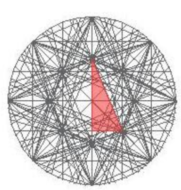
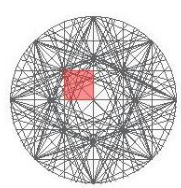
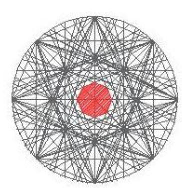
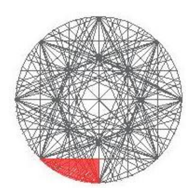
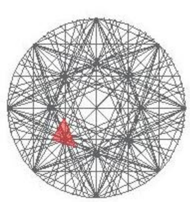
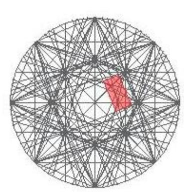
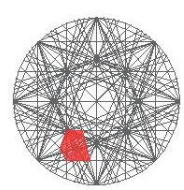
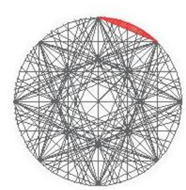
Medium size= 1\16 of circle area = 0.0625 circle area

Large size= 1/8 of circle area = 0.125 circle area

Table 16 illustrates the different choices of elements that fall under each element size category.

Table 16: examples of element selection according to type and size variables.

Source: author.

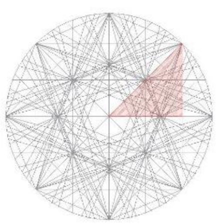
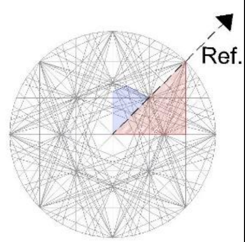
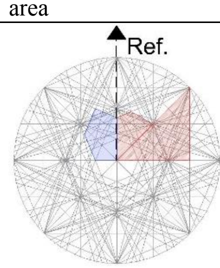
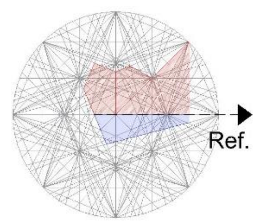
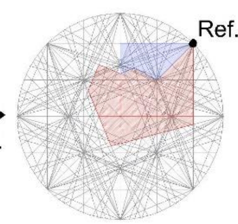
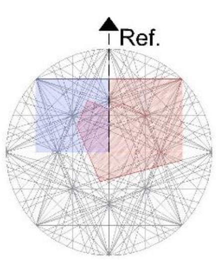
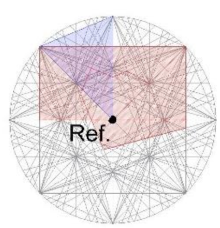
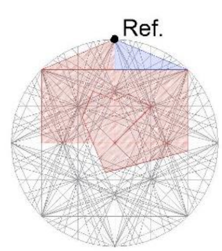
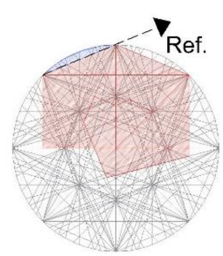
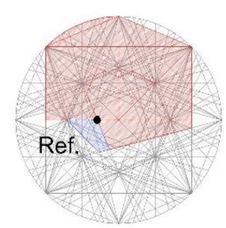
	Element type				
Element size		Triangle	Rectangular	polygon	circle segment
	large				
	medium				
	small				

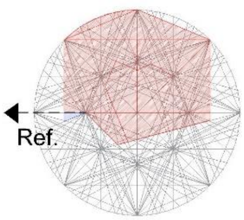
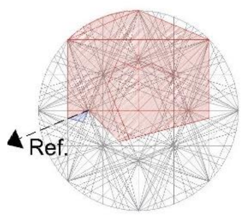
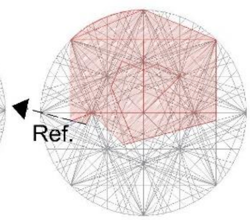
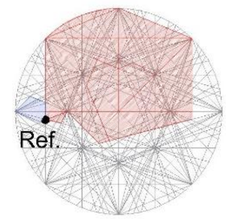
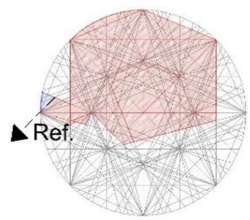
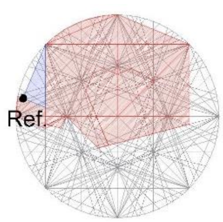
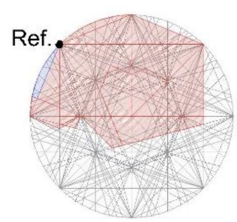
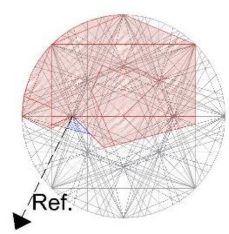
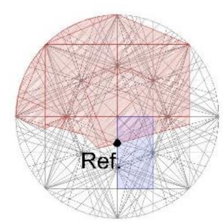
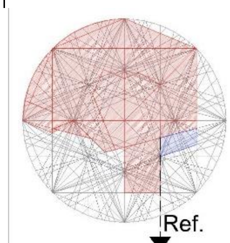
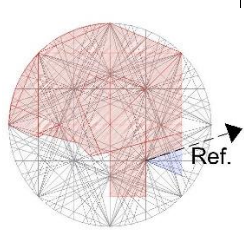
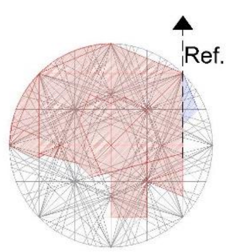
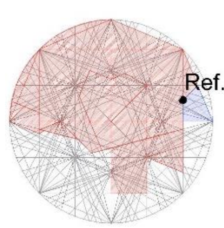
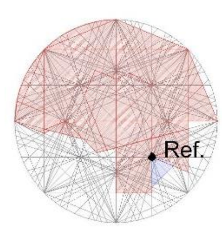
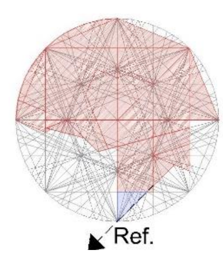
5.2.1 Layout formation process

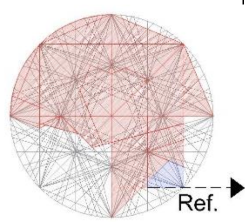
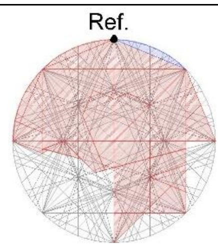
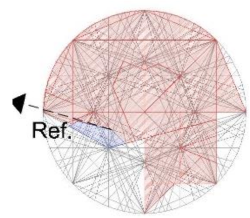
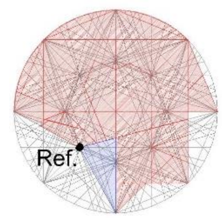
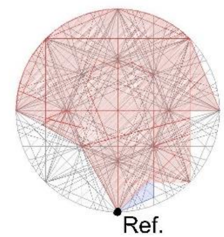
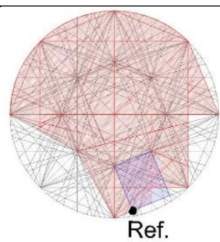
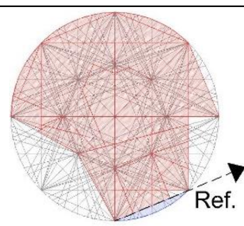
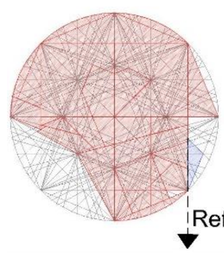
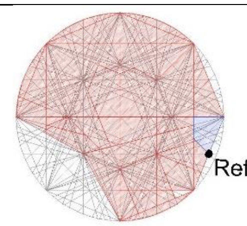
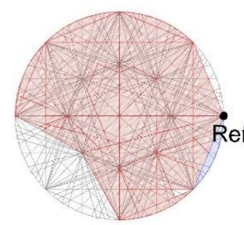
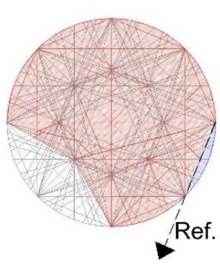
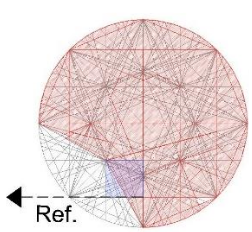
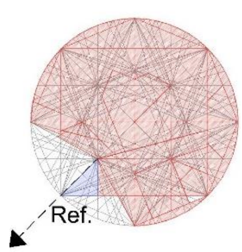
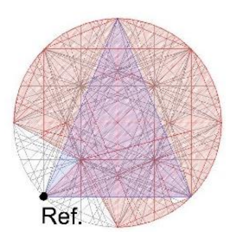
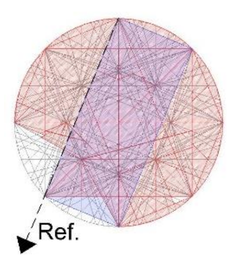
Table 17 presents the results of the first process (layout formation process). As discussed before, this process does not require any input or feedback from the designer. Each variable control the process is digitally picked up from a defined range of variables values. As shown in the table, in each repetition an element is randomly selected from the base pattern according to its type and size then it is connected to other selected one. The proposed stoppage condition for the process allowed the generation of 41 layout solutions.

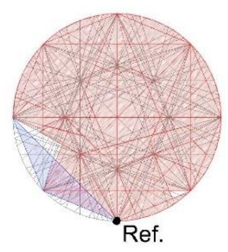
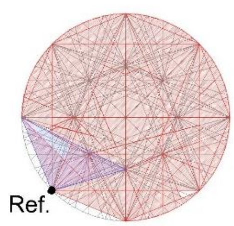
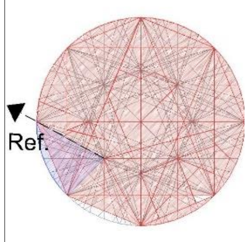
Table 17: Example of layout formation process.

Source: author.

First run		**Repeat 1**	**Repeat 2**	**Repeat 3**
1) Select element Type: triangle Size: large **2) Check condition:** Area= 0.08 of circl0e area	**3) Select reference** Type: side **4) Select element** Type: polygon Size: small **5) Check condition:** Area= 0.11 of circle area	**6) Select reference** Type: side **7) Select element** Type: polygon Size: medium **8) Check condition:** Area= 0.15 of circle area	**9) Select reference** Type: side **10) Select element** Type: polygon Size: medium **11) Check condition:** Area= 0.20 of circle area	**12) Select reference** Type: node **13) Select element** Type: polygon Size: medium 14) **Check condition**: Area: 0.25 of circle area
				
Repeat 4	**Repeat 5**	**Repeat 6**	**Repeat 7**	**Repeat 8**
15) Select reference Type: side **16) Select element** Type: rectangle Size: large 17) **Check condition**: Area= 0.37 of circle area	**18) Select reference** Type: node **19) Select element** Type: triangle Size: large **20) Check condition:** Area: 0.40 of circle area	**21) Select reference** Type: node **22) Select element** Type: triangle Size: medium **23) Check condition:** Area= 0.44 of circle area	**24) Select reference** Type: side **25) Select element** Type: circle segment Size: small **26) Check condition:** Area: 0.45 of circle area	**27) Select reference** Type: node **28) Select element** Type: polygon Size: small **29) Check condition:** Area: 0.47 of circle area
				

Repeat 9	Repeat 10	Repeat 11	Repeat 12	Repeat 13
30) Select reference Type: side **31) Select element** Type: triangle Size: small **32) Check condition:** Area: 0.47 of circle area	**33) Select reference** Type: side **34) Select element** Type: triangle Size: small **35) Check condition:** Area: 0.47 of circle area	**36) Select reference** Type: side **37) Select element** Type: triangle Size: small 38) **Check condition**: Area: 0.49 of circle area	**39) Select reference** Type: node **40) Select element** Type: polygon Size: small **41) Check condition:** Area: 0.49 of circle area	**42) Select reference** Type: side **43) Select element** Type: circle segment Size: small **44) Check condition:** Area: 0.51 of circle area
				
