VISION FOR ADVANCED BUILDING DESIGN TECHNIQUES – COMPUTATIONAL MODELING AND GENERATIVE DESIGN

By

SIVA REDDY KONDAMADUGULA

A THESIS PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN CONSTRUCTION MANAGEMENT

UNIVERSITY OF FLORIDA

2021

To my mother, Mrs. Tulasi, father, Mr. Jayarami Reddy, and brother, Leela Sankar Reddy for all the hardships you have been through and showing me nothing but unparallel love

Amma, Nana, Anna – For all the sacrifices you have been making for me. I am forever indebted.

ACKNOWLEDGMENTS

I thank the M.E. Rinker, Sr. School of Construction Management initially for giving me the opportunity to enroll into their master's program. I am grateful for all the faculty that have shared their immense knowledge to us students so seamlessly.

I thank Dr. R. Raymond Issa, Dr. Larry C. Muszynski, and Dr. Idris Jeelani, for serving as my committee members. I specially thank Dr. R. Raymond Issa for his and guidance and knowledge that were critical to the completion of my thesis.

I would like to thank my friends and well-wishers, for their constant support. This would not have been possible without their unwavering support.

TABLE OF CONTENTS

page

LIST OF FIGURES

Abstract of Thesis Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Master of Science in Construction Management

VISION FOR ADVANCED BUILDING DESIGN TECHNIQUES – COMPUTATIONAL MODELING AND GENERATIVE DESIGN

By

Siva Reddy Kondamadugula

August 2021

Chair: - R. Raymond Issa
Major: - Construction Management

Although parametric modeling can solve some exciting and challenging design problems, it still has some heuristic barriers to cross reach its full potential for end-users to solve these problems and step into advanced modeling techniques using computational design. This research utilizes the available ready-to-use nodes in Dynamo and Grasshopper for creating custom scripts in Python, which opens new doors to modeling/designing and controlling the Revit modeling environment.

Revit also has an API (Application Programming Interface) that will enable software developers/designers/engineers to link their sources/applications with Revit's data sources—accessing Revit API through Python in Dynamo interface for generating and manipulating the elements in the Revit environment, that allows the user to have complete control on how to represent the elements and their relationships. This research intends to outline a novel approach to how computational modeling can be used in Building Information Modeling (BIM). The goals are to design spaces in BIM software effectively while taking advantage of relationships between elements in the BIM world. This approach aims to explore design problem scenarios and generate solutions using Parametric Modeling using visual programming and python scripts.

Parametric modeling through Revit has made great advances in the AEC industry. However, to advance past this phase, the design must be modeled computationally (Computational Modeling), inputting a set of instructions by developing a path through which the data can flow. The next phase of computational modeling would be Generative Design, Generation, and Design Optimization, both of which fall under the tree of Generative design.

After developing models to solve specific design problems (through Experiments and Case Studies), the results of the Computational Modelling approach show that Computational Modelling can be applied from a smaller scale to a larger scale. A smaller scale is defined as manipulating relations between Revit/Rhino elements, for example, creating a function that allows selecting all doors (or windows or other categories) once the process is activated. This is achieved through scripts, and the user interface (UI) for this function can be created under Revit UI to activate it when clicked upon. A larger scale is developing a plugin that takes in Rhino/Revit geometry (IFC exported file) as input and converts this geometry into parametric Revit elements. BIM managers and Project Managers can create plugins if that would save time in the long run of company's life cycle. If not, they can generate shorter and quick solutions(scripts) to reduce monotonous and repetitive tasks, resulting in a significant amount of time savings.

CHAPTER 1
INTRODUCTION

There have been many advancements in the field of Building Information Modelling in recent years. After introducing the most advanced software platform – Revit by Autodesk, in 2000, the construction industry was never the same. Considering all the tools and data representation that this advanced software allows the designers to use, some challenges hinder the construction project's modeling process. Even though BIM has pushed the AEC industry's limits, the real values of BIM largely depend upon effective information.

In Traditional modeling or Traditional Design, the architects or the designers were acquainted with representing ideas/designs in form lines through software such as AutoCAD. The biggest hold-up for this Traditional Design is that each element(line) has no relation to other elements (line, point, text). After Autodesk realized that there are doors to be opened in this field, it purchased a startup company that mainly focused on creating a construction technology whose core fundamentals are 'relationships' between the elements created in its software environment. A couple of years fast forward it, and Revit has started achieving what it was intended to. This Parametric Design approach lets the user define these relationships that the elements have with each other. Such as Door is a family that can only be hosted on to a wall category. The developers have been making constant updates to add new relationships that the end-user can utilize. Unlike the Traditional Design environment, where one change to an element does not differ from another element. In Parametric Design, once design change of one element will automatically affect depending on the relation that this primary element possesses with other elements in the environment.

But in general, the direct relationships that these systems are mostly limited. To created furthermore relationships that are not pre-existing, the best approach is to use a graphical algorithm editor such as Dynamo. This Computational modeling extends the Building Information Modeling (BIM) by laying a logic in which the data flows in the virtual environment.

This study aims to research the existing level of modeling available in Revit, focusing on extents of parametric modeling utilization and general research on how construction companies are utilizing this in their project workflow by interviewing on the challenges they still face in creating construction documents and model design of their projects. Exploring Generative Design techniques that can serve as a solution to complex design problems. Studying how Information/data is being flown around various disciplines in a project, defining problems like information loss, extra time /procedures required for an efficient information transformation between these disciplines.
The research will be presented using appropriate BIM case study projects that are used as examples for modeling techniques. To concrete the statement that this new technology is transforming the 21st-century design practices and making advancements in Architecture, Engineering, and Construction (AEC) disciplines.

1.1 Assumptions Of The Study

Utilizing technology (software) in various stages of architectural layout maybe perhaps not only for visualization and drafting, however in different stages of design phase; including as for example operational connections, form-finding, and industrial and manufacturing generation.

A machine (computer) might function as a generative instrument which would possibly be utilized at the design phase by dependent upon the algorithmic method of

designing since there must become described as considered a synergetic connection among your individual intellect and the laptop method. This type of stride is potential simply with using predictive plans that guarantee a mathematical and dialectic partnership involving your human beings to comprehend, conquer, and fundamentally transcend their physical and mental limits.

1.2 Procedures Used to Execute The Work

This Research has three Phases. Phase 1 is to brief out on the existing circle of knowledge of parametric modeling, various technologies that are available in the market that allow the designers to arrive at solutions to their design problems. Phase 2 of the paper's aim is to focus on the potential of algorithm design where the designer can make further use of these elements that possess relationships with other elements, by writing logic to solve a design situation. Phase 3 of Aim is to research where Generative Design stands in the practical world, the plateau at where option generation and Design optimization stand.

To carry out the Aim of this research paper, a detailed list of objectives will be carried out. Steps and methods that are going to be followed down the line by the researcher. Literature Review will be initially carried out to better understand the boundaries where the knowledge in this topic. Reviewing books on parametric modeling techniques in Revit in various disciplines. Reviewing the computation modeling techniques that are possible without the box nodes, in Dynamo for Revit and Grasshopper for Rhino. Reviewing and understanding the challenges/problems that are faced by designers, architects, engineers using BIM to transfer ideas into construction documents, by reading the technological support forums, internet blogs, articles. Create solutions to

and explore new techniques to model such as generative design and application of visual scripting to model a given design.

To breakdown and understand the Generative Design process at a macro level and solve a complex design/modeling scenario using generative design along with the help of visual scripting such as Dynamo or Fusion. Experiment and document the results of 'Project Refinery' which is an Autodesk beta platform for Generative Design. Set the inputs/criteria that fed into the generative design system and let the design tools crunch through all the possible scenarios and letting the user can run an analysis and finalize the outcomes/outputs. To develop a technique, after reasoning the choice of technique chosen, let the designer(researcher) define, and explore the alternatives for this automation process.

The methodology of Generative design is that it utilizes not just a single application and a single technique. As technology advances, Generative design will become advanced just like machine learning and Artificial intelligence.

The scope of the BIM case study projects (3D models) mentioned in this research paper is to show that computational design is a part of the design process or BIM framework which can help in achieving a more intelligent, comprehensive, and integrated building design, such a design that can be achieved in parallel to the intrinsic architectural concepts of the building's design. And further, explore the capabilities of Generative Design which will essentially save an enormous amount of time for designers.

1.3 Objective Of Study

The Significance of this research is to study the proven uses of Generative Design in the field of Architectural design, and test how useful Computational modeling

can be for designers, engineers, and modelers them help understand which areas computational modeling can be advantageous in and its limitations. The intent of this research is also to how to break down design into small set of rules and actions, and the logic that connects them together. There exist scenarios where a designer must do a certain task in Revit a thousand times repeatedly. This research works on few problems like that, and how we can use VSL and TSL to design an efficient solution. This research also gives the reader access to files and projects (3D Models and scripts) along with detailed description of how the reader can apply the same logic that is used to solve these design situations.

In an industry where every 1 million dollars invested in the industry creates 12 jobs, it is an important responsibility for professionals to find solutions that can push this industry further with the help of these advanced technologies. Sometimes to fix something the system down into its fundamental parts, and that is the approach this research takes in seeking solutions to these interesting problems. While working on this vision for Generative Design, the significance of this research shows in the answer to questions addressing the transition process from Traditional through Parametric and Computational to Generative Design how to utilize technologies like Computational Modeling and Generative Design in addressing them.

1.4 Literature Review And Methodology

Starting with stating the topic briefly, that gives the reader a clear understanding of the area this research is aimed at. Then a Literature review will be carried out, as most of the questions are answered in some sort or form in a literature review. The first focus of this section is on an introduction to terminology and the objectives of BIM and focusing on BIM in parametric and computational design. Then the content and Design

Intent of Building Information Modelling is discussed followed by a categorized introduction to the various technologies that are available out there to build a space in a virtual environment. A brief description of the existing modeling techniques that are being followed in the AEC industry, with help of images and concepts from the respective BIM platform such as Autodesk Revit, ArchiCAD, Bentley Architecture, Tekla Structures, Rhinoceros, Dynamo, Grasshopper.

The following sections focuses on the BIM platforms and technologies, by going through the software packages - user interface and understand the type of modeling approach these software packages take. Then the research jumps into examples that show the reader a problem scenario that is could be encountered when working on a project. Solutions to these problems will be clearly illustrated, the thought process to solve such problems/issues. So, exploring those reviews and understanding what solutions/approaches are available for the modeling problems that this paper aims to research. The main intent is to work on technologies available in the current day and try pushing them to the limits by solving complex design problems. Work at an advanced level with technologies such as Revit, Project Refinery, Dynamo, Python. For Generative Design, the intention is to set criteria for the model, and to design a algorithm to simulate multiple options possible for the given geometry model, and once all the options are generated, how we can analyze, using the UI on the software and compare various outcomes by manipulating the desirable outcomes. By understanding these underlying concepts, through Experiments/Projects and Case studies of projects, to fully understand the potential and limitations of these concepts.

CHAPTER 2
LITERATURE REVIEW

2.1 Generative Design Introduction

The corner stone of architectural design development are the computational systems, that have emerged in the past decade. This signals the growth of research of a new subject that involves design scheme, bolstered by computational models and generative design principles (Gero and Tyugu 1994). The core views of Gero and Tyugu in the field of computer aided design are “the representation and production of the geometry and topology of designed objects” and “the representation and use of knowledge to support or carry the synthesis of designs”. (Gero and Tyugu 1994) Their first view is related to the overall usage of over-the-counter CAD tools which aim to boost the efficacy and to reach the goal of automating the construction and design activities. The later view, has laid path for researching the generative approaches that respect computation all through the design phase, aiming to explore several design possibilities in the same amount of time or even less.

Generative design methods permit the creation of sophisticated compositions, the two conceptual and formal, during the execution of some easy group of parameters and operations. This brand fresh comprehending marks the development of advanced manners of layout believing. The most important challenge is inside the farming of computation for an instrument which matches the designer's own capacities inside the conceptualization and manufacturing of style artifacts from the architectural program (Ahlquist and Menges 2011). Generative Design encompasses the idea of exploring an idea(design) to quickly generate and evaluate high-performing design alternatives. There are multiple Generative Design tools that are available, but the problem is that

most of them require the end user to program/code and have a comprehensive understanding of how algorithms work, to take advantage of these technologies. Generative Design would be your action of trying to find design choices and possibilities throughout the evolution of the style and style about certain limitations and abilities from the circumstance of prospective construction that is, structure as stuff training; the look mining version could be your domain name at which in fact the limitations have been based and their consequences to potential layout choices have been researched. A generative way of creating calls for employing a small collection of principles to create all potential varieties.

Generative Design investigation calls for a specific method of coming in the plan intent (geometric shape). It entails inventing design-related problems, capabilities, and limitations at the plan of the plan procedure and iteratively investigating resultant style and layout alternatives utilizing appropriate and relevant technical instruments and methods. Computational instruments and methods permit the processing of advice, also is digital or physical. Compared to normal layout and style, by which designers will instantly attract the shape of the construction design exploration with computational applications calls for the programmer having a style and layout model with all the goal the subsequent interaction using this specific style and layout model may allow to its discovery of potential design in tents related towards the invented design difficulties. Now, electronics are normally used to get computation and creating that the mining version, however, architects like Antoni Gaudi and Frei Otto have designed and built the mining version - wonderfully, Gaudi’s hanging series version and Otto's soap-bubble machine enabled the designers to socialize using a version and research other variants which couldn't readily be built and experimented with traditional projective resources.

Though an investigation version might be generated utilizing either bodily or electronic technical instruments, advancements in electronic technical power, applications, and calculations have enabled for intricate calculations and the processing of rather considerable levels of info. Utilizing these high-level tools for design investigation, the procedure for hunting for other style and style outcomes would be a cooperation between machine and human (such as computation). This alliance has resulted in style outcomes which had been unthinkable a couple of short ages back. This informative article can be located from the circumstance of this design style and layout and computer-controlled manufacture of structure. It's all about generative layout explorations that develop substantial creations and support the programmer from ideation. Inch The thesis explores a couple of fruitful generative mining types for architects using computation to create shape and make properties.

2.2 Generative Design Definitions

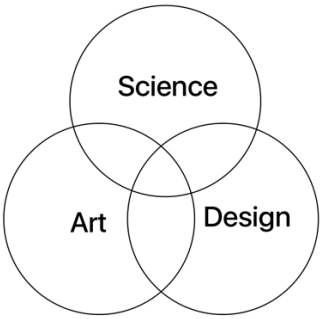


Figure 2-1. Elements of Generative Design

Designers or engineers can input design goals into specifically created generative design software, along with design parameters like materials, manufacturing methods, past creations, and budget limitations. Then, use high performing computers to explore the various possibilities of a solution by quickly generative more design alternatives than ever thought possible.

Generative Design is the way of using one's computing power as a stakeholder to create/generate designs which could be potential solutions, that are analyzed by the user to align with their needs. Generative design is combined with computation through a generative design system using mechanisms (Tang and Chang 2005).