Repeat 14	**Repeat 15**	**Repeat 16**	**Repeat 17**	**Repeat 18**
45) Select reference Type: node **46) Select element** Type: triangle Size: medium **47) Check condition:** Area: 0.53 of circle area	**48) Select reference** Type: node **49) Select element** Type: circle segment Size: small **50) Check condition:** Area: 0.53 of circle area	**51) Select reference** Type: side **52) Select element** Type: triangle Size: small **53) Check condition:** Area: 0.59 of circle area	**54) Select reference** Type: node **55) Select element** Type: rectangle Size: large **56) Check condition:** Area: 0.60 of circle area	**57) Select reference** Type: side **58) Select element** Type: polygon Size: small **59) Check condition:** Area: 0.62 of circle area
				
Repeat 19	**Repeat 20**	**Repeat 21**	**Repeat 22**	**Repeat 23**
60) Select reference Type: side **61) Select element** Type: triangle Size: small **62) Check condition:** Area: 0.63 of circle area	**63) Select reference** Type: side **64) Select element** Type: triangle Size: small **65) Check condition:** Area: 0.65 of circle area	**66) Select reference** Type: node **67) Select element** Type: polygon Size: small **68) Check condition:** Area: 0.66 of circle area	**69) Select reference** Type: node **70) Select element** Type: triangle Size: large **71) Check condition:** Area: 0.68 of circle area	**72) Select reference** Type: side **73) Select element** Type: triangle Size: small **74) Check condition:** Area: 0.70 of circle area
				

Repeat 24	Repeat 25	Repeat 26	Repeat 27	Repeat 28
75) Select reference Type: side **76) Select element** Type: triangle Size: small **77) Check condition:** Area: 0.71 of circle area	**78) Select reference** Type: node **79) Select element** Type: circle segment Size: small **80) Check condition:** Area: 0.73 of circle area	**81) Select reference** Type: side **82) Select element** Type: polygon Size: small **83) Check condition:** Area: 0.77 of circle area	**84) Select reference** Type: node **85) Select element** Type: triangle Size: medium **86) Check condition:** Area: 0.78 of circle area	**87) Select reference** Type: node **88) Select element** Type: polygon Size: small **89) Check condition:** Area: 0.79 of circle area
				
Repeat 29	**Repeat 30**	**Repeat 31**	**Repeat 32**	**Repeat 33**
90) Select reference Type: node **91) Select element** Type: rectangle Size: medium **92) Check condition:** Area: 0.79 of circle area	**93) Select reference** Type: side **94) Select element** Type: circle segment Size: small **95) Check condition:** Area: 0.80 of circle area	**96) Select reference** Type: side **97) Select element** Type: triangle Size: small **98) Check condition:** Area: 0.81 of circle area	**99) Select reference** Type: node **100) Select element** Type: polygon Size: small **101) Check condition:** Area: 0.83 of circle area	**102) Select reference** Type: node **103) Select element** Type: circle segment Size: small **104) Check condition:** Area: 0.85 of circle area
				
Repeat 34	**Repeat 35**	**Repeat 36**	**Repeat 37**	**Repeat 38**
105) Select reference Type: side **106) Select element** Type: circle segment Size: small **107) Check condition:** Area: 0.86 of circle area	**108) Select reference** Type: side **109) Select element** Type: rectangle Size: medium **110) Check condition:** Area: 0.87 of circle area	**111) Select reference** Type: side **112) Select element** Type: triangle Size: small **113) Check condition:** Area: 0.89 of circle area	**114) Select reference** Type: node **115) Select element** Type: triangle Size: large **116) Check condition:** Area: 0.91 of circle area	**117) Select reference** Type: side **118) Select element** Type: circle segment Size: large 119) **Check condition**: Area: 0.93 of circle area
				

Repeat 30	Repeat 40	Repeat 41
120) Select reference Type: node **121) Select element** Type: triangle Size: large **122) Check condition:** Area: 0.96 of circle area	**123) Select reference** Type: node **124) Select element** Type: triangle Size: large **125) Check condition:** Area: 0.97 of circle area	**126) Select reference** Type: side **127) Select element** Type: circle segment Size: large **128) Check condition:** Area: 0.99 of circle area
		

The process resulted in various layouts geometries and the repetition of the whole process with the same variables will also generates more layouts geometries even if the base pattern kept the same.

5.2.1.1 Grasshopper/ Rhino3d application to illustrate the system

Forward layout formation process also was scripted using grasshopper/ rhino 3d interface. A sub model has been added to backward triangulation stage model as shown in Figure 40 and Figure 41.

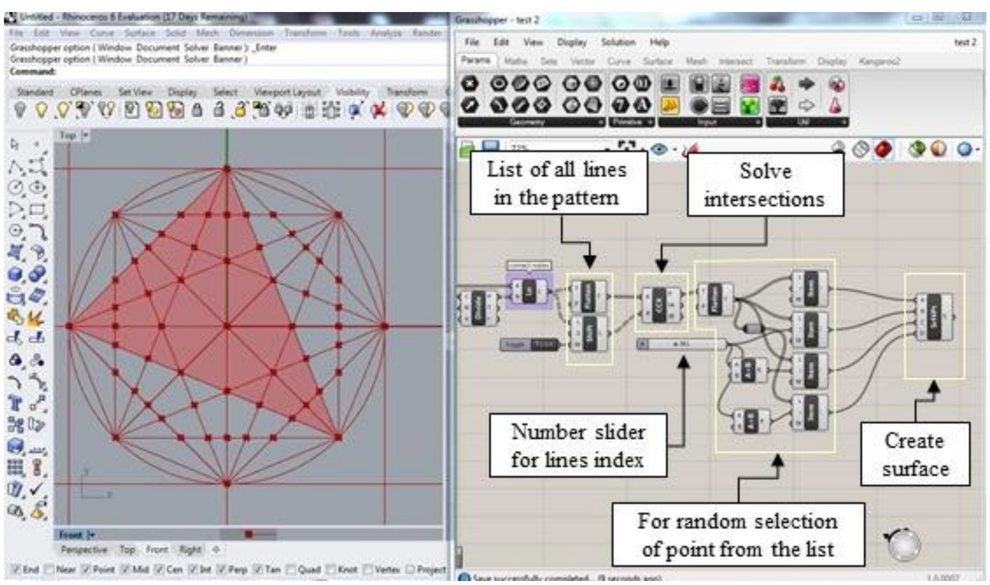


Figure 40: Grasshopper visual scripting components for forward layout formation stage- select element task. Source: author.

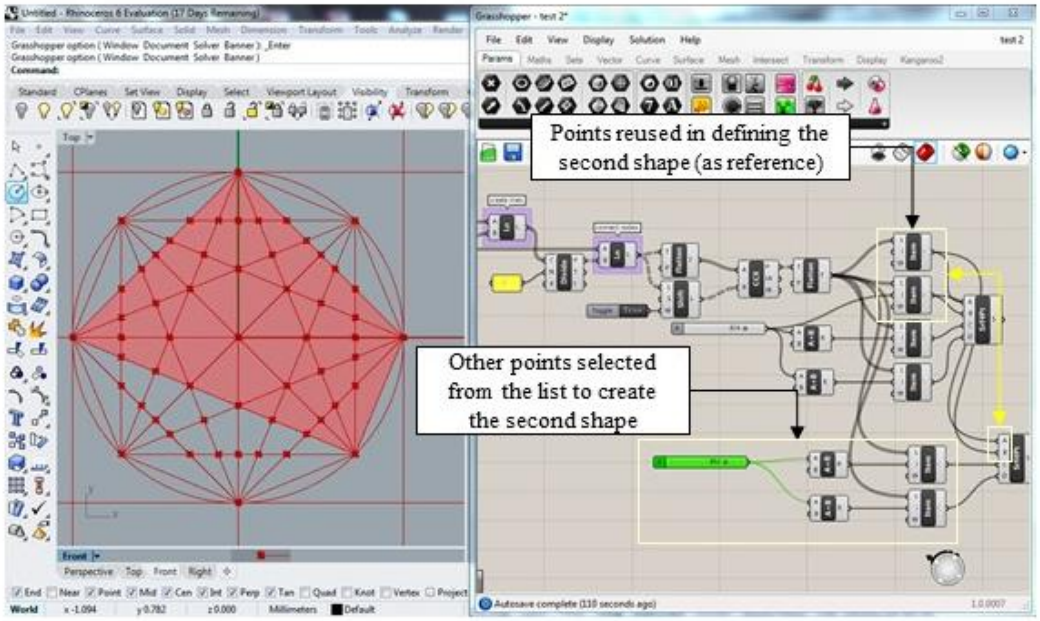


Figure 41: Grasshopper visual scripting components for forward layout formation stage- select reference and second element task.

Source: author.

For scripting the proposed tasks in the forward layout formation, first we had to deal with formulating the base pattern into list of points to be use as an input to direct elements selection task. For this purpose, the pattern were deconstructed into list of lines then the “solve intersection event” component was used to find all points of intersection and put it in a list.

Various points were randomly selected from the list to create a surface. The random selection was done be using “random select” component connected with number slider, as an input for points' index value in the “select item” component. For another element selection, one or two points of the previous selection are reused to provide a reference (node or side). Figure 42 illustrates different element selections resulted from changing the value of the random input using the slider number. Figure 43 illustrates different results of second element selections by defining side as reference while Figure 44 the reference was defined as a node (only one point was reused from the previous element selection).

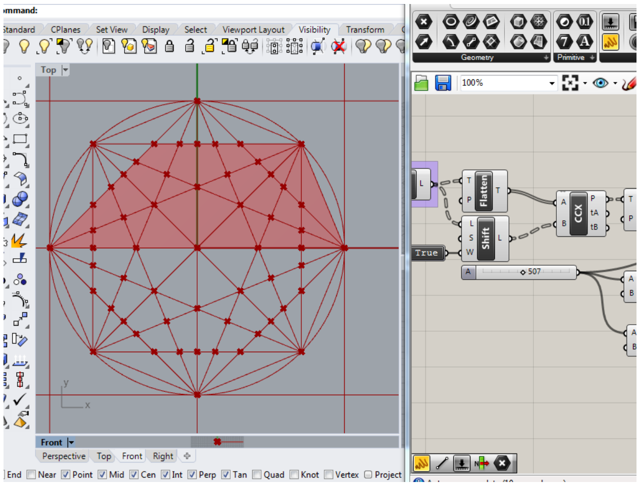
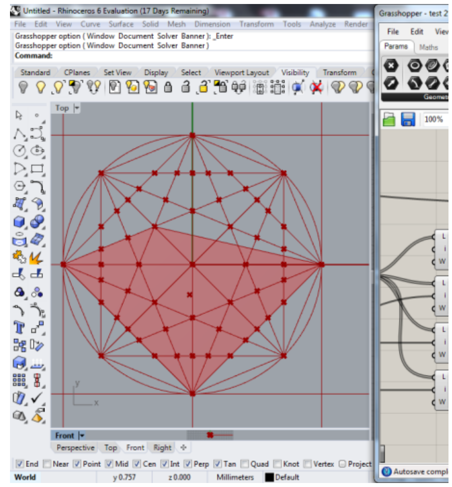

(a)

(b)

Figure 42: element selection task results using grasshopper model.

(a) And (b) different results of selecting element task by changing the values of input of "random select" component using slider number. Source: author.

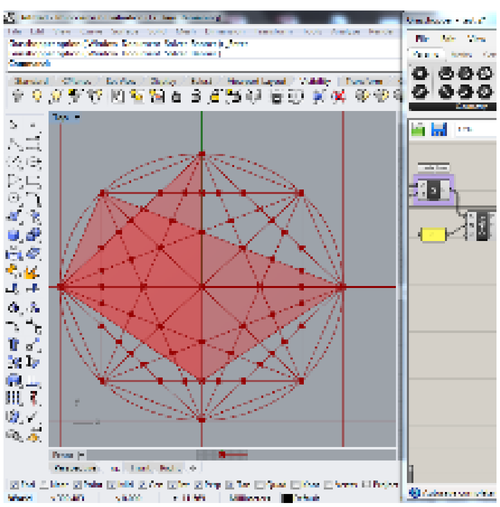
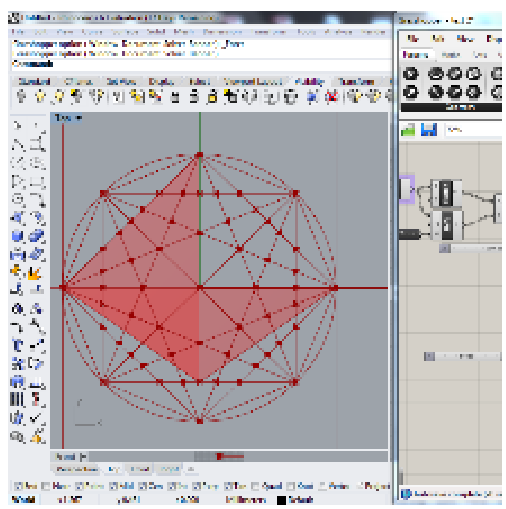

(a)

(b)

Figure 43: second element selection task with defining side as reference results using grasshopper.

(a) And (b) different results of second element selecting task by changing the values of input of "random select" component using slider number. Source: author.

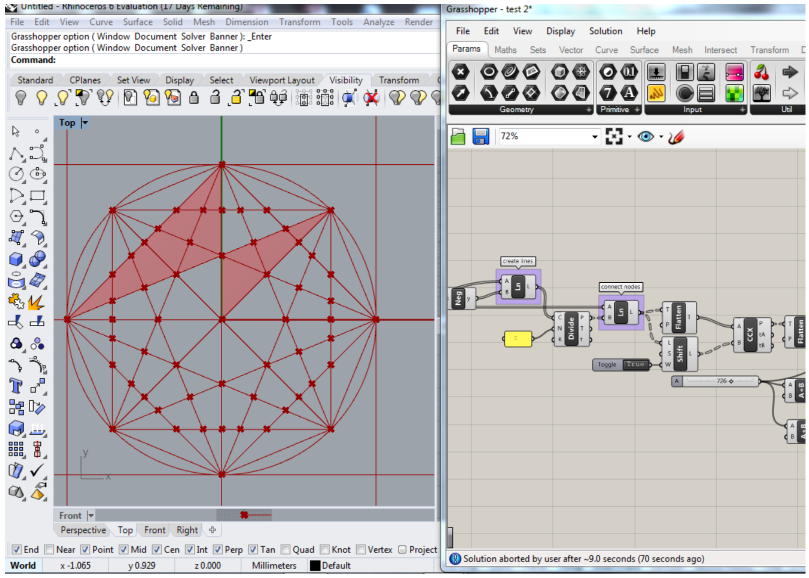


Figure 44: second element selection task with defining nodes as reference results using grasshopper.

Source: author.

The stoppage condition is been checked by incorporating math components as shown in figure. However, further work is needed to link the condition to the process so it works as stoppage condition as well as a script to specify type and size of elements as a condition control the selection which is exceed our scope of work. Generally, the main aim of scripting using grasshopper is to unfold the logic of the proposed tasks from computer point of view as well as to validate the process.

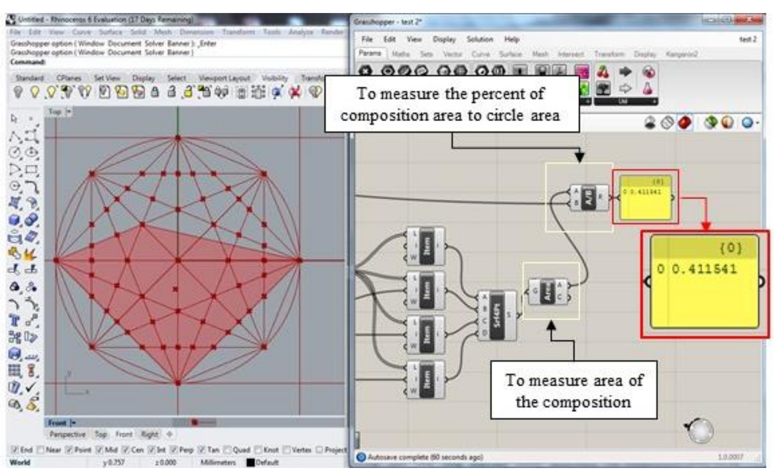


Figure 45: Checking the stoppage condition of layout formation in grasshopper

Source: author.

5.2.2 3D Transformation process

To continue with the second process of this stage (3d transformation process), plot geometry, setbacks and other project requirements is used to define the spatial limitation for the building by generating the generic template presented in Figure 46.

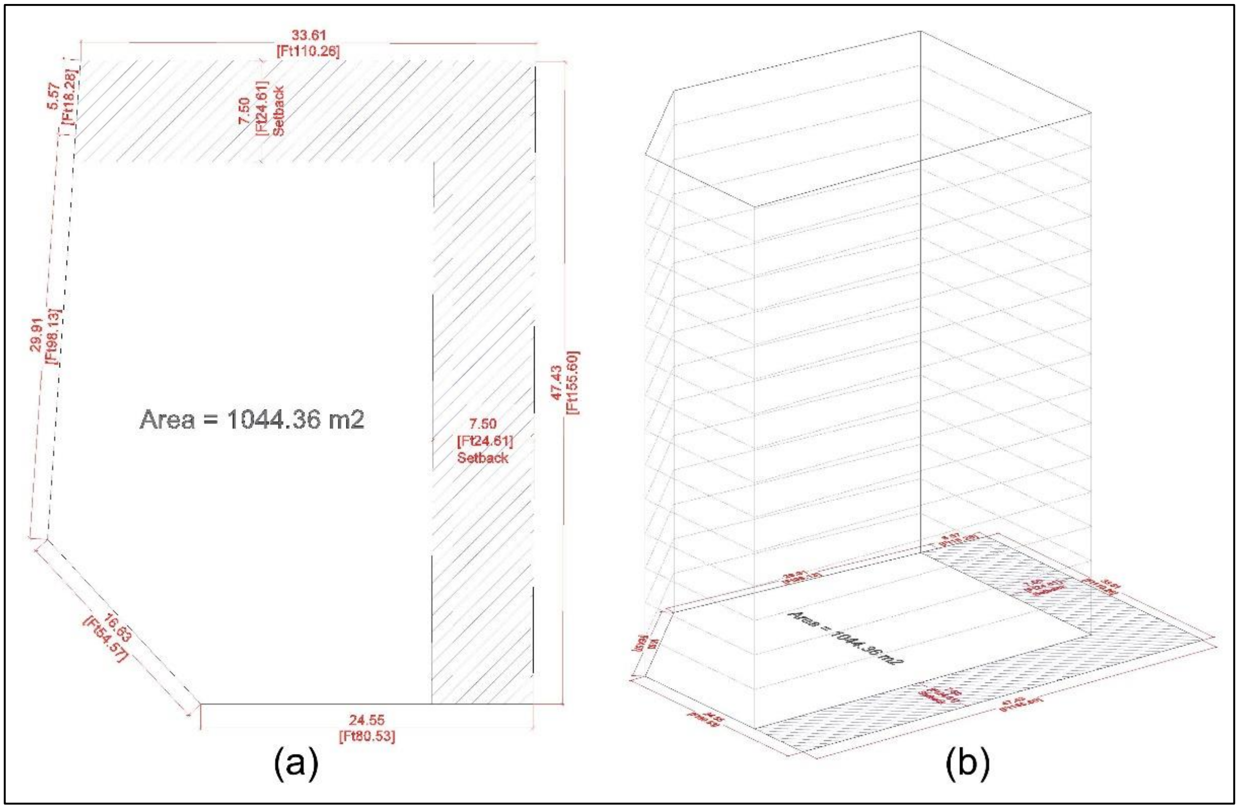


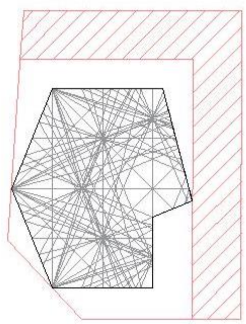
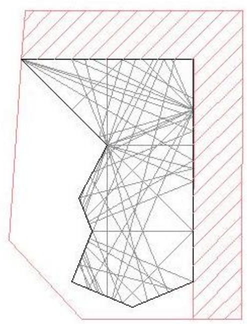
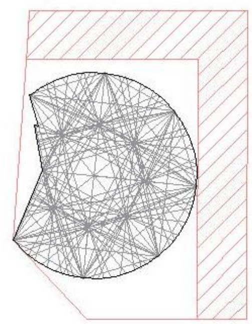
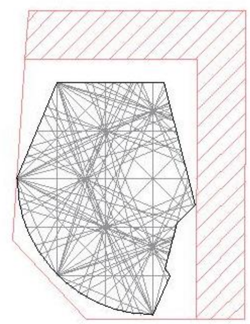
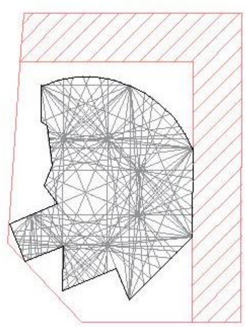
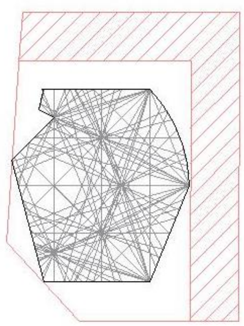
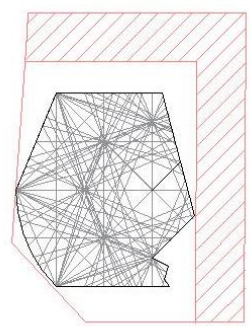
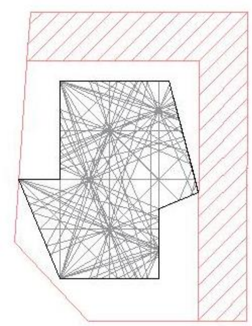
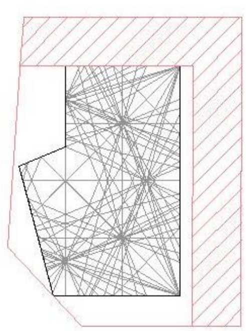
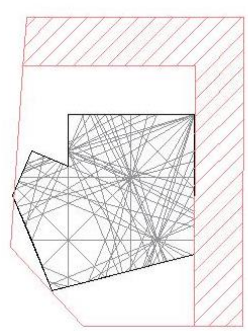
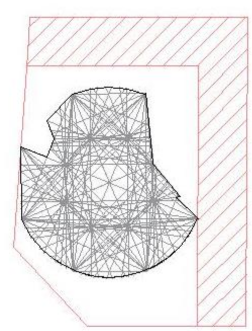
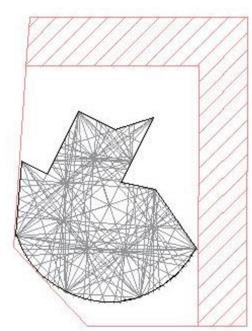
Figure 46: generic templet for the project.

Source: author.

A group of layouts (12 layout) from the 41 layout resulted in the previous process were randomly selected to be used as input data to continue the generation process of the hypothetical project. Each layout is scaled to fit the boundary of the plot area after counting off the setbacks as shown in Table 18.

Table 18: Examples of layouts fitted to the plot boundary.

Source: author.

Layout no. 1	Layout no. 2	Layout no. 3	Layout no. 4
			
Layout no. 5	Layout no. 6	Layout no. 7	Layout no. 8
			
Layout no. 9	Layout no. 10	Layout no. 11	Layout no. 12
			

To create the third dimension for the building, a transformation type is selected. The following figures illustrate possibilities of solutions generated from selecting each transformation type, where Figure 47 illustrate forms generated using extrusion transformation, and combined with scale transformation in Figure 48 and Figure 49, while combined with twist transformation in Figure 50 and finally multiple extrusions were done to create forms in Figure 51. Similarly, Figure 52 illustrate forms generated using blend transformation, combined with scale transformation in Figure 53, Figure 54 and multiple blends were done to create forms in Figure 55.

Figure 47: Sample of results generated using extrusion transformation
Source: author.

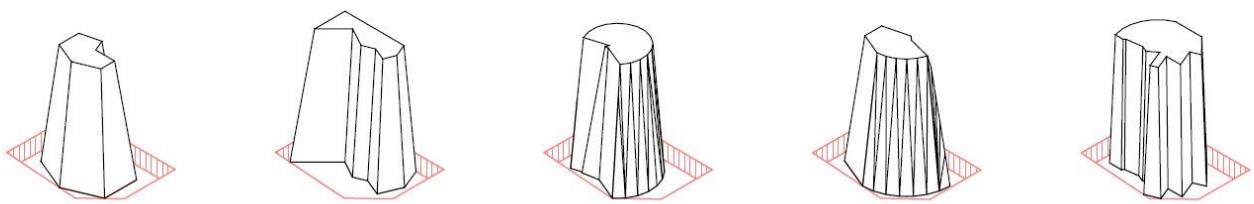


Figure 48: Sample of results generated using extrusion with scaling transformation
Source: author.