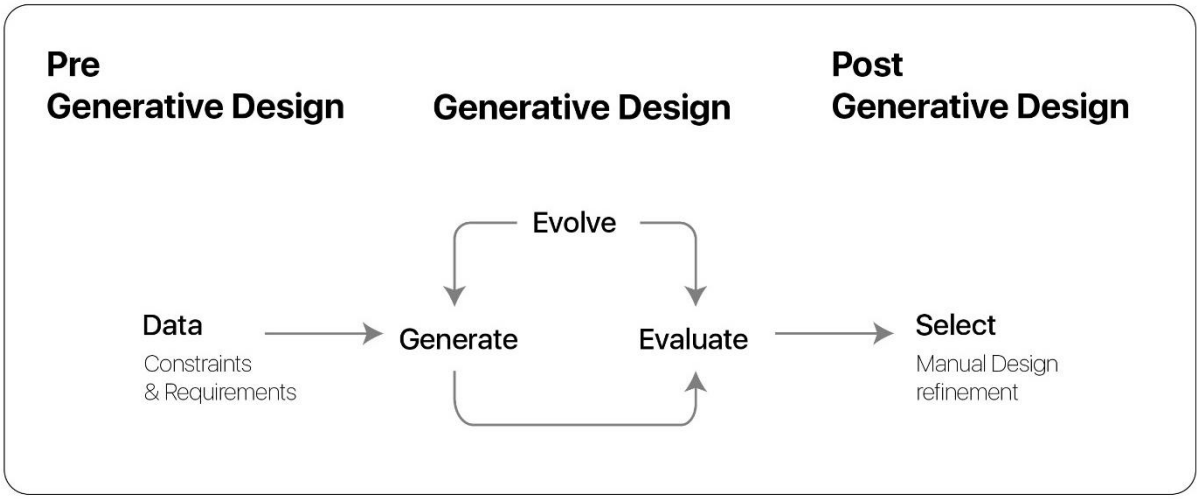


Figure 2-2. Breakdown of Generative Design stages

Generative design emerged from the search for Strategies to Facilitate the exploration of different options in Design, using software technology as options creating machines to navigate large alternative designs and to think of alternatives which were unforeseen. In generative design, algorithms are often used to create a variety of alternative choices based on predefined goals and constraints, the designer then assesses to decide on the best option or intriguing variant. Design options that need a more context-based understanding and detail oriented will be determined by the designer (Negroponte 1975).

2.3 Generative Design Properties

Most of the Generative Design concepts involves steps like those shown in Figure 2-2, that is being applied over many design fields – Architecture, Product Design, Communication Design, Art. The fundamental principle of Generative Design requires:

1. A structural outline of the criteria requirements.
2. A method of producing variants.
3. A method of picking desired results.

Dependent on these sorts of attributes, Generative Design methods deliver substantial advantages of design in the conceptual stages whilst the accent will be on exploration of alternate options. But, probably perhaps one among the absolute most essential rewards is the fact that Generative design surroundings are lively and interactive, so providing real visual responses, whilst the dimensional and geometric variations are all manipulated (Guidera 2011). Several strategies utilize genetic algorithms to produce variants. Some usage only arbitrary amounts. Generative layout and style were motivated by conventional design procedures, where layouts have been manufactured as hereditary variants throughout mutation and crossovers (Guidera 2011).

2.4 Generative Design in Architectural Design

Lots of designers and engineers such as Herbert A. Simon, Lionel March, Yahuda E. Kalay, and many others discussed that the thought of why (generate-test) style and layout and style loops. They identified layout for an outcome compiled by just two generative design engines, one is included with production and the other is for diagnosis. In design studios research and data sites are accumulated and a concept regarding that individual’s architectural design is created, subsequently opportunities are analyzed predicated over a variety of standards set by the designer and clients. This action might be widely displayed as exhibited under in a three-node diagram. On the list of limitations within this treatment seems within the variety of choices that the “design terminology" node may make. It really is commonly quite couple, or none.

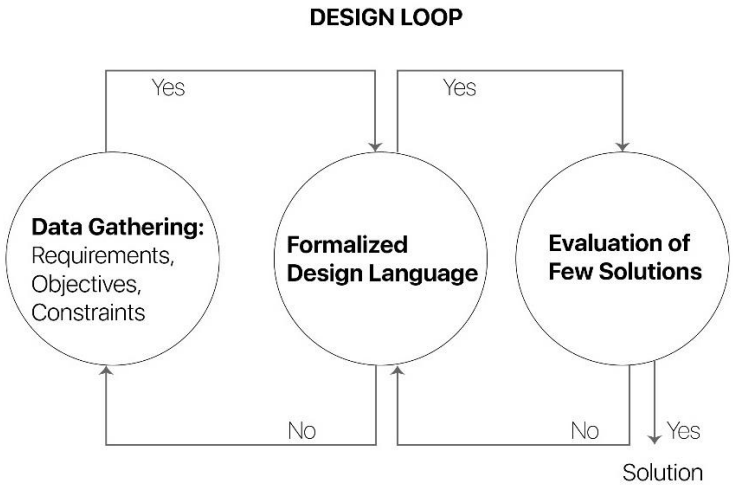


Figure 2-3. Design Loop (Adapted from El-Khaldi 2007)

According to historic design evolution, it is possible to know the way that events rust or mature, last or end, break or sustain. Most pioneers in architecture outsourced their design scheme after constructing a suitable working environment / platform, setting up certain style, exploring two or three or more architectural elements for functions. In terms of computation, we discovered the way L systems, and cellular automata, are used for creating simulation and several other systems were produced for generating designs. The following example indicates a revised possible diagram for integrating generative systems to the plan. A fourth node, “manufacturing", is based between the test and the idea. The most important benefit of integrating such systems to some process is the capability to test “many" options made to compare them. This permits you to catch more possible schematic options for every schematic design. In Figure 2-4 Generative systems and their solution analysis is the final node loop before coming to an agreement with the ultimate solution, for that design challenge.

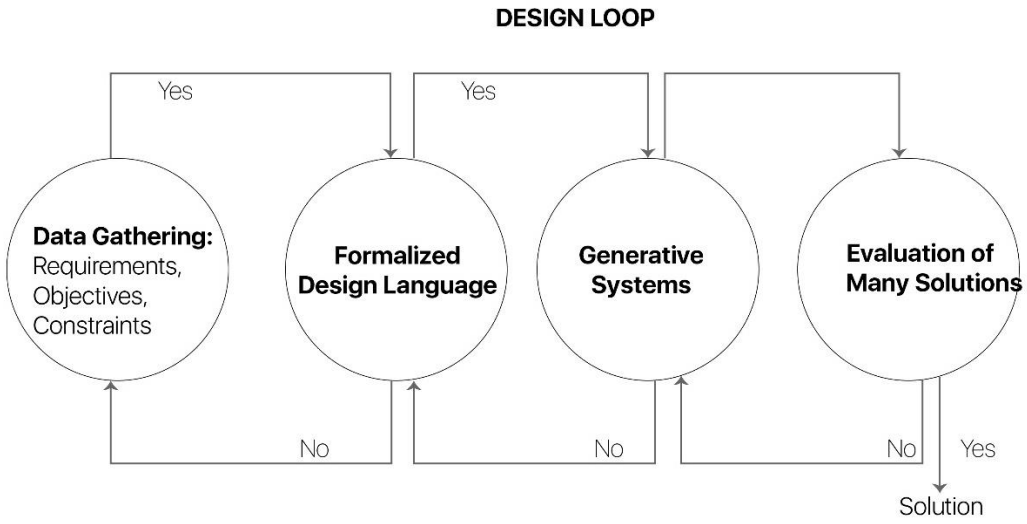


Figure 2-4. Design Loop (Adapted from El-Khaldi 2007)

Generative Design understanding is simply innovative to style and design. Louis Sullivan's plates (Figure 2-6) that make clear processes for copying floral ornamentation predicated on geometrical constructs. An even recent example is Peter Eisenman, which utilized analogue transformational fundamentals in architectural style and layout and style enhancer. Eisenman displays this practice around the plan of an assortment of residences by which he states," the house is not a thing inside the standard feeling - which may be actually the last result of this course of action -- even accurately a series of some techniques" (Eisenman 1977). This accent about the task at the final solution, and the action of conceiving of architectural sort signifies a generative theory as the exact crucial motorist throughout atomic regeneration.

The procedure for generative formation requires four components:- the beginning requirements and parameters (input), a generative mechanism (principles, calculations etc.), the action of creation of these variations (output), and the choice of the ideal version. The plan artifact does not materialize before the fourth measure; hence, a

generative system is thought of as a manufacturing system instead of a representational build. Additionally, "The generative purpose of new electronic techniques is accomplished via the programmer's simultaneous interpretation and interpretation of a computational build. The ability of electronic, computational architectures to create “new" designs is, therefore, highly determined by the designer's perceptual and cognitive capabilities, as constant, dynamic processes floor the emerging form" (Kovacevic 2003)

A linguistic technique can be just a grammar-based formalism by that a set of compositional guidelines (syntax) regulate and contour that exactly the design (semantics). The computational execution of ancestral generative devices mainly manifests itself in shape grammars. Twist grammars specify and employ a couple of alteration rules on a starter thing (a contour) whilst to produce new layout that is complex. As stated by Knight, contour grammars are both more descriptive and invisibly in a means in the alteration rules describe the different kinds of those generated layouts and crank out or calculate styles (Knight 2000). Biological generative style and layout techniques, on the opposite side, adopt a distinct generative design pattern, that necessitates naturel and intricate dwelling organisms because being a precedent and uses its own fundamentals in the derivation and transformation of architectural design (Hensel 2010). Vincent (2009) was more afield onto the emphasis put about the becoming of this proper execution rather than the consequence in itself. Natural development, describing the manners complex organic approaches evolve, both self-organize, and grow, donate to architectural comprehension production into the production of architectural, and especially performative design (Weinstock 2010). Therefore, a more profound engagement with all the personality will be pursued, which

explores the many manners that the fundamental theories like functional integration, performance capacity, and substance resourcefulness (Ahlquist and Menges 2011).

2.4.1 Generative Design System – Definition

According to Leach (2009), to understand the concept of a GD (Generative Design) System, it is better to look at it as a form of production system. But this production system will not be responsible for any specifications of the design intent. But it does specify on how the design intent itself is made. The GD must form the GD system before making the form. This is simply understood as defining the logic of building or modeling something instead of modeling/building it.

Gursel's (2012) noted that the Generative Design system should be given an input, which serve as the building blocks of the design intent. These building blocks form the Generative Design System, which creates a canvas filled with multiple potential solution variations. GD(Generative Design) systems do possess the skill of automating certain design-oriented tasks, the ruleset is used to train the system by the designer, and the system will create variants based on the ruleset. But the ultimate decision is still by designer who gets to decide the final design solution that the system is intended for. The GD system visualizes the set of solutions that are filtered through the designer(human) restrictions/logic.

2.4.2 History Of Generative Design Systems

To understand how technology that supports Generative Design has come to a fascinating phase, we should look at how software has given Designers and Architects to explore design solutions like it. The Figure 2-5 is a linear representation of timelines on four technical aspects, for the reader to comprehend the connections between each of these technologies.

Back in 1975, Architect Benoit B. Mandelbrot has coined the expression "Fractal". After he described the concept of fractal, the term has been unchanged since then. It is a mathematical Nightmare, including Computer Scientists that they were used to Create self-similar structures (Figure 2-8) posing queries on measurements and topology (El-Khaldi 2007).

The very first approach to generative design dates to 1821, by Durand in this book “Partie graphique des cours d’architecture” by Neoclassical Architecture through two design stages. One design going from bottom-top and the other going from top-bottom.

Almost after 10 decades, Louis Sullivan has designed a series of plates that are known as “A system of Architectural Ornament, according with a Philosophy of Man’s Powers”. These drawings show a detailed process on achieving ornamental patterns through geometrical constructs.

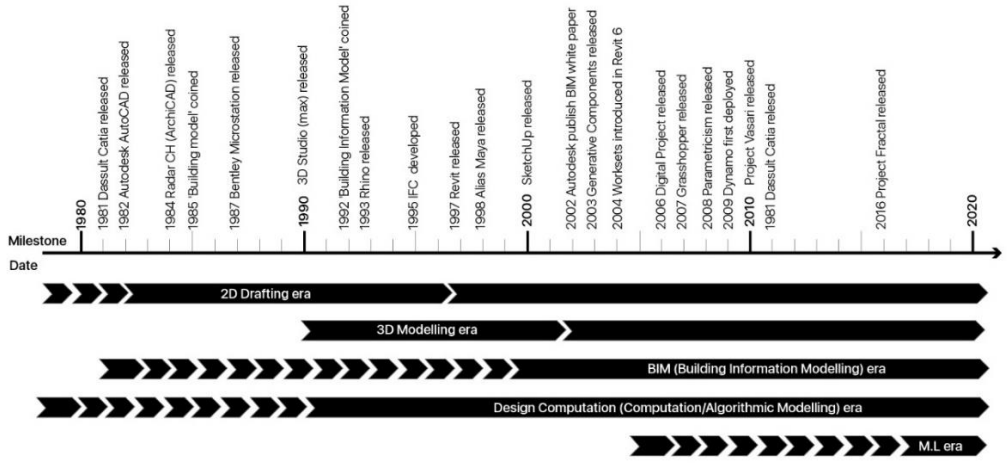


Figure 2-5. Timeline of Architectural Software development

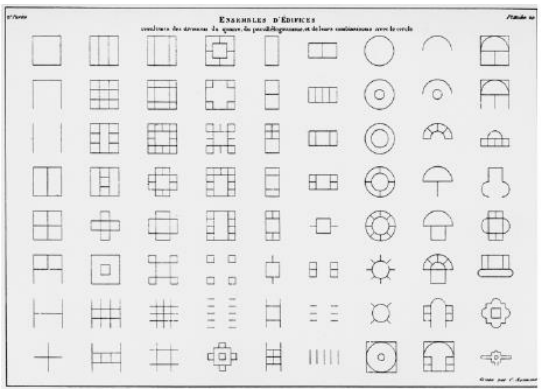


Figure 2-6. Durand's Plate 20 (Getty Research Institute for the History of Art and the Humanities 2000)

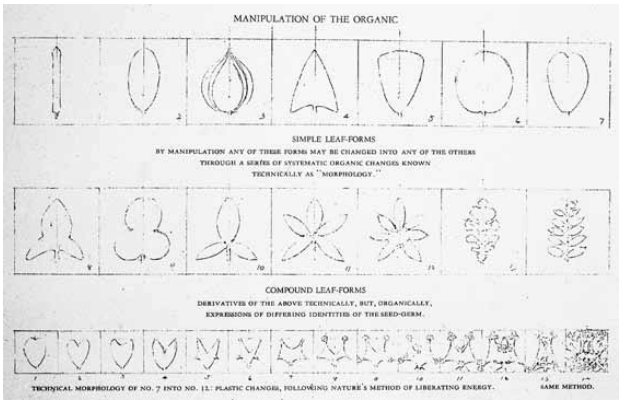


Figure 2-7. Plate 2 – Louis Sullivan (AIA Press 1924)

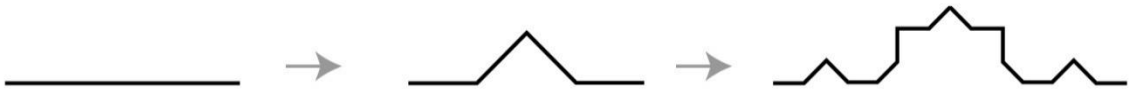


Figure 2-8. Fractal Formalism Visual representation from a Line (Adapted from El-Khaldi 2007)

Most of these concepts reappeared in design to form combinations through the inception of fresh architecture-oriented tools. Their introduction prompted a series of fresh ideas on concept, form, application, tectonics, culture, etc. As an instance, it had been following the introduction of computers when Eisenman explained, "You can establish a string of rule arrangements for entering to the computer not knowing a priori what the proper results would be". (El-Khaldi 2007).

Then the procedure becomes one of analyzing algorithms against potential formal outcomes. The writing and adjusting of those algorithms become among the activities of design. This was the time when William J. Mitchell printed The Logic or Architecture introducing the notions of algorithms, development, grammars, and logic to structure design procedure. Then they began the study of theories about forces dividing applications, or scooping out types, or deforming landscapes and so on. (El-Khaldi 2007).

Forster partners chose that algorithmic design strategy to another degree from the Swiss re undertaking. They utilized parametric systems in which a continuous (relation) connection is established amongst all the different tiers of geometry and the values that impacted how these geometries are modeled. Their system was decided by parametric modeling applications which translates a spreadsheet's numerical data into geometric elements. The logic (geometric and numerical) conveyed a simpler, and remarkably precise design procedure. Since it was parametric, the model was always live and responds to changes almost instantly, which meant a flexibility in design procedure for the architectural firm, which was not known before.