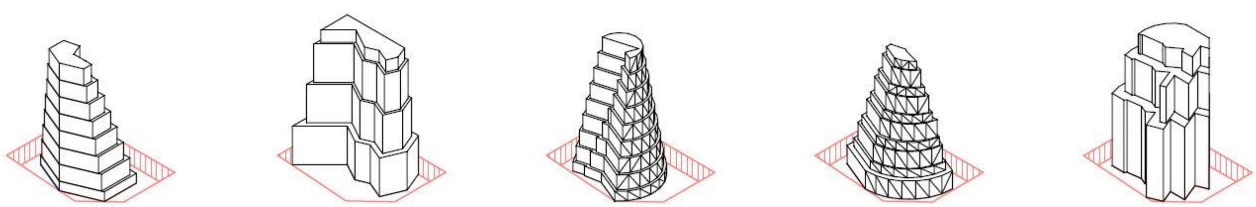


Figure 49: Sample of results generated using extrusion with scaling transformation
Source: author.

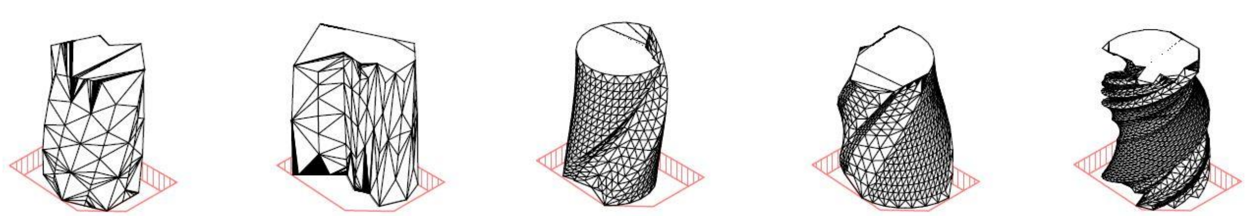


Figure 50: Sample of results generated using extrusion with twist transformation
Source: author.

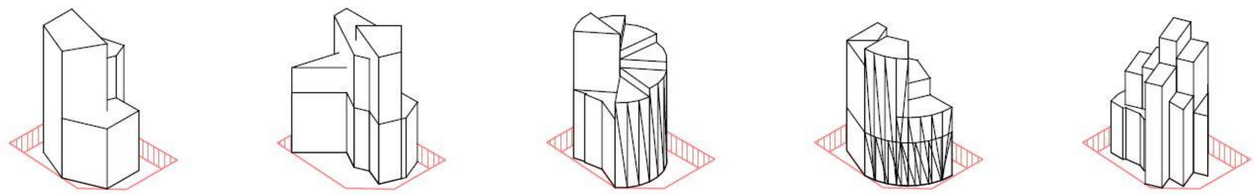


Figure 51: Sample of results generated using multiple extrusion transformation.
Source: author.

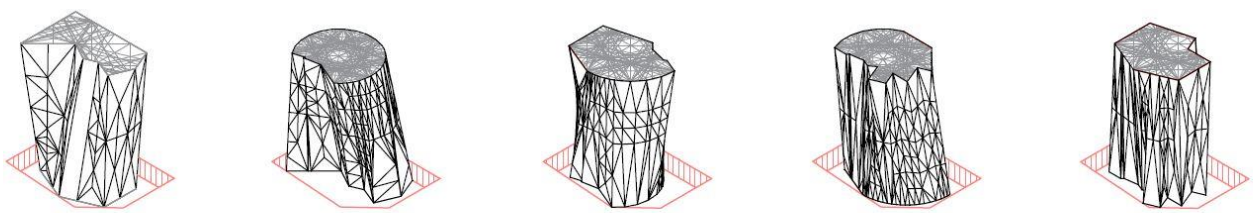


Figure 52: Sample of results generated using blend transformation.
Source: author.

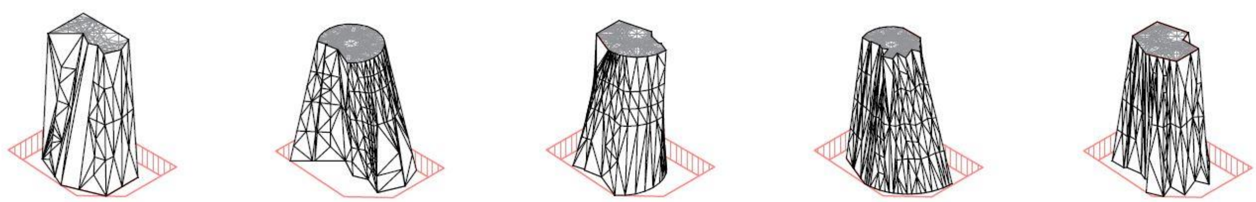


Figure 53: Sample of results generated using blend with scaling transformation.
Source: author.

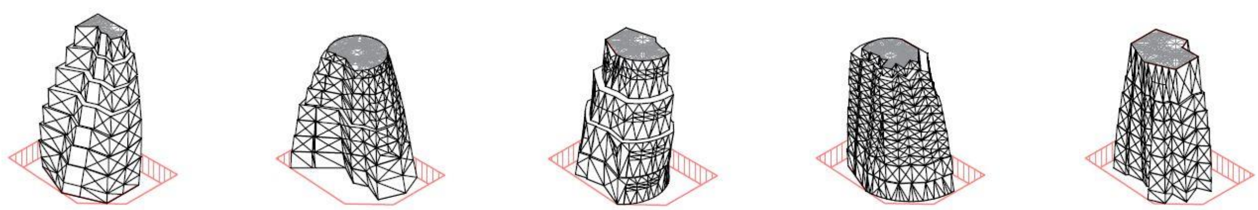


Figure 54: Sample of results generated using blend with scaling transformation.
Source: author.

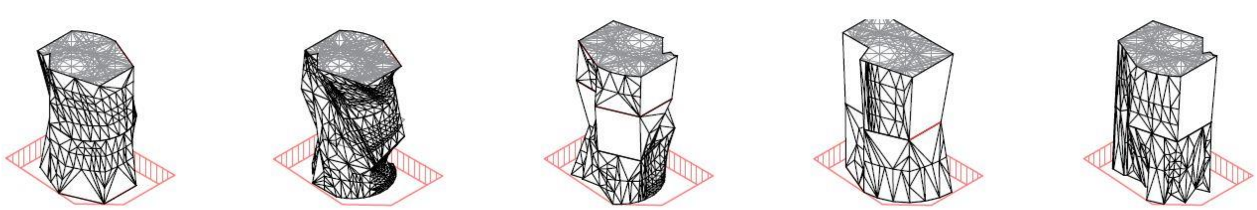


Figure 55: Sample of results generated using multiple blends transformation.
Source: author.

For each transformation type, 5 examples were developed with taken in consideration the specified requirements. Comparing all the 45 generated solutions to each other, none of them is duplicated and no high degree of similarity can be capture among the solutions even in the same group. This support the argument that the methodology is capable of creating alternative that have different architectural styles.

- **Editing the roof shape by changing edges**

The final step in the transformation process is editing the roof shape by changing edges levels within the range of the specified roof height in the project requirements. Figure 56 presents examples of editing roof edges levels.

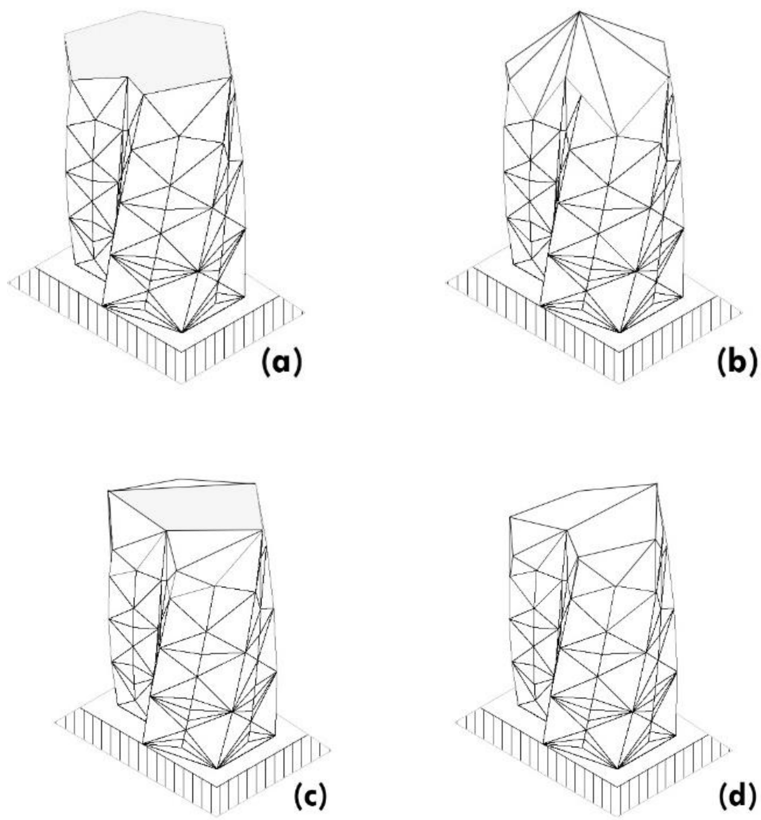


Figure 56: example of editing top according to the "top height" specified in designer input.

(a) form generated with flat roof. (b, c, d) results of editing the levels of roof edges. Source: author.

5.2.3 Structural identification process

The final process in this stage is structural identification process. An example is presented in Figure 57 to suggest structural layouts for solution number 3. Shared nodes between the floors was identified then the distances between them were measured to find a grid match the specified span in the input (6- 8) m.

The use of computer as an operator of the process allows structure identification for more complex forms that include twist or blend types of transformation. This can be done in grasshopper by using "solve intersection event" component where each layout used in the formation process is taken as separate input then the resulted list of intersection points can be sorted based on the distance that represents the span between the identified points using "sort list" component.

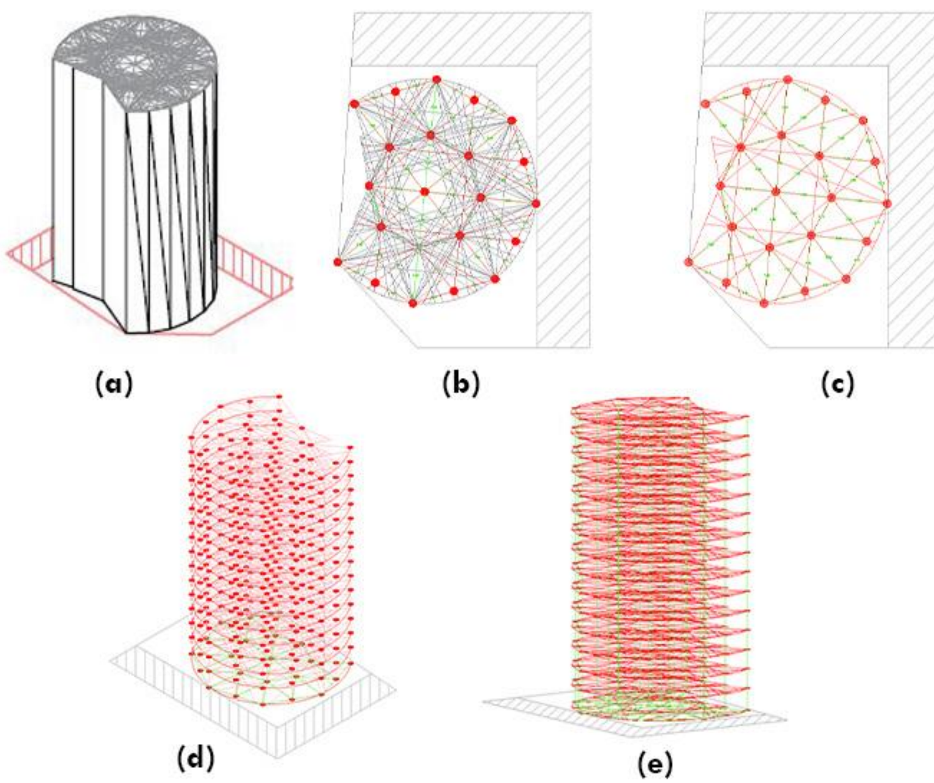


Figure 57: example of structural identification for the generated forms.

(a) Example of building form generated in the stage. (b) Measuring distance between nodes and identify best nodes create grid with the specified structural spam. (c) Proposed structural layout for the building. (d) And (e) 3d views for the proposed structure layout consist of columns and beams. Source: author.

5.3 Evaluation stage

All solutions generated from the previous stage used as an input data for developing the evaluation stage examples. The information needed for each evaluation criteria is presented in Table 19 to Table 27. These information is used to rank the solutions as shown in Table 28.

During the evaluation of the solutions, similarities between the generated forms in term of their areas and complexity index were observed. This is due to the fact that the 45 solutions were generated from using only 12 layouts. The reduplication in the use of the same layout for different transformation types produced spatial similarities among solutions that have various geometrical configuration. For example, solutions number 2, 7, 12, 17, 22, 27, 32 and 37 have the same footprint as a result of sharing the same ground floor layout.

Moreover, some of the solutions almost have the same total built up area although the layouts and the transformation types used in the process of generating each of them are different, for example, solution number 35 and solution number 36 have total built up area of 7207.807 m² and 7139.199 m² respectively. This is can be related to the process of scaling each layout to fit plot boundary before doing the transformation which reduce the difference between layouts areas.

Another observation during the evaluation is that some of transformation types such as blend and twist have resulted in exceeding the spatial limitation of the generic template especially in the middle floors. This affected the coverage area, however, it doesn't consider to be problem unless the maximum coverage area is specified in plot regulation then either the solutions is excluded or it can be solved be rescale the solution again to fit the boundary of the generic template.

Table 19: evaluation of solutions generated from extrusion transformation type. Source: author.

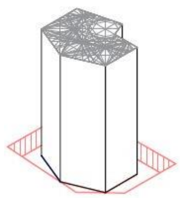
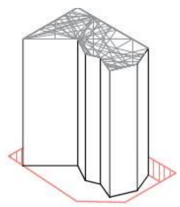
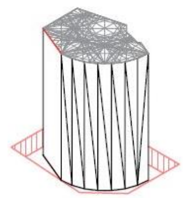
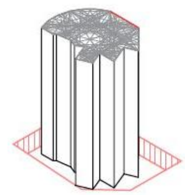
Solution number	1	2	3	4	5
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1st floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
2nd floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
3 floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
4th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
5th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
6th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
7th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
8th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
9th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
10th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
11th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
12th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
13th floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
Roof area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
Total built up area	9658.848	9626.462	10354.77	11032.64	9818.222
Coverage area	643.9232	641.7641	690.3177	735.5092	654.5481
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.9232	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	1	1	1	1	1
Number of layout edges	3.5	4	2.5	4	7
Number of operations	1	1	1	1	1
Type of operation	1	1	1	1	1
Total number of point	6.5	7	5.5	7	10
complexity index	0.36	0.39	0.31	0.39	0.56

Table 20: Evolution of solutions generated from extrusion with scale transformation type. Source: author.

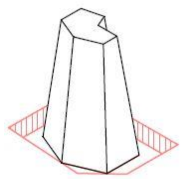
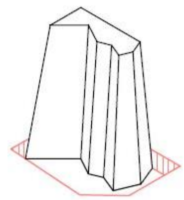
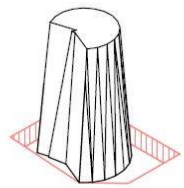
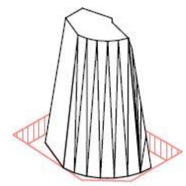
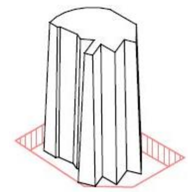
Solution number	6	7	8	9	10
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1st floor area (m)	613.2932	616.8241	665.9377	698.3692	636.7481
2nd floor area (m)	582.6632	591.8841	641.5577	661.2292	618.9481
3 floor area (m)	552.0332	566.9441	617.1777	624.0892	601.1481
4th floor area (m)	521.4032	542.0041	592.7977	586.9492	583.3481
5th floor area (m)	490.7732	517.0641	568.4177	549.8092	565.5481
6th floor area (m)	460.1432	492.1241	544.0377	512.6692	547.7481
7th floor area (m)	429.5132	467.1841	519.6577	475.5292	529.9481
8th floor area (m)	398.8832	442.2441	495.2777	438.3892	512.1481
9th floor area (m)	368.2532	417.3041	470.8977	401.2492	494.3481
10th floor area (m)	337.6232	392.3641	446.5177	364.1092	476.5481
11th floor area (m)	306.9932	367.4241	422.1377	326.9692	458.7481
12th floor area (m)	276.3632	342.4841	397.7577	289.8292	440.9481
13th floor area (m)	245.7332	317.5441	373.3777	252.6892	423.1481
Roof area (m)	215.0657	292.499	348.992	215.4322	405.2539
Total built up area	6442.661	7007.656	7794.86	7132.821	7949.127
Coverage area	643.9232	641.7641	690.3177	735.5092	654.5481
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.9232	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	1	1	1	1	1
Number of layout edges	3.5	4	2.5	4	7
Number of operations	2	2	2	2	2
Type of operation	2	2	2	2	2
Total number of point	8.5	9	7.5	9	12
complexity index	0.47	0.50	0.42	0.50	0.67

Table 21: Evolution of solutions generated from extrusion with scale transformation type.
Source: author.

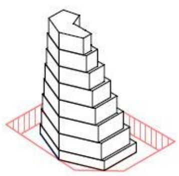
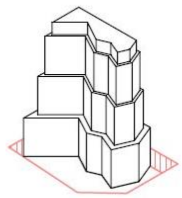
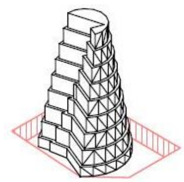
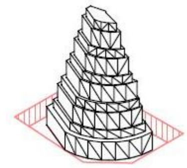
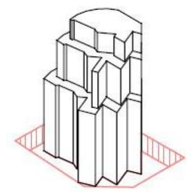
Solution number	11	12	13	14	15
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1st floor area (m)	613.2932	616.8241	665.9377	698.3692	636.7481
2nd floor area (m)	582.6632	591.8841	641.5577	661.2292	618.9481
3 floor area (m)	552.0332	566.9441	617.1777	624.0892	601.1481
4th floor area (m)	521.4032	542.0041	592.7977	586.9492	583.3481
5th floor area (m)	490.7732	517.0641	568.4177	549.8092	565.5481
6th floor area (m)	460.1432	492.1241	544.0377	512.6692	547.7481
7th floor area (m)	429.5132	467.1841	519.6577	475.5292	529.9481
8th floor area (m)	398.8832	442.2441	495.2777	438.3892	512.1481
9th floor area (m)	368.2532	417.3041	470.8977	401.2492	494.3481
10th floor area (m)	337.6232	392.3641	446.5177	364.1092	476.5481
11th floor area (m)	306.9932	367.4241	422.1377	326.9692	458.7481
12th floor area (m)	276.3632	342.4841	397.7577	289.8292	440.9481
13th floor area (m)	245.7332	317.5441	373.3777	252.6892	423.1481
Roof area (m)	215.0657	292.499	348.992	215.4322	405.2539
Total built up area	6442.661	7007.656	7794.86	7132.821	7949.127
Coverage area	643.9232	641.7641	690.3177	735.5092	654.5481
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.9232	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	1	1	1	1	1
Number of layout edges	3.5	4	2.5	4	7
Number of operations	2	2	2	2	2
Type of operation	2	2	2	2	2
Total number of point	8.5	9	7.5	9	12
complexity index	0.47	0.50	0.42	0.50	0.67

Table 22: Evolution of solutions generated from extrusion with twist transformation type. Source: author.

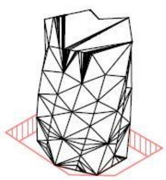
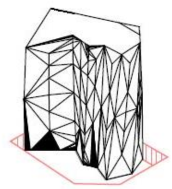
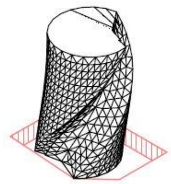
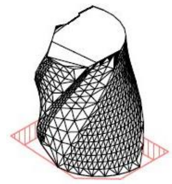
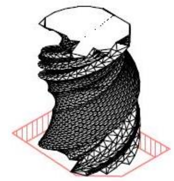
Solution number	16	17	18	19	20
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1^{st} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
2^{nd} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
3 floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
4^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
5^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
6^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
7^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
8^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
9^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
10^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
11^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
12^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
13^{th} floor area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
Roof area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
Total built up area	9658.848	9626.462	10354.77	11032.64	9818.222
Coverage area	833.651	845.4696	936.8316	1134.5143	1072.566
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.9232	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	1	1	1	1	1
Number of layout edges	3.5	4	2.5	4	7
Number of operations	2	2	4	3	5
Type of operation	5	5	5	5	5
Total number of point	11.5	12	12.5	13	18
complexity index	0.64	0.67	0.69	0.72	1.00

Table 23: Evolution of solutions generated from multi extrusion transformation type.

Source: author.

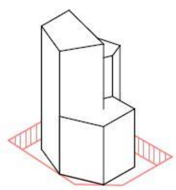
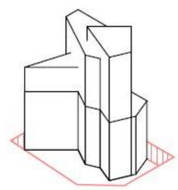
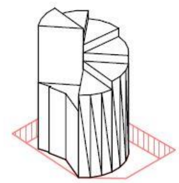
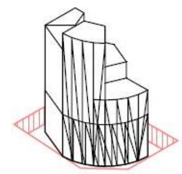
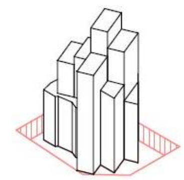
Solution number	21	22	23	24	25
					
Total built up area evaluation					
Ground area (m)	643.9232	641.74	690.3177	735.5	654.5481
1st floor area (m)	643.9232	641.74	690.3177	735.5	654.5481
2nd floor area (m)	643.9232	641.74	690.3177	735.5	654.5481
3 floor area (m)	643.9232	641.74	690.3177	735.5	654.5481
4th floor area (m)	643.9232	641.74	690.3177	735.5	654.5481
5th floor area (m)	643.9232	641.74	690.3177	735.5	654.5481
6th floor area (m)	643.9232	641.74	690.3177	660.2185	621.4496
7th floor area (m)	442.84	533.96	690.3177	660.2185	588.3492
8th floor area (m)	442.84	533.96	690.3177	660.2185	540.1192
9th floor area (m)	442.84	533.96	601.36	525.6	460.2092
10th floor area (m)	442.84	457.59	501.1365	525.6	380.2992
11th floor area (m)	442.84	457.59	400.9092	323.4018	300.38
12th floor area (m)	327.65	457.59	300.6819	323.4018	220.47
13th floor area (m)	327.65	205.17	200.4546	134.62	140.56
Roof area (m)	327.65	205.17	100.2273	134.62	60.65
Total built up area	7704.612	7877.17	8317.629	8360.899	7239.775
Coverage area	643.9232	641.74	690.3177	735.5	654.5481
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.9232	641.74	690.3177	735.5	654.5481
Complexity evaluation					
Number of layouts	1	1	1	1	1
Number of layout edges	3.5	4	2.5	4	7
Number of operations	1.5	1.75	2.5	1.75	3
Type of operation	1	1	1	1	1
Total number of point	7	7.75	7	7.75	12
complexity index	0.39	0.43	0.39	0.43	0.67

Table 24: Evolution of solutions generated from blend transformation type. Source: author.

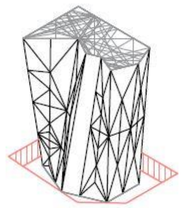
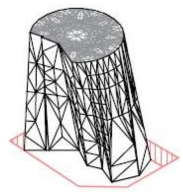
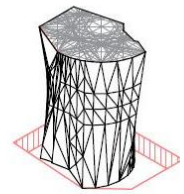
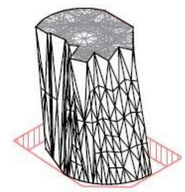
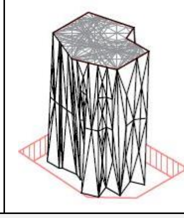
Solution number	26	27	28	29	30
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1st floor area (m)	643.7732	645.2241	693.5377	729.8092	653.7981
2nd floor area (m)	643.6232	648.6841	696.7577	724.1092	653.0481
3 floor area (m)	643.4732	652.1441	699.9777	718.4092	652.2981
4th floor area (m)	643.3232	655.6041	703.1977	712.7092	651.5481
5th floor area (m)	643.1732	659.0641	706.4177	707.0092	650.7981
6th floor area (m)	643.0232	662.5241	709.6377	701.3092	650.0481
7th floor area (m)	642.8732	665.9841	712.8577	695.6092	649.2981
8th floor area (m)	642.7232	669.4441	716.0777	689.9092	648.5481
9th floor area (m)	642.8732	672.9041	719.2977	684.2092	647.7981
10th floor area (m)	642.4232	676.3641	722.5177	678.5092	647.0481
11th floor area (m)	642.2732	679.8241	725.7377	672.8092	646.2981
12th floor area (m)	642.1232	683.2841	728.9577	667.1092	645.5481
13th floor area (m)	641.9732	686.7441	732.1777	661.4092	644.7981
Roof area (m)	641.7641	690.3177	735.5092	654.5481	643.9232
Total built up area	9643.039	9989.875	10692.98	10432.98	9739.347
Coverage area	885.295	894.8837	805.9185	800.9921	829.5864
Ground/ plot %	0.616572063	0.614504673	0.660995921	0.704267877	0.626745662
Foot print	643.7732	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	2	2	2	2	2
Number of layout edges	4	4	4	7	7
Number of operations	1	1	1	1	1
Type of operation	3	3	3	3	3
Total number of point	10	10	10	13	13
complexity index	0.56	0.56	0.56	0.72	0.72

Table 25: Evolution of solutions generated from blend with scale transformation type. Source: author.