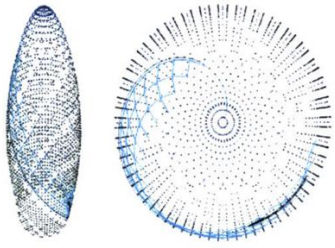
A   B

Figure 2-9. Driving Geometry Points generated from spreadsheet. A) Points plotted using Excel (Verb Matters by Jaime Salazar 2004). B) Present Swiss Re Tower, London, UK (Photo by Alexander Klink)

2.4.3 Process Of A Generative Design

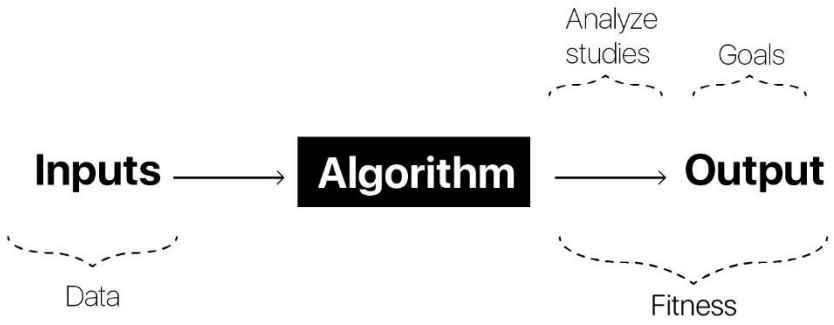


Figure 2-10. Process of Generative Design

Comparing to the origins of generative design idea in other fields such as Music, Literature it is relatively new to Architecture and Design (Mitchell 1979). As stated by Hanna and Barber, “Jean-Nicolas-Louis Durand adopted an analogue generative tactic for its production of neo classical structure by implementing distinct mixes of various elements of the building” (Hanna and Barber 2001). An even recent instance is Peter Eisenman, that utilized analogue transformational guidelines in architectural style and layout and style enhancer. Eisenman's design theory works over a platform (a terminology) that allows creative actions, building an endless range of utterances and producing unlimited use of finite implies (Hays 2000). Eisenman displays this clinic around the plan of some succession of homes (Home I - X), by which he says, "Home is not just a final result of a continuous process, but also a record of the process itself” (Eisenman 1977). This notion about focusing on the process rather than the final product, and the action of conceiving of architectural design during the process implies a generative approach throughout architectural synthesis.

To understand Generative Design and its applications in the AEC industry,

- Developing a script

- Setting up a study
- Running the Study
- Analysis of results

2.4.4 Generative Design Systems Categories

The major classification of Generative Design systems involves two categories:-

1. Linguistic Generative Design System
2. Biological Generative Design System

Linguistic Generative Design System:- This kind of system is based on grammar, which means there is a syntax (a set of pre-defined sensitive rules) that outline the semantics. The computational execution of ancestral generative programs manifests itself chiefly in grammars of silhouette. Form Grammars specify and employ some of alteration rules within a first thing (a contour) to build innovative fresh style. As stated by Knight, shape grammars are more both descriptive and invisibly at a way the modification policies clarify different kinds of their established concepts (Knight and Stiny 2001).

Biological Generative Design System:- Biological generative design systems, alternatively, embrace a various generative plan, that takes nature and elaborate living organisms because being a precedent and utilizes its own fundamentals at the derivation and transformation of their architectural design (Hensel and Menges and Weinstock 2010). Vincent (2009) additionally emphasized the contour rather than the resultant shape itself. Natural evolution, describing that the manners that intricate organic approaches evolve, either the self-organize and increase, subscribe to architectural comprehension formation into the production of architectural, and also especially performative style and layout and style (Weinstock 2010). Hence a deeper engagement with character is closely pursued, that investigates the manners that the

principles of naturel gift helpful concepts such as functional integration, performative capability, and also substance resourcefulness (Ahlquist and Menges 2011).

The notion of computational layout and style is linked to your variety of design areas such as, siphoned layout, design layout, algorithmic style, and computational imagination. Generative design techniques use two different technical approaches - Natural analogy and Logical basis (Knight 2002 and Shea 2004).

Chomsky's (1975) classified the syntax through an approach of consequent string re-writing. In the early 1960 computer programming has advanced to a new level, with the invention of OOP (Object Oriented Programming). In OOP the 'objects' take the properties of their parent 'class'. This method of defining objects led the development of many animation software packages and parametric modeling software. At the end of 1960's a formalism was introduced, L-Systems. L-Systems can be seen as a modified version of the formalism grammar proposed by Chomsky (1975), because in L-Systems are created to simulate biological patterns. L-Systems formalism is built on the idea of parallel string rewriting, whereas Chomsky's grammar was linear or sequential.

2.4.5 Generative Design Systems

Let us specify Generative approaches as arrangements with the capacity of creating lots of outcomes. These approaches have been written of lipoic which spell sequential or parallel or arbitrary procedures. Let us also specify formalisms being a distinctive kind of the generative technique formalized with guidelines. The approaches and formalisms will screen a certain representation.

A generative technique could include a single or more bi-directional associations (connections). One-directional institutions crank out hierarchical buildings exactly wherever aspects are put inside of just positions. Usually Called a Parent-Child

connection. Within this construction, inheritance gets potential for possessions surfaced in several centuries in a category of the elements. From the subsequent six segments, I'll attempt and highlight big theories whom I come across highly relevant into this reach of the thesis. Each department will probably present you generative technique containing background info, experimental implementations inside of just style and layout and style, and also a brief outline. It's important not to the experiments I'm revealing for every single platform proved additionally constructed to tackle similar style and layout issues for people exhibited from the very first chapter (construction envelopes). Each experimentation is organized to utilize routine having hardly any test. The main reason why I picked this a specified circumstance to look was to strip down each system to the own bones and also focus on the conversation about its own arrangement.

Formalism is a technique to provide a clear boundary and structure to something. The various Formalisms for categorizing Generative Design are L-Systems, Cellular Automata, Fractals, Shape Grammars as shown in Figure 2-11.

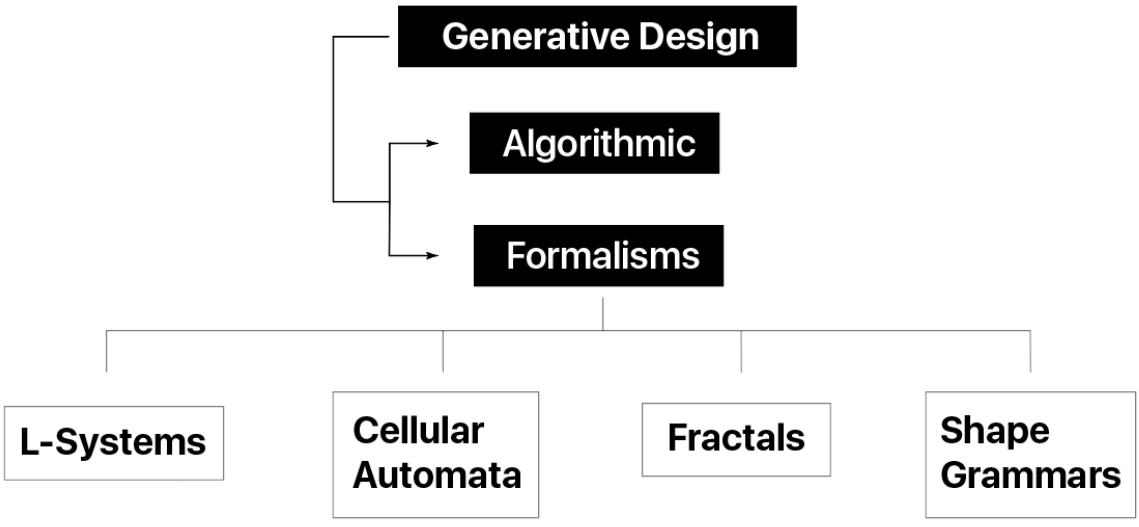


Figure 2-11. Types of Generative Design

2.4.5.1 Algorithmic generative design – logic based medium

2.4.5.1.1 Algorithmic generative systems

Algorithmic programs would be the fundamental parts in all the systems. The Algorithmic Generative System are not rigid to custom-made, they are flexible in terms of their relationships, structure. The idea of architectural design through a algorithm is not entirely new. The concept of developing a algorithm involves describing the series of the steps, which is a great mapping method for the designer to breakdown the idea/design. (El-Khaldi 2007).

2.4.5.1.2 Algorithm – concept

Figure 2-12 by Meibodi (2012) explained the structure of an algorithm for making N cups of tea. The algorithm on the left is given 0 input. The algorithm on the right is given an integer input.

Algorithm can be understood as a set of instructions that are defined by a human; these instructions are quantifiable in size. These instructions have a goal, which is received by following logic and computation. An algorithm in turn can be programmed / written to generate another algorithm, but the base architecture is defined by a human. The creation of a good algorithm is very challenging and is unique in the field of programming. Inspiration for creating algorithm can be obtained by observing natural phenomena. The idea of an algorithm is not new. The very first algorithm was written to find the greatest common divisors for a given number, known as Euclid's algorithm (Knuth 1981, p. 318). The concept of an algorithm goes even further back in time, to the 9th century mathematician named al-Khwarizmi, from Persia. who "laid out the basic methods for adding, multiplying, and diving numbers – even extracting square roots and calculating digits of **π** (Pi) (Papadimitriou et al. 2011).

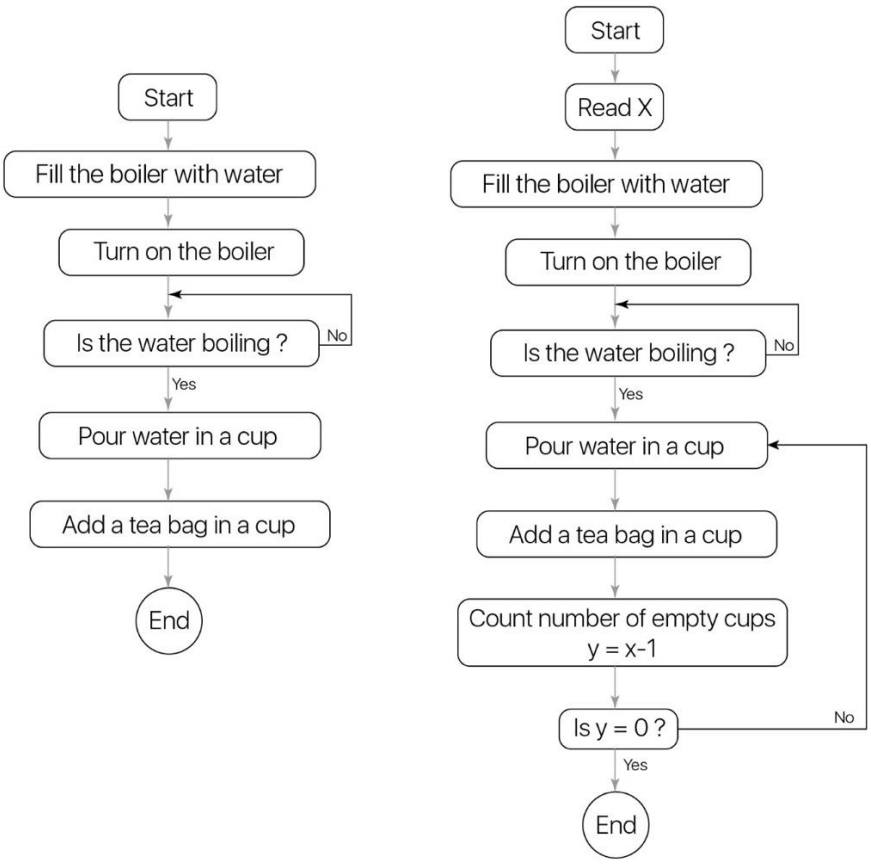


Figure 2-12. **IF** logic in the Generative Design Algorithm using tea boiling example (Adapted from El-Khaldi 2007)

The work done by Demaine and Lubiw (1992) in Figure 2-13 breaks down the steps(instructions) of countable size, for making a hyperbolic paraboloid from a sheet of square paper. These set of geometric instructions are the algorithm, and the paper itself is the computer.

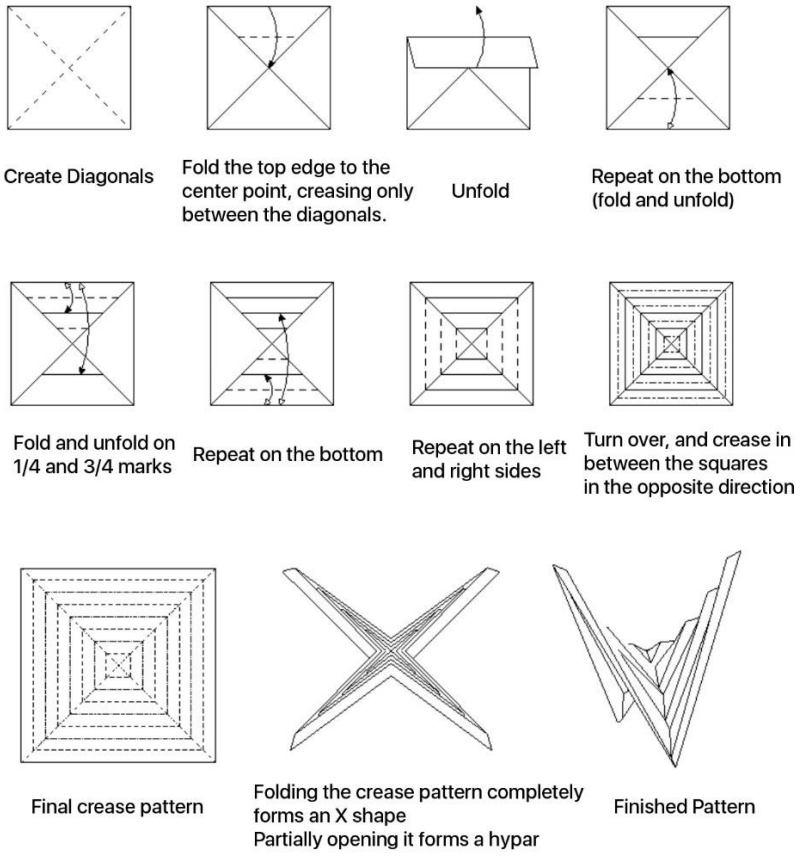


Figure 2-13. Illustration of Origami pattern using paper (Adapted from Demain 1992)

The instructions and methods he laid to solve these set of problems has aided next generation humas to be able to solve problems that “were precise, unambiguous, mechanical, efficient, correct-in short, they were algorithms” (Papadimitriou et al. 2011). But these preliminary methods do not have a concrete definition for an algorithm. The features and structure of an algorithm are defined by Rogers (1967).

1. Algorithm has a set of instructions with finite size.
2. Usually, a human is the computing agent, who can react to the instructions and carry out the computations.
3. There are facilities for making, storing, and retrieving steps in a computation.
4. Let P be a set of instructions as in *1 and L be a computing agent as in *2. Then L reacts to P in such a way that, for any given input, the computation is carried out in a discrete stepwise fashion, without use of continuous methods or analogue devices.

5. L reacts to P in such a way that a computation is carried forward deterministically, without resort to random methods or devices, e.g., a Dice. (Rogers 1967)

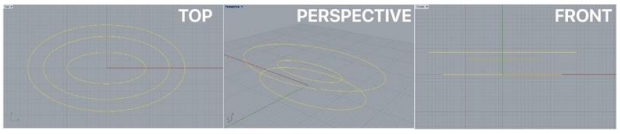
2.4.5.1.3 Architectural potential using algorithms

The main objective of this section is to showcase how algorithms work, and how they can be used in architectural design process, to make the design phase more technical in terms of data flow. Rather drafting and engineering steel envelope to hold an organic façade, defining a set of rules on how the façade shall be designed and this envelope in turn serves as an input to design the structural elements around it.

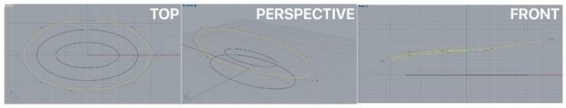
2.4.5.1.4 Experiment E.A - Grasshopper

In the Rhino's Grasshopper canvas, create three ellipses of different sizes, and then the elevations of each of these curves are altered as shown in Step 1.

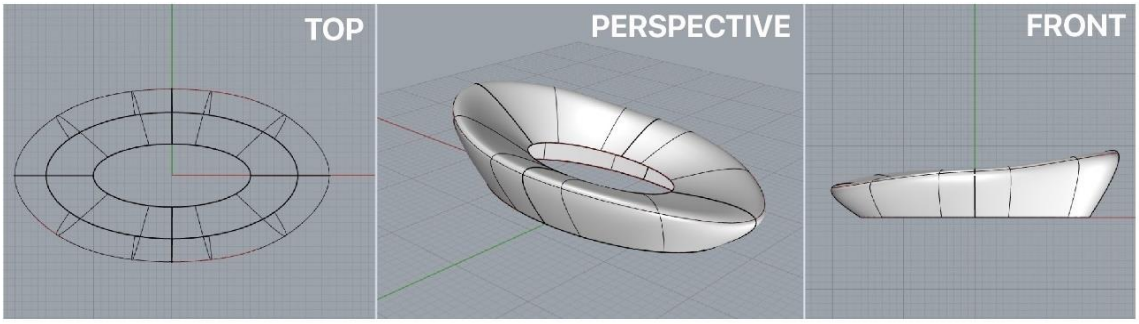
Then with the 'Control Points On' either one or two of the top-level curves are changed, as displayed in Step2. In the next set of instructions (Step 3), the Loft command in Grasshopper is used. While lofting through all the three curves, select the bottom most curve and then the topmost curve, in t the middle curve last. This creates a surface lofting through all the three curves.



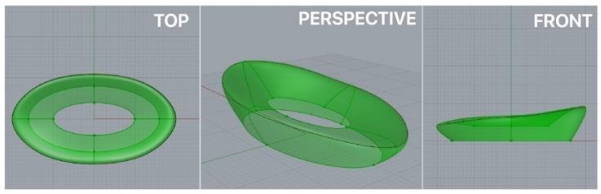
A



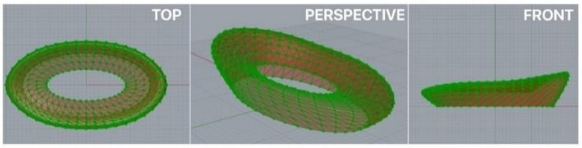
B



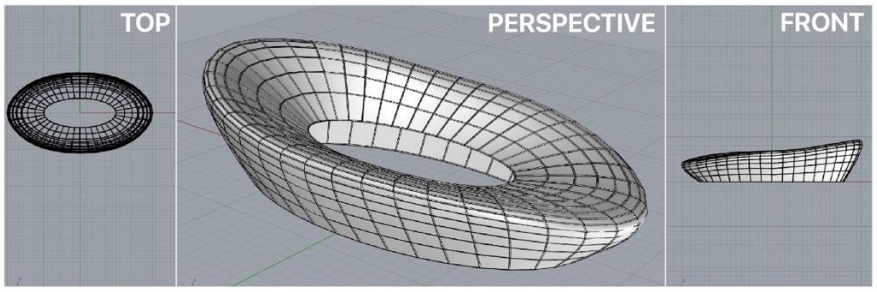
C



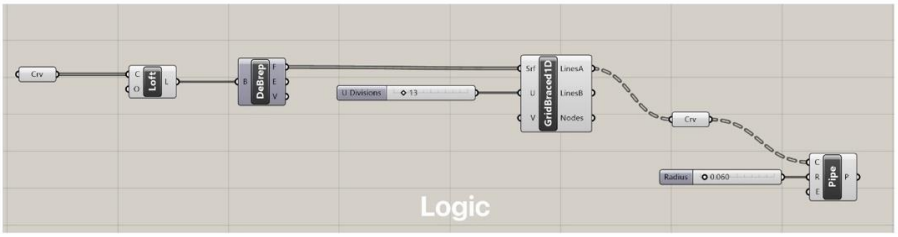
D



E



F



G

Figure 2-14. Form transformation using Grasshopper Logic (Experiment E.A)

Then input the Loft's output into *DeBrep*, this Deconstructs the *Brep* into it inherits. Which gives outputs of Faces, Edges and Vertices. The faces output is what is needed to plug in to the *BracedGrid* node. The logic is shown in Figure 2-14 G.

The output from *BracedGrid* node is converted into a Curve, but this not just a single curve. A closer look at the output of the Curve node, shows a family of multiple curves.

The number of curves that the surface is being split into can be increased through the number slider on the *BracedGrid* node.

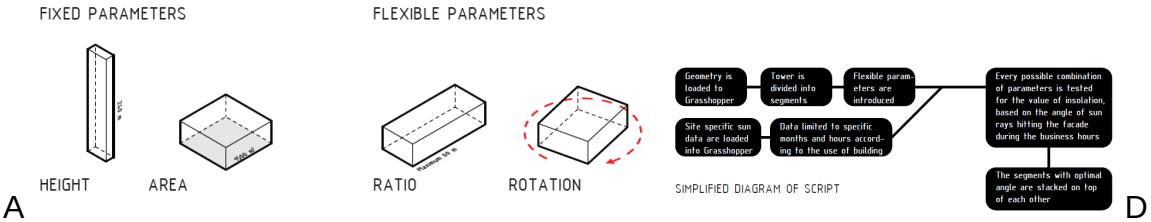
2.4.5.1.5 Experiment E.B – Insolation study

Sunlight may be preferred, in some situations, and not as much in other situations. . Within the Instance of high-tech workplace buildings, even the price for

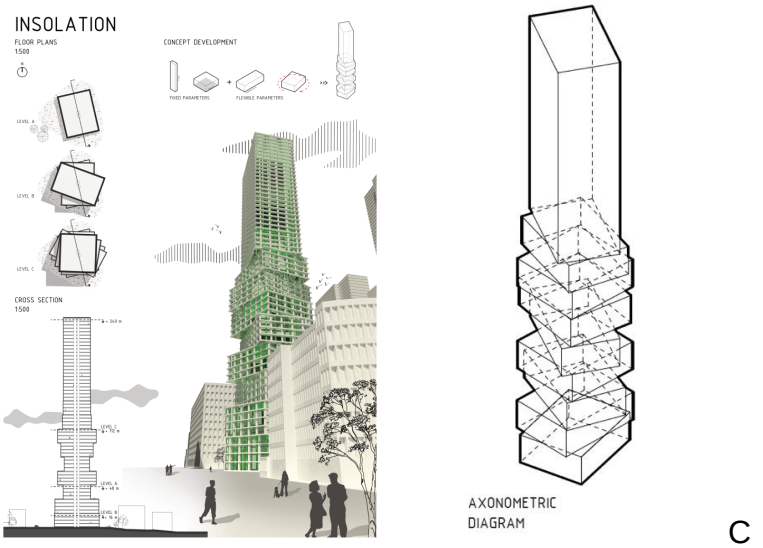
cooling tends to be much greater compared to the price of heating system. For that reason, one would like to lessen the vulnerability of these tiled surfaces onto the facade. But at an identical period, you would like to acquire too much daylight as feasible. Thus, in such a specific event we plan to discover the right proportions of this high-rise, so that your own sunlight beams perhaps maybe never hitting on the facade right back.

From the very first instance, there are selected parameters which can be mended. The elevation of this construction and floor space as designers/programmers desire to make the most of the useable area for the leasing. Nevertheless, the storyline lets turning of their floors plus a particular level of liberty from the ratio between the width and duration of this construction. For that reason, just two adjustable parameters must be analyzed.

The top layer of the tower has been subdivided into sections. Every segment includes four floors, which might possibly rotate separately. Those sections were analyzed for occlusion from the nearby properties, to ensure that adjacent properties are not obstructing. Panels that are struck by sunlight are quantified along with the amounts have been all inserted with each other. Evolutionary solver Galapagos (Grasshopper plugin) assesses the mixes and then adapt its own reckoning by assessing previous iterations.



A D



B C

Figure 2-15. A) Problem's Parameters, B) Plan, Section, Isometric View, C) Axonometric Diagram, D) Simplified Logic (Ondrej Slunecko 2020)

For analyzing we place our own sun to provide a variety of worth to its warmer section of this season.

Final results:- The results show that there are a total of 360 sections may be redeemed by and somewhere around 50 distinct ratios we are able to examine. Combined you will find 1000 chances to examine. When we needed to complete it manually, then we would spend few times. The script was made in grasshopper, which required approximately an hour or so to examine the chances for every single portion.

2.4.5.1.6 Summary of algorithmic generative design

Algorithmic devices resemble the heart of most generative systems. They pack a succession of responsibilities in a variety of constructions, representations, or even rules. Using calculations at designing procedures may possibly be confused because of limit for they are able to simply manage namely defined elements, or even since they violate procedure into different activities, etc., In reality, "algorithmic thinking" helps designers set their own aims and describe their own aims and plans. Mapping a style and style course of action in a whole lot of ways may be difficult or guide, nonetheless it

may help externalize logic, and thereby assemble a strategy capable of building an assortment of remedies rather than only just one or even perhaps a couple of remedies.

2.4.5.2 Parametric generative design

Parametric systems are treated as a branch of Algorithmic System's. There are two contextual usages in the field of Architecture by means of Parametric Generative systems:

(1) Geometric Design
(2) Animation Design

The first Geometric Design is carried out by modeling software's like Solid works, Revit, CATIA, Rhino's – Grasshopper. All these software's create data in the order of hierarchy. The data is nothing but the model in the modeler's interface. Relationships and dependencies are built across elements all through the software's cycle. The later, Animation design is carried out by Animation modelers such as Cinema-4D, Maya, 3Ds Max, there are connections and relationships in these modelers too, but the difference is how they are related with each other. Mainly they are linked through Dynamics and inverse kinematics solvers.

Any modeler or system that has the capacity to relate the elements (Data) that it takes in as inputs, modifies, and outputs by having dependencies and connections with each other, can be considered a Parametric Modeler.

The first and foremost to introduce parametric concept into computer science are W. Orchard, G.B. Manne and S.I. Gass (El-Khaldi 2007). Figure 2-16 shows the shapes Lynn (1990) came up with for residential spaces. It was during 1990s that a new set of animation software was developed, this caught the architect's eye almost immediately and they started using this software to something more than just drafting/documenting their work, but for exploring form and design. These set of parametric software

packages allowed the user to create, modify and interact with relations and the constraints of the elements. One of the famous series of projects was done by Lynn (1990), where he used this animation software to design shape morphing.

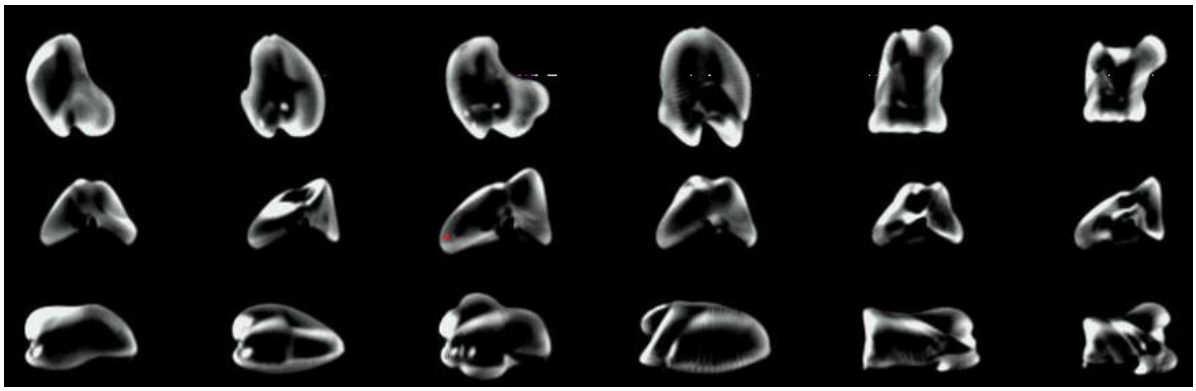
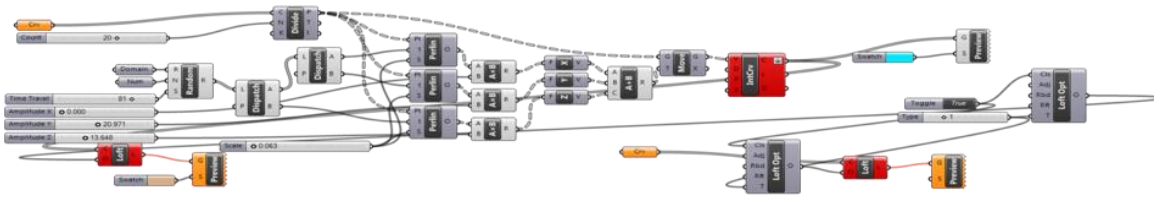


Figure 2-16. Parametric Shape finding using Logic (Lynn 1990)

2.4.5.2.1 Experiment E.C – Ripple surface

To understand the Parametric Generative Design is to look at an experiment. In Figure 2-17 A, the logic is shown how a plain 2D surface can be sent through a series of instructions to achieve along the lines of ripple effect.

The Number/Integer sliders once toggled, the 3D geometry changes instantly. Because the links between the nodes is logical and unbroken. But one key thing to understand is that data can only from Left/Top to Right/Bottom, but not the other way



A

B

Figure 2-17. A) Grasshopper Logic for Ripple Effect B) Rendered output (Experiment E.C)

round in the VSL. Though there are plugins and script that lets the designer achieve ‘loops’ for dynamic designs. The concept of ‘while’ and ‘if’ loops are not native functions.

The output is shown in Figure 2-17 B, is the rendered output from Rhino and Grasshopper, but designers now have access to these powerful computational design tools, that let you achieve which was treated as challenging before.

2.4.5.3 L-Systems in generative design

It is crucial to be aware these formalisms had been made to mimic very special occurrences rather than supply a functional platform such as Algorithmic or even Parametric devices. As an example, L-systems have been used to mimic botanic expansion, Cellular automata had been made to mimic reproduction," Fractals are made to mimic self-similarity in character, and contour grammars had been intended to mimic human capability to determine or calculate visually (El-Khaldi 2007).

Parametric devices really are a particular example of systems, (people together with institutions). L-Systems tend to be somewhat more special algorithmic programs. All these are rule-based procedures, and it can be described as formalisms. Rules are also often exhibited like a left-hand, arrow, and an ideal aspect.

2.4.5.4 Cellular Automata

Cellular-automata methods provide a richer atmosphere because of its own symbols since they are not restricted by a kind of significance. A logo in CA (mobile may refer to “coloration" having its own versions (black, whitened, etc.), or dimension (with different amounts), site (roughly quite a few axes), etc.) maybe different items. L-systems would be the earliest of those 4 formalisms since it had been that at the very least elastic of systems. Its logos are all confined by a form of significance, alphabets (El-Khaldi 2007).

2.4.5.5 Fractal Design

L-systems and Mobile Automata keep the magnitude of these tiniest components. Rule’s re-place alphabets or cells without even dividing them to smaller sized kinds. The idea of this “smallest component" just isn't related to Fractal devices because they rely on mathematical types of recursions. Fractal calculations may recursively fracture parts initially, then replace them with fresh kinds (El-Khaldi 2007).