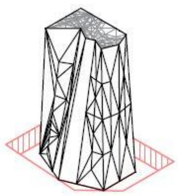
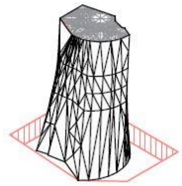
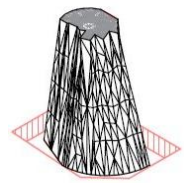
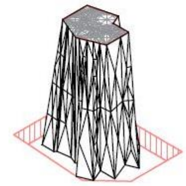
Solution number	31	32	33	34	35
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	663.8427	654.5481
1st floor area (m)	616.6732	621.4641	663.8427	637.3677	629.6881
2nd floor area (m)	589.4232	601.1641	637.3677	610.8927	604.8281
3 floor area (m)	562.1732	580.8641	610.8927	584.4177	579.9681
4th floor area (m)	534.9232	560.5641	584.4177	557.9427	555.1081
5th floor area (m)	507.6732	540.2641	557.9427	531.4677	530.2481
6th floor area (m)	480.4232	519.9641	531.4677	504.9927	505.3881
7th floor area (m)	453.1732	499.6641	504.9927	478.5177	480.5281
8th floor area (m)	425.9232	479.3641	478.5177	448.3252	455.6681
9th floor area (m)	398.6732	459.0641	452.0427	412.4272	430.8081
10th floor area (m)	371.4232	438.7641	425.5677	376.5292	405.9481
11th floor area (m)	344.1732	418.4641	399.0927	340.6312	381.0881
12th floor area (m)	316.9232	398.1641	372.6177	304.7332	356.2281
13th floor area (m)	289.6732	377.8641	346.1427	268.8352	331.3681
Roof area (m)	262.38	357.5625	319.6562	232.9321	306.3936
Total built up area	6797.555	7494.96	7574.879	7263.343	7207.807
Coverage area	659.9901	821.9031	686.0642	734.3374	661.1533
Ground/ plot %	0.616572	0.614505	0.660996	0.704268	0.626746
Foot print	643.9232	641.7641	690.3177	663.8427	654.5481
Complexity evaluation					
Number of layouts	2	2	2	2	2
Number of layout edges	4	4	4	7	7
Number of operations	2	2	2	2	2
Type of operation	6	6	6	6	6
Total number of point	14	14	14	17	17
complexity index	0.78	0.78	0.78	0.94	0.94

Table 26: Evolution of solutions generated from blend with scale transformation type. Source: author.

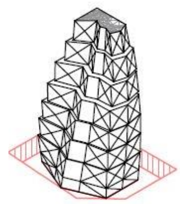
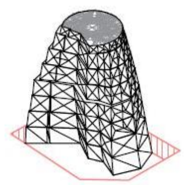
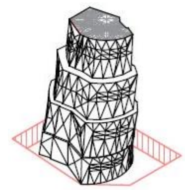
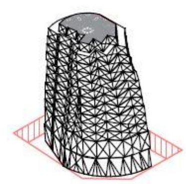
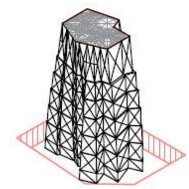
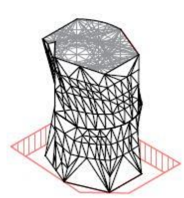
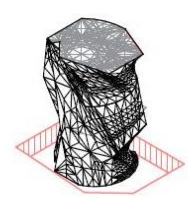
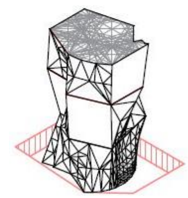
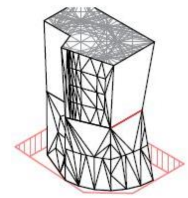
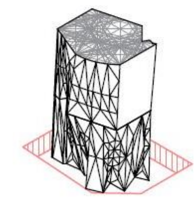
Solution number	36	37	38	39	40
					
Total built up area evaluation					
Ground area (m)	643.9232	641.7641	690.3177	735.5092	654.5481
1st floor area (m)	645.0781	645.83	688.1027	725.7592	650.38
2nd floor area (m)	645.42	636.45	685.8877	716.0092	646.233
3 floor area (m)	647.488	637.7975	687.5952	711	630.0834
4th floor area (m)	622.1957	614	689.0952	685.25	622.322
5th floor area (m)	593.01	612.73	690.5952	659.5	596.35
6th floor area (m)	565.2296	610.41	692.0952	643	580.35
7th floor area (m)	549.86	615.0947	567.7491	608	565.0622
8th floor area (m)	448.2324	530.7	555.5891	580.8206	554.8722
9th floor area (m)	442.5467	518.836	543.4291	552.9322	490.8456
10th floor area (m)	374.9478	467.2088	531.2691	518.37	480.1453
11th floor area (m)	314.9	450.52	349.5322	485	422.9915
12th floor area (m)	260.202	391.4	328.2589	450.483	374.472
13th floor area (m)	234.0065	373.9226	306.9856	415.12	315.5224
Roof area (m)	149.159	294.1875	264.439	218	315.5224
Total built up area	7136.199	8040.851	8270.941	8704.753	7899.7
Coverage area	763.8843	821.8658	752.618	756.0994	681.4298
Ground/ plot %	0.616572	0.614505	0.660996	0.704268	0.626746
Foot print	643.9232	641.7641	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	2	2	2	2	2
Number of layout edges	4	4	4	7	7
Number of operations	2	2	2	2	2
Type of operation	6	6	6	6	6
Total number of point	14	14	14	17	17
	0.78	0.78	0.78	0.94	0.94

Table 27: Evolution of solutions generated from multi blend transformation type. Source: author.

Solution number	41	42	43	44	45
					
Total built up area evaluation					
Ground area (m)	643.9232	558.5988	690.3177	735.5092	654.5481
1st floor area (m)	628.989	563.7648	665.1533	723.2236	653.8291
2nd floor area (m)	614.05	568.9308	639.9889	710.938	653.1101
3 floor area (m)	599.11	574.0968	614.8245	698.6524	652.3911
4th floor area (m)	569.25	581.85	589.6634	686.3668	651.6721
5th floor area (m)	554.315	589.5988	603.9994	674.0812	650.9531
6th floor area (m)	539.38	609.7428	618.3354	661.7956	650.2341
7th floor area (m)	524.4522	629.8868	632.6714	649.5099	649.5099
8th floor area (m)	540.08	650.0308	647.0074	660.1656	640.7957
9th floor area (m)	555.715	670.1748	661.3352	670.8213	632.0815
10th floor area (m)	571.35	690.3177	646.7752	681.477	623.3673
11th floor area (m)	586.98	680.1152	632.2152	692.1327	614.6531
12th floor area (m)	602.6125	669.9127	617.6552	702.7884	605.9389
13th floor area (m)	618.245	659.7102	603.0952	713.4441	597.2247
Roof area (m)	633.89	649.5077	588.5082	724.0998	588.5105
Total built up area	8782.3419	9346.2387	9451.5456	10385.0056	9518.8193
Coverage area	743.7028	770.9458	787.5378	846.7326	757.05
Ground/ plot %	0.616572063	0.534871883	0.660995921	0.704267877	0.626745662
Foot print	643.9232	558.5988	690.3177	735.5092	654.5481
Complexity evaluation					
Number of layouts	3	4	4	3	3
Number of layout edges	4.5	6	4	4	7
Number of operations	2	3	3	2	2
Type of operation	3	3	3	3	3
Total number of point	12.5	16	14	12	15
complexity index	0.69	0.89	0.78	0.67	0.83

In Table 28 solutions are ranked based on its fitness to each evaluation criteria comparing to the other solutions. For built up area, solutions are ranked from 1 (smallest area) to 44 (largest area. While for coverage area, solution are ranked from 1 (smallest area) to 30 (largest area) since some of the solution has the same coverage area. For best fit and foot print both area ranked from 1 (smallest value) and 6 (largest value). And finally for complexity, the solution that has the largest number of points was give value 1 and an index for the other solution was calculated accordingly.

Table 28: Solutions final evaluation and ranking.
Source: author.

Final evaluation and ranking										
solution number	Built up area		Coverage area		Ground/ plot %		Foot print		Complexity	
	area (m²)	rank	area (m²)	rank	percentage	rank	area (m²)	rank	points	index
1	9658.848	33	643.9232	2	61.66%	3	643.9232	3	6.5	0.36
2	9626.4615	30	641.7641	1	61.45%	2	641.7641	2	7	0.39
3	10354.7655	39	690.3177	8	66.10%	5	690.3177	5	5.5	0.31
4	11032.638	44	735.5092	10	70.43%	6	735.5092	6	7	0.39
5	9818.2215	36	654.5481	3	62.67%	4	654.5481	4	10	0.56
6	6442.6605	1	643.9232	2	61.66%	3	643.9232	3	8.5	0.47
7	7007.6564	4	641.7641	1	61.45%	2	641.7641	2	9	0.50
8	7794.8598	15	690.3177	8	66.10%	5	690.3177	5	7.5	0.42
9	7132.821	6	735.5092	10	70.43%	6	735.5092	6	9	0.50
10	7949.1273	19	654.5481	3	62.67%	4	654.5481	4	12	0.67
11	6442.6605	2	643.9232	2	61.66%	3	643.9232	3	8.5	0.47
12	7007.6564	5	641.7641	1	61.45%	2	641.7641	2	9	0.50
13	7794.8598	16	690.3177	8	66.10%	5	690.3177	5	7.5	0.42
14	7132.821	7	735.5092	10	70.43%	6	735.5092	6	9	0.50
15	7949.1273	20	654.5481	3	62.67%	4	654.5481	4	12	0.67
16	9658.848	34	833.651	23	61.66%	3	643.9232	3	11.5	0.64
17	9626.4615	31	845.4696	24	61.45%	2	641.7641	2	12	0.67
18	10354.7655	40	936.8316	28	66.10%	5	690.3177	5	12.5	0.69

19	11032.638	44	1134.5143	30	70.43%	6	735.5092	6	13	0.72
20	9818.2215	37	1072.566	29	62.67%	4	654.5481	4	18	1.00
21	7704.6124	14	643.9232	2	61.66%	3	643.9232	3	7	0.39
22	7877.17	17	641.7641	1	61.45%	2	641.7641	2	7.75	0.43
23	8317.6288	23	690.3177	8	66.10%	5	690.3177	5	7	0.39
24	8360.8991	24	735.5092	10	70.43%	6	735.5092	6	7.75	0.43
25	7239.775	10	654.5481	3	62.67%	4	654.5481	4	12	0.67
26	9643.0389	32	885.295	26	61.66%	3	643.9232	3	10	0.56
27	9989.8751	38	894.8837	27	61.45%	2	641.7641	2	10	0.56
28	10692.977	43	805.9185	19	66.10%	5	690.3177	5	10	0.56
29	10432.9769	42	800.9921	18	70.43%	6	735.5092	6	13	0.72
30	9739.3466	35	829.5864	22	62.67%	4	654.5481	4	13	0.72
31	6797.5548	3	659.9901	4	61.66%	3	643.9232	3	14	0.78
32	7494.9599	12	821.9031	21	61.45%	2	641.7641	2	14	0.78
33	7574.879	13	686.0642	7	66.10%	5	690.3177	5	14	0.78
34	7263.3429	11	734.3374	9	70.43%	6	735.5092	6	17	0.94
35	7207.807	9	661.1533	5	62.67%	4	654.5481	4	17	0.94
36	7136.199	8	763.8843	15	61.66%	3	643.9232	3	14	0.78
37	8040.85115	21	821.8658	20	61.45%	2	641.7641	2	14	0.78
38	8270.941	22	752.618	12	66.10%	5	690.3177	5	14	0.78
39	8704.7534	25	756.0994	13	70.43%	6	735.5092	6	17	0.94
40	7899.7001	18	681.4298	6	62.67%	4	654.5481	4	17	0.94
41	8782.3419	26	743.7028	11	61.66%	3	643.9232	3	12.5	0.69
42	9346.2387	27	770.9458	16	53.49%	1	558.5988	1	16	0.89
43	9451.5456	28	787.5378	17	66.10%	5	690.3177	5	14	0.78
44	10385.0056	41	846.7326	25	70.43%	6	735.5092	6	12	0.67
45	9518.8193	29	757.05	14	62.67%	4	654.5481	4	15	0.83

5.4 Selection stage

In this proposed technique, the selection stage is left entirely to the designers, using their accumulated knowledge, experience and creativity, to decide the optimum solution that best matches the design goals and criteria based on his/ her conviction. This section will attempt to further explore this process by taking the role of the designer, setting design goals, and selecting a number of solutions, where each one of these solutions achieves the goals and/or satisfy the aspects that affect the designer decision.