2.4.5.6 Shape Grammar

The only type of formalism among all the formalisms - L-systems, Mobile Automata, and Fractals the location of any component is identified as unique, not having a relationship with the other components in terms of component’s location. All these processes were assembled to catch visual improvement procedures in Design. To imitate this system for a complex real-world design can get extremely complex and would be possible with advanced algorithms.

Figure 2-18 explains Shape Grammar in practice, in image A every square has two attributes, an anchor (the dot) and an arrowhead (represents the vector). The position of these two attributes defines the rotation of the square (Counterclockwise in

this case). Figure 2-18 B shows a new square formed by changing the position of the attributes.

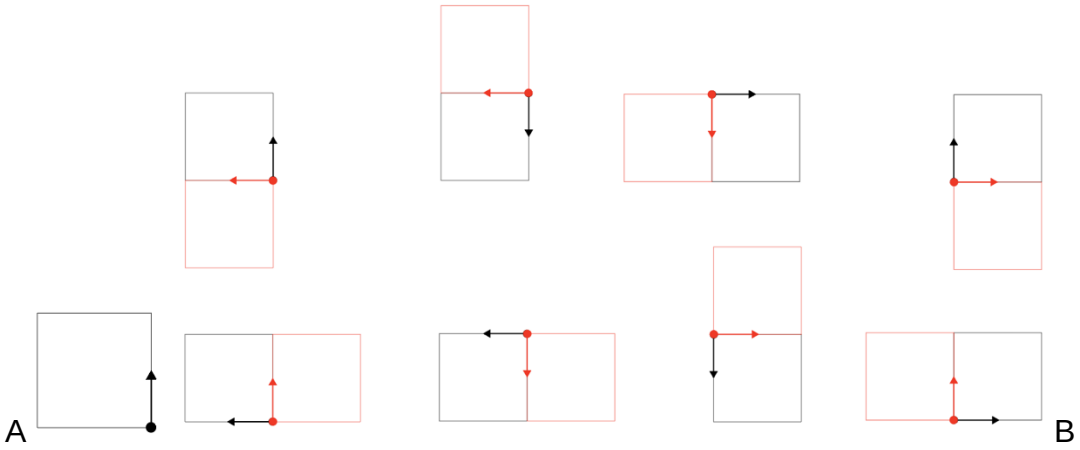


Figure 2-18. A) Anchor and Vector B) Rotation variations

CHAPTER 3
COMPUTATIONAL MODELING

3.1 Introduction

Designing through algorithms will permit the user to solve i) Problem set that is very well defined ii) Complex problem set that has weak definitions, and computational modeling through algorithms solves the well-defined problems (i) to detail, but the latter is where this method truly represents its principles – by exploring solution variants. Any design problem has no clear, correct solution; instead, it has a set of suitable solutions (Simon 1969). Hence, it is in the hands of the designer to evaluate the existing solution, and redefining the problem, redesigning new and better solutions. However, speculating a broad set of solutions is very time-consuming to verify if the new design is parallel with the solution standards.

The chief aim of mainstream CAD is the rendering of the final design. Designers work towards the finalized design, with no room for exploring variants, which is why they rely on single-state style and design style. On the flip side, parametric modeling for a style enhances the designer's procedure. It permits him/her to diverge the plan space to check out various variations of the exact multifunctional product. Therefore, the fundamental principle showcased with parametric modeling is that the relationships between the elements in a design space make it possible to revisit the past timeline's design variants and work on them to make them better based on the practical results. With provincial relationships, the programmer can research various design selections through the years, reevaluate previous style and style alternatives, and enhance the plan artifact through the planned method (Aish and Woodbury 2005).

The algorithmic design approach is a dynamic design procedure that has multiple input values. To design spaces through algorithms that let data flow in a direction for

achieving the final design demands a very high fidelity to represent physical structure/space in terms of logic.

To best understand the Computational modeling concept, the different variations of British Petrol headquarters at Sunbur designed by the P.art team at ADK (Adams Kara Taylor) can be studied. P.art is a research team at Adams Kara Taylor, which consists of designers and architects researching creative problems in architectural design. Figure 3-1 shows the roof design variations for the British Petrol headquarters in Sunbury. The options in Figure 3-1 result from geometric modification in the data input and the algorithm's connections. The roof is a combination of triangles at two levels; the variations are achieved by modifying both the triangles at the lower and upper levels. The complete geometric form is designed using toroid geometry. A toroid is an orientable polyhedron without intersecting in its geometry; this means that the geometric form has either one or more voids. These voids are used as the opening for light sources and changing the levels between the two levels of the triangles changes the overall opening in the roof. Furthermore, the length of the radius of the toroids defines the curvature of the roof surface. To achieve a model that is as complex as this concept and to possess the freedom to see the resultant design instantly is not possible with traditional design programs, the parameters should have consistent relations that the designer can manipulate to achieve the desired design output.

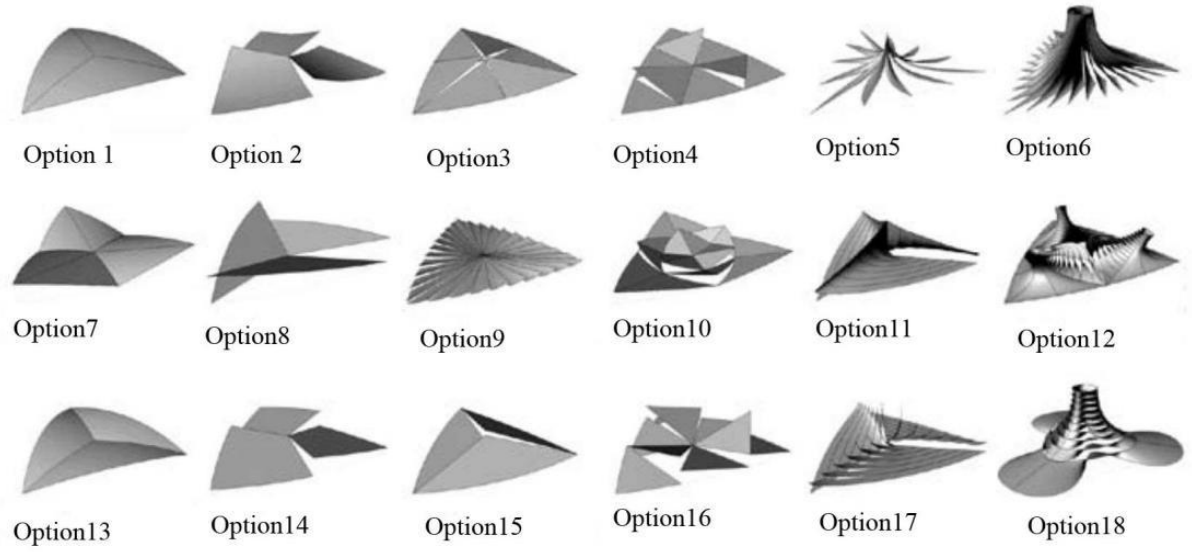


Figure 3-1. British Petrol Headquarters in Sunbury by Adams Kara Taylor (Sakamoto and Ferre 2008)

3.2 Literature Review And Experiments

3.2.1 Design Technologies

The concept of design itself is complex enough, but to articulate these designs by the designers, CAD (Computer-aided Design) tools have been developed. CAD tools help the designer to work on the designs in the software's canvas for achieving design variations. To truly comprehend and review the CAD technology is beyond possible for the scope of this research. Based on personal experience, the following three trajectories were drawn for technology in design. The reason for taking these three trajectories method, is to understand the different paths for the development of the tools, which eventually impacts the designer who will utilize these tools to explore design.

- Digital representation of this construction procedure to control the flow of info reasonably, improve efficient communication, and ease a significant change within the total construction procedure.

- Usage of CAD tools to explore the form of a space and how this form-finding process affects architectural design.
- It is the usage of CAE (Computer-Aided Engineering) that impacts how we analyze, generate, and create tools that impact design sequences to the exploration of form and architectural outcome throughout formation procedure.

The first trajectory is to represent any building/construction information in digital form. This path has led to several keywords – BIM (Building Informational Modeling), Virtual Building. The main objective of this path is that the data flows in a controlled way, that is rational. The key feature of this trajectory is that it allows design professionals and construction professionals to store and maintain huge amounts of data in the model. It is not just storing this data/information, but also to manage/edit, process and share it with other personnel. The first development in this path is ArchiCAD by Graphisoft which was first deployed in 1982. Later the major deployment was Revit in 1997 by Revit Technology Corporation. It took some time for academic curriculum to incorporate Revit, but ArchiCAD was introduced to students in design class of architectural school's beginning in 1980.

The second trajectory is how CAD is being integrated into the design process itself, with a desire to model/design complex spaces, topology, and geometry. In the early introduction of 3D modelling there were many limitations and after researching more on how to create a no paper studio, researchers and developers have built software packages like Softimage, Wavefront that have opened doors for many architectural designers. Softimage was not intended for architectural modeling, it was intended for film industry usage, but eventually caught the eyes of the architects in very less time.

The third trajectory is the generative approach, utilizing computational power and process to analyze and optimize solutions/variants. In the past decade with many

updates, the distinction between the second and third trajectories has softened. The idea behind the third trajectory is to create a set of rules (ruleset). Constraints and forces are modelled within this ruleset to create and find form. Figure 3-2 shows a form-finding example by Lewis(2010) using ROBOT to find the optimum shape. The lengths of the individual linear components are inputted through Excel, and with the change of the lengths in Excel, the design can now output a massive range of possible shapes.

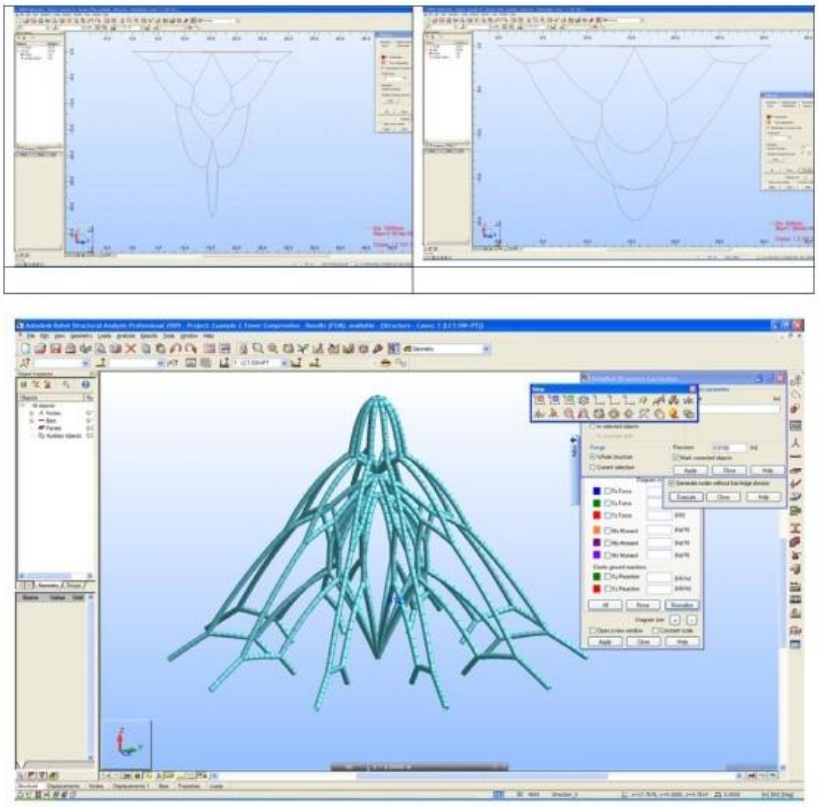


Figure 3-2. A hanging chain example, form-finding (Lewis 2010)

It is critical to understand these three paths because the fundamentals underlying these paths have shaped how CAD and computational technologies are being used, furthermore the areas to focus on for future developments. This research has its core at the second path and explores the foundations of the third path. Nevertheless, the first

path is critical as it shapes how the designer gets to explore the design in software's canvas.

3.2.1.1 Algorithms and Parametricism for CAD

The second path has evolved in a way that the usage of computation in design has totally changed. Prior to CAD the designer used software as a tool to present the idea/design that he/she already had in mind, in other words CAD (Figure 3-3) was mostly seen as a digital drafting platform, that represents information digitally. But with the latest advancements in the software and its architecture, the designer can now explore the design itself before finalizing it.

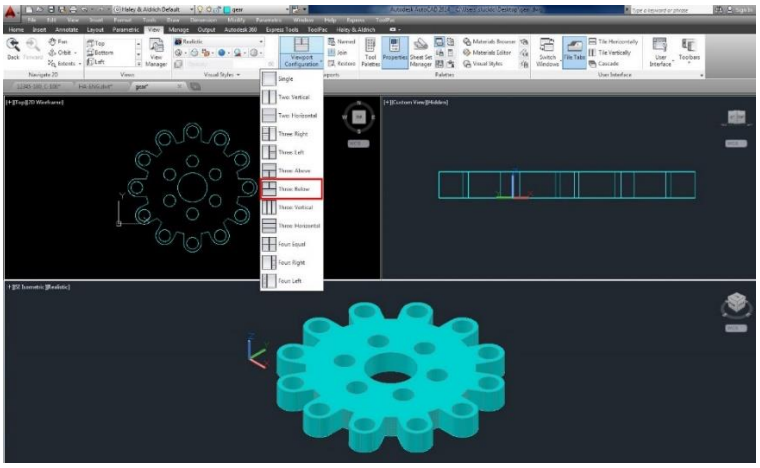


Figure 3-3. AutoCAD 3D modelling canvas

When a design software is equipped with parametric system, it allows the user/designer to not just design strategically but also intuitively (Aish et al. 2003). It is the way of linking the values such as geometric properties to the geometry itself. Just like in programming, the designer must think through the logic before building the relations. Then you debug for any errors, which is ensuring that the design works with all the parameter's limits. With the help of scripting tools such as Grasshopper in Rhino, Visual Basic and Python in Rhino, Dynamo and Python nodes for Revit, Java for

processing and GC Scripts for Generative components, Maya Embedded language for Maya, 3D Max Script for 3DS MAX, the designer can create their own interdependent relations instead of working with the design's software packages out of the box capabilities.

For designers and users who are not proficient or introduced to IDE or algorithmic environment, the best approach is to take advantage of the VSL (Visual Scripting Language) environment. Rhino for instance, is used in many architectural schools in the US and an inbuilt VSL plugin for Rhino is Grasshopper (Figure 3-4), and for industry standard software – Revit, its VSL is Dynamo, both VSLs are very user friendly. Any user with no prior experience in programming can use them and take full advantage of the technology.

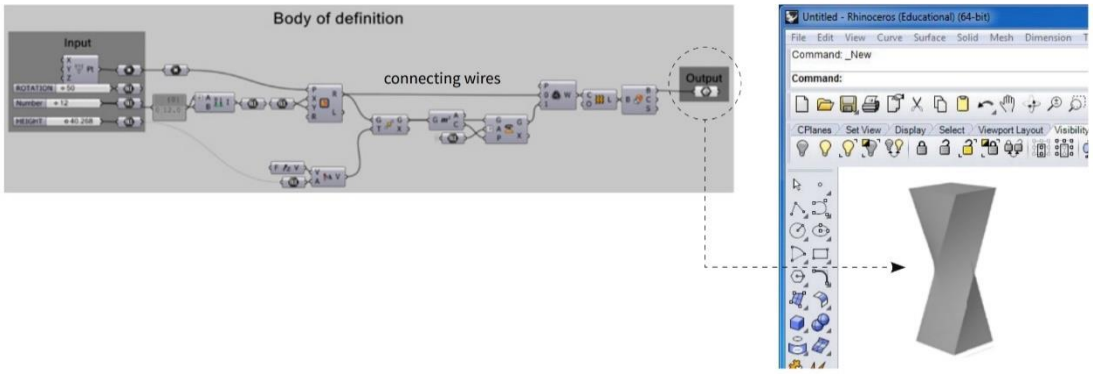


Figure 3-4. Rhino(right) and Grasshopper(left) Interface

Grasshopper is a widely used VSL in the architectural design space, though Dynamo has constantly been gaining more users due to much wider parent platform, Revit. Grasshopper's elements are known as 'Components' and the algorithm are built using these components, which will serve as input's, output's, mostly both. The idea of components is from JavaScript, a widely popular language and is best known for component-based architecture, since Grasshopper is written in JavaScript the concept

of Components in JS (JavaScript) gets carried into Grasshopper. These components are blocks of code, that performs the specified task, for example a line component creates a line, based on two inputs – starting point XYZ co-ordinates and the End point XYZ co-ordinates, these co-ordinates are also components. Once these input co-ordinate components are connected to the line component, a line is shown in the Grasshopper interface, and it can be ‘baked’ into Rhino’s interface (Figure 3-5).

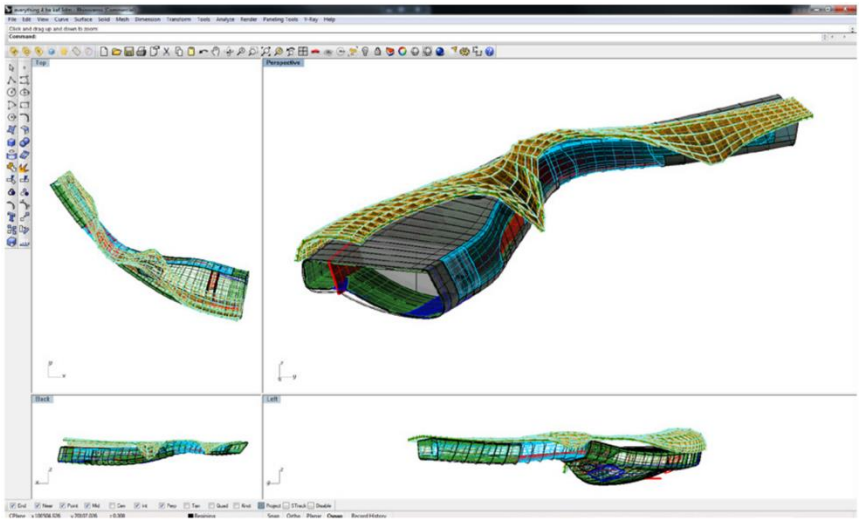


Figure 3-5. Rhino’s design canvas

Figure 3-5 is created using Rhinoceros along with Grasshopper. This project is based on exploring form. In Revit there are preconfigured objects with inbuilt parameters, in Rhino the end user defines the parametric connections between the elements. The logic that is used to model the form is developed in Grasshopper, while the geometry of the form is shown in the Rhinoceros window’s (Top, Perspective, Back, Left).

3.2.2 Experiment E1.1 - Dynamic Co-Circular Grid

Computational modeling has progressed so far that we are at a phase where it is more ideal to go into a 3D modeler interface and place an element (such as a wall), rather the designer defining the properties and conditions of the element.

In this experiment, explore the extents of dynamic modelling in Dynamo is explored. A virtual space can be created to carry out dynamic experiments, this can be understood very well through Arches. According to physics-based form finding, the most ideal position can be understood if this arch is imagined as a chain/thread.

If a thread or chain is draped down from two supports with a certain span, the natural curve of this thread is the optimum curve when built upside down. This whole dynamic can be built in the virtual environment, in the familiar Dynamo interface.

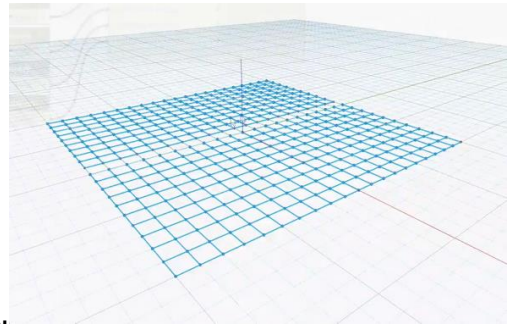
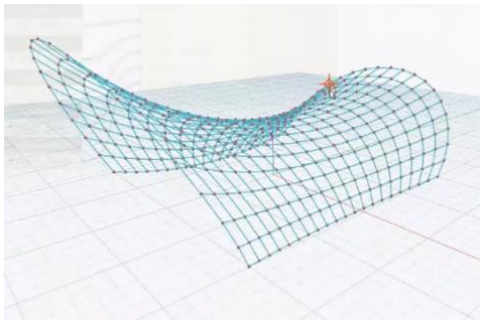
A.   B

Figure 3-6. A) Dynamo grid, B) Grid created using DynaShape

With the help of open-source packages developed by engineers and professionals from across the globe, some of mind-bending features of Dynamo can be accessed. In this experiment the ‘DynaShape’ and ‘Mesh ToolKit’ packages are used, and they were downloaded from the packet manager in Dynamo.

Figure 3-7 shows the relations created on to the DynaShape nodes, Stage 1 is to create the Grid, with an input of X, Y units. Stage 2 is defining the DynaShape goals that are to be achieved. In the final stage, the Solver.Execute node takes the combined

list input (Goals of the problem), along with the Boolean values that shape the execution. (Project files found in Experiment E1.1).

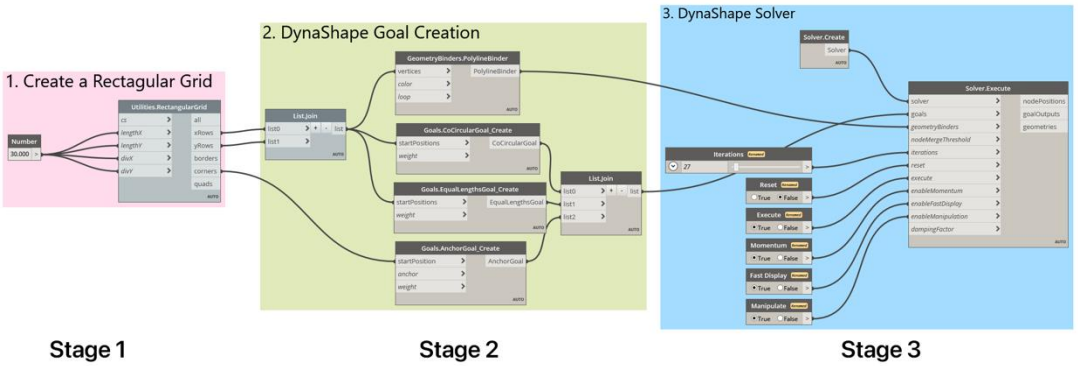


Figure 3-7. Nodes for DynaShape Solver and their connections (Experiment E1.1)

3.2.3 Experiment E1.2 - Shell star Pavilion by MATSYS

The famous Shell Star Pavilion design by Matsys design studio could also be achieved through computational modelling toolsets in Dynamo and Revit interface by creating a logic for the geometric elements, in order to achieve this organic design.

Figure 3-8. Shellstar by Matsys and Riyad Joucka (Photo by Dennis Lo)

Figure 3-9 shows the logic built in Dynamo to achieve this design. Stage 1 is to create the Rectangular Grid, using the Mesh ToolKit3.0 plugin.

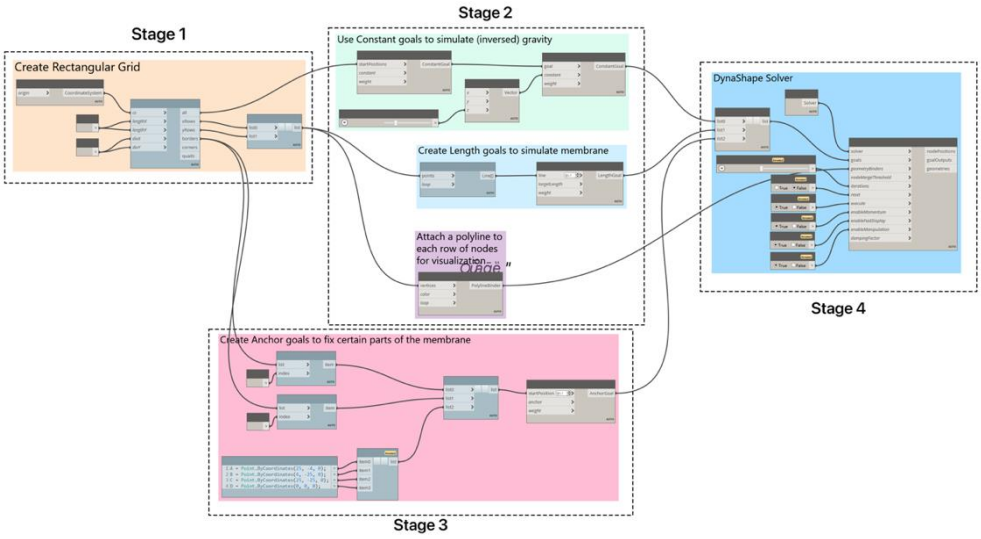


Figure 3-9. Logic for Shellstar Pavilion (Experiment E1.2)

Stage 2 simulates gravity, the actual model is not static once the nodes are connected and get to work. The model is dynamic, a draped down fabric material. This is a beautiful visualization achieved through the DynaShape plugin, which has a C# (C-Sharp) loop, that runs over and over to simulate the gravity dynamic.

The way to look at what Stage 3 is solving is how the nodes create the Anchor points, from these Anchor locations the grid from Group1 Nodes will be hanging inversely. Just like Lewis (2010) in his experiment found the optimal shape of the chains by inversing the whole model and letting the chains find the optimum position in ROBOT. Just like Experiment E1.2, the anchor goals are defined in Stage 2. Also, by using the DynaShape's length and inverted gravity goal nodes, the input for Solver.Execution node is generated. Once all the lists are combined as shown in Figure 3-9, Stage 2 output becomes the input for Solver.Execution, which creates the final dynamo output shown in Figure 3-10, it is dynamic and interactive, thanks to the Java loop built in the Solver.Execute node of DynaShape. Figure 3-10 shows geometry that

results at one instance of time at an anchor point once the nodes are connected as shown in the Experiment E1.2 logic.

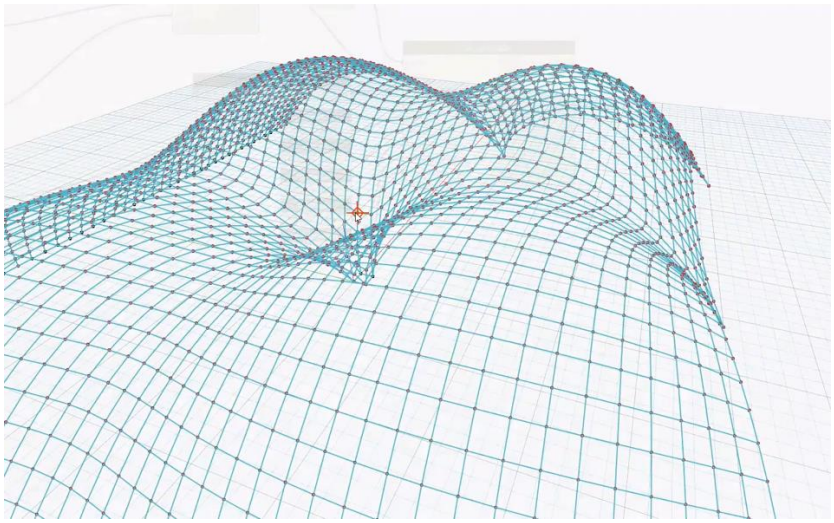


Figure 3-10. Fluid Dynamo output – Shellstar pavilion

3.2.4 Experiment E1.3 – Tensile Tent

Although the Dynamo logic for the Tensile Tent Experiment E1.3 looks more complicated, it is the simplest of the three experiments E1.1, E1.2 and E1.3. Stage 1 is similar to the first two experiments, the rectangular grid is generated through the 'Mesh ToolKit3.0' plugin, this output is first transferred through a list node. The nodes in Stage 2 in Figure 3-11 show the four anchor points, for the tensile tent. These are the Anchor goals which is output as a list.

The integer sliders are the inputs of point coordinates for the anchor points, which change dynamically where the tensile tent's anchor points are located. The Anchor goals and the Length goal nodes output are combined into a list, by the list combination node.

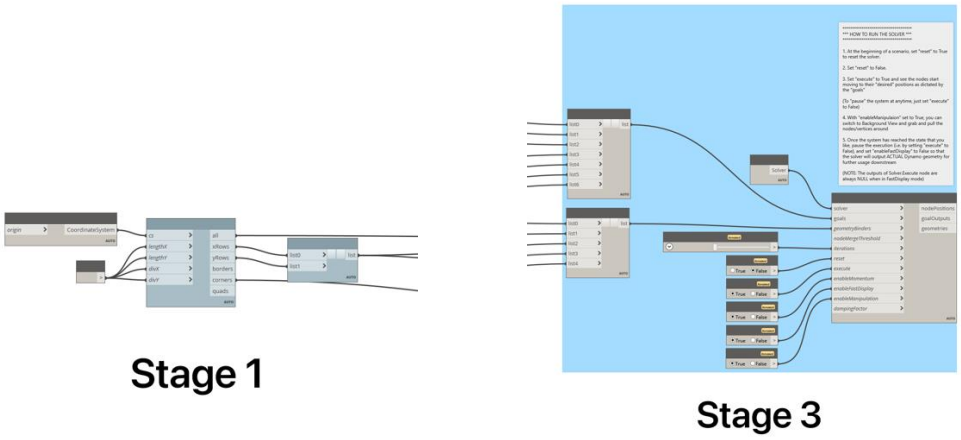


Figure 3-11. Creating Anchor point nodes using DynaShape Solver (Experiment E1.3)

Stage 3 is the DynaShape solver, the list that contain the information on the Anchor goals, Length Goals, Constant Goals, Line bender values which are all fed as inputs into the respective input of the DynaShape solve nodes. Once the connections are made, there are a set of steps to get the DynaShape solver to display what is attempted to be achieved, first the “reset” Boolean node is set to False. Immediately thereafter, the Execute is set to True. While the “Enable Manipulation” input is set to true, the user can switch to Background view and grab and pull the nodes/vertices around the space and see how the model interacts in reality.

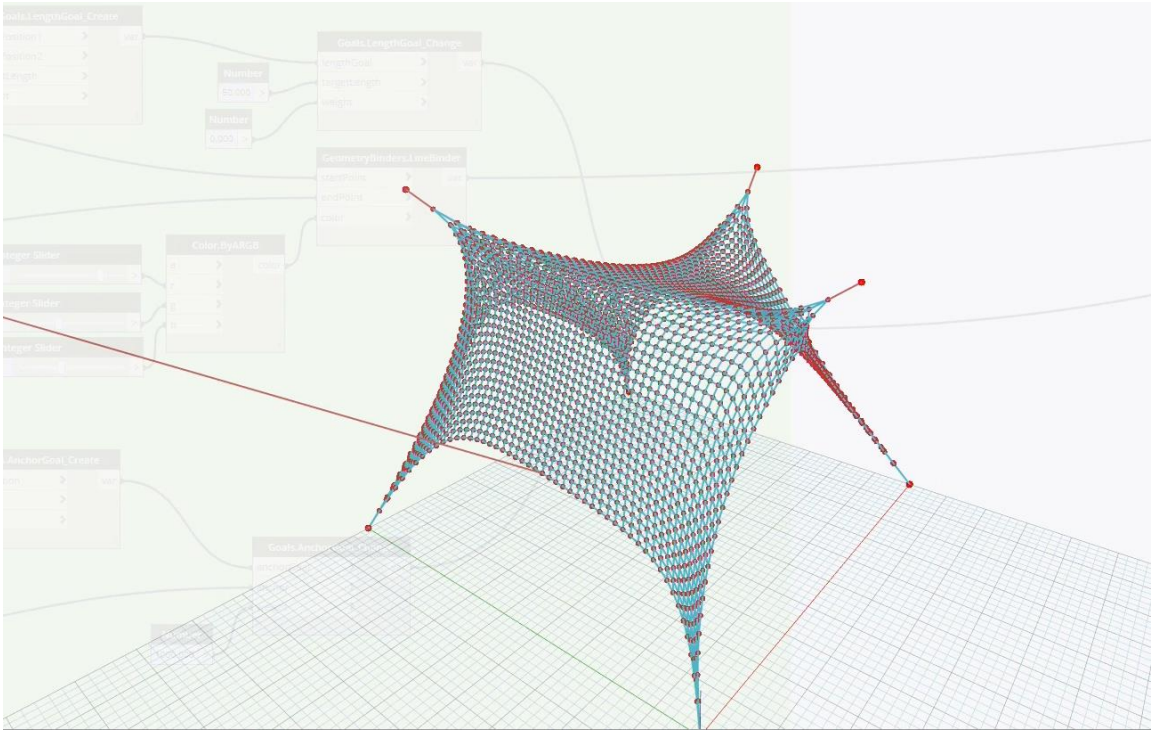


Figure 3-12. Final Dynamo model output (Experiment E1.3)

3.2.5 Experiment E2 - Randomize Panels on a Selected Curtain Panel

Selecting every single Panel on a Curtain Panel Wall and Changing each panel’s material/finish is extremely time dependent and it is not very feasible for a designer to

have to manually go back and forth and make changes. This can be achieved by using Dynamo and computing a random list of curtain wall Panels (which needs to be iterated through to find the optimal curtain wall).