Goal 1: low footprint, low cost. An initial look may suggest that solution number 42 is the best solution that achieve this goal since it has the least footprint, however, when we take into consideration the fact that it also has one of the highest complexity index as well as a large total built up area which directly increase building costs, solution No. 42 is no longer the best fit. On the other hand, solution number 2 has one of the lowest footprint values among the solutions and a relatively low complexity index; which makes it a more preferable solution.

The evaluation process, in this thesis, brought out the idea of evaluation in the generative design methodologies from evaluation based on goal optimization to allow a comparison between a numbers of goals. This approach equipped the designer with a deeper understanding of the consequences of selecting each solution, which makes it more convenient in the design early stage.

Goal 2: large built up area, low complexity. In this case, Solution number 4 and solution number 19 have the largest built up area (110321.638 m^2), however, solution number 4 has lower complexity index (0.41) than solution number 19 (0.71). Other solutions may also be considered such as solution number 3 which has a large built up area (10354.76 m^2) with a very low complexity index (0.32).

Goal 3: large built up area with allowing areas for landscaping. Solutions that have lowest “best fit” rank have a better potential to incorporate landscape. However, most of these solutions have large built up area rank as well, such as solutions number 4, 19, 44, 29 and 28. Since the main goal is to find solutions that satisfy multiple criteria and not optimization, solution number 27 is the best solution that achieve a good rank for the two criteria at the same time.

CHAPTER SIX

CONCLUSION

Chapter 6 Conclusion

The conclusion brings together the outcomes in the view of the objectives of this dissertation, both in terms of generative design approach from architectural perspective and more detailed aspects concerning the developed technique. In addition, it shows the main contributions of this dissertation to the existing knowledge and propose further research areas related to the topic. The main objectives that were introduced at the beginning of the research are:

- ***Developing a design algorithm based on a generative design approach for generating and automating the concept design of a building form. An algorithm that enables the exploration of many design solutions or ideas according to a defined set of requirements.*** The development of the technique initiated by proposing a procedural structure in chapter 3, formulating an algorithm in chapter 4 and finally implementing the technique for validation in chapter 5.
- ***Examining the existing generative design approaches and identifying the main challenges that limit their implementations. And Identifying the potential and implications of generative design system for generating building form.*** In chapter 2 (literature review), seven common generative design strategies were discussed and analysed as well as examples for each strategy were presented. In the end of literature review chapter, the challenges and limitation were identified.
- ***Establishing an architectural framework for adopting generative design approach for building form exploration process.*** Chapter 3 presents the methodological framework.

6.1 Summary of the research

This dissertation sets out to investigate the use of generative design approach for design exploration in the context of building form finding. It recognizes generative design potential to explore form integrally with instant evaluation process and questions the potential of enabling the generation of creativity aspects of the design process in such trajectory. The main focus of the dissertation is to develop a novel design technique for generating building form in the early stage of the design. The development of the technique was carried out by systemizing and structuring the design procedure and exploring how incorporating multiple constraints affects the generation process.

In general, the dissertation started by literature review chapter that presents studies on design thinking, design procedure and the influence of computation and the invention of computers on architectural design. The chapter also defined generative design and the current strategies developed by researchers to employ this approach for automating specific design tasks. Throughout the literature, the main challenge facing the computation and automation of design tasks has been identified by understanding the ill-structure nature of design problems and the creativity aspect of the design procedure. The examination of current generative design strategies showed the tendency of utilizing this approach for evaluating and optimizing building forms with respect to performance criteria or for generating building form alternatives based on specific architectural style. All the strategies focused on the geometrical transformation method by developing algorithm that depends heavily on designer inputs and feedback.

After identifying the main challenges in the current practices, the development of the technique is initiated by proposing an exploratory methodology where the structure of the design procedure was systemized and reorganized by defining it into tasks, each are characterized by input, relationships, constrains, and output. Some of designs tasks were hard or impossible to be characterized as long as they are related to the human intelligence. Their input is undefined due to the fact that it various according to each designer knowledge, taste and experience. Their results in the other hand, are unexpected and unmeasurable. This tasks is problematic if technique operator is a machine (computer). In case of human operator, the technique will act as a guidance and it's enough to put the undefined tasks into its place in the technique operations sequence because the human brain is able to solve the task without detailed instructions. However, in case of machine operator, solving these tasks will either require a feedback and interaction from the designer, or will require replacing these tasks with well identified tasks that can result in the same type of outputs. Requesting a feedback from the designer, with respect to this research purpose, was not an option to be used in the proposed technique since it indicates a failure in exceeding the current applications of the use of generative design approaches in the design procedure.

A second trajectory for the research was incorporated. The trajectory is how to replace the undefined design tasks to well defined ones that the machine (computer) can solve. For designing a building, the undefined tasks are associated with producing design ideas and inspirations to create the geometry of the layouts or the overall building. The method used to formulate this type of tasks depends on analysing the outputs of the traditional tasks of creating design ideas and suggesting a new approach for creating similar outputs by a pure geometric abstraction of basic spatial relations. The challenge was to maintain aspects related to design creativity and avoid falling in the problem of architecture style. In this manner, the principle of randomness in selecting variables was used as a way for supporting creativity.

Moreover, considerations were taken for the constraints that control the generation process determined by the designer inputs and computer preference variables. Designer inputs were limited to realistic constraints and goals that the designer deals with in the traditional design procedure; mostly specified according to the project site and client requirements. However, constraints of the mechanisms that are related to the creativity have not included any inputs from the designer. This is to ensure that the generated solutions are not affected by the designer taste or experience, in order to have a wider range of alternatives that include a new designs probably which cannot be developed by the designer in the normal design process.

The designer intentions were considered in the evaluation process. The concept of evaluation is shifted in the proposed technique from being an optimization process to become a tool that shows the real variation between solutions other than the geometrical composition. The purpose of evaluation in the purpose technique is equipping the solutions with the necessary information. The evaluation stage also integrated qualitative criteria by converting it into quantitative measurements such as the proposed Archi-complexity index to measure the complexity of the generated solutions. In this way, designer intentions can be fulfilled in the selection stage when the generated solutions are presented to the designer with their rank according to multiple quantitative and qualitative design criteria.

The theoretical framework of the technique was successfully tested out and validated by implementing it over a serious of examples categorized into three groups:

- Examples directed to test out the proposed variables in each process,
- Examples directed to validate the ability of whole technique to generate forms for a hypothetical project requirements and

- Examples of scripting the mechanisms using the algorithmic editor interface Grasshopper incorporated with visual programming language Rhino 3d.

6.2 Contributions of the research

The most important contribution of this research is the development of a new design method in generative design paradigm that will generates building forms. The method is supported by the formulation of new mechanisms to enable the exploration of building forms by generating a wide spectrum of possible outputs that cannot be developed manually by designer.

The ***"Backward triangulation"*** mechanism developed by the researcher presents unprecedented idea to initiate the building generation process. Unlink the existing generative design methods that require at least the specification of geometrical units or basic layouts from the designer to start the process, this mechanism generates its own geometrical base pattern in unconventional way that is driven by mathematical relationships and the concept of randomness. The role of *backward triangulation* mechanism is not limited to feeding the process with geometrical compositions but also provide a spatial organizational logic that allows structural system identification.

The ***"Building layout formation"*** is another novel mechanism that was invented in this research to generate the first layer of building form which is layouts. The mechanism has successfully extracted unlimited number of alternatives of building layouts from the geometrical base pattern generated by *backward triangulation* mechanism. To this extend, these two mechanisms have significantly supported the diversity of designs outputs where the units used in the generation process, for the first time, are not predefined by the designer.

This is continued in the generation of the third dimension of the building by introducing a multiple transformation types of extrusion, blend, scale and twist or a combination of them where the concept of randomness in selecting type of transformation for each alternative is playing an important role in supporting diversity of outcomes and producing a large number of designs.

The ***"Structural node identification"*** process provide integrity between the architectural form design and the structural design by building the structural system up on the geometrical base pattern that is been used in the generation process. Here, the form of the building moves away from being a limitation in designing the structural system to being an inspiration for generating multiple solutions for the building structural system.

The evaluation process, in the proposed technique, brought out the idea of evaluation in the generative design approaches from evaluation based on goal optimization to allow a comparison between numbers of goals by incorporating the principles of **exclusion and ranking**. This approach equipped the designer with deeper understanding of the consequences of selecting each solution which make it more convenient in the early stage of the design.

The ***"Archi-complexity index"*** proposed in the evaluation process has transferred a qualitative measurement into quantitative one that the computer can calculate. By this, the research technique has set an examples of the possibilities of quantifying the qualitative important aspects to achieve better evaluation process.

The generalization sense of the technique is significant to broaden its implementations in the architectural field and to be more convenient for the users. The proposed generative technique is "novel" because of its capability in generating building forms despite the building type or function as well as its inclusion to all parts of the design procedure. The generalization was achieved by defining the input by site and client requirements. The researcher argues that generalization derived by critical choice of inputs data is needed to bridge the gap between the research and the practice in this topic.

6.3 Research limitations

The main limitation is related to the nature of generative design as interdisciplinary approach that requires collaboration between numbers of experts in the field of architectural design, structural design, computer science, mathematics and philosophy. Even though, this dissertation investigates the topic in architectural perspective, the feasibility of the proposed technique depends on programming aspects. In the dissertation, architectural-friendly visual scripting interface (grasshopper) was used to verify key aspects and concepts, however, a further development can only be done under a team of experts.

Another limitation has been addressed at the end of 3d transformation process is that some variables can only be validated or developed by observing the system behaviour during the operation after fully scripting the proposed technique or by examining the technique over a large number of examples. For example, the minimum number of solutions that must be generated before the exclusion part in the evaluation process to ensure a sufficient number of solutions reach the selection stage.

6.4 Future work

The further research should evolves around scripting the proposed method in the purpose of developing a software that use commercial available CAD , e.g. AutoCAD, to extract the building proprieties as form is done.

Transformation types proposed for this technique have been chosen with respect to the assumption that the geometrical 2d composition generated from “layout formation” stage are layouts. Further development can propose a different type of transformation with taken the geometrical 2d composition as sections. Transformation types such as revolve and sweep may be more convenient in this case.

Further development can be done to customize the technique for single-story building forms where other ways of generating the third dimension may be more suitable for this case.

Further development can be done on “*archi-complexity index*” by incorporating other factors.

References

Aamodt, A. (1994). Case-based reasoning : foundational issues , methodological variations , and system approaches. *AI Communications*, vol. 7 (1), pp. 39–59.

Agkathidis, A. (2015). Generative design methods. *Real Time: Extending the Reach of Computation, eCAADe 33*, vol. 2, pp. 47–55.

Akin, Ö. (2002). Case-based instruction strategies in architecture. *Design Studies*, vol. 23 (4), pp. 407–431.

Archer, L. B. (1979). Whatever became of design methodology. *Design Studies*, pp. 17–18.

Ault, H. K. (1999). Using geometric constraints to capture design intent, vol. 3 (1), pp. 39–45.