A B

Figure 3-13. A) Revit model with default curtain panels script B) Final output from running the script

Packages needed to achieve this Dynamo Script – 'LunchBox', Dynamo Player.

In Figure 3-14, the logic is broken down into groups on how to achieve the result. First portion of the problem is to select all the 'Panels' of a particular 'Curtain Wall' configuration as follows:

- In Stage 1 all the elements that have 'Family Symbol' are selected and since the 'Panels' have as a Family Symbol a unique name, in this case, 5 panels of different colors were created that have to be randomized on the selected Curtain Panel.
- In Stage 2.1, all the elements from Stage 1 are input, and then only the elements that start with the string (It is an object in Programming that is used to represent a combination of characters, in other words text) 'Random' are selected, and then the "FilterByBoolMask" will split all those that have the word 'Random' from the elements that do not, which in this case are a total of 6. (These are the total number of panel types created with unique finishes for the selected curtain wall configuration)
- In Stage 2.2, the Curtain Panel configuration that needs to include these random Panels is input by selecting and right-clicking on the node (see Figure 3-14), which lets the user simply select the wall when the script is run in Dynamo Player.

- Stage 3 – is where the ‘LunchBox’ node is executed. The list of the filtered curtain panels (6) is inputted to the Lunchbox’s Randomize node.
- In the final stage 4, setting the type of elements for the chosen family instance. The Family instance is inputted through the Stage 2.2 output, and the Family Type is inputted from the Stage 3 output.

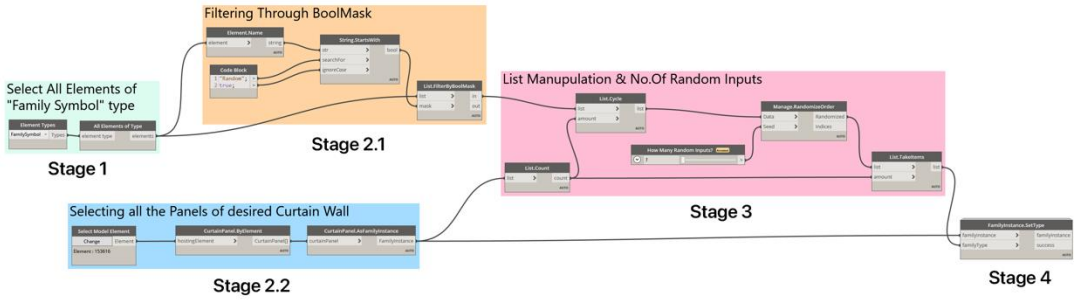


Figure 3-14. Logic for Curtain Panels Experiment.

Figure 3-15 shows multiple Curtain Panel Walls that are selected and applied with randomize wall panel patterns using only a couple of clicks, with the help of Dynamo Player and selecting which Curtain Panel Wall to use and what pattern to apply to.

Figure 3-15. Dynamo Player for Curtain Panels Experiment

3.2.6 Experiment E3 - Inserting text Above and Below Dimensions using Python

The steps to create a simple Python node in the Dynamo Environment is to first right click on the menu (See Figure 3-16 A) to open search and search for “Python Script”, this will create a small node initially with a single input – IN[0]. When you clik + it adds more inputs. These inputs serve as the data inputs to this node.

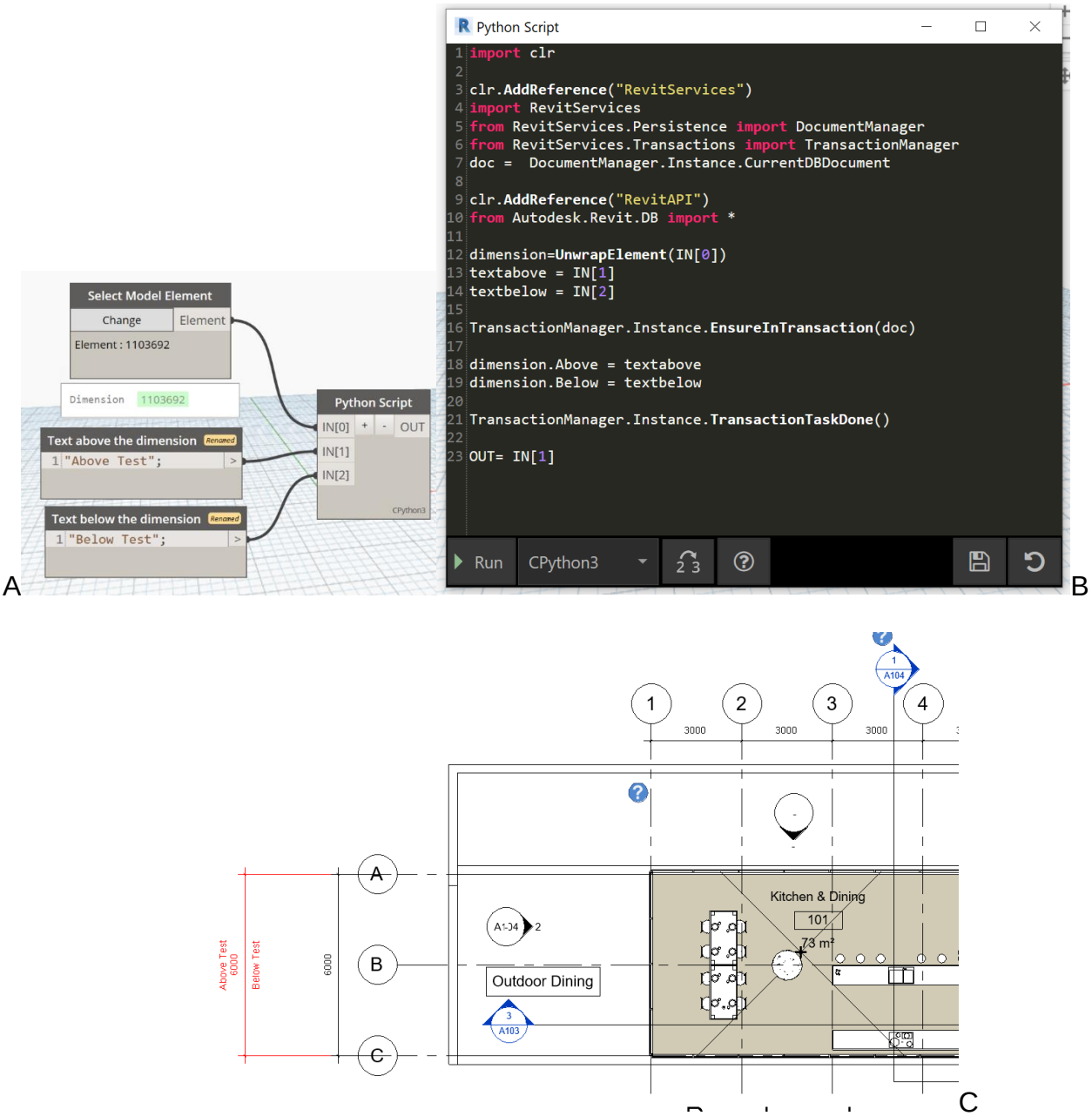


Figure 3-16. Python Script to set Dimension’s above and below text

The user can define the type of inputs that are accepted in the node, by specifying these types in the Python IDE (Integrated Development Environment). Each line of the Python script (Figure 3-16 B) is explained below, as to what it infers:

- Import CLR - Common Language Runtime, this is the bear bone of the node.
- Add Revit API to it. -To communicate and work with Revit and its elements
- Add RevitServices to it. -This enables something called as transactions.
- Import DocumentManager – This points to work on the current Revit file that is open. It is a good practice to work on a single project file in Revit interface to ensure that the nodes point to the existing document.
- UnwrapElement-This convers the element to internal Revit type element.
- Inputs – As you can see on Figure 3-16 A, first input takes the element, and second input takes the name that has to be set above the dimension, and the third input takes the name that has to be set below the selected dimension.
- TransactionManager.Instance.EnsureIntransaction(doc) –Start the changes
- Dimension.Above = IN[1] – Binding the second input as the text that goes above this dimension.
- Dimension.Below = IN[2] – Binding the third input as the text that goes below this dimension.
- TransactionManager.Instance.TransactionTaskDone–Make sure that the transaction is complete.
- OUT – Outputs the result

The results of the changed dimension is shown in Figure 3-16 C

3.3 CASE STUDY – ZHA B.I.M Workflow

In the Autodesk University convention at Las Vegas in 2018(Zaha Hadid Architects) shared how they take advantage of technology in such a fascinating way that creates the illusion of architectural magic. I have been able to study these cross-platform collaborations by ZHA through ZHA's BIM manager 'Eckart Schwerdtfeger' presentations and conferences in the AutodeskUniversity.com

The best way to incorporate technology to solve problems is that we should have a problem solving with technology, not fit the technology available to solve a non-existent problem. ZHA has a clean framework on how to achieve their design goals. ZHA first starts with setting up the goals, these goals must first be defined. Goals for any designer or firm or GC could be what they are trying to achieve with BIM and computational modeling. They need to determine whether a parametric model needed to be shown to a client, whether schedules are needed, whether quantity take offs are needed, whether clash detection is needed, d whether VR/AR capabilities are needed, whether Time and Cost are needed as the 6^{th} and 7^{th} dimensions, whether a system is needed to maintain and operate the facility, or some of these or all of these are needed. These are some of the important questions any BIM engineer or VDC manager should ask to successfully reach their goals.

ZHA Design Goals- The ZHA works are unique but in terms of their technological need perspective, for their Preliminary design stages 'Rhino' has been the best solution so far, due to the way it is structured and moreover reinforced by a powerful tool – Grasshopper'. So, they need Rhino and Grasshopper to design the space. Now this model needs to be carried over the cycle, to actively make changes back and forth while having parametric features among the design elements. Though there are technologies like ArchiCAD that meets the parametric modelling and documentation, but the most extensive used software is Revit, due to the way it is built and can be integrated smoothly with interoperable needs.

Figure 3-17 shows how data is flowing from Rhino to Revit, while it is being push-pulled and modified with scripts and plugins to meet the firm's need. Initially all the geometry in Rhino is referred to as 'Eckart Schwerdtfeger' (REFERNCE) dead

geometry because they are geometry without any properties or in other words meta data.

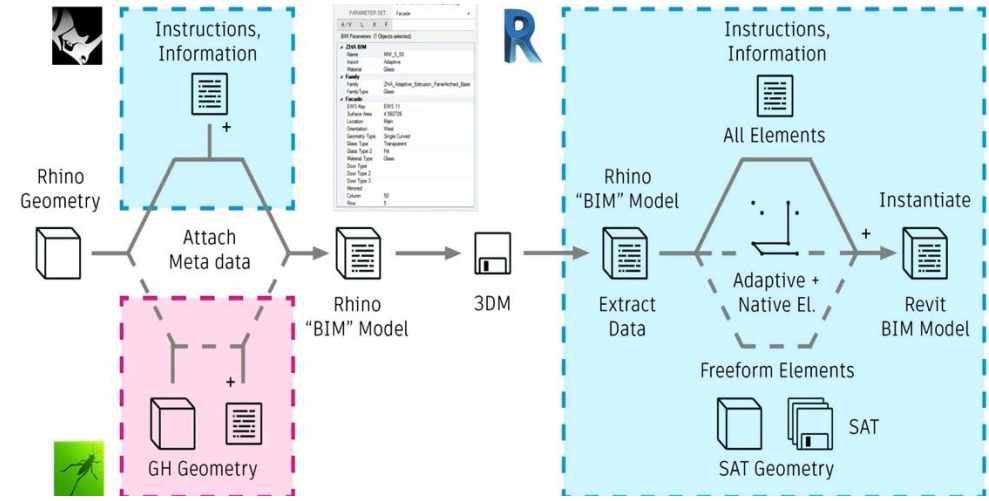


Figure 3-17. Data Flow from Rhino and Grasshopper to Revit

So, to work around this, the way they have initially been creating geometry with the help of Grasshopper, they attach a couple of additional nodes to the existing Grasshopper logic (Figure 3-18), to add Meta Data to the Rhino Model, which becomes the 'Rhino BIM Model'.

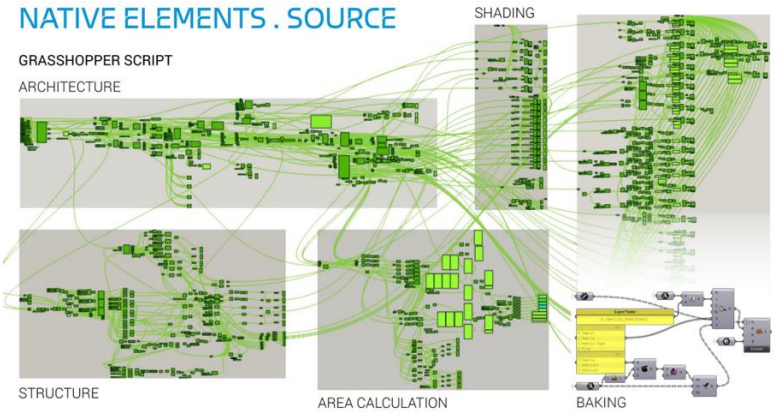
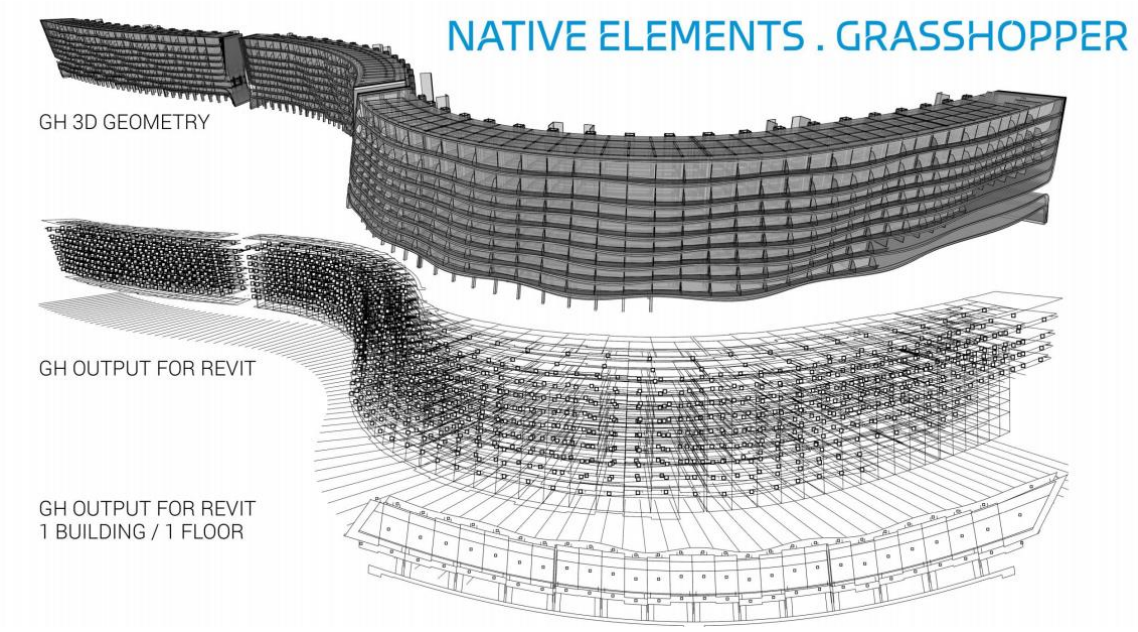


Figure 3-18. Grasshopper Logic to add Parametric Information to Rhino Geometry

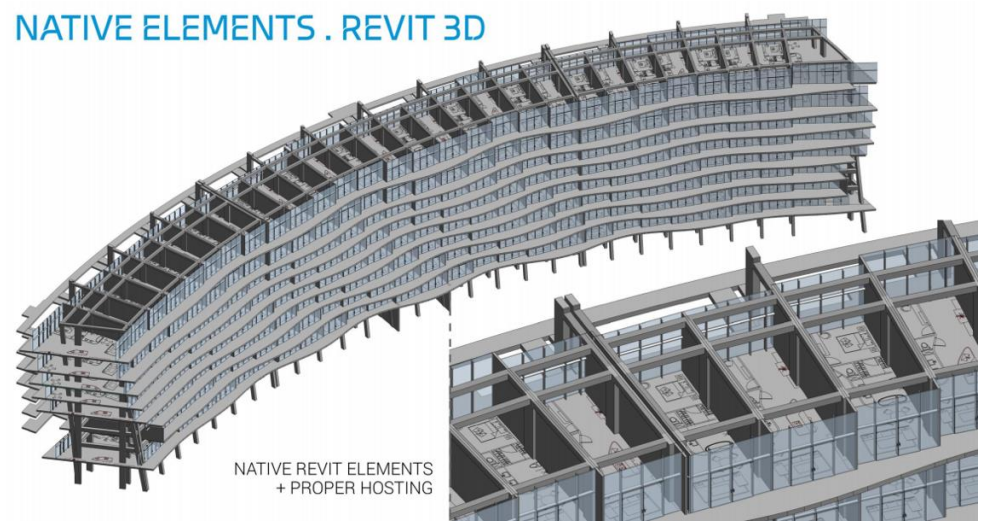
To utilize this Geometric information and the Meta Data that, is in Revit's canvas, ZHA with the help of Dynamo and ZHA's private Revit plugin, converts all the previous so-called Dead Geometry to full Parametric Revit Model elements. If they decided to

have more parameters for the elements, they would add it in the Grasshopper logic, and the ZHA plugin which is designed to convert this Geometry+MetaData into Parametric Revit Elements, is going to make the conversion.

To understand a step deeper on how the final Grasshopper output looks like before going into Revit's interface, look at Figure 3-19 A, the top shows the Grasshopper's 3D geometry, and before adding the Meta Data to GH (Grasshopper) elements, the elements are simplified. Simplification is breaking the GH geometry to a level where it has only what is needed for Revit as input data for Revit's parametric elements to replace this GH output. So, for example, for a wall to be placed only a line is enough, since Revit's wall element can be placed in with reference to the line (on edge, at core boundary, as the user wishes).



A



B

Figure 3-19. G A) Process of Revit elements being created, B) Final Revit out – fully parametric elements

The second image is the simplified GH output of the entire building that serves as input for Revit, the third image shows how simple the input is for the entire 1st floor of the building. The final Revit model after parsing the final GH output into Revit ZHA plugin is shown in Figure 3-19 B

CHAPTER 4
RESULTS

This research has shown Generative and Parametric ways of approaching generative technologies in architectural style and layout and style. Parametric modeling and design methods differentiate themselves out of additional generative approaches by how that they make it possible for a stepwise control across the proper execution throughout the design practice, that turns out to be used specially throughout designing investigation. Their responsiveness and adaptability to inside and outside stimulation, especially the dynamicity of their design procedure and different neurological requirements, create the processes for an ideal strategy whilst designing complex design settings (Experiment 1C – Tensile Tent). Additionally, simplifying exploitation of shape can be also very invaluable in performance-driven designing processes, easing the quick cycling of operation design and analysis synthesis within an incorporated course of action.

After reviewing the literature, conducting experiments, and the case studies from the previous chapters, it might possibly be reasoned that computers have penetrated the design process, and it is now part of this from the brand newest structure since it tends to emulate or expand the individual wisdom therefore that it helped generate a design layout strategy that is different from the traditional design practices. (Experiment 1B – Shell Star Pavilion).

This sort of designing clearly depends upon the algorithmic idea that is referred to as algorithmic style and layout and style. It empowers the part of the programmer to become changed from Architecture programming into Programming Architecture, as a way to be determined by calculations in the structure. It is necessary to have a background in programming possibly programming or Analysis of Algorithms and which

is needed to cope with all those programming languages VPLs or even TPLs. It was discovered that the contemporary day TPLs (Text Programming Language's) could be more productive than VPLs (Visual Programming Language's), since the modern TPLs offer many linguistic features that are designed to overcome the traditional TPL. Davis (2020) breaks it down as quality being greater than quantity in terms of design. Imagine a designer must produce a design proposal to a client. Depending on the size of the design team, resources, experience, the number of designs that are made would not be more than 10 or even less. Coming up with hundreds of design options with different design schemes is more comprehensive.

Generative Design allows hundreds of variations are created/generated to choose from, depending on the design schemes that are fed as inputs for the design algorithm. Algorithms do what they are intended to, they do not have an Artificial Intelligent NN (Neural Network) that would reiterate the algorithm itself for giving the best design. In fact, what is the best design for one person might not be the same, in most cases. So, there is no way that algorithms cannot differentiate between good ideas, bad ideas, better ideas. Programming an algorithm by defining which design scheme is seemed better, good, bad for the user, then it will do that, but which is not the case.

Generative Design Summary

Once assessing this phase, one might reason that Generative Design can be just a new system of designing based upon computers that act as part of their extraction procedure since they work as an instrument. Software for producing most varieties of designs and which can be accomplished by precisely delivering the programmer's notion in creating variants in every possible outcome based on the factors and variables

given as input, through a system architecture such as - parametric or formalisms that split to four different forms, L-Systems, Mobile Automata approach Fractal devices, Form Grammars.

Each of these procedures has an algorithmic foundation, and these foundations are the core for any generative design system. Utilizing this set of algorithms in design should be seen as an advantage, rather than look at as a hinderance. Because the fundamental of this approach is that they break the entire process into distinct tasks. This truly helps the user/designer to define the design goal more clearly. To map the entire real-world design into a series of steps is clearly not a simple task, but the by taking this approach the real benefit is that it lets the designer manifest the logic. But in case the outcome failed to match up with the designer's gratification, it could be simple for the programmer to accomplish a few changes from the composed calculations or at the worthiness of these factors to build most of the discretionary alternatives before the programmer chooses the desired outcome.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

The way to look at Generative Design in terms of Architectural design, is not as a tool that can replace the entire design process. Rather as a smart and intelligent way to iterate hundreds and thousands of design schemes in a relatively very short amount of time and review them by groups that seem fit to the user's needs. It accomplishes more design iterations in each time, but not to be interpreted as better designs.

Computational modeling might be a different approach to design things to a traditional designer, Architect. The learning curve for building nodes, and logic that will practically save time, and money is steep. Imagine a scenario, where there are couple of walls, that will be clashing with the structure. The designer is aware of this, and the team decides make openings in these walls, to avoid clashes with the structural components. Now, for documenting these changes (openings in these walls), the designers must first make openings in the model, then create sheet views or highlight these changes in the existing sheet view. For a couple of clashes, we can do it in a few hours or a day. But what if there are hundreds of walls that have-to-have openings, it is not at all reliable for the designer to go through them one by and one and make the changes.

Writing a Dynamo script, with the power of computational modeling, allows these specific structural elements, and make an offset around them, and these offset curves will serve as inputs for wall opening locations. Once the logic is built without errors, manipulating also becomes extremely easy, which would save double the time in case there are changes in the size of openings.

After reviewing the case studies, literature reviews, and carrying out experiments by building models and projects, the researcher recommends new and existing

designers, Architects, BIM engineers to get comfortable with computational modeling through Grasshopper and Dynamo, which have a huge design potential based. Learning to program, with high level languages is a great skill that will be handy in multiple phases of the project. Instead of looking for what the software has to offer, a programmer can bend the software's capabilities to solve his/her specific problems.

APPENDIX
SOFTWARE REFERENCES AND VERSIONS

AutoCAD 2021 - https:-//www.autodesk.com/products/autocad/overview

Revit 2021 - https:-//www.autodesk.com/products/revit/overview

Dynamo2.3, 2.5, 2.10 – Dynamo for Revit - https:-//dynamobim.org/

Navisworks2020 - https:-//www.autodesk.com/products/navisworks/overview

Inventor – https:-//www.autodesk.com/products/inventor

Project Refinery – https:-//www.autodesk.com/campaigns/refinery-beta

Rhino – Rhinoceros 3D – Version 6.0, 7.0

Rhino-Grasshopper – V6

CATIA – V2018

ArchiCAD - 2018

Autodesk – Inventor 2021

Python 3.9.2

Dynamo Packages - DynaShape, Mesh ToolKit3.0, Lunchbox

Models - https:-//uflorida-my.sharepoint.com/:-f:-/g/personal/skondamadugula1_ufl_edu/EqewD4h33wlMrQ_CkF1OtgsBJyMO6ZlIznfKBb8EEiExiA?e=SMHbWB

APPENDIX
LIST OF REFERENCES

Aghaei Meibodi, Mania 2012b. “Technological Advances in Design and Construction: Bridging the Gap between the Conception Stage and the Manufacturing Process.” In Essays on Construction, Annual International Conference on Construction, Athens Institute for Education and Research, Athens, Greece, October 2012.

Aish, R., Woodbury, R.(2005) Multi-level interaction in parametric design, in A. Butz, B. Fisher, A. Krüger and P. Oliver (eds.) SmartGraphics, 5th Int. Symp., SG2005, Lecture Notes in Computer Science, Springer, Berlin.

Ahlquist, S., & Menges, A. (2011). Computational Design Thinking. Computation Design Thinking. New Jersey:- John Wiley and Sons Ltd. The Building Information Modeling and its Use for Data Transformation in the Structural Design Stage - Zhao-Qiu Liu, Fei Zhang 1and Ji Zhang

BIM-Enabled Structural Design:- Impacts and Future Developments in Structural Modelling, Analysis and Optimisation Processes - Hung-Lin Chi · Xiangyu Wang · Yi Jiao

Noam Chomsky (1975) The Logical Structure of Linguistic Theory

Cross, N. (2001b) Designerly ways of knowing:- design discipline versus design science. Design issues, 17:-3; 49-55.

Demaine, Erik D., Martin L. Demaine, and Anna Lubiw. 1999. “Polyhedral Sculptures with Hyperbolic Paraboloids.” In Proceedings of the 2nd Annual Conference of BRIDGES:- Mathematical Connections in Art, Music, and Science, 91–100

Eisenman, P. D., Gass, W., & Gutman, R. (1977, June 6). House VI (Frank Residence) in Cornwall, Connecticut. Progressive Architecture, pp. 57-67.

El-Khaldi, M. (2007). Mapping boundaries of generative systems for design synthesis. Unpublished Master of Science Thesis. Cambridge, Massachusetts, USA:- MIT

Framework for Structural Design - Nawari O. Nawari, Michael Kuenstle

Gero, J., & Tyugu, E. (1994). Formal Design Methods for CAD. Amsterdam:- Elsevier.

Gursel, İ. D. (2012). Creative Design Exploration by Parametric Generative Systems In Architecture. Journal of the Faculty of Architecture, Middle East Technical University, 207-224.

Guidera, S. (2011). Conceptual Design Exploration in Architecture Using Parametric Generative Computing:- A case Study. Connecting Concepts in Sustainable Design and Digital Fabrication:- A Project-Based Learning Case Study (pp. 2728-2748). Bowling Green:- American Society for Engineering Education.

Hays, K. M. (2000). Architecture theory since 1968. Cambridge:- MIT Press.

Hensel, M., & Menges, A. (2008). Morpho-Ecologies (Towards an Inclusive Discourse on Heterogeneous Architecture). Architectural Association, 20

Hensel, M., Menges, A., & Weinstock, M. (2004). Emergence:- Morphogenetic Design Strategies. London:- Academy Press

Hensel, M., Menges, A., & Weinstock, M. (2010). Emergent technologies and design:- towards a biological paradigm for architecture. London:- Routledge

Kolarevic, B., & Malkawi, A. (2005). Performative Architecture:- Beyond Instrumentality. New York:- Routledge.

Knight, T., & Stiny, G. (2001). Classical and Non-Classical Computation. Architectural Research Quarterly, 355-372.

Neil Leach, (2009). Digital Morphogenesis. Architectural Design, 32-37

Louis Durand, (trans.) David Britt, Precis of the Lectures on Architecture:- with Graphic Portion of the Lectures on Architecture, (Getty Trust Publications:- Getty Research Institute for the History of Art and the Humanities, 2000).

LIU, Y.C., CHAKRABARTI, A., BLIGH, T. (2003) Towards an ideal approach for concept generation, Design Studies, 24:-4;341-55.

Mitchell, W. J. (1978). The Theoretical Foundation of Computer-Aided Architectural Design. Environment and Planning B Journal, 127-150.

Negroponte, N. Reflections on Computer Aids to Design and Architecture (pp. 17-26). New York:- Petrocelli / Charter

Rogers, Hartley Jr. 1967. The Theory of Recursive Functions and Effective Computability.

Runberger, Jonas. 2012. “Architectural Prototypes II:- Reformations, Speculations and Strategies in the Digital Design Field” PhD diss., KTH School of Architecture and the Built Environment, Royal Institute of Technology, Stockholm, Sweden.

Simon, H. A. (1969). The Sciences of the Artificial. Cambridge:- MIT Press.

Shea, K. (2004). Directed randomness. In N. Leach, D. Turnbull, & C. Williams, Digital Tectonics (pp. 10-23). London:- Academy Press

Tang, P.-Y., & Chang, T.-W. (2005). Mutating 3D Generative Form with Co-Evolve Approach. Computer Aided Architectural Design Research in Asia (pp. 266- 271). New Delhi:- CAAD.

Vincent, J. (2009). Biomimetic Patterns in Architectural Design. Architectural Design , 74-81.

Weinstock, M. (2010). The Architecture of Emergence:- The Evolution of Form in Nature and Civilization. Oxford:- Wiley & Sons.

BIOGRAPHICAL SKETCH

With a Bachelor in Architecture degree from Sathyabama Institute of Science and Technology, Siva found his passion for design. After working as a Junior Architect in India. He is pursuing his master's degree from the M. E. Rinker, Sr. School of Construction Management, University of Florida, USA. He was driven towards complex computational modeling problems and working on new methods to achieve solutions to such problems.

With some experience in front-end web development in his undergraduate studies, he used it as a bridge to learn Python. Currently he is working towards creating Python scripts that can automate the building construction documentation phase, to save time and money, while advancing his modeling skills in Revit – in Architectural and Structural discipline. Siva will be graduating with his Master of Science in Construction management in August 2021.