Bentley, P. & Wakefield, J. (1997). Generic evolutionary design. *Soft Computing in Engineering Design …*, pp. 1–10.

Bhardwaj, R. & Upadhyay, A. (2017). Cellular automata: *Journal of Organizational and End User Computing*, vol. 29 (1), pp. 42–50.

Borrego, M., Douglas, E. & Amelink, C. (2009). Quantitative, qualitative, and mixed research methods in engineering education. *Journal of Engineering Education*, vol. 98 (January), pp. 53–66.

Braha, D. & Maimon, O. (1997). The design process : properties , paradigms , and structure. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS*, vol. 27 (2), pp. 146–166.

Cenani, S. & Cagdas, G. (2007). A shape grammar study: form generation with geometric islamic patterns. *Proceedings of 10th Generative Art Conference GA2007*, pp. 1–7 [online].Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:A+Shape+Grammar+Study+:+Form+Generation+with+Geometric+Islamic+Patterns#0.

Chakrabarti, A., Shea, K., Stone, R., Cagan, J., Campbell, M., Hernandez, N. V. & Wood, K. L. (2011). Computer-based design synthesis research: an overview. *Journal of Computing and Information Science in Engineering*, vol. 11 (2), p. 21003.

Colakoglu, B. & Keskin, G. (2010). Form generator: a cad tool for conceptual design development. *28th eCAADe Conference Proceedings*, pp. 411–418.

Cross, N. (1993). Science and design methodology: a review. *Research in Engineering Design*, vol. 5 (2), pp. 63–69.

Cross, N. (2011). *Design Thinking: Understanding How Designers Think and Work*. First Edit. New York: Berg Publisher.

Dİno, İ. G. (2012). Creative design exploration by parametric generative systems in architecture. *METU JOURNAL OF THE FACULTY OF ARCHITECTURE*, vol. 1 (29), pp. 207–224.

Duarte, J. P. (2001). Customizing mass housing : a discursive grammar for siza's malagueira houses. *Architecture*, vol. 14 (2), pp. 1–536.

Eiben, A. E., Michalewicz, Z., Schoenauer, M. & Smith, J. E. (2007). Parameter control in evolutionary algorithms. *Parameter Setting in Evolutionary Algorithms*, vol. 54 (2), pp. 19–46.

Flemming, U. (1987). More than the sum of parts: the grammar of queen anne houses. *Environment and Planning B: Planning and Design*, vol. 14 (3), pp. 323–350.

Ganzerli, S., Pantelides, C. P. & Reaveley, L. D. (2000). Performance-based design using structural optimization. *Earthquake Engineering & Structural Dynamics*, vol. 29 (11), pp. 1677–1690.

Garza, A. G. de S. & Maher, M. Lou. (2001). Using evolutionary methods for design case adaptation. *ACADIA 01 Reinventing the Discourse*, pp. 180–191.

Gerber, H., Anthony, O. & Abrams, S. (2007). Mixed methods research. *Journal of Mixed Methods Research*, vol. 1 (2), pp. 112–133.

Gero, J. S. & Chase, S. (2001). A generative design system applied to siza's school of architecture at oporto luisa caldas and joão rocha. *Caadria*, (1999), pp. 253–264.

Goel, V. & Pirolli, P. (1992). The structure of design problem spaces, vol. 429, pp. 395–429.

Groat, L. & Wang, D. (2007). *architctural research methods*. Second Edi. *Journal of Experimental Psychology: General* Second Edi. Edited by J. W. & Sons. Hoboken, New Jersey.

Grobman, Y. J., Yezioro, A. & Capeluto, I. G. (2009). Computer-based form generation in architectural design — a critical review. *International Journal of Architectural Computing*, vol. 7 (4), pp. 535–553.

Guidera, S. (2011). Conceptual design exploration in architecture using parametric generative computing: a case study. *2011 Asee Annual Conference & Exposition*.

Guo, Z. & Li, B. (2017). Evolutionary approach for spatial architecture layout design enhanced by an agent-based topology finding system. *Frontiers of Architectural Research*. Elsevier B.V., vol. 6 (1), pp. 53–62.

Herr, C. M. & Arch, D. (2002). Generative architectural design and complexity theory. *Proceedings of the Generative Art Conference*, pp. 1–13 [online].Available at: http://cumincad.scix.net/cgi-bin/works/Show?_id=ga0213&sort=DEFAULT&search=%2B2002 Potential Performative Effects Contemporary Techniques in Architecture&hits=488.

Heylighen, A. & Neuckermans, H. (1993). (learning from experience) 2 : promises , problems and side-effects of cbd in architecture. , in *Architectural Computing from Turing to 2000*. Liverpool, UK: eCAADe, pp. 567–575.

Jabi, W. & Grochal, B. (2013). The potential of evolutionary methods in architectural design. *Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe.*, vol. 2, p. 217- [online].Available at: http://orca.cf.ac.uk/49324/.

Jonassen, D. H. (1997). Instructional design models for well-structured and ill-structured problem-solving learning outcomes. *ETR&D*, vol. 48 (0), pp. 1042–1629.

Jonassen, D. H. (2000). Toward a design theory of problem solving. *ETR&D*, vol. 48 (4), pp. 63–85.

Kim, B., Rudin, C. & Shah, J. (2015). The bayesian case model: a generative approach for case-based reasoning and prototype classification [online].Available at: http://arxiv.org/abs/1503.01161.

Kim, J., Ahn, C. & Lee, S. (2018). Modeling handicapped pedestrians considering physical characteristics using cellular automaton. *Physica A: Statistical Mechanics and its Applications*. Elsevier B.V., vol. 510, pp. 507–517.

Kolarevic, B. (2009). Digital morphogenesis. *Architectural Design*, vol. 79 (1), pp. 11–28.

Koning, H. & Eizenberg, J. (1981). The language of the prairie: frank lloyd wright's prairie houses. *Environment and Planning B: Planning and Design*, vol. 8 (3), pp. 295–323.

Krawczyk, R. J. (2002a). Architectural interpretation of cellular automata. *Proceedings of the Generative Art Conference*, p. 7.1-7.8 [online].Available at: http://www.generativearts.com.

Krawczyk, R. J. (2002b). Experiments in architectural form generation using cellular automata. *Proceedings of the 20th eCAADe Conference*, (1970), pp. 552–555.

Krish, S. (2011). A practical generative design method. *CAD Computer Aided Design*. Elsevier Ltd, vol. 43 (1), pp. 88–100.

Lawson, B. (1979). Cognitive strategies in architectural design. *Ergonomics*, vol. 22 (1), pp. 59–68.

Lawson, B. (2006). *How designers think : the design process demystified*. 4th edn. Amsterdam.

Lobos, D. & Donath, D. (2010). The problem of space layout in architecture: a survey and reflections. *Arquitetura Revista*, vol. 6 (2), pp. 136–161.

Lowgren, J. (1995). Applying design methodology to software development. , in *Proceedings of the 1st conference on Designing interactive systems: processes, practices, methods, & techniques*. Michigan, USA: Ann Arbor, pp. 87–95.

Lynn, G. (1999). *Animate form*. 1 st. New York: Princeton Architctural press.

Matchett, E. (1968). Control of thought in creative work. *Chartered Mechanical Engineer*, vol. 14 (4).

Menges, A. & Ahlquist, S. (2011). *Computational design thinking*. First Edit. Chichester, UK: John Wiley & Sons.

Merrell, P., Schkufza, E. & Koltun, V. (2010). Computer-generated residential building layouts. *ACM SIGGRAPH Asia 2010 papers on - SIGGRAPH ASIA '10*, vol. 29 (6), p. 1.

Michalek, J., Choudhary, R. & Papalambros, P. (2002). Architectural layout design optimization. *Engineering Optimization*, vol. 34 (5), pp. 461–484.

Möller, O., Foschi, R. O., Quiroz, L. M. & Rubinstein, M. (2009). Structural optimization for performance-based design in earthquake engineering: applications of neural networks. *Structural Safety*. Elsevier Ltd, vol. 31 (6), pp. 490–499.

Neumann, J. (1951). The general and logical theory of automata. *Cerebral mechanisms in behavior; the Hixon Symposium*, vol. 10 (3), pp. 1–41.

Oxman, R. (2008). Performance-based design: current practices and research issues. *International Journal of Architectural Computing*, vol. 6 (1), pp. 1–17.

Oxman, R. (2009). Performative design: a performance-based model of digital architectural design. *Environment and Planning B: Planning and Design*, vol. 36 (6), pp. 1026–1037.

Packard, N. H. & Wolfram, S. (1985). Two-dimensional cellular automata. *Journal of Statistical Physics*, vol. 38.

Petruševski, L., Devetakovic, M. & Mitrovic, B. (2009). Self-replicating sys tems in spatial form generation - th e concept of cellular, (January), pp. 8–14.

Plack, M. M. (2005). The reflective practitioner: reaching for excellence in practice. *Pediatrics*, vol. 116 (6), pp. 1546–1552.

Prusinkiewicz, P. & Streibel, D. (2007). Constraint − based modeling of three − dimensional shapes. , in *Constraint − based modeling of three − dimensional shapes*.

Rodrigues, E., Gaspar, A. R. & Gomes, Á. (2013). An approach to the multi-level space allocation problem in architecture using a hybrid evolutionary technique. *Automation in Construction*. Elsevier B.V., vol. 35, pp. 482–498.

Salih, A. G. & Ahmed, H. A. (2014). The effective contribution of software applications in various disciplines of civil engineering. *International Journal of Civil Engineering and Technology*, vol. 5 (12), pp. 976–6308.

Schmitt, G., Dave, B., Faltings, B. & Smith, I. (1994). Case based design in architecture. *Artificial Intelligence in Design- AID '94*, (Section 3), pp. 145–162.

Shi, X. (2010). Performance-based and performance-driven architectural design and optimization. *Frontiers of Architecture and Civil Engineering in China*, vol. 4 (4), pp. 512–518.

Si, F. & Wang, T. (2015). Building massing optimisation in early design stage. , in Y. Ikeda, C. M. Herr, D. Holzer, S. Kaijima, M. J. Kim. M, A, S. (ed.) *Emerging Experience in Past, Present and Future of Digital Architecture, Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA*. Hong kong: The Association for Computer-Aided Architectural Design Research, pp. 583–592.

Simon, H. A. (1973). The structure of ill structured problems. *Artificial Intelligence*, vol. 4 (3–4), pp. 181–201.

Singh, V. & Gu, N. (2012). Towards an integrated generative design framework. *Design Studies*. Elsevier Ltd, vol. 33 (2), pp. 185–207.

Stiny, G. & Mitchell, W. J. (1978). The palladian grammar. *Environment and Planning B: Planning and Design*, vol. 5 (1), pp. 5–18.

Suzuki, H., Ando, H. & Kimura, F. (1990). Geometric constraints and reasoning for geometrical cad systems. *Computers and Graphics*, vol. 14 (2), pp. 211–224.

Terzidis, K. (2004). Algorithmic design : a paradigm shift in architecture ? *digital design tools 2*, pp. 201–207.

Walliman, N. (2011). *Research Methods the basics*. First edit. 270 Madison Avenue, New York: the Taylor & Francis e-Library.

Wang, Y. & Duarte, J. P. (2001). Automatic generation and fabrication of designs. *Automation in Construction*, vol. 11 (3), pp. 291–302.

Watson, I. & Perera, S. (1997). Case-based design: a review and analysis of building design applications. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, vol. 11 (1), pp. 59–87 [online].Available at: http://journals.cambridge.org/abstract_S0890060400001840.

Wong, S. S. Y. & Chan, K. C. C. (2009). Evoarch: an evolutionary algorithm for architectural layout design. *CAD Computer Aided Design*. Elsevier Ltd, vol. 41 (9), pp. 649–667.

Yi, Y. K. & Malkawi, A. M. (2012). Site-specific optimal energy form generation based on hierarchical geometry relation. *Automation in Construction*. Elsevier B.V., vol. 26, pp. 77–91.

Zhang, L., Zhang, L. & Wang, Y. (2016). Shape optimization of free-form buildings based on solar radiation gain and space efficiency using a multi-objective genetic algorithm in the severe cold zones of china. *Solar Energy*. Elsevier Ltd, vol. 132, pp. 38–50.

Zhou, M., Chen, Z., He, W. & Chen, X. (2010). Representing and matching simulation cases: a case-based reasoning approach. *Computers and Industrial Engineering*. Elsevier Ltd, vol. 59 (1), pp. 115–125